ADVANCED WEB PROGRAMMING
ROUTING, CONTROLLER, AND VIEW

KHEN MUHAMMAD CAHYO

244107023002

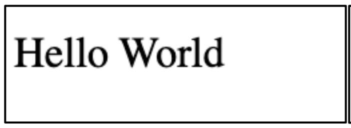
TI-2I

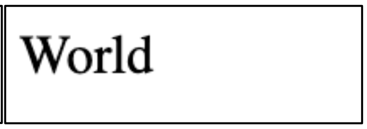
STATE POLYTECHNIC OF MALANG

INFORMATION TECHNOLOGY DEPARTMENT

PRACTICUM ROUTING

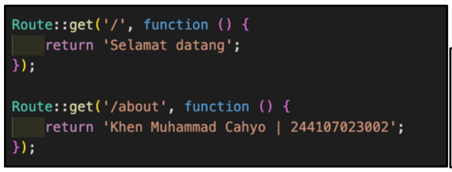
- **Basic Routing (b-e)**

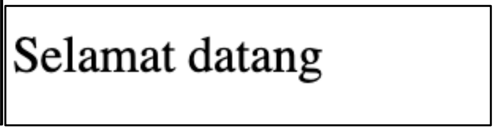

Hello World

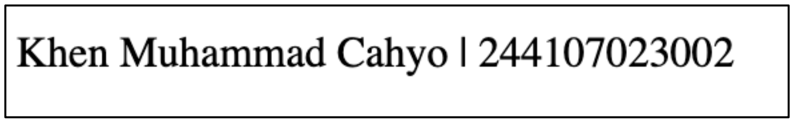

World

The code defines two routes in a Laravel application using the Route::get method, which handles HTTP GET requests. The first route listens for requests at the /hello URL and returns the string "Hello World", while the second route listens at /world and returns the string "World". These routes demonstrate how Laravel can handle multiple endpoints by mapping URLs to closures that return simple responses without requiring a controller.

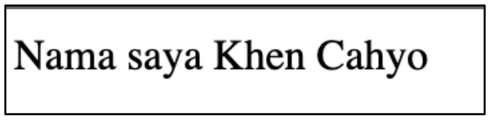
- **Basic Routing (f-g)**


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

Route::get('/about', function () {
    return 'Khen Muhammad Cahyo | 244107023002';
});
```


Selamat datang

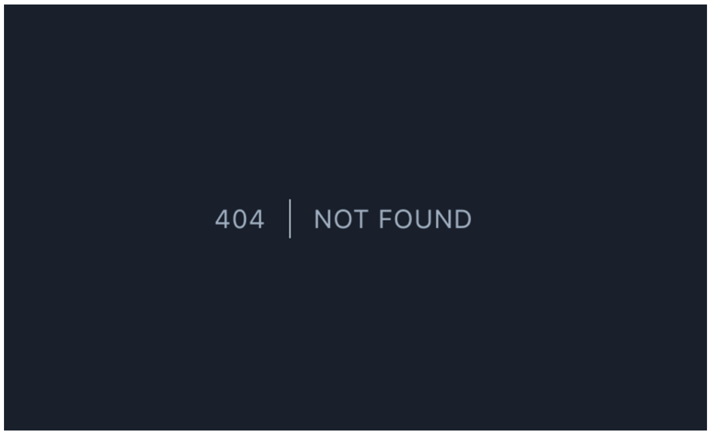

Khen Muhammad Cahyo | 244107023002

- **Route Parameters (a-b)**


Nama saya Khen Cahyo

The code defines a dynamic route in Laravel using Route::get, which listens for HTTP GET requests at the /user/{name} URL. The {name} part is a route parameter, meaning any value passed in that position will be captured and assigned to the $name variable. When the route is accessed (e.g., /user/KhenCahyo), Laravel will execute the provided closure and return the string "Nama saya KhenCahyo". This demonstrates how Laravel handles route parameters to create dynamic and customizable URLs.

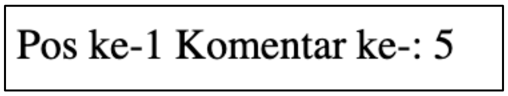
- **Route Parameters (c)**



```
404 | NOT FOUND
```

When we try to access /user without providing a {name} parameter, Laravel will return a 404 Not Found error page. This happens because the route is defined to require a {name} parameter, and without it, Laravel does not match the request to any defined route.

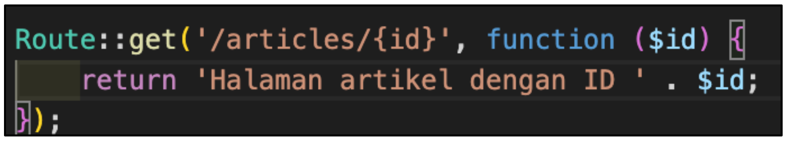
- **Route Parameters (d-e)**



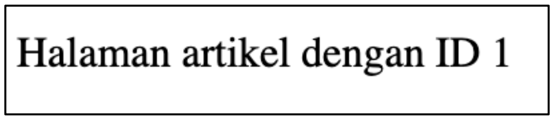
```
Pos ke-1 Komentar ke-: 5
```

The code defines a Laravel route that handles HTTP GET requests for URLs following the pattern /posts/{post}/comments/{comment}. It includes two route parameters: {post} and {comment}, which capture dynamic values from the URL. These values are then passed as arguments $postId and $commentId to the closure function. When accessed with a URL like /posts/3/comments/7, Laravel assigns 3 to $postId and 7 to $commentId, returning the response "Pos ke-3 Komentar ke-: 7". This demonstrates how Laravel supports multiple route parameters to handle nested resources dynamically. However, to ensure the function works correctly, the parameters inside the closure should match the route placeholders, meaning $post and $comment should be used instead of $postId and $commentId.

- **Route Parameters (f)**

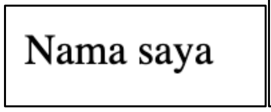


```
Route::get('/articles/{id}', function ($id) {
    return 'Halaman artikel dengan ID ' . $id;
});
```

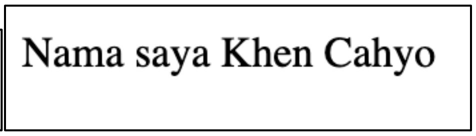


```
Halaman artikel dengan ID 1
```

- **Optional Parameters (a-c)**



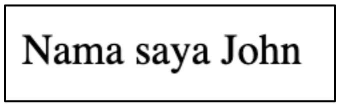
```
Nama saya
```



```
Nama saya Khen Cahyo
```

When visiting /user with no name provided, the {name?} parameter is optional, so the $name variable defaults to null, and the response will be "Nama saya " without any name. However, when visiting /user/Khen Cahyo, the {name} parameter is supplied with the value "Khen Cahyo", and the closure receives that value in the $name variable, resulting in the response "Nama saya Khen Cahyo". If you want to handle the case where no name is provided more gracefully, you can set a default value for $name, such as "Guest", so the response would be "Nama saya Guest" when accessing /user without a name.

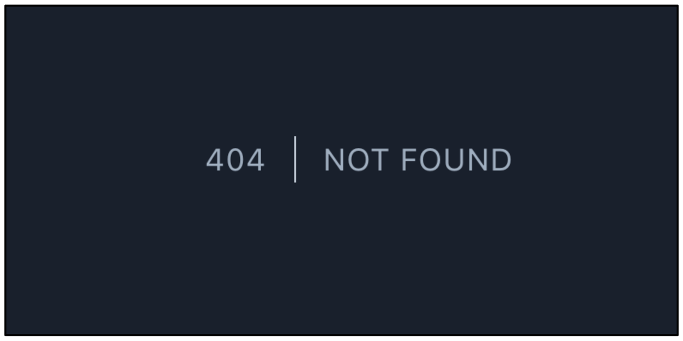
- **Optional Parameters (d-e)**



```
Nama saya John
```

The code defines a Laravel route that listens for HTTP GET requests at /user/{name?}, where {name?} is an optional route parameter. The $name variable in the closure has a default value of 'John', so if no name is provided in the URL (e.g., /user), the response will be "Nama saya John". If a name is provided (e.g., /user/Khen Cahyo), the route will use the provided name, and the response will be "Nama saya Khen Cahyo". This setup ensures that a default name is used when the parameter is not given, providing a more user-friendly fallback.

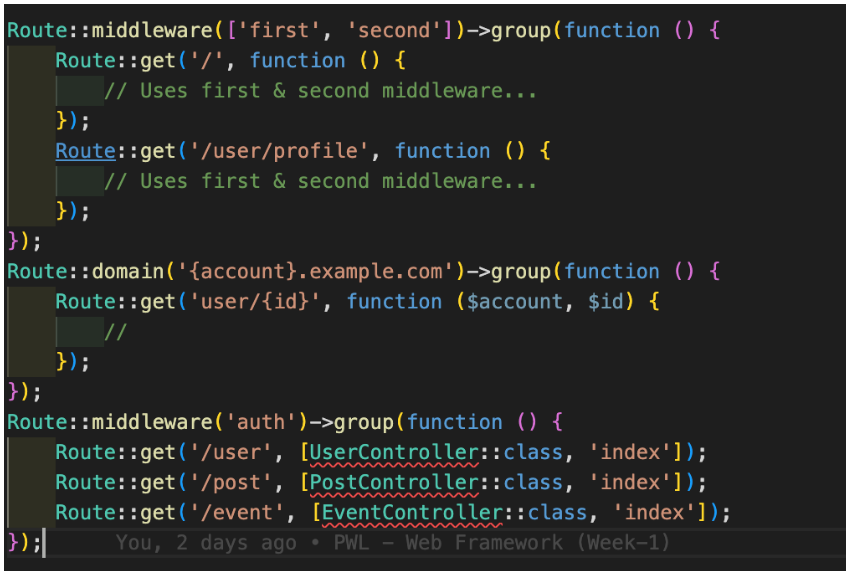
- **Route Name (a-b)**



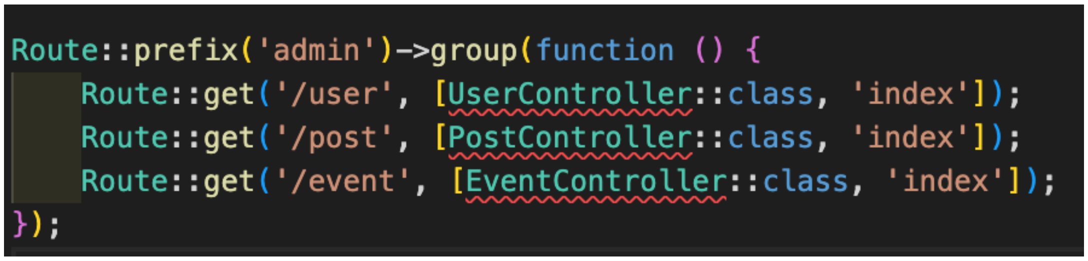
```
404 | NOT FOUND
```

The code defines a Laravel route that listens for HTTP GET requests at the /user/profile URL and associates it with a named route profile using the name('profile') method. The route currently does not return any response because the closure is empty. When trying to visit the URL /profile, it will not match this route, as it is specifically defined for /user/profile. If you want to access the route with /profile, you would need to change the URL pattern in the route definition to /profile or adjust your link to point to /user/profile.

- **Route Group and Route Prefixes**

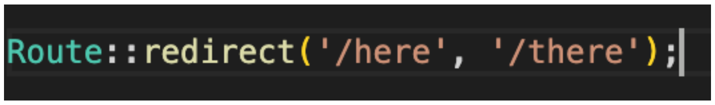


```
Route::middleware(['first', 'second'])->group(function () {
    Route::get('/', function () {
        // Uses first & second middleware...
    });
    Route::get('/user/profile', function () {
        // Uses first & second middleware...
    });
});
Route::domain('{account}.example.com')->group(function () {
    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']);
});      You, 2 days ago • PWL - Web Framework (Week-1)
```



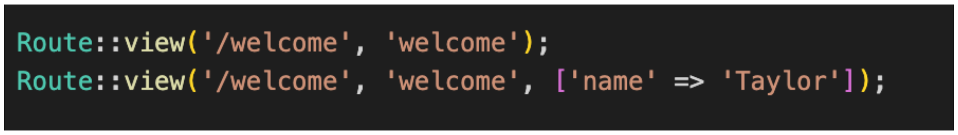
```
Route::prefix('admin')->group(function () {
    Route::get('/user', [UserController::class, 'index']);
    Route::get('/post', [PostController::class, 'index']);
    Route::get('/event', [EventController::class, 'index']);
});
```

- **Route Redirect**



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

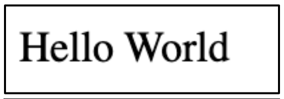
- **View Routes**



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

PRACTICUM CONTROLLER

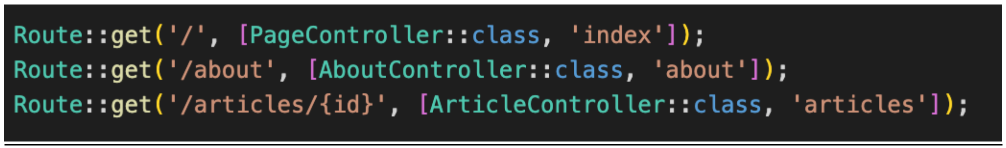
- **Creating Controller (a-e)**



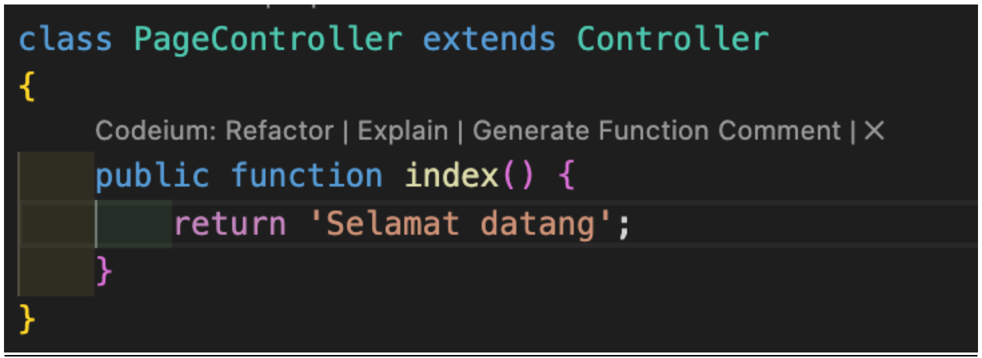
```
Hello World
```

The code defines a route in Laravel that listens for HTTP GET requests at the /hello URL. Instead of using a closure, it maps the route to the hello method in the WelcomeController class using the WelcomeController::class syntax. When the route is accessed, Laravel invokes the hello method within the WelcomeController, which returns the string "Hello World". This demonstrates how to organize application logic in a controller rather than using closures directly in the route definition.

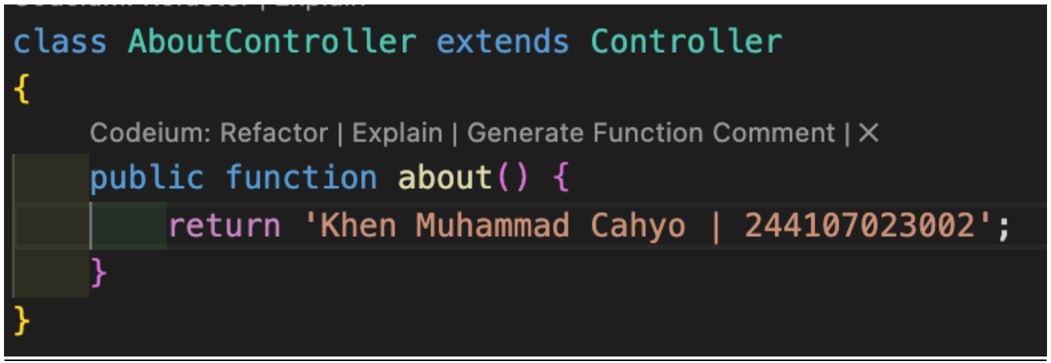
- **Creating Controller (f-g)**



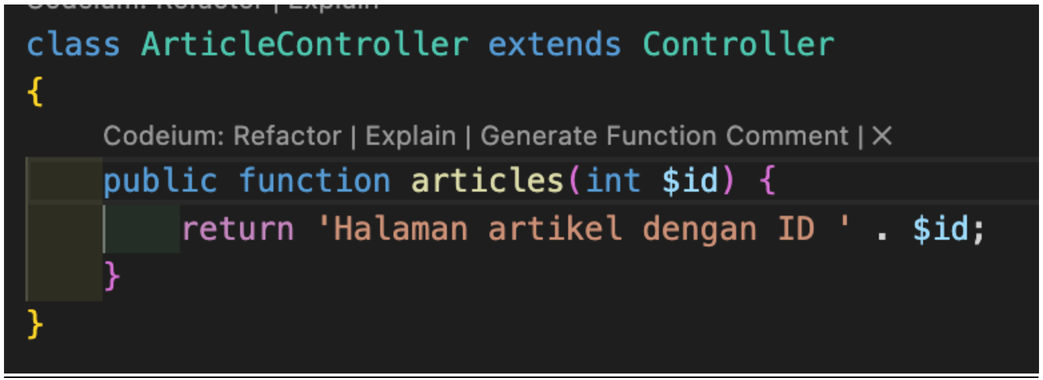
```php
Route::get('/', [PageController::class, 'index']);
Route::get('/about', [AboutController::class, 'about']);
Route::get('/articles/{id}', [ArticleController::class, 'articles']);
```



```php
class PageController extends Controller
{
    Codeium: Refactor | Explain | Generate Function Comment | X
    public function index() {
        return 'Selamat datang';
    }
}
```

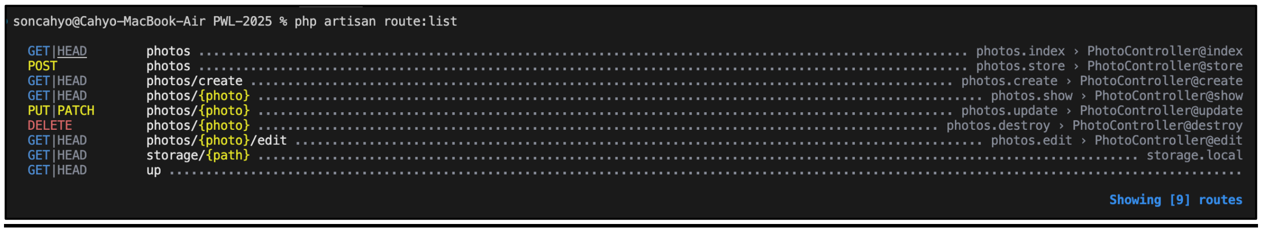


```
class AboutController extends Controller
{
    Codeium: Refactor | Explain | Generate Function Comment | ✕
    public function about() {
        return 'Khen Muhammad Cahyo | 244107023002';
    }
}
```

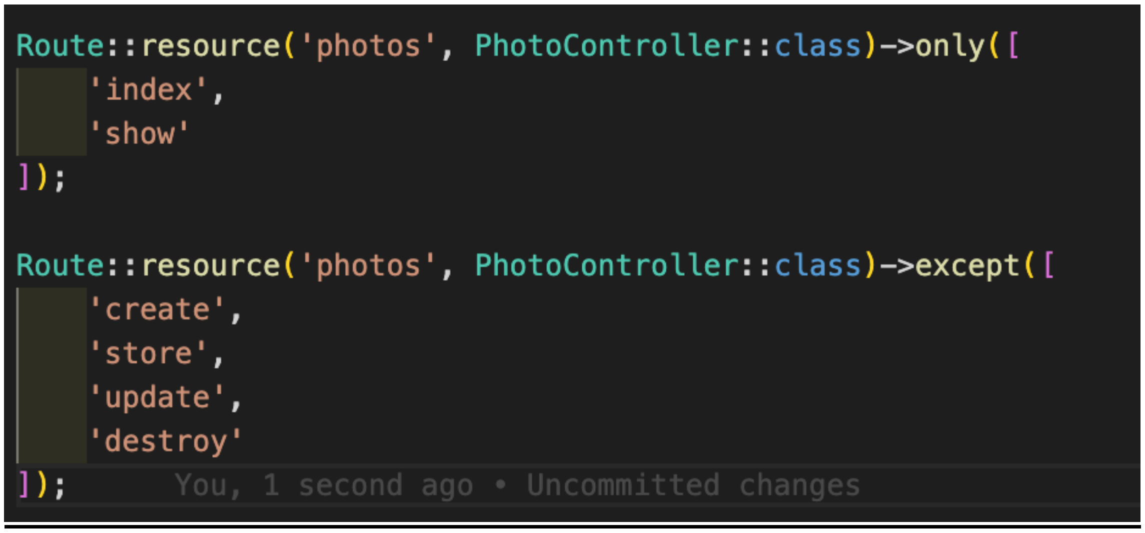


```
class ArticleController extends Controller
{
    Codeium: Refactor | Explain | Generate Function Comment | ✕
    public function articles(int $id) {
        return 'Halaman artikel dengan ID ' . $id;
    }
}
```

- **Resource Controller (a-d)**



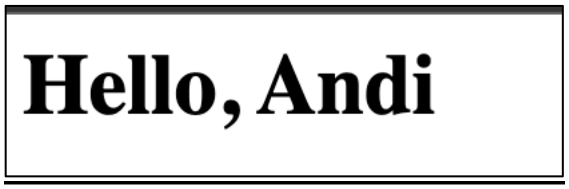
```
soncahyo@Cahyo-MacBook-Air PWL-2025 % php artisan route:list

  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        storage/{path} ........................................................................................................ storage.local
  GET|HEAD        up ...........................................................................................................................

                                                                                                                                          Showing [9] routes
```



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

Route::resource('photos', PhotoController::class)->except([
    'create',
    'store',
    'update',
    'destroy'
]);        You, 1 second ago • Uncommitted changes
```

PRACTICUM VIEW

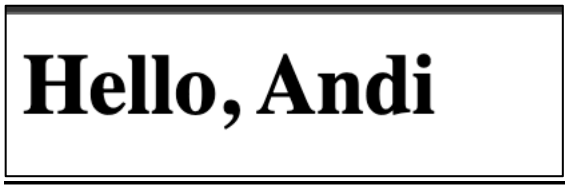
- **Creating View (a-c)**



Hello, Andi

The code defines a route in Laravel that listens for HTTP GET requests at the /greeting URL. When accessed, the route returns a view named hello from the resources/views/hello.blade.php file. The view is passed a variable name with the value 'Andi'. Inside the Blade view, the variable {{ $name }} is used to dynamically display the value of name, so the resulting HTML will show "Hello, Andi" in an <h1> element. This demonstrates how to pass data from a route to a Blade view for dynamic content rendering.

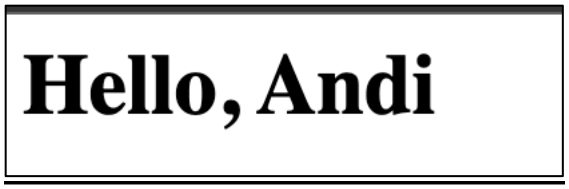
- **View Inside Directory (a-d)**



Hello, Andi

The updated code defines a route in Laravel that listens for HTTP GET requests at the /greeting URL. When accessed, it returns a view named hello located in the blog folder inside the resources/views directory, specifically at resources/views/blog/hello.blade.php. The view is passed a variable name with the value 'Andi', which is used in the Blade template to dynamically display the name. The Blade syntax {{ $name }} in the view will output "Hello, Andi". This demonstrates how to organize views in subdirectories within the resources/views folder and pass data to those views.

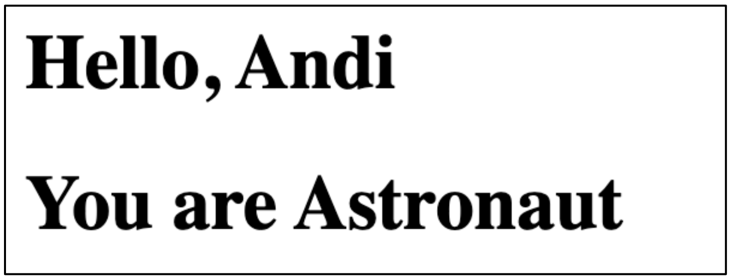
- **Showing View From Controller (a-c)**



Hello, Andi

The code defines a Laravel route that listens for HTTP GET requests at the /greeting URL. Instead of using a closure, the route maps to the greeting method in the WelcomeController class. In the greeting method, the view() function is used to return the hello view located in the blog folder (resources/views/blog/hello.blade.php), passing a variable name with the value 'Andi'. Inside the Blade view, the {{ $name }} syntax dynamically displays the name, so the rendered HTML will show "Hello, Andi." This demonstrates how to structure a controller method to handle route logic and pass data to a Blade view.

- **Forwards Data to View (a-c)**



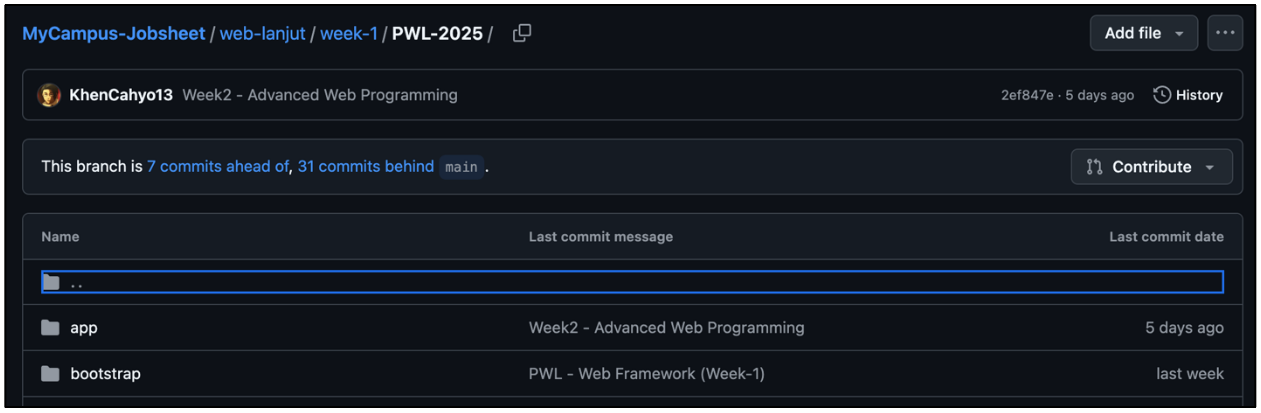
Hello, Andi

You are Astronaut

The code defines a Blade view in resources/views/blog/hello.blade.php that expects two variables: name and occupation. The greeting method in the controller returns this view and uses the with() method to pass the variables to it—'name' with the value 'Andi' and 'occupation' with the value 'Astronaut'. In the Blade template, the variables are displayed using the {{ $name }} and {{ $occupation }} syntax, which will output "Hello, Andi" and "You are Astronaut" respectively in <h1> tags when the view is rendered. This shows how to pass multiple pieces of data from the controller to a Blade view using the with() method.

PRACTICAL QUESTIONS

ANSWER QUESTION 1



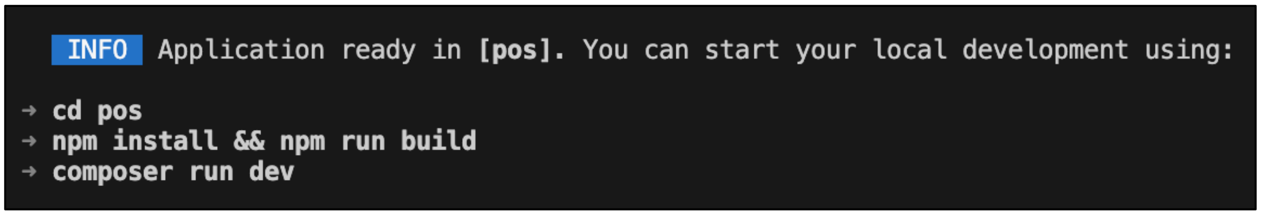
MyCampus-Jobsheet / web-lanjut / week-1 / PWL-2025 /

KhenCahyo13 Week2 - Advanced Web Programming 2ef847e · 5 days ago History

This branch is 7 commits ahead of, 31 commits behind main.

Name	Last commit message	Last commit date
..		
app	Week2 - Advanced Web Programming	5 days ago
bootstrap	PWL - Web Framework (Week-1)	last week

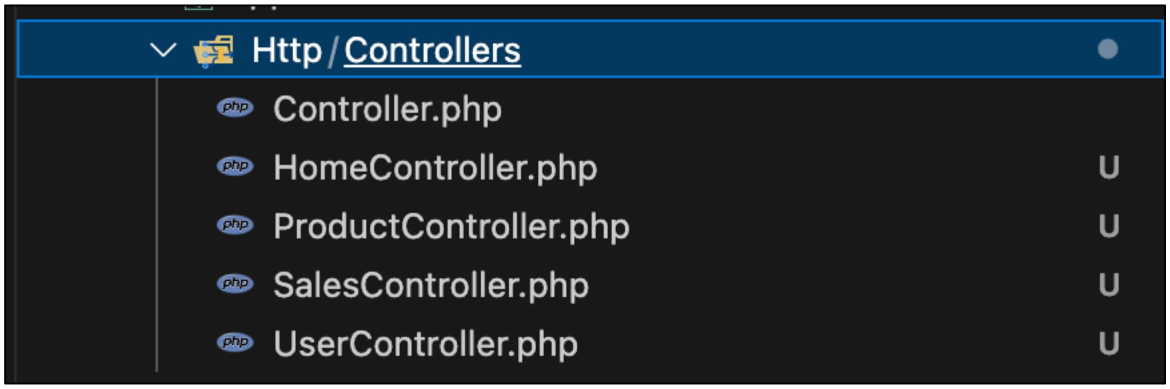
ANSWER QUESTION 2



```
  INFO  Application ready in [pos]. You can start your local development using:

➜ cd pos
➜ npm install && npm run build
➜ composer run dev
```

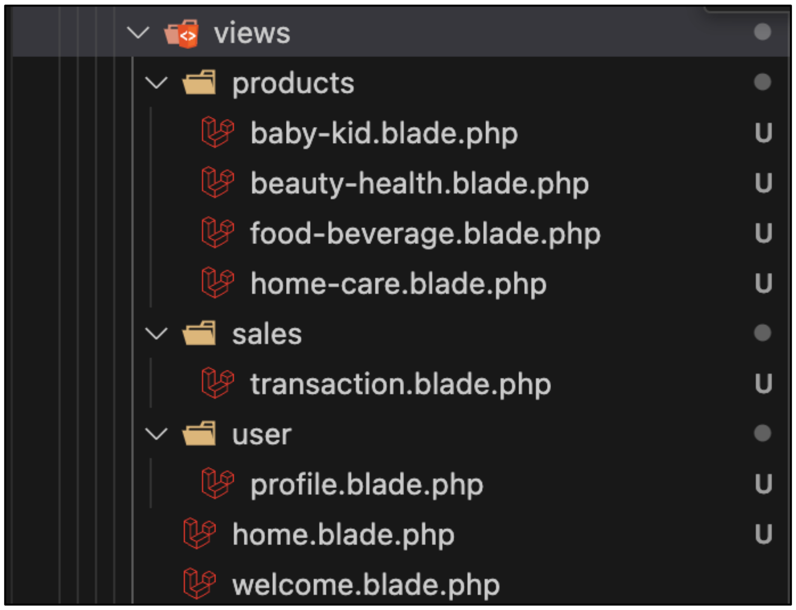
ANSWER QUESTION 3

- **Creating Controller**



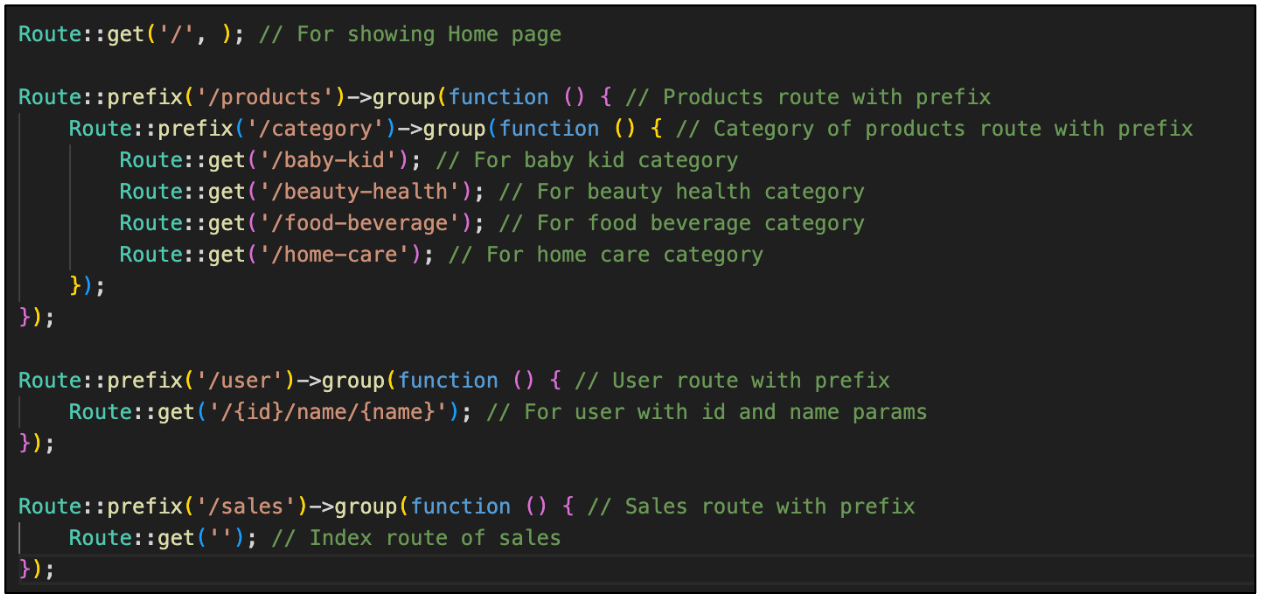
```
Http / Controllers
    Controller.php
    HomeController.php        U
    ProductController.php     U
    SalesController.php       U
    UserController.php        U
```

- **Creating View**



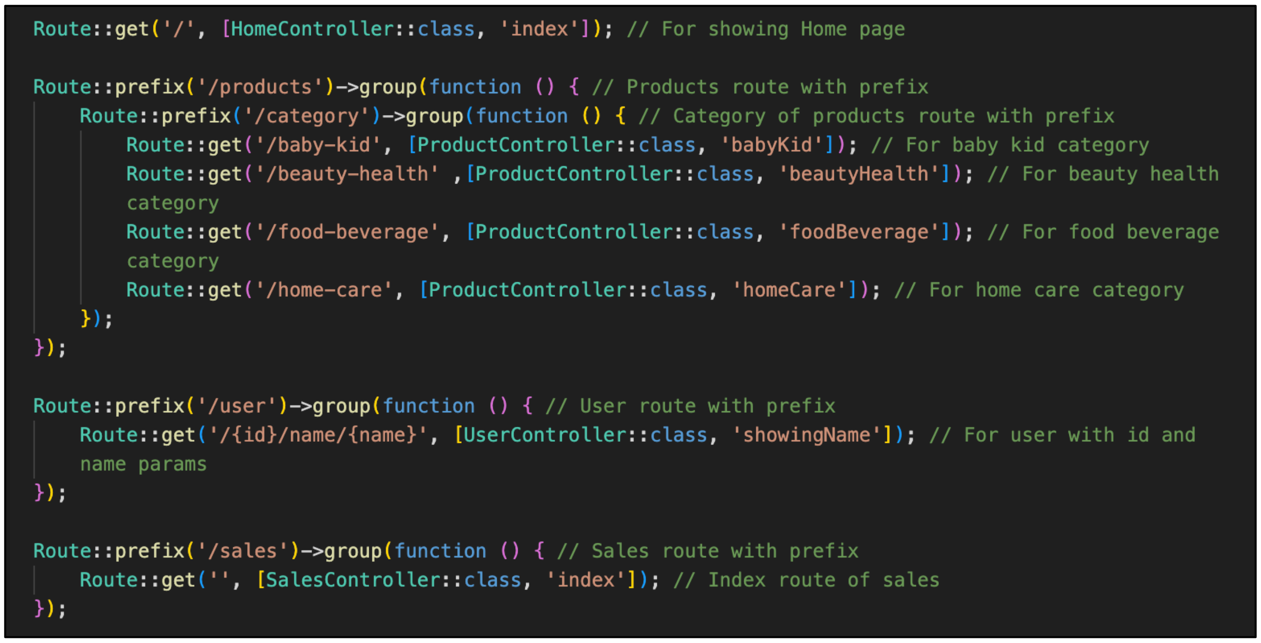
```
views
  products
    baby-kid.blade.php            U
    beauty-health.blade.php       U
    food-beverage.blade.php       U
    home-care.blade.php           U
  sales
    transaction.blade.php         U
  user
    profile.blade.php             U
  home.blade.php                  U
  welcome.blade.php
```

- **Creating Route**



```php
Route::get('/', ); // For showing Home page

Route::prefix('/products')->group(function () { // Products route with prefix
    Route::prefix('/category')->group(function () { // Category of products route with prefix
        Route::get('/baby-kid'); // For baby kid category
        Route::get('/beauty-health'); // For beauty health category
        Route::get('/food-beverage'); // For food beverage category
        Route::get('/home-care'); // For home care category
    });
});

Route::prefix('/user')->group(function () { // User route with prefix
    Route::get('/{id}/name/{name}'); // For user with id and name params
});

Route::prefix('/sales')->group(function () { // Sales route with prefix
    Route::get(''); // Index route of sales
});
```

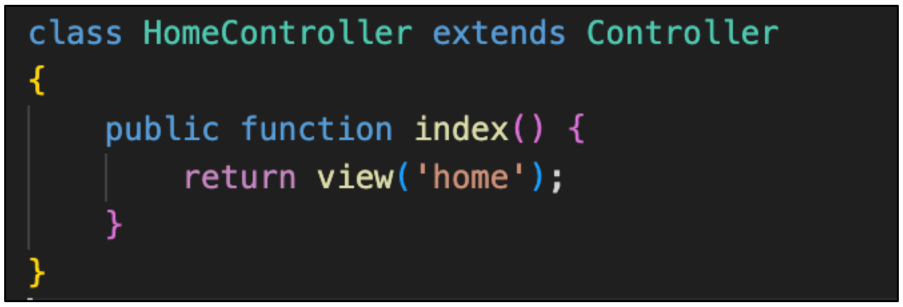
ANSWER QUESTION 4



```
Route::get('/', [HomeController::class, 'index']); // For showing Home page

Route::prefix('/products')->group(function () { // Products route with prefix
    Route::prefix('/category')->group(function () { // Category of products route with prefix
        Route::get('/baby-kid', [ProductController::class, 'babyKid']); // For baby kid category
        Route::get('/beauty-health' ,[ProductController::class, 'beautyHealth']); // For beauty health
        category
        Route::get('/food-beverage', [ProductController::class, 'foodBeverage']); // For food beverage
        category
        Route::get('/home-care', [ProductController::class, 'homeCare']); // For home care category
    });
});

Route::prefix('/user')->group(function () { // User route with prefix
    Route::get('/{id}/name/{name}', [UserController::class, 'showingName']); // For user with id and
    name params
});

Route::prefix('/sales')->group(function () { // Sales route with prefix
    Route::get('', [SalesController::class, 'index']); // Index route of sales
});
```

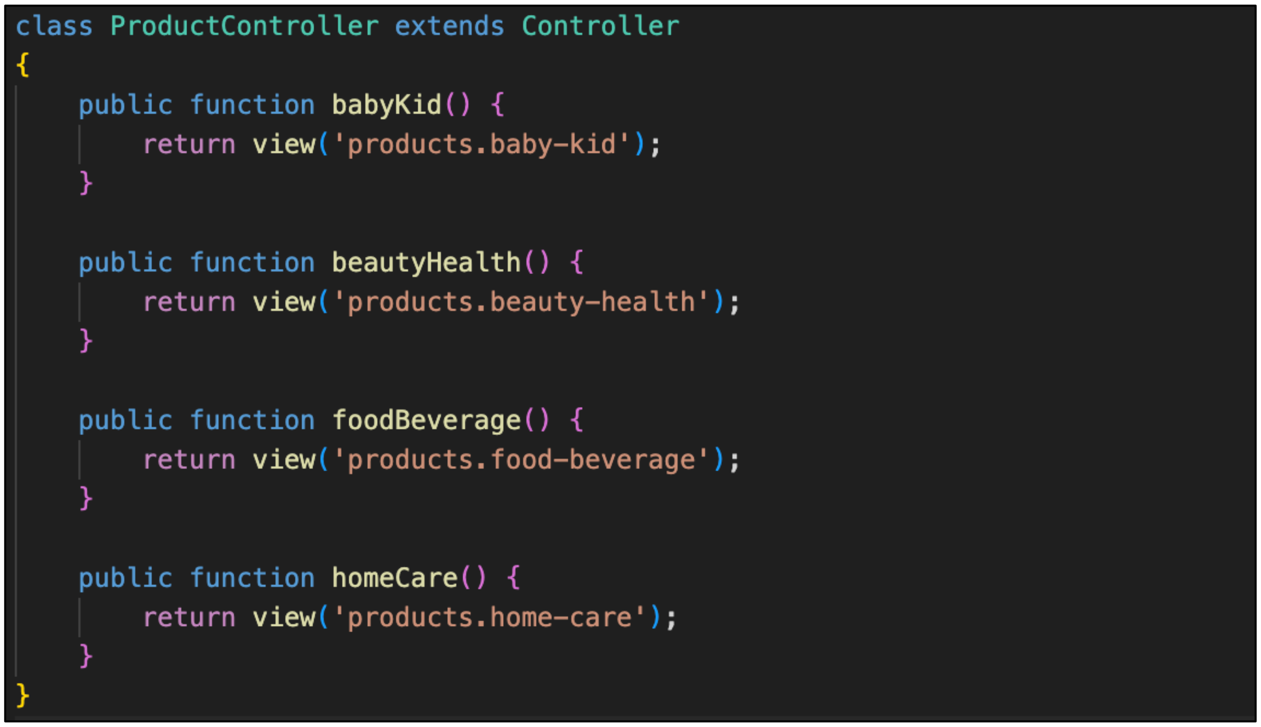
ANSWER QUESTION 5

- **Home Controller**



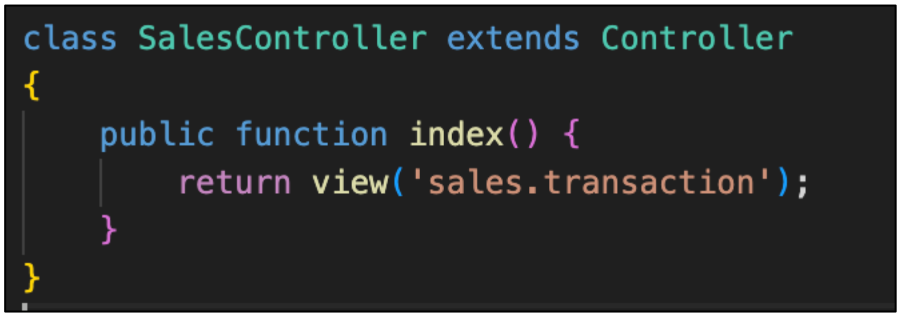
```
class HomeController extends Controller
{
    public function index() {
        return view('home');
    }
}
```

- **Product Controller**



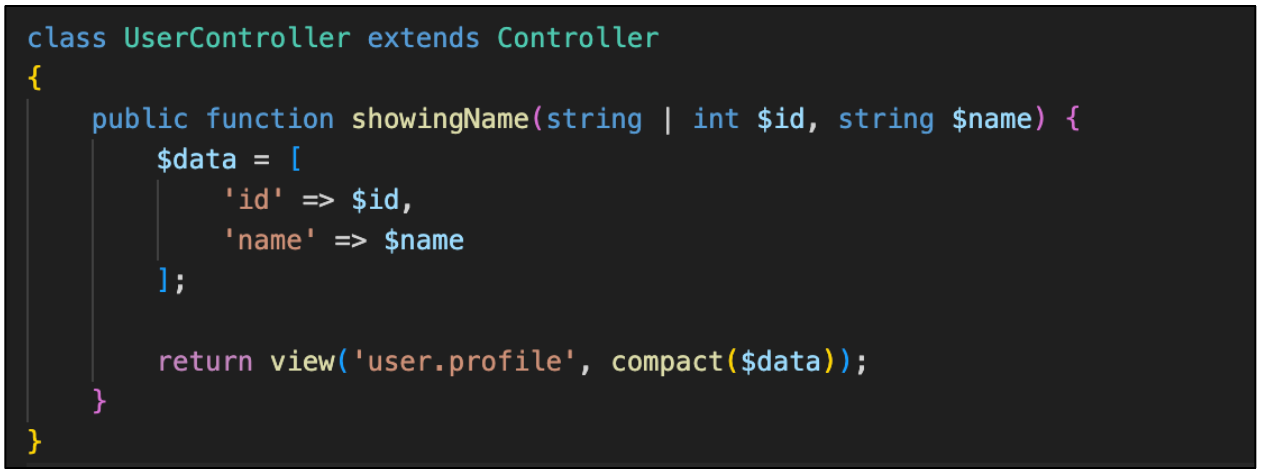
```php
class ProductController extends Controller
{
    public function babyKid() {
        return view('products.baby-kid');
    }

    public function beautyHealth() {
        return view('products.beauty-health');
    }

    public function foodBeverage() {
        return view('products.food-beverage');
    }

    public function homeCare() {
        return view('products.home-care');
    }
}
```

- **Sales Controller**



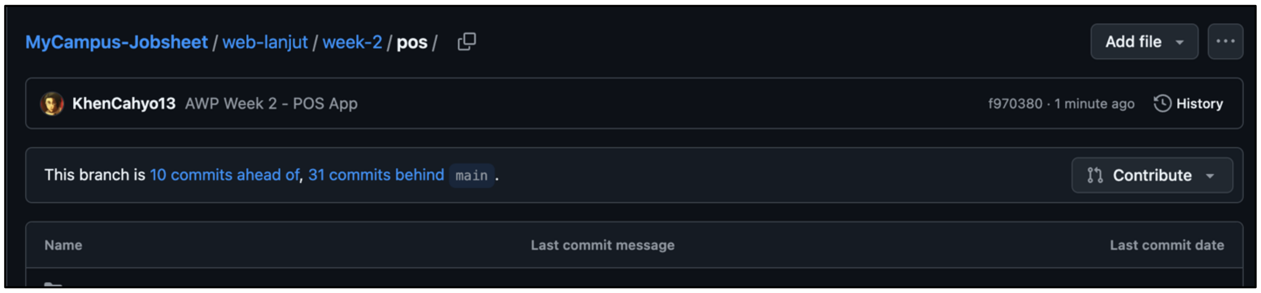
```php
class SalesController extends Controller
{
    public function index() {
        return view('sales.transaction');
    }
}
```

- **User Controller**



```php
class UserController extends Controller
{
    public function showingName(string | int $id, string $name) {
        $data = [
            'id' => $id,
            'name' => $name
        ];

        return view('user.profile', compact($data));
    }
}
```

ANSWER QUESTION 6



MyCampus-Jobsheet / web-lanjut / week-2 / pos /

KhenCahyo13 AWP Week 2 - POS App

f970380 · 1 minute ago

This branch is 10 commits ahead of, 31 commits behind `main`.