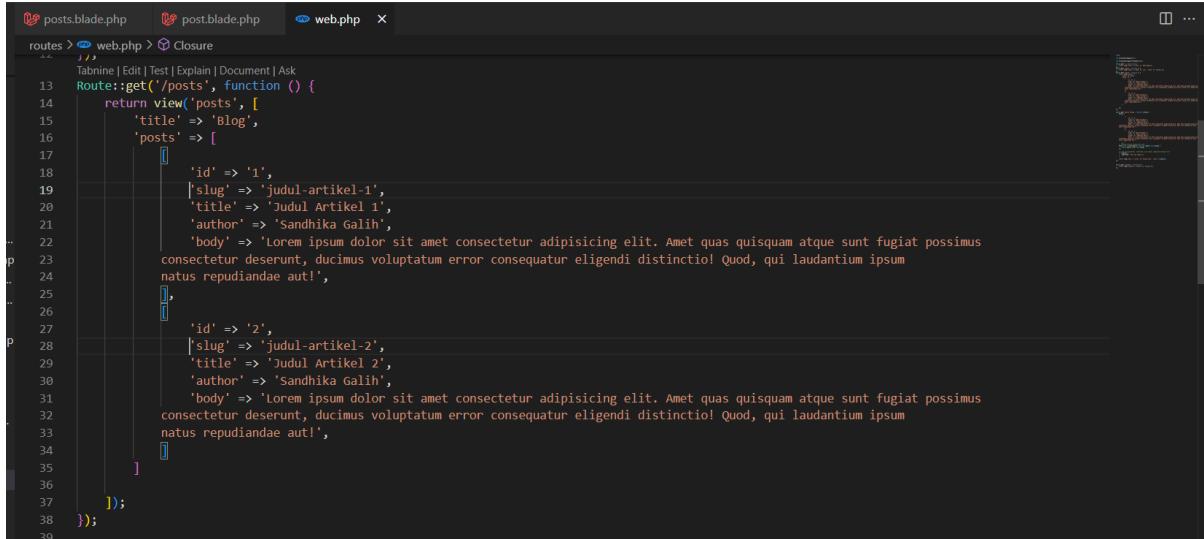
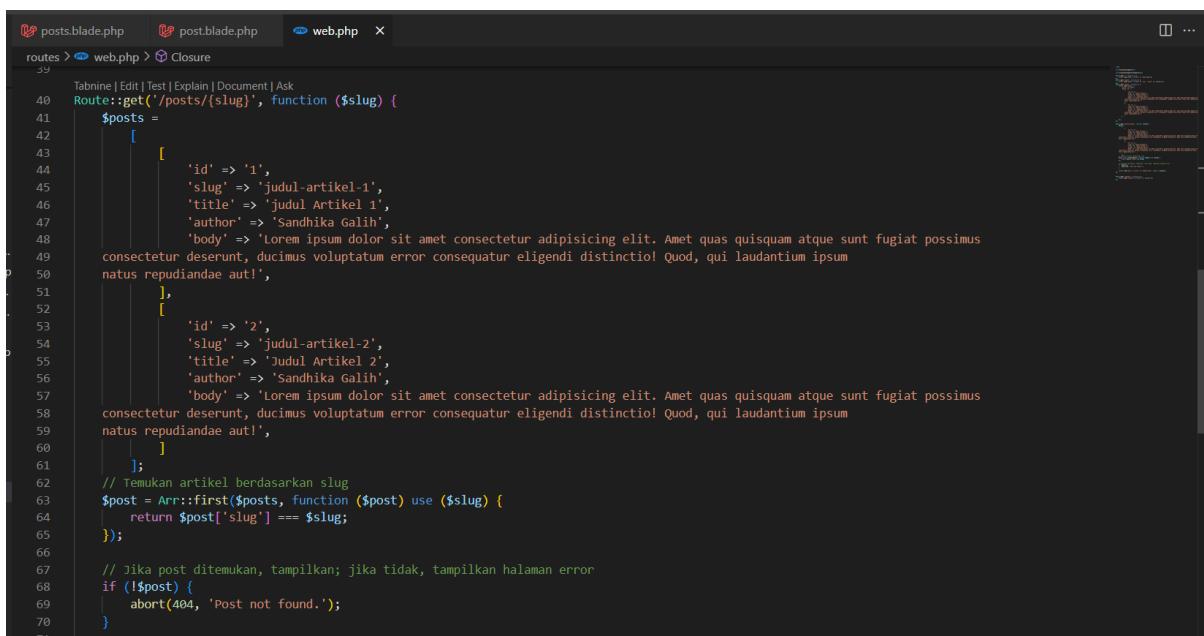


MODEL



```
routes > web.php > Closure
routes > web.php > Closure
Tabnine | Edit | Test | Explain | Document | Ask
13 Route::get('/posts', function () {
14     return view('posts', [
15         'title' => 'Blog',
16         'posts' => [
17             [
18                 'id' => '1',
19                 'slug' => 'judul-artikel-1',
20                 'title' => 'judul Artikel 1',
21                 'author' => 'Sandhika Galih',
22                 'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus
consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum
natus repudiandae aut!',
23             ],
24             [
25                 'id' => '2',
26                 'slug' => 'judul-artikel-2',
27                 'title' => 'judul Artikel 2',
28                 'author' => 'Sandhika Galih',
29                 'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus
consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum
natus repudiandae aut!',
30             ],
31         ],
32     ]);
33 });
34 });
35 });
36 });
37 });
38 });
39 );
```



```
routes > web.php > Closure
routes > web.php > Closure
Tabnine | Edit | Test | Explain | Document | Ask
39
40 Route::get('/posts/{slug}', function ($slug) {
41     $posts =
42     [
43         [
44             'id' => '1',
45             'slug' => 'judul-artikel-1',
46             'title' => 'judul Artikel 1',
47             'author' => 'Sandhika Galih',
48             'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus
consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum
natus repudiandae aut!',
49         ],
50         [
51             'id' => '2',
52             'slug' => 'judul-artikel-2',
53             'title' => 'judul Artikel 2',
54             'author' => 'Sandhika Galih',
55             'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus
consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum
natus repudiandae aut!',
56         ],
57     ];
58     // Temukan artikel berdasarkan slug
59     $post = Arr::first($posts, function ($post) use ($slug) {
60         return $post['slug'] === $slug;
61     });
62     // Jika post ditemukan, tampilkan; jika tidak, tampilkan halaman error
63     if (!$post) {
64         abort(404, 'Post not found.');
65     }
66 });
67 });
68 });
69 });
70 });
```

1. Memperbaiki route agar menggunakan 1 sumber data.

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows the project structure under "LARAVEL11". Key files include:
 - routes/web.php:** Contains the main route definitions.
 - post.blade.php:** A Blade template file.
 - web.php 2:** Another version of the web route file.
- routes/web.php Content:**

```

1 <?php
2
3 use Illuminate\Support\Arr;
4 use Illuminate\Support\Facades\Route;
5
6 class Post{
7     Tabnine | Edit | Test | Explain | Document | Ask
8     public static function all()
9     {
10         return [
11             [
12                 'id' => '1',
13                 'slug' => 'judul-artikel-1',
14                 'title' => 'Judul Artikel 1',
15                 'author' => 'Sandhika Galih',
16                 'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus
17                 consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum
18                 natus repudiandae aut!',
19             ],
20             [
21                 'id' => '2',
22                 'slug' => 'judul-artikel-2',
23                 'title' => 'Judul Artikel 2',
24                 'author' => 'Sandhika Galih',
25                 'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus
26                 consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum
27                 natus repudiandae aut!',
28             ]
29         ];
30     }
31 }
32 
```
- Route Definitions:**
 - `Route::get('/', function () { return view('home', ['title' => 'Home Page']); });`
 - `Route::get('/about', function () { return view('about', ['nama' => 'Lia', 'title' => 'Contact']); });`
 - `Route::get('/posts', function () { return view('posts', ['title' => 'Blog', 'posts' => Post::all()]);}); syntax error, unexpected token "]"`
 - `Route::get('/posts/{slug}', function ($slug) { $posts = [
51 [
52 'id' => '1',
53 'slug' => 'judul-artikel-1',
54 'title' => 'judul Artikel 1',
55 'author' => 'Sandhika Galih',
56 'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus
57 consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum
58 natus repudiandae aut!',]]; });`

2. Dengan membuat sebuah class yang nanti akan mengembalikan data array posts.

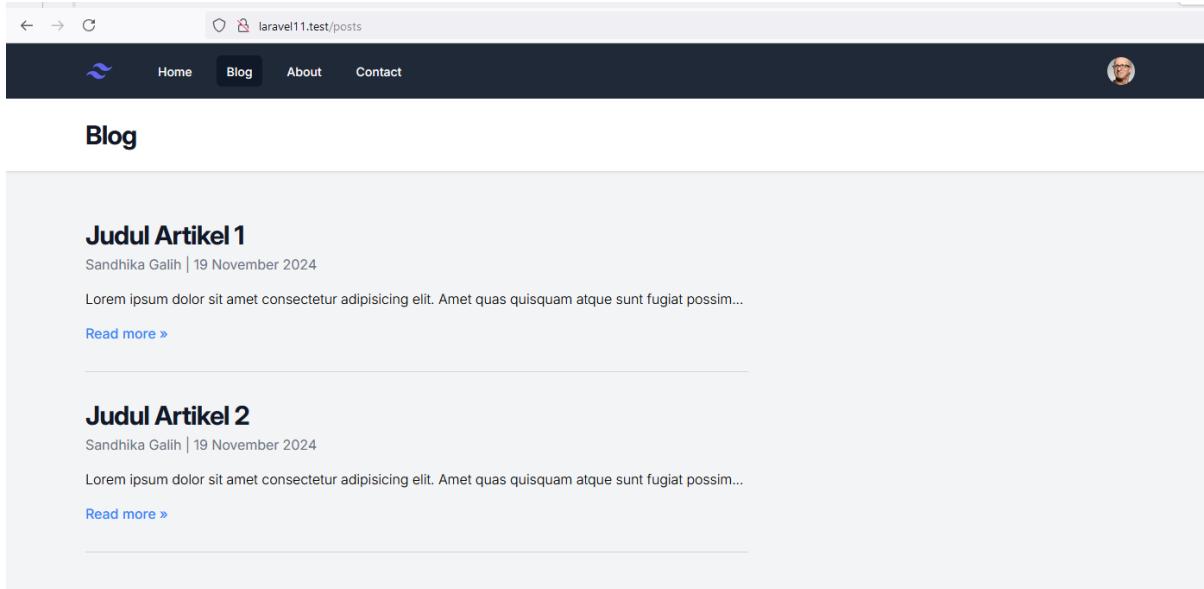
The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows the project structure under "LARAVEL11". Key files include:
 - routes/web.php:** Contains the main route definitions.
 - post.blade.php:** A Blade template file.
 - web.php 2:** Another version of the web route file.
- Post Class Definition:**

```

Tabnine | Edit | Test | Explain | Document | Ask
Route::get('/posts', function () {
    return view('posts', [
        'title' => 'Blog',
        'posts' => Post::all()
    ]);
}); 
```

3. Data yang dikirim kita ambil dari class Post dengan memanggil method staticnya ::all().



4. Yang mana kalau dijalankan masih sama.

```
Tabnine | Edit | Test | Explain | Document | Ask
Route::get('/posts/{slug}', function ($slug) {
    // Temukan artikel berdasarkan slug
    $post = Arr::first(Post::all(), function ($post) use ($slug) {
        return $post['slug'] === $slug;
    });

    // Jika post ditemukan, tampilkan; jika tidak, tampilkan halaman error
    if (!$post) {
        abort(404, 'Post not found.');
    }

    return view('post', ['title' => 'Single Post', 'post' => $post]);
});
```

A screenshot of a web browser window. The address bar shows 'laravel11.test/posts/judul-artikel-1'. The page title is 'Single Post'. The main content area displays a post titled 'Judul Artikel 1' by 'Sandhika Galih'. The post content is placeholder text: 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum natus repudiandae aut!'. Below the post is a link '« Back To Posts'.

Single Post

Judul Artikel 1

Sandhika Galih

Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum natus repudiandae aut!

[« Back To Posts](#)

A screenshot of a web browser window. The address bar shows 'laravel11.test/posts'. The page title is 'Blog'. The main content area displays two posts. The first post is titled 'Judul Artikel 1' by 'Sandhika Galih | 19 November 2024'. The post content is placeholder text: 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...'. Below the post is a link 'Read more »'. The second post is titled 'Judul Artikel 2' by 'Sandhika Galih | 19 November 2024'. The post content is placeholder text: 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...'. Below the post is a link 'Read more »'.

Blog

Judul Artikel 1

Sandhika Galih | 19 November 2024

Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...

[Read more »](#)

Judul Artikel 2

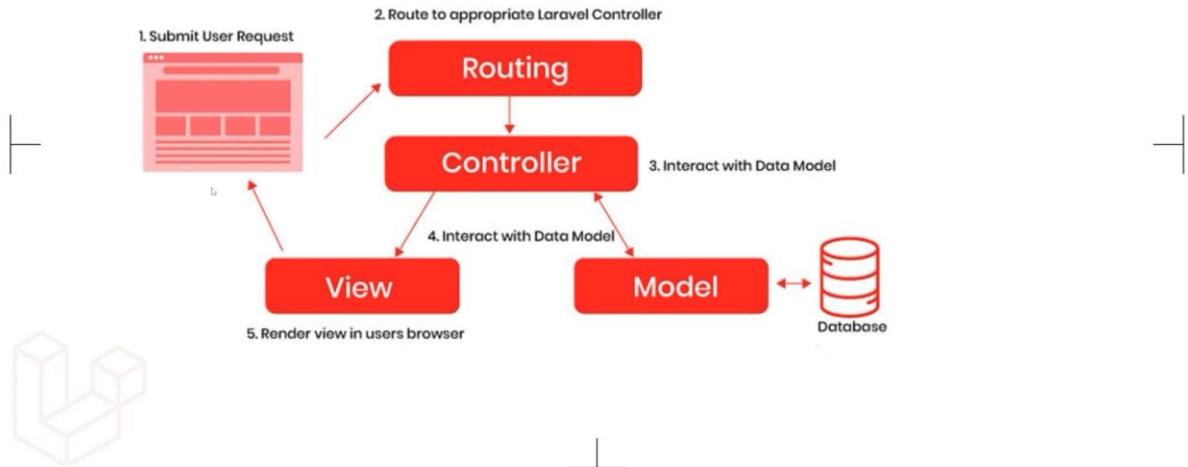
Sandhika Galih | 19 November 2024

Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...

[Read more »](#)

5. Kita lakukan hal yang sama jika mau mengambil data di single post nya. Namun ada 1 masalah, yaitu kita tidak mungkin menaruh class post di route yang mana peruntukannya juga sudah berbeda. Route digunakan untuk penjaluran dari request.

MODEL VIEW CONTROLLER (MVC)



6. Berdasarkan konsep MVC, Class post kita pindahkan ke folder model.

The screenshot shows a code editor interface with the following details:

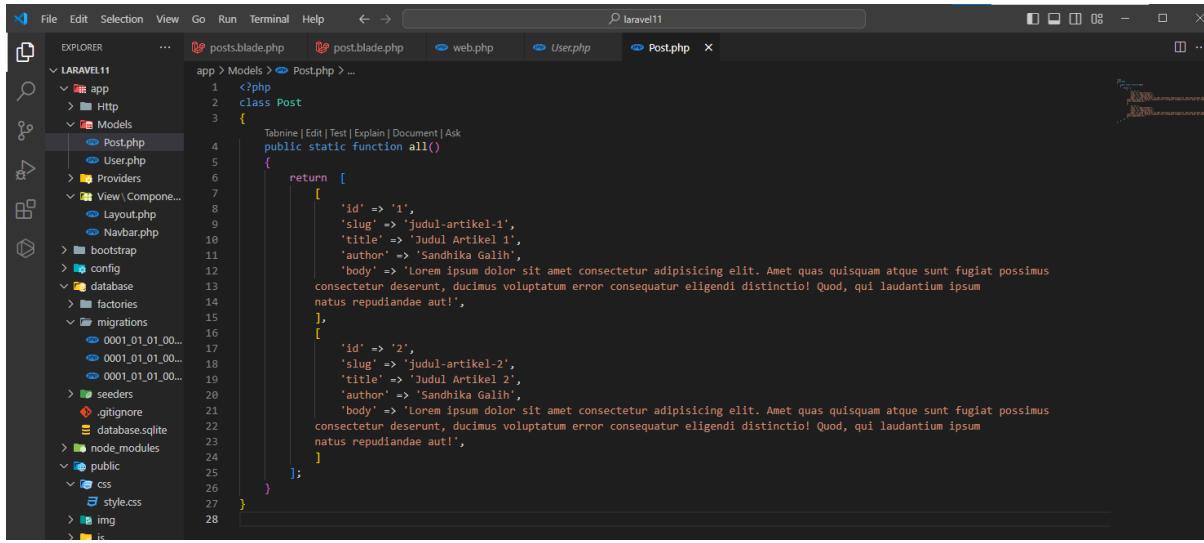
- File Explorer (Left):** Shows the project structure under "LARAVEL11". Key folders include "app", "Models" (containing "User.php"), "View", "Components", "bootstrap", "config", "database" (with "factories" and "migrations" subfolders), "node_modules", "public" (with "css", "img", "js", ".htaccess", "favicon.ico", "hot", "index.php", and "robots.txt"), "resources" (with "css" and "js" subfolders), and "views" (with "components" containing various blade files like "header.blade.php", "layout.blade.php", etc.).
- Code Editor (Right):** Displays the "User.php" file content. The code defines a User model extending the Illuminate\Auth\Access\Authorizable trait. It includes imports for App\Models, Illuminate\Contracts\Auth\MustVerifyEmail, Illuminate\Database\Eloquent\Factories\HasFactory, Illuminate\Foundation\Auth\User, and Illuminate\Notifications\Notifiable. The class has protected properties for \$fillable (name, email, password) and \$hidden (password, remember_token). It also defines a casts() method to handle casting of attributes.

```

1 <?php
2
3 namespace App\Models;
4
5 // use Illuminate\Contracts\Auth\MustVerifyEmail;
6 use Illuminate\Database\Eloquent\Factories\HasFactory;
7 use Illuminate\Foundation\Auth\User as Authenticatable;
8 use Illuminate\Notifications\Notifiable;
9
10 class User extends Authenticatable
11 {
12     /**
13      * The attributes that are mass assignable.
14      *
15      * @var array<int, string>
16      */
17     protected $fillable = [
18         'name',
19         'email',
20         'password',
21     ];
22
23     /**
24      * The attributes that should be hidden for serialization.
25      *
26      * @var array<int, string>
27      */
28     protected $hidden = [
29         'password',
30         'remember_token',
31     ];
32
33     /**
34      * Get the attributes that should be cast.
35      *
36      * @return array<string, string>
37      */
38     protected function casts(): array
39     {
40         return [
41             'email_verified_at' => 'datetime',
42             'password' => 'hashed',
43         ];
44     }
45
46 }
47
48 }
49

```

7. Pada class ini akan me-representasikan tabel yang ada di database. Class ini singular dari tabel yang biasa punya pasangan di migration. Ada migration default yang namanya users. Migration ini biasanya untuk membuat tabel users dan pasangan modelnya adalah user.



The screenshot shows a code editor interface with a dark theme. On the left is the 'EXPLORER' sidebar, which lists the project structure of 'LARAVEL11'. The 'Models' folder contains 'Post.php' and 'User.php'. The 'routes' folder contains 'posts.blade.php', 'post.blade.php', 'web.php', and 'Post.php'. The 'Post.php' file is currently open in the main editor area. It contains PHP code defining a 'Post' class with a static method 'all()' that returns an array of two objects, each representing a post with properties like id, slug, title, author, and body.

```
<?php  
class Post  
{  
    public static function all()  
    {  
        return [  
            [  
                'id' => '1',  
                'slug' => 'judul-artikel-1',  
                'title' => 'Judul Artikel 1',  
                'author' => 'Sandhika Galih',  
                'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus  
consequetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum  
natus repudiandae aut!',  
            ],  
            [  
                'id' => '2',  
                'slug' => 'judul-artikel-2',  
                'title' => 'Judul Artikel 2',  
                'author' => 'Sandhika Galih',  
                'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus  
consequetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum  
natus repudiandae aut!',  
            ]  
        ];  
    }  
}
```

8. Dengan contoh tadi jika kita membuat class post yang isinya diambil dari class post yang tadi di route. Namun route kita tidak kenal model post yang sudah kita buat. Jadi model post nya tidak otomatis bisa langsung kita gunakan.

Internal Server Error

Error

Class "Post" not found

GET laravel11.test

PHP 8.2.25 — Laravel 11.30.0

```
C:\laragon\www\laravel11\routes\web.php :17

 Illuminate\View\Middleware\ShareErrorsFromSession handle

 Illuminate\Pipeline\Pipeline:183
 Illuminate\Pipeline\{closure}

 Illuminate\Session\Middleware\StartSession:121
 handleStatefulRequest

 Illuminate\Session\Middleware\StartSession:64
 handle

 Illuminate\Pipeline\Pipeline:183
 Illuminate\Pipeline\{closure}

 Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse handle

 Illuminate\Pipeline\Pipeline:183
 Illuminate\Pipeline\{closure}
```

```
12     return view('about', ['nama' => 'Lia', 'title' => 'Contact']);
13 });
14 Route::get('/posts', function () {
15     return view('posts', [
16         'title' => 'Blog',
17         'posts' => Post::all()
18     ]);
19 });
20
21 Route::get('/posts/{slug}', function ($slug) {
22     // Temukan artikel berdasarkan slug
23     $post = App::first(Post::all(), function ($post) use ($slug) {
24         return $post['slug'] === $slug;
25     });
26
27     // Jika post ditemukan, tampilkan; jika tidak, tampilkan halaman error
28     if (!$post) {
```

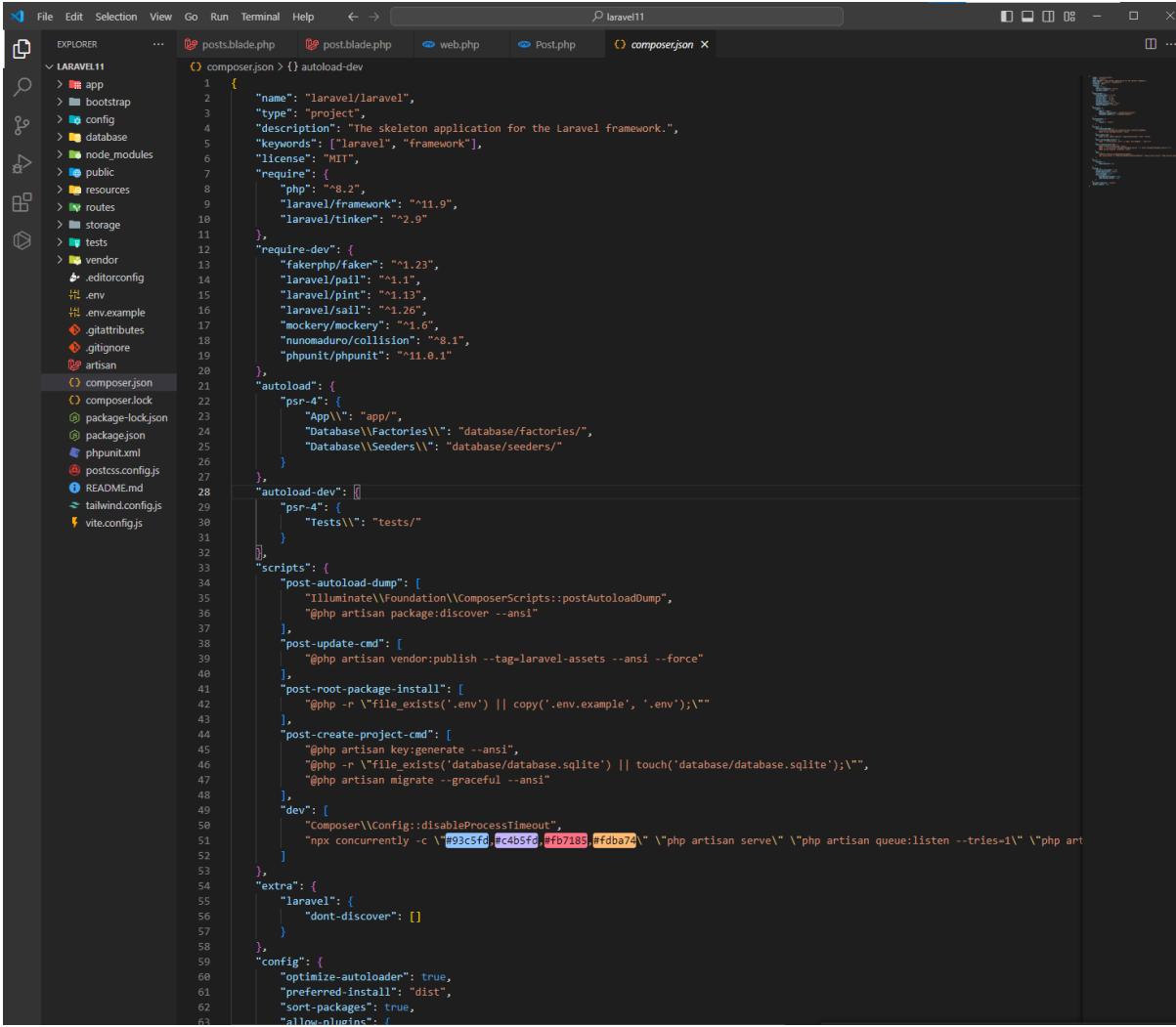
Request

GET /posts

Headers

| | |
|-------------------------|-----|
| priority | u=4 |
| upgrade-insecure-req... | 1 |

9. Jika kita balik ke halaman web, maka akan error.



The screenshot shows a code editor interface with the title bar "laravel11". The left sidebar displays a file tree for a Laravel 11 project, including files like .env, .env.example, .gitignore, artisan, composer.json, composer.lock, package-lock.json, package.json, phpunit.xml, postcss.config.js, README.md, tailwind.config.js, and vite.config.js. The main editor area is focused on the "composer.json" file, which contains the following JSON code:

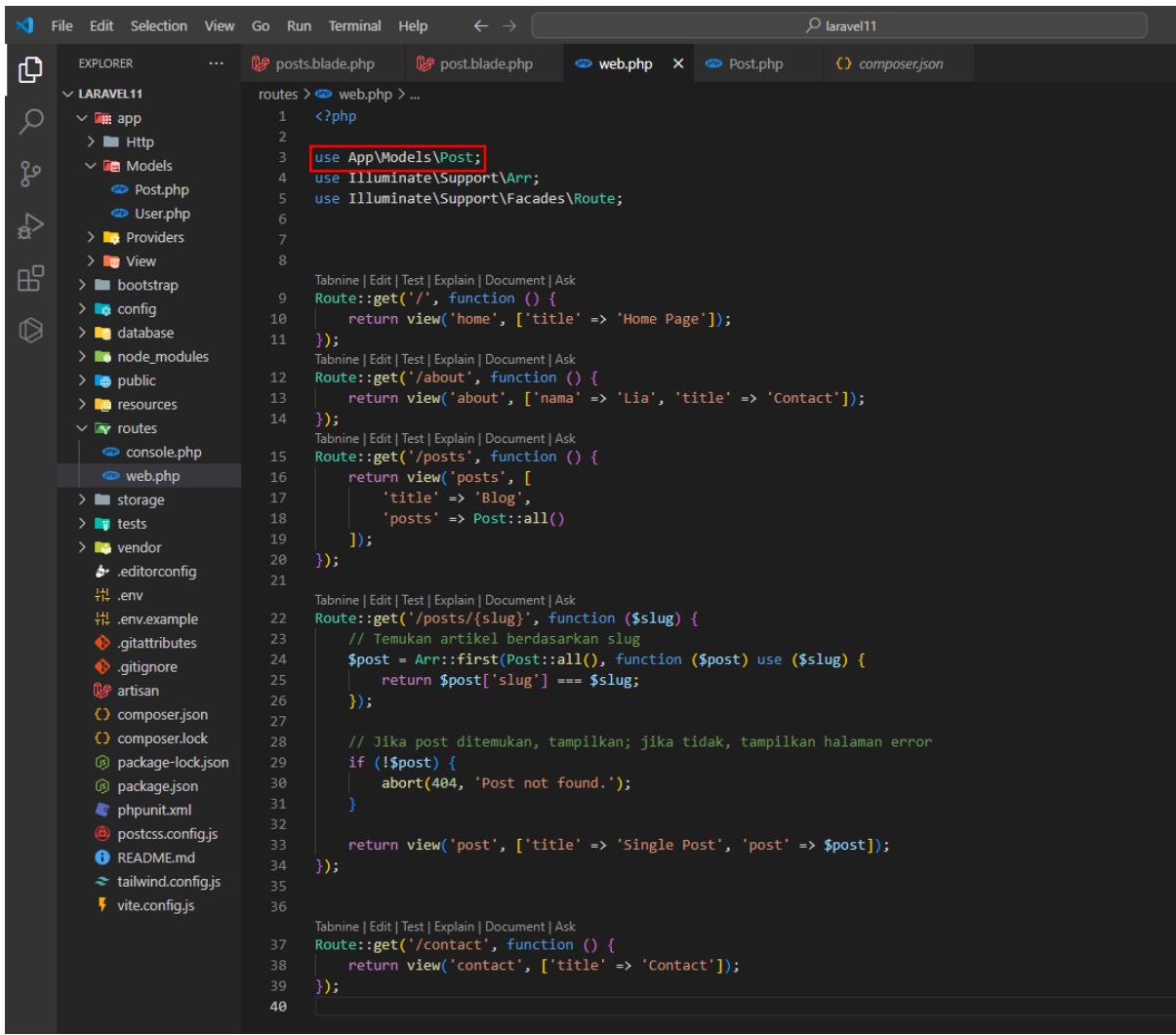
```
1 {  
2     "name": "laravel/laravel",  
3     "type": "project",  
4     "description": "The skeleton application for the Laravel framework.",  
5     "keywords": ["laravel", "framework"],  
6     "license": "MIT",  
7     "require": {  
8         "php": "^8.2",  
9         "laravel/framework": "^11.9",  
10        "laravel/tinker": "^2.9"  
11    },  
12    "require-dev": {  
13        "fakerphp/faker": "^1.23",  
14        "laravel/pall": "^1.1",  
15        "laravel/pint": "^1.13",  
16        "laravel/sail": "^1.26",  
17        "mockery/mockery": "^1.6",  
18        "nunomaduro/collision": "8.1",  
19        "phpunit/phpunit": "11.0.1"  
20    },  
21    "autoload": {  
22        "psr-4": {  
23            "App\\": "app/",  
24            "Database\\Factories\\": "database/factories/",  
25            "Database\\Seeders\\": "database/seeders/"  
26        }  
27    },  
28    "autoload-dev": [  
29        "psr-4": {  
30            "Tests\\": "tests/"  
31        }  
32    ],  
33    "scripts": {  
34        "post/autoload-dump": [  
35            "Illuminate\Foundation\ComposerScripts::postAutoloadDump",  
36            "@php artisan package:discover --ansi"  
37        ],  
38        "post-update-cmd": [  
39            "@php artisan vendor:publish --tag=laravel-assets --ansi --force"  
40        ],  
41        "post-root-package-install": [  
42            "@php -r \"file_exists('.env') || copy('.env.example', '.env');\""  
43        ],  
44        "post-create-project-cmd": [  
45            "@php artisan key:generate --ansi",  
46            "@php -r \"file_exists('database/database.sqlite') || touch('database/database.sqlite');\"",  
47            "@php artisan migrate --graceful --ansi"  
48        ],  
49        "dev": [  
50            "Composer\Config::disableProcessTimeout",  
51            "npx concurrently -c \"#93c5fd:#c4b5fd:#f67188:#fdbd74\" \"php artisan serve\" \"php artisan queue:listen --tries=1\" \"php art  
52        ]  
53    },  
54    "extra": {  
55        "laravel": {  
56            "dont-discover": []  
57        }  
58    },  
59    "config": {  
60        "optimize-autoloader": true,  
61        "preferred-install": "dist",  
62        "sort-packages": true,  
63        "allow-plugins": true  
64    }  
65}
```

10. Kita akan memanfaatkan proses autoloading yang ada di laravel. Class post kita tidak terbaca karena belum tersimpan di dalam sebuah namespace. Namespace adalah sebuah teknik untuk menyimpan class ke dalam folder yang spesifik agar tidak bentrok dengan class lain yang namanya sama.

The screenshot shows the VS Code interface with the Laravel 11 project open. The Explorer sidebar on the left lists files and folders such as .env, .env.example, .gitignore, artisan, composer.json, composer.lock, package-lock.json, package.json, phpunit.xml, postcss.config.js, README.md, tailwind.config.js, and vite.config.js. The main editor area displays the Post.php file under the app\Models directory. The code in Post.php is as follows:

```
<?php  
namespace App\\Models;  
class Post  
{  
    public static function all()  
    {  
        return [  
            [  
                'id' => '1',  
                'slug' => 'judul-artikel-1',  
                'title' => 'Judul Artikel 1',  
                'author' => 'Sandhika Galih',  
                'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus  
consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum  
natus repudiandae aut!',  
            ],  
            [  
                'id' => '2',  
                'slug' => 'judul-artikel-2',  
                'title' => 'Judul Artikel 2',  
                'author' => 'Sandhika Galih',  
                'body' => 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus  
consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum  
natus repudiandae aut!',  
            ]  
        ];  
    }  
}
```

11. Namespace ini untuk memberitahu bahwa class post berada di dalam folder App\folder Model.



The screenshot shows the VS Code interface with the file `routes/web.php` open. The code defines several routes using the `Route::get` method. Route 1 handles the root path ('/'). Route 2 handles the '/about' path. Route 3 handles the '/posts' path, which returns a view named 'posts' with a title of 'Blog' and a list of posts. Route 4 handles the '/posts/{slug}' path, which finds a post by slug and returns its details. Route 5 handles the '/contact' path. The code uses Laravel's `use Illuminate\Support\Arr;` and `use Illuminate\Support\Facades\Route;` imports.

```
<?php

use App\Models\Post;
use Illuminate\Support\Arr;
use Illuminate\Support\Facades\Route;

Route::get('/', function () {
    return view('home', ['title' => 'Home Page']);
});

Route::get('/about', function () {
    return view('about', ['nama' => 'Lia', 'title' => 'Contact']);
});

Route::get('/posts', function () {
    return view('posts', [
        'title' => 'Blog',
        'posts' => Post::all()
    ]);
});

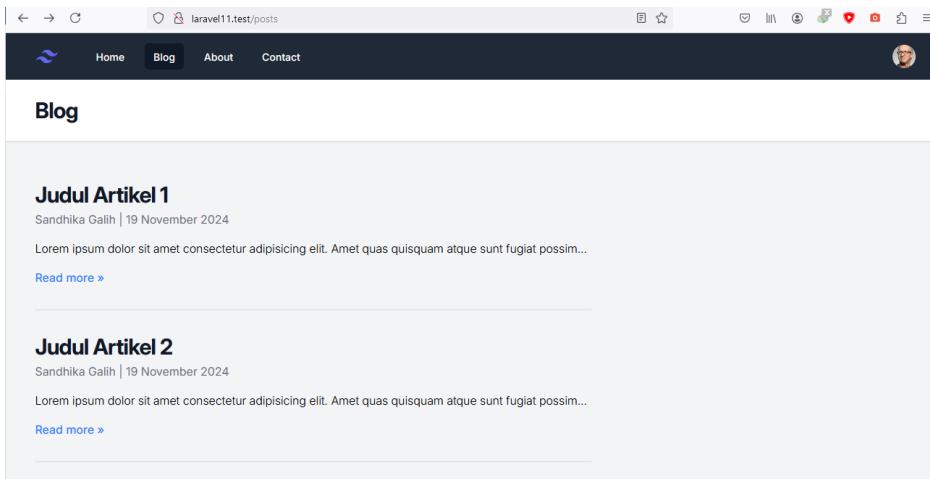
Route::get('/posts/{slug}', function ($slug) {
    // Temukan artikel berdasarkan slug
    $post = Arr::first(Post::all(), function ($post) use ($slug) {
        return $post['slug'] === $slug;
    });

    // Jika post ditemukan, tampilkan; jika tidak, tampilkan halaman error
    if (!$post) {
        abort(404, 'Post not found.');
    }

    return view('post', ['title' => 'Single Post', 'post' => $post]);
});

Route::get('/contact', function () {
    return view('contact', ['title' => 'Contact']);
});
```

12. Tinggal kita panggil di file web.



The screenshot shows a browser window displaying a Laravel application. The URL is `laravel11.test/posts`. The page has a dark header with a logo, a navigation bar with links for Home, Blog, About, and Contact, and a user profile icon. The main content area shows two articles. Article 1 is titled "Judul Artikel 1" and was posted by Sandhika Galih on 19 November 2024. Article 2 is titled "Judul Artikel 2" and was posted by Sandhika Galih on 19 November 2024. Both articles have a short preview and a "Read more »" link.

Blog

Judul Artikel 1
Sandhika Galih | 19 November 2024
Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...
[Read more »](#)

Judul Artikel 2
Sandhika Galih | 19 November 2024
Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...
[Read more »](#)

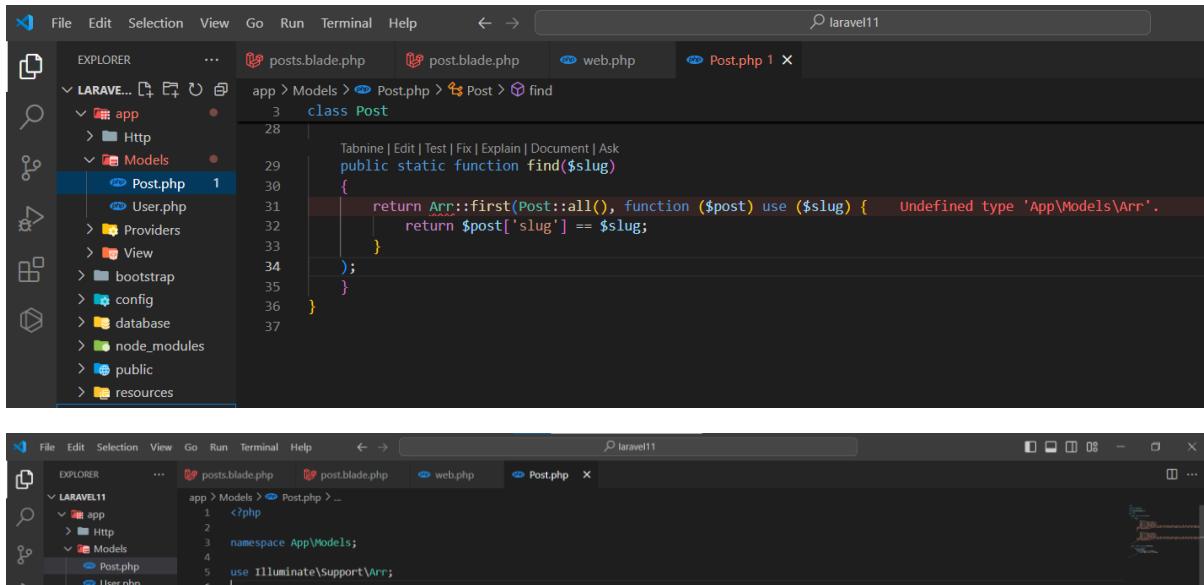
The screenshot shows a web browser window with the URL `laravel11.test/posts/judul-artikel-1`. The page has a dark header with navigation links for Home, Blog, About, Contact, and a user profile icon. The main content area is titled "Single Post" and displays the title "Judul Artikel 1". Below the title is the author's name, Sandhika Galih, and a short paragraph of placeholder text (Lorem ipsum). At the bottom left, there is a link to "« Back To Posts".

13. Hasilnya jika kita kembali ke halaman web.

The screenshot shows a code editor with the file `routes/web.php` open. The code defines several routes in the `Route::get` method. The route `/posts/{slug}` is highlighted with a red box. The code within this block is as follows:

```
Route::get('/posts/{slug}', function ($slug) {
    // Temukan artikel berdasarkan slug
    $post = App\Models\Post::first();
    if (!$post) {
        abort(404, 'Post not found.');
    }
    return view('post', ['title' => 'Single Post', 'post' => $post]);
});
```

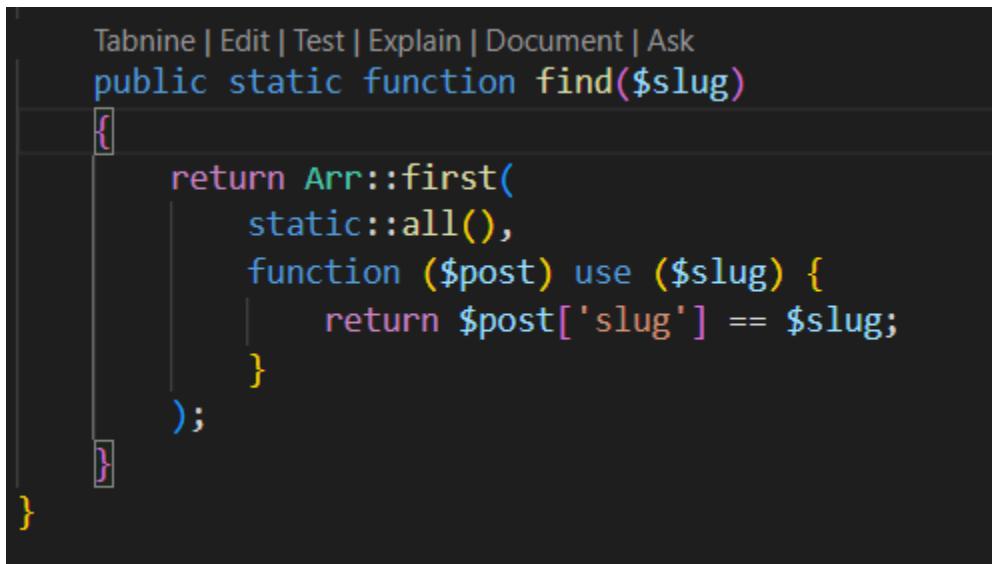
14. Sekarang kita perbaiki model agar dapat melakukan pencarian secara spesifik atau pencarian single post menggunakan static method telah kita buat. Untuk sekarang proses pencarian dilakukan di route. Pada kotak merah seharusnya itu tugas dari model kasrena melakukan pencarian atau merubah data masuk ke kategori bisnis logic yang seharusnya dibebankan pada model.



```
public static function find($slug)
{
    return Arr::first(Post::all(), function ($post) use ($slug) {
        return $post['slug'] == $slug;
    });
}
```

```
namespace App\Models;
use Illuminate\Support\Arr;
```

15. Kita buat funct baru atau funct static.



```
public static function find($slug)
{
    return Arr::first(
        static::all(),
        function ($post) use ($slug) {
            return $post['slug'] == $slug;
        }
    );
}
```

16. Kita ganti post nya menjadi static, karena ada di class yang sama. Kalau dalam sebuah method memanggil method yang lain di dalam class yang sama itu pake **this**.

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar showing the project structure of 'LARAVEL11' with 'app' selected, containing files like View, bootstrap, config, database, node_modules, public, resources, routes, console.php, and web.php. The main editor area displays the contents of 'routes/web.php'. The code defines several routes:

```
Route::get('/', function () {
    return view('about', ['nama' => 'Lia', 'title' => 'Contact']);
});
Route::get('/posts', function () {
    return view('posts', [
        'title' => 'Blog',
        'posts' => Post::all()
    ]);
});
Route::get('/posts/{slug}', function ($slug) {
    // Temukan artikel berdasarkan slug
    $post = Post::find($slug);
    return view('post', ['title' => 'Single Post', 'post' => $post]);
});
Route::get('/contact', function () {
    return view('contact', ['title' => 'Contact']);
});
```

17. Jadi kita tinggal memanggil class Post method find.

The screenshot shows a browser window with the URL 'laravel11.test/posts'. The page title is 'Blog'. It displays two articles:

Judul Artikel 1
Sandhika Galih | 19 November 2024
Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...
[Read more »](#)

Judul Artikel 2
Sandhika Galih | 19 November 2024
Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...
[Read more »](#)

At the bottom left, there is a small footer text: 'laravel11.test/posts/judul-artikel-1'.



Single Post

Judul Artikel 2

Sandhika Galih

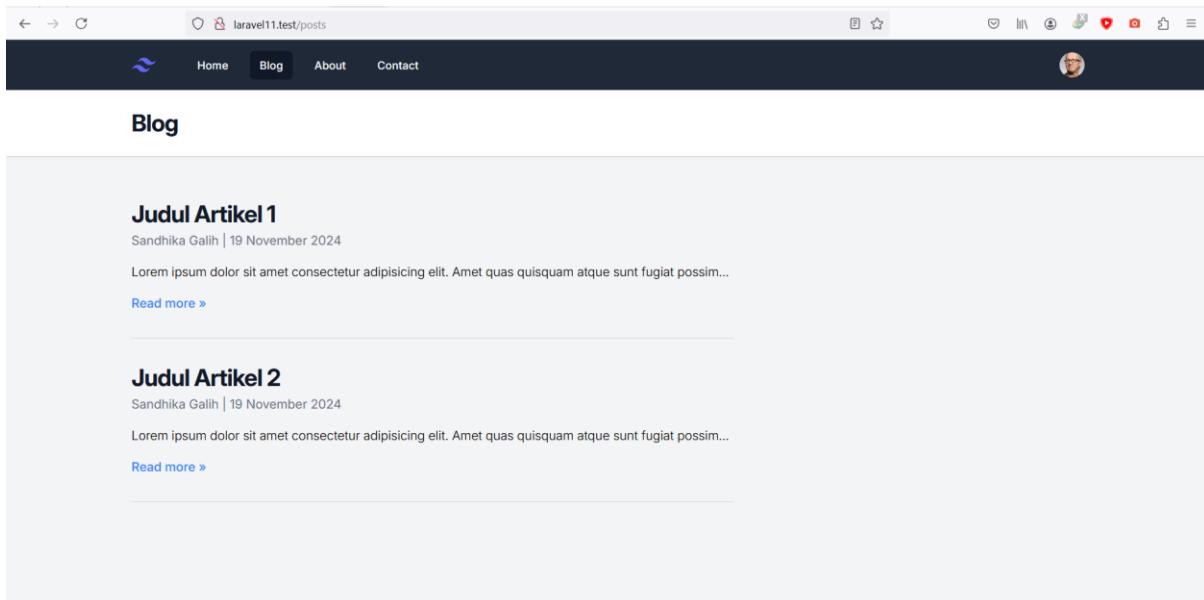
Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum natus repudiandae aut!

[« Back To Posts](#)

18. Hasil run nya.

A screenshot of a code editor with the file 'Post.php' open. The code defines a static method 'find(\$slug)' which uses Laravel's collection methods like 'first()' and 'all()' to return a post by its slug. It also shows an arrow function 'fn (\$post) => \$post['slug'] == \$slug'.

19. Kita menggunakan teknik **arrow funct** jika tidak ingin menggunakan **use**.



A screenshot of a web browser displaying a Laravel application. The URL is `laravel11.test/posts/judul-artikel-1`. The page has a dark header with a logo, navigation links for Home, Blog, About, Contact, and a user profile icon. Below the header is a section titled "Single Post". Inside this section, there is a heading "Judul Artikel 1" and a paragraph of placeholder text: "Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possimus consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum natus repudiandae aut!". At the bottom of this section is a link "[« Back To Posts](#)".

20. Hasil saat di run masih sama. Kenapa kita buat manual pakai class terus namanya find dan all ini supaya merepresentasikan ke depannya kita akan menggunakan fungsi laravel dengan nama method yang sama.

A screenshot of a web browser displaying a Laravel application. The URL is `laravel11.test/posts/asdadadas`. The page has a dark header with a logo, navigation links for Home, Blog, About, Contact, and a user profile icon. Below the header is a section titled "Single Post". Inside this section, there is a message "Post not found.".

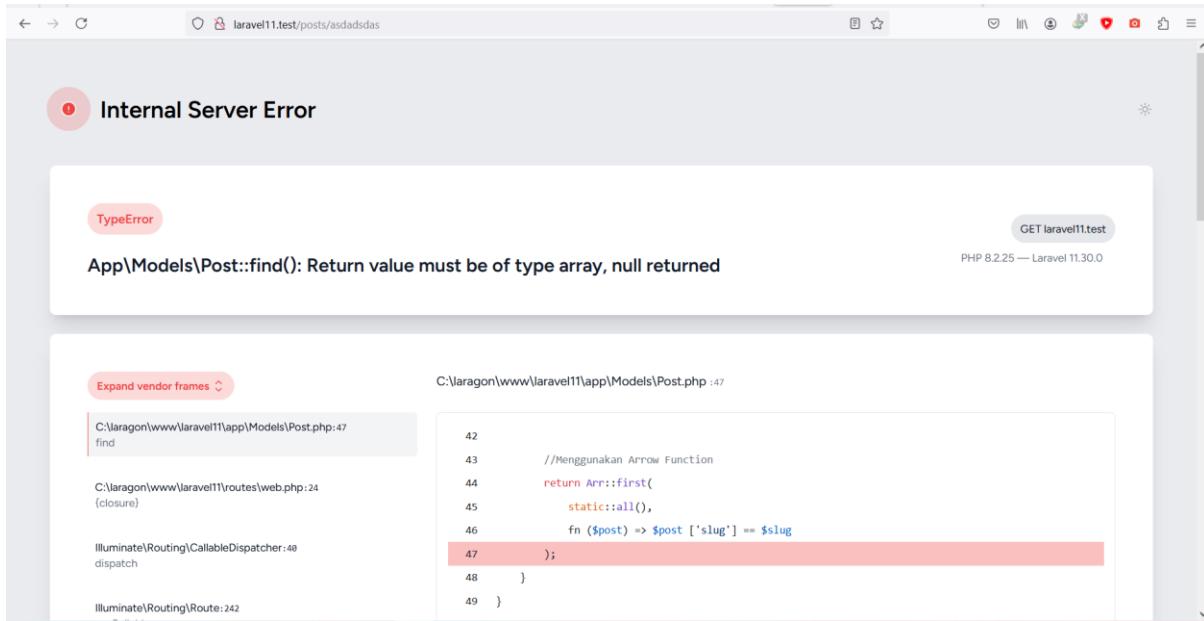
21. Gimana kalau misalnya postnya nggak ada, di url kita ketik asal maka akan terjadi error.

A screenshot of a code editor showing the `Post.php` file from the `Laravel11` project. The code defines a `Post` model with two static methods: `all()` and `find($slug) : array`. The `all()` method uses a callback function to filter posts by slug. The `find($slug) : array` method uses an arrow function to filter posts by slug. The code editor interface includes tabs for other files like `posts.blade.php`, `post.blade.php`, and `web.php`.

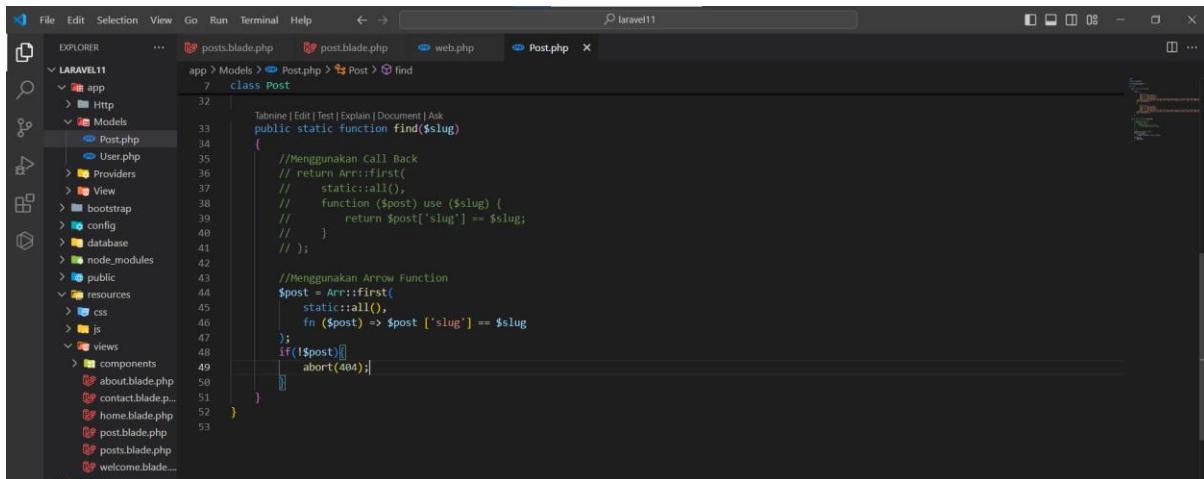
```
public static function all()
{
    //Menggunakan Call Back
    // return Arr::first(
    //     static::all(),
    //     function ($post) use ($slug) {
    //         return $post['slug'] == $slug;
    //     }
    // );
}

//Menggunakan Arrow Function
return Arr::first(
    static::all(),
    fn ($post) => $post ['slug'] == $slug
);
```

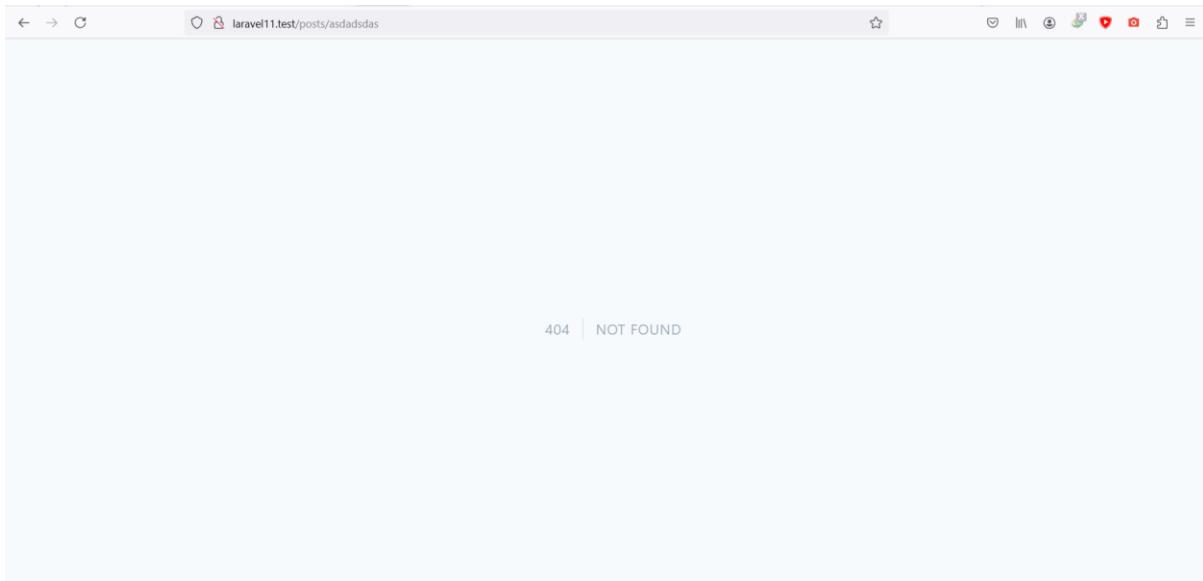
22. Kita kasih alamat 404 aja ketika posnya nggak ketemu. Kalau misalkan mau sedikit memperbaiki supaya lebih yakin dengan apa yang dikembalikan oleh method, Kita bisa ngasih return type berupa array.



23. Langsung ketahuan salahnya di mana gitu di sini



24. Pertama kita akan Simpan dulu ke dalam sebuah variabel, jika posnya kosong bisa tulis kasih halaman 404.

A screenshot of a blog application. The header includes a logo, navigation links for Home, Blog, About, and Contact, and a user profile picture. The main content area is titled 'Blog' and lists two posts:

- Judul Artikel 1**
Sandhika Galih | 19 November 2024
Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...
[Read more »](#)
- Judul Artikel 2**
Sandhika Galih | 19 November 2024
Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet quas quisquam atque sunt fugiat possim...
[Read more »](#)

25. Jika url kosong maka halaman tetap berjalan.

A screenshot of a blog application. The header includes a logo, navigation links for Home, Blog, About, and Contact, and a user profile picture. The main content area is titled 'Single Post' and displays the message 'Post not found.'

26. Namun jika read more di klik halaman akan error karena belum dilakukan return.

27. Lakukan return \$post

Single Post

Judul Artikel 1

Sandhika Galih

Lorem ipsum dolor sit amet consectetur adipisciing elit. Amet quas quisquam atque sunt fugiat possimus consectetur deserunt, ducimus voluptatum error consequatur eligendi distinctio! Quod, qui laudantium ipsum natus repudiandae aut!

[« Back To Posts](#)

Code Editor Screenshot:

```

File Edit Selection View Go Run Terminal Help ← → laravel11
EXPLORER LARAVEL11
app
  Http
  Models
    Post.php
    User.php
  Providers
  View
  bootstrap
  config
  database
  node_modules
  public
  resources
    css
    js
  views
    components
      about.blade.php
      contact.blade.php
      home.blade.php
      post.blade.php
      posts.blade.php
      welcome.blade.php
posts.blade.php post.blade.php web.php Post.php
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55

```

The code editor shows the `Post.php` file with the following content:

```

public static function find($slug)
{
    //Menggunakan Call Back
    // return Arr::first(
    //     static::all(),
    //     function ($post) use ($slug) {
    //         return $post['slug'] == $slug;
    //     }
    // );

    //Menggunakan Arrow Function
    $post = Arr::first(
        static::all(),
        fn ($post) => $post ['slug'] == $slug
    );
    if (!$post){
        abort(404);
    }
    return $post;
}

```

28. Hasil setelah dilakukan return \$post