

# LAPORAN PRAKTIK MEMBUAT TAMPILAN INTERFACE WEB DASHBOARD IOT

*Yasiva Nurul Ramadhan – 233140707111067 – T4A – Teknologi Informasi*

*Fakultas Vokasi, Universitas Brawijaya*

*Email: [ysvlhn@student.ub.ac.id](mailto:ysvlhn@student.ub.ac.id)*

Praktikum ini membahas pembuatan tampilan *interface web dashboard* menggunakan Laravel. *Dashboard* dirancang untuk menampilkan grafik berdasarkan data yang diakses melalui API. Praktikum dilakukan dengan memanfaatkan fitur *HTTP client* pada Laravel dan ditampilkan ke halaman *web* melalui *view*. Hasilnya adalah tampilan grafik yang dapat dilihat melalui browser secara langsung.

Kata kunci— Laravel, PHP, Web, Dashboard, API

## 1. Pendahuluan

### 1.1 Latar belakang

Pemanfaatan API dalam sistem IoT memungkinkan data ditampilkan secara fleksibel melalui antarmuka web. Laravel menyediakan kemudahan dalam melakukan permintaan HTTP dan menampilkan hasilnya ke dalam tampilan yang interaktif.

Praktikum ini bertujuan membuat dashboard web sederhana yang mengambil data dari API dan menampilkannya dalam bentuk grafik. Semua proses dilakukan menggunakan Laravel dan PHP, dengan memisahkan logika pengambilan data dan tampilan.

### 1.2 Tujuan Eksperimen

Praktikum ini bertujuan untuk:

1. Membangun tampilan dashboard menggunakan Laravel.
2. Mengakses data melalui API menggunakan HTTP client Laravel.
3. Menampilkan data API dalam bentuk grafik di halaman web.

## 2. Metodologi

### 2.1 Alat dan Bahan

Dalam praktikum ini, alat dan bahan yang digunakan meliputi:

1. Komputer/Laptop
2. Akses internet untuk koneksi Wi-Fi
3. Visual Studio Code
4. Laravel

5. Browser (Chrome atau Firefox)

## 2.2 Langkah Implementasi

Berikut langkah-langkah yang dilakukan dalam praktikum ini:

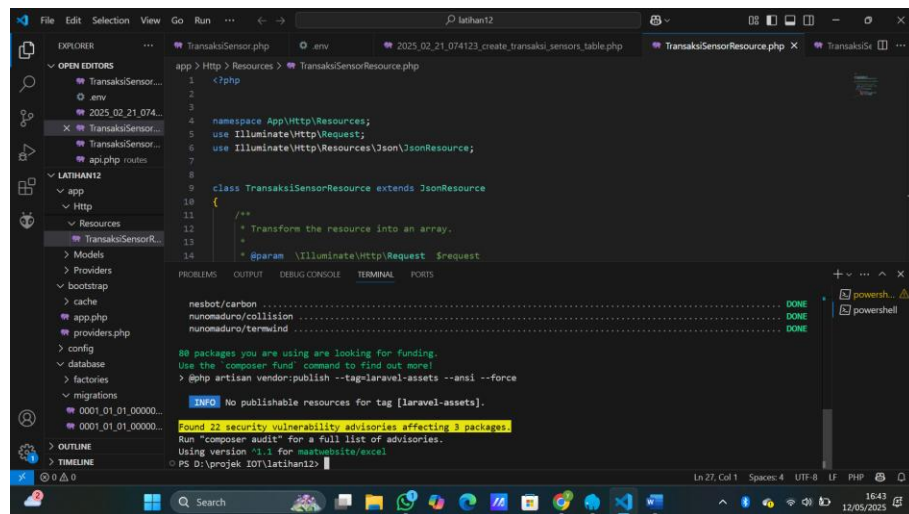
1. Membuka atau membuat project Laravel.
2. Menjalankan perintah php artisan make:controller GraphController.
3. Menambahkan kode untuk mengambil data dari API pada controller tersebut.
4. Membuat file view graph.blade.php untuk menampilkan data dalam bentuk grafik.
5. Menambahkan routing pada web.php agar dapat mengakses tampilan grafik.
6. Menjalankan Laravel menggunakan perintah php artisan serve.

## 3. Hasil Dan Pembahasan

### 3.1 Hasil Eksperimen

Hasil dari praktikum ini adalah sebagai berikut:

1. Buka folder laravel yang sudah dibuat pada Praktik 12 pada VSCode. Buka terminal dan jalankan code berikut:



The screenshot shows the Visual Studio Code interface with a Laravel project. The Explorer panel on the left shows the project structure, including the 'Resources' folder. The main editor displays the 'TransaksiSensorResource.php' file, which contains a class extending 'JsonResource'. The terminal at the bottom shows the output of the 'php artisan vendor:publish --tag=laravel-assets' command, indicating that no publishable resources were found for the 'laravel-assets' tag. The status bar at the bottom shows the file is in 'Ln 27, Col 1' and the encoding is 'UTF-8'.

```
1 <?php
2
3
4 namespace App\Http\Resources;
5 use Illuminate\Http\Request;
6 use Illuminate\Http\Resources\Json\JsonResource;
7
8 class TransaksiSensorResource extends JsonResource
9 {
10     /**
11      * Transform the resource into an array.
12      *
13      * @param \Illuminate\Http\Request $request
14      */
15 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

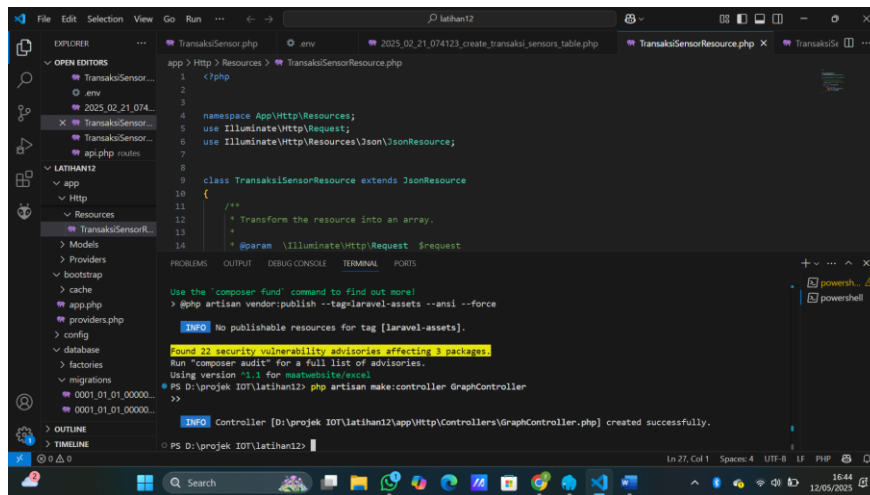
nesbot/carbon ..... DONE  
nunomaduro/collision ..... DONE  
nunomaduro/termwind ..... DONE

80 packages you are using are looking for funding.  
Use the composer fund command to find out more!  
> @php artisan vendor:publish --tag=laravel-assets --ansi --force

[INFO] No publishable resources for tag [laravel-assets].

Found 22 security vulnerability advisories affecting 3 packages.  
Run "composer audit" for a full list of advisories.  
Using version "1.1" for mastamaste/excel

Ln 27, Col 1 Space: 4 UTF-8 LF PHP



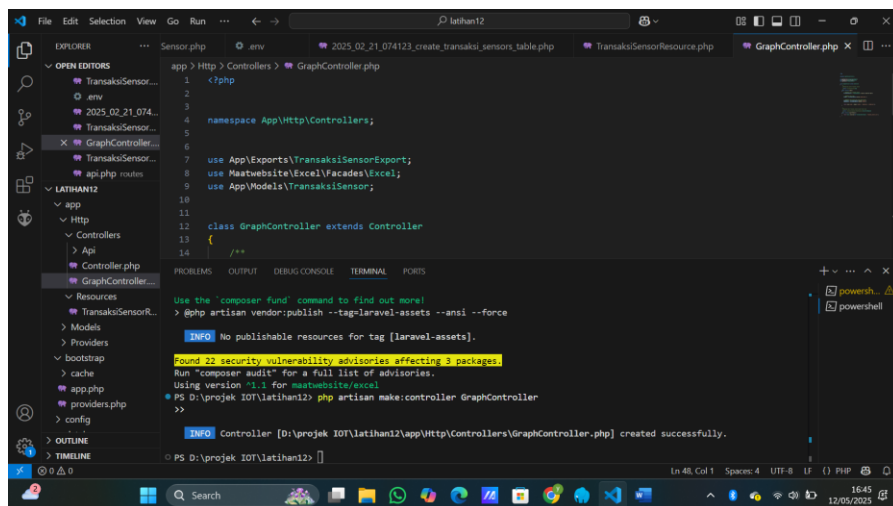
The screenshot shows the Visual Studio Code editor with a file explorer on the left. The file explorer shows the project structure for 'latihan12', including 'app', 'resources', 'models', 'providers', 'bootstrap', 'cache', 'app.php', 'providers.php', 'config', 'database', 'factories', 'migrations', and '0001\_01\_01\_000000...'. The 'app' folder is expanded, showing 'Http' and 'Resources'. The 'Resources' folder is also expanded, showing 'TransaksiSensorResource.php'. The main editor window shows the code for 'TransaksiSensorResource.php'. The code is as follows:

```
1 <?php
2
3
4 namespace App\Http\Resources;
5 use Illuminate\Http\Request;
6 use Illuminate\Http\Resources\Json\JsonResource;
7
8 class TransaksiSensorResource extends JsonResource
9 {
10     /**
11      * Transform the resource into an array.
12      *
13      * @param \Illuminate\Http\Request $request
14      */
15 }
```

The terminal window at the bottom shows the output of the command 'php artisan make:controller GraphController'. The output is:

```
PS D:\projek IoT\latihan12> php artisan make:controller GraphController
>>
INFO Controller [D:\projek IoT\latihan12\app\Http\Controllers\GraphController.php] created successfully.
```

2. Menjalankan perintah php artisan make:controller GraphController.



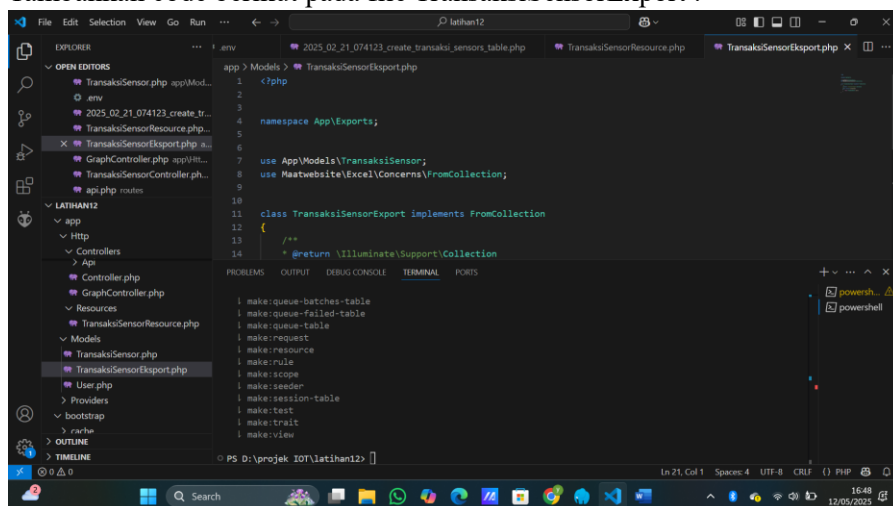
The screenshot shows the Visual Studio Code editor with a file explorer on the left. The file explorer shows the project structure for 'latihan12', including 'app', 'resources', 'models', 'providers', 'bootstrap', 'cache', 'app.php', 'providers.php', 'config', 'database', 'factories', 'migrations', and '0001\_01\_01\_000000...'. The 'app' folder is expanded, showing 'Http' and 'Controllers'. The 'Controllers' folder is also expanded, showing 'GraphController.php'. The main editor window shows the code for 'GraphController.php'. The code is as follows:

```
1 <?php
2
3
4 namespace App\Http\Controllers;
5
6 use App\Exports\TransaksiSensorExport;
7 use Maatwebsite\Excel\Facades\Excel;
8 use App\Models\TransaksiSensor;
9
10 class GraphController extends Controller
11 {
12     /**
13      *
14      */
15 }
```

The terminal window at the bottom shows the output of the command 'php artisan make:controller @GraphController'. The output is:

```
PS D:\projek IoT\latihan12> php artisan make:controller @GraphController
>>
INFO Controller [D:\projek IoT\latihan12\app\Http\Controllers\GraphController.php] created successfully.
```

3. Tambahkan code berikut pada file TransaksiSensorExport :



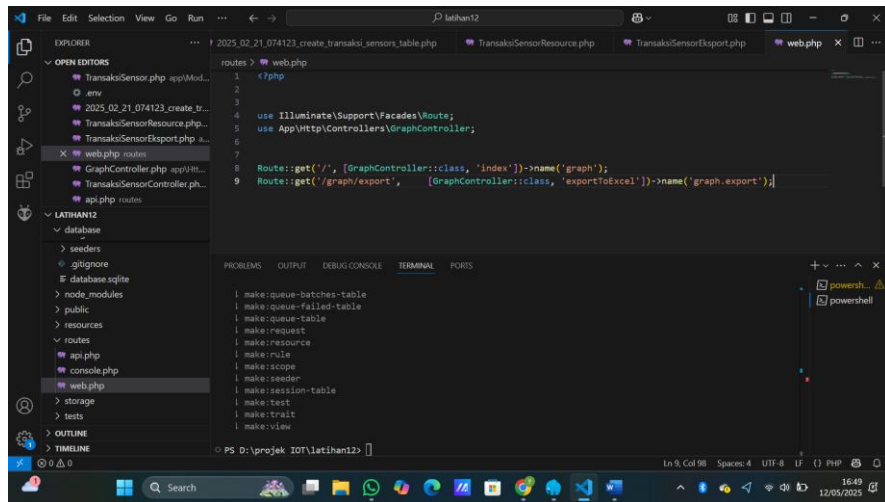
The screenshot shows the Visual Studio Code editor with a file explorer on the left. The file explorer shows the project structure for 'latihan12', including 'app', 'resources', 'models', 'providers', 'bootstrap', 'cache', 'app.php', 'providers.php', 'config', 'database', 'factories', 'migrations', and '0001\_01\_01\_000000...'. The 'app' folder is expanded, showing 'Http' and 'Exports'. The 'Exports' folder is also expanded, showing 'TransaksiSensorExport.php'. The main editor window shows the code for 'TransaksiSensorExport.php'. The code is as follows:

```
1 <?php
2
3
4 namespace App\Exports;
5
6 use App\Models\TransaksiSensor;
7 use Maatwebsite\Excel\Concerns\FromCollection;
8
9 class TransaksiSensorExport implements FromCollection
10 {
11     /**
12      *
13      */
14     public function collection()
15     {
16         return Illuminate\Support\Collection::make([
17             //
18         ]);
19     }
20 }
```

The terminal window at the bottom shows the output of the command 'php artisan make:controller @GraphController'. The output is:

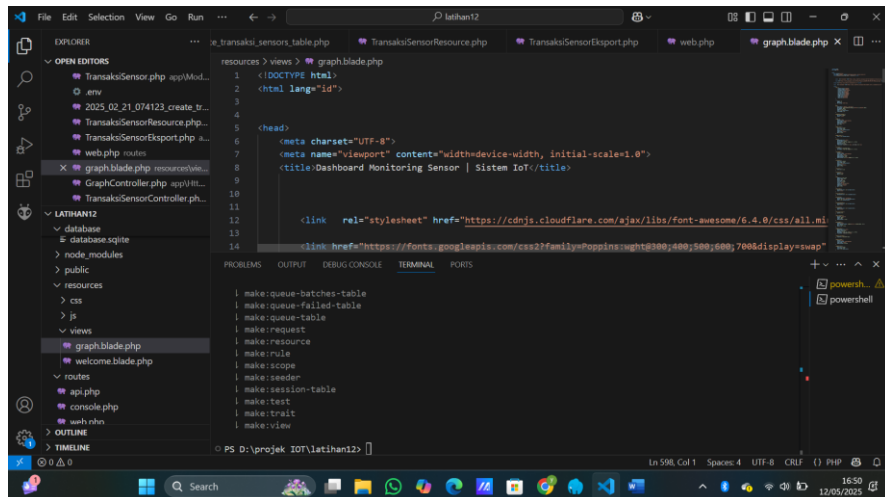
```
PS D:\projek IoT\latihan12> php artisan make:controller @GraphController
>>
INFO Controller [D:\projek IoT\latihan12\app\Http\Controllers\GraphController.php] created successfully.
```

4. Setelah itu, edit file web.php yang berada di folder routes menjadi seperti berikut:

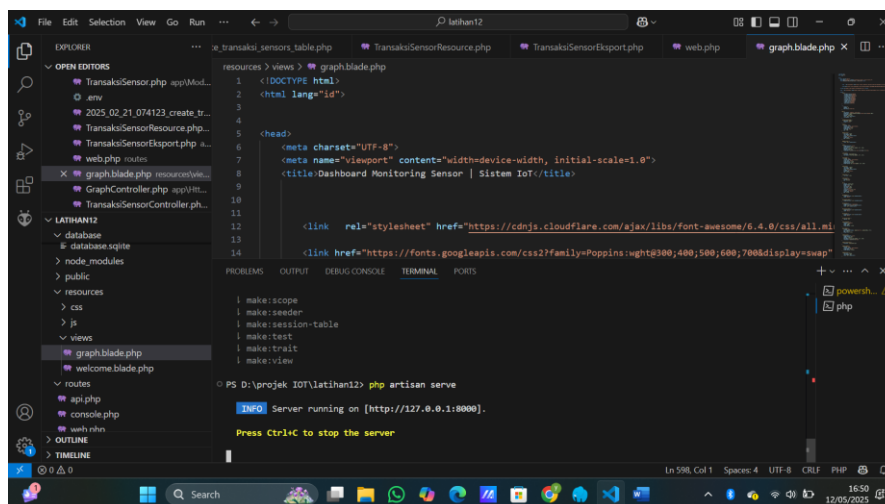


```
1 <?php
2
3
4 use Illuminate\Support\Facades\Route;
5 use App\Http\Controllers\GraphController;
6
7
8 Route::get('/', [GraphController::class, 'index'])->name('graph');
9 Route::get('/graph/export', [GraphController::class, 'exportToExcel'])->name('graph.export');
```

5. Buat file graph.blade.php pada folder resources/views dan tambahkan code berikut:



```
1 <!DOCTYPE html>
2 <html lang="id">
3
4
5 <head>
6 <meta charset="UTF-8">
7 <meta name="viewport" content="width=device-width, initial-scale=1.0">
8 <title>Dashboard Monitoring Sensor | Sistem IoT</title>
9
10
11 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
12 <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&display=swap">
13
14
```



```
1 <!DOCTYPE html>
2 <html lang="id">
3
4
5 <head>
6 <meta charset="UTF-8">
7 <meta name="viewport" content="width=device-width, initial-scale=1.0">
8 <title>Dashboard Monitoring Sensor | Sistem IoT</title>
9
10
11 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
12 <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&display=swap">
13
14
```

PS D:\projek IoT\latihan2> php artisan serve

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

Press Ctrl+C to stop the server

## 6. Proses Berhasil

