

# PRAKTIK PENGEMBANGAN BACK END

06 November 2024

## FUNGSI CREATE DI LARAVEL

---

### a. Apa itu CRUD

CRUD adalah singkatan dari Create, Read, Update, dan Delete, yang merupakan empat fungsi dasar yang digunakan untuk mengelola data dalam basis data dan penting untuk sistem manajemen basis data. Fungsi-fungsi ini sering digunakan sebagai dasar untuk membangun aplikasi web.

Pertimbangkan panel admin atau backend apa pun, CRUD akan menjadi fungsi pentingnya untuk mengelola data.

### Membuat / create

Operasi create digunakan untuk **menambahkan data baru ke dalam basis data**. Misalnya, jika Anda memiliki basis data pelanggan dan ingin menambahkan pelanggan baru ke dalam basis data, Anda dapat menggunakan operasi create. Hal ini dapat melibatkan penambahan rekaman baru ke tabel pelanggan dengan nama pelanggan, alamat, dan informasi lainnya.

### Membaca / read

Operasi baca digunakan untuk **mengambil data dari basis data**. Misalnya, jika Anda ingin mengambil daftar semua pelanggan dalam basis data, Anda akan menggunakan operasi baca. Ini dapat melibatkan permintaan ke tabel pelanggan untuk mengambil semua rekaman dan menampilkannya di situs web atau aplikasi.

### Memperbarui / update

Operasi pembaruan digunakan untuk **mengubah data yang ada dalam basis data**. Misalnya, jika Anda ingin memperbarui alamat pelanggan dalam basis data, Anda akan menggunakan operasi pembaruan. Ini dapat melibatkan perubahan catatan dalam tabel pelanggan dengan informasi alamat baru.

## Menghapus / delete

Operasi hapus digunakan untuk **menghapus data dari basis data** . Misalnya, jika Anda ingin menghapus pelanggan dari basis data, Anda akan menggunakan operasi hapus. Ini dapat melibatkan penghapusan rekaman dari tabel pelanggan.

### b. Apa Itu Model

Model menjadi salah satu bagian penting dari konsep [MVC](#) (Model-View-Controller) pada framework Laravel yang bertanggung jawab untuk mengatur interaksi antara aplikasi dengan database. Dalam konsep MVC, Model dipadukan dengan Controller dan View untuk membentuk sebuah fitur atau halaman pada aplikasi Laravel kita. Model bertanggung jawab untuk memproses data dari database dan mengembalikan data tersebut ke Controller.

### c. Apa itu Controller

Dalam konteks Laravel, controller adalah kelas PHP yang bertanggung jawab untuk memproses permintaan HTTP dari pengguna dan mengembalikan respons yang sesuai. Controller menyatukan logika aplikasi dan memastikan bahwa permintaan dari pengguna diarahkan dengan benar ke model yang sesuai atau ke tampilan yang tepat.

### d. Apa itu Routes / routing

Routing adalah salah satu komponen inti dalam aplikasi web yang memungkinkan Anda mendefinisikan alamat URL yang akan dipetakan ke controller tertentu dalam aplikasi Anda. Dalam Laravel, routing digunakan untuk mengarahkan semua request HTTP ke handler / method yang tepat.

Routing di Laravel sangat fleksibel dan memberi Anda kontrol penuh atas setiap request ditangani oleh aplikasi Anda. Ini

memungkinkan struktur yang rapi dan mudah dikelola untuk aplikasi yang kompleks.

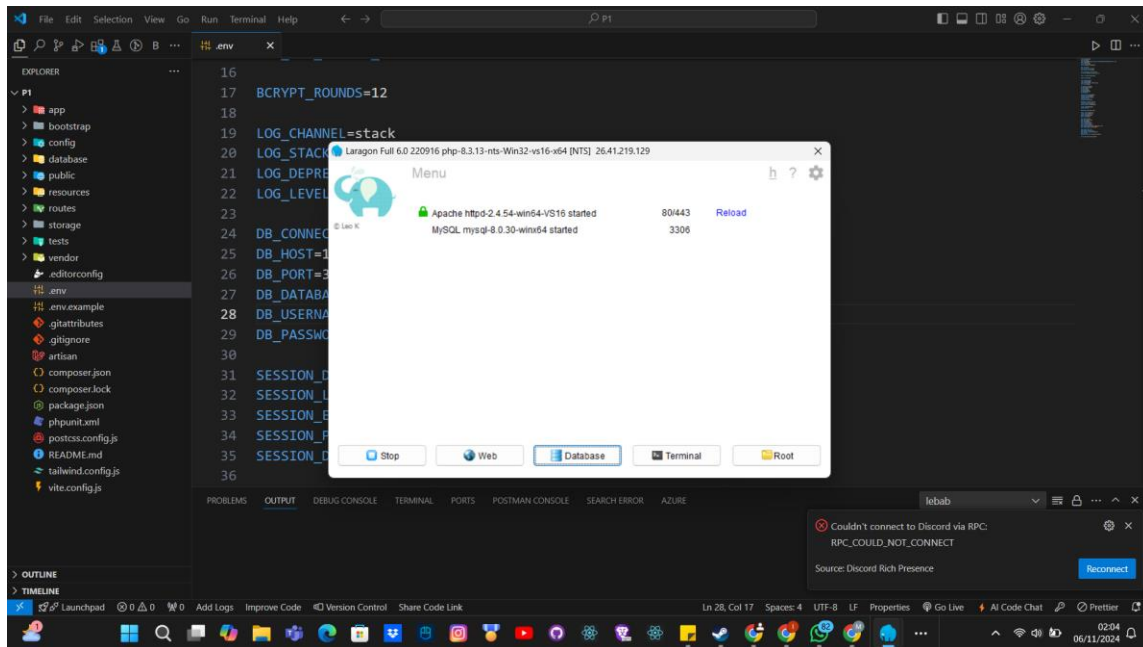
e. Konsep MVC (Model, View, Controller)

MVC adalah sebuah pendekatan perangkat lunak yang memisahkan aplikasi logika dari presentasi. MVC memisahkan aplikasi berdasarkan komponen-komponen aplikasi, seperti : manipulasi data, controller, dan user interface.

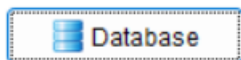
1. Model, Model mewakili struktur data. Biasanya model berisi fungsi-fungsi yang membantu seseorang dalam pengelolaan basis data seperti memasukkan data ke basis data, pembaruan data dan lain-lain.
2. View, View adalah bagian yang mengatur tampilan ke pengguna. Bisa dikatakan berupa halaman web.
3. Controller, Controller merupakan bagian yang menjembatani model dan view.

## e. Proses Pembuatan Fungsi Create di Laravel

1. Buatlah database di phpMyAdmin, Untuk dipercobaan ini menggunakan Nama database : Percobaan1



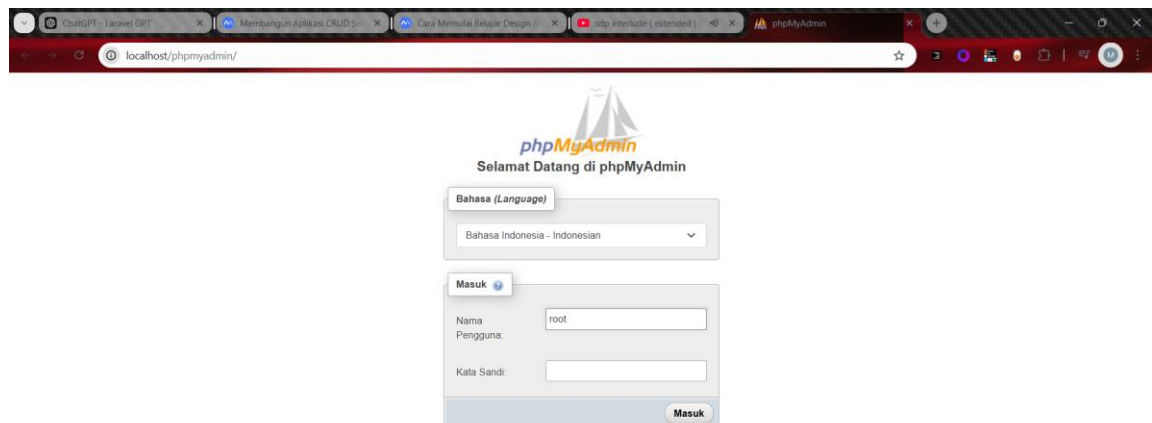
- a) Klik Database



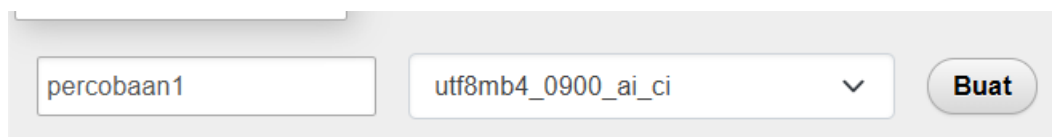
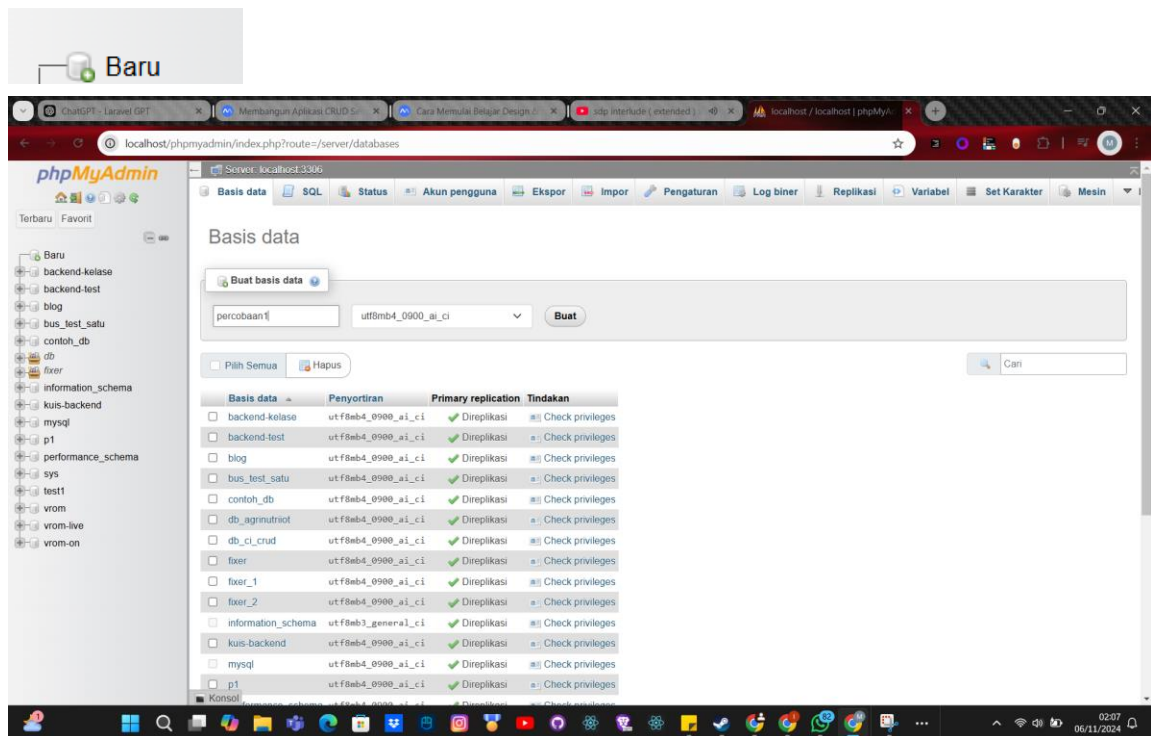
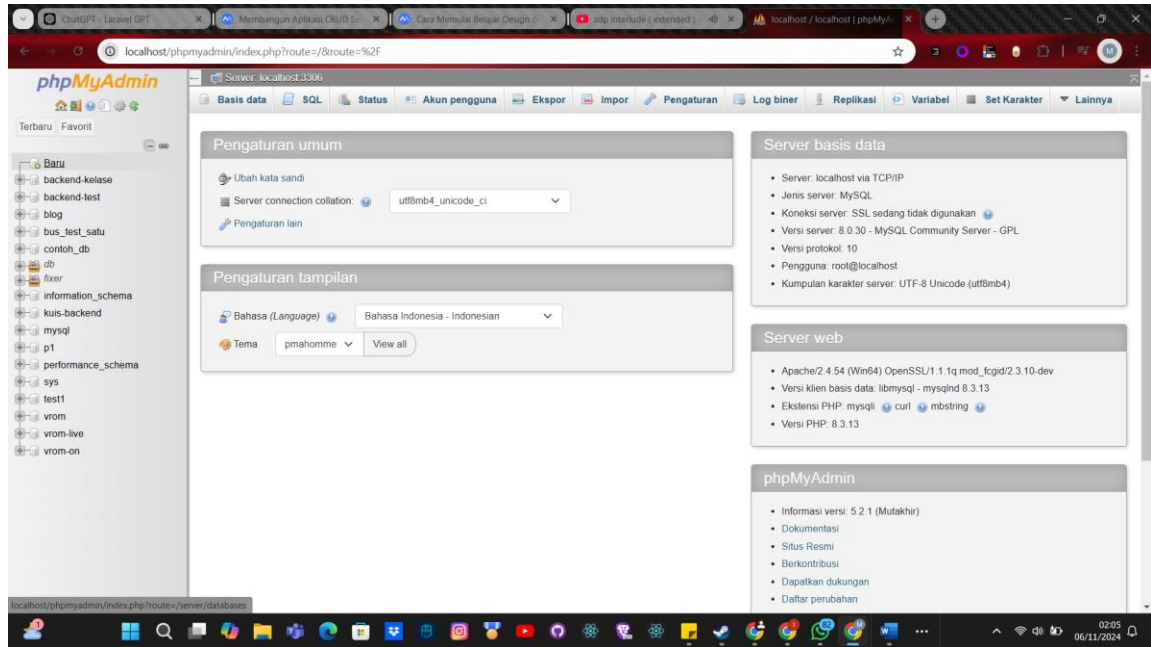
- b) Kemudian masuk ke phpmyadminnya

Username : root

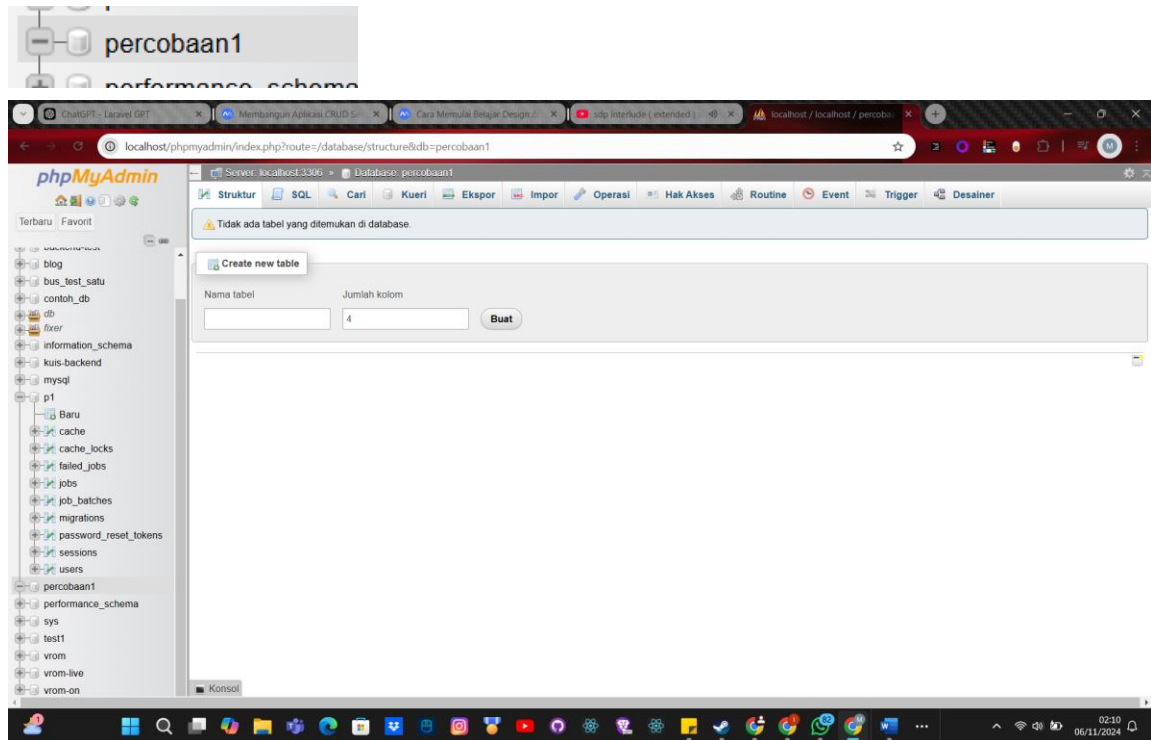
Pw :



c) Setelah itu buatlah databasenya dengan nama percobaan1



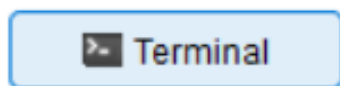
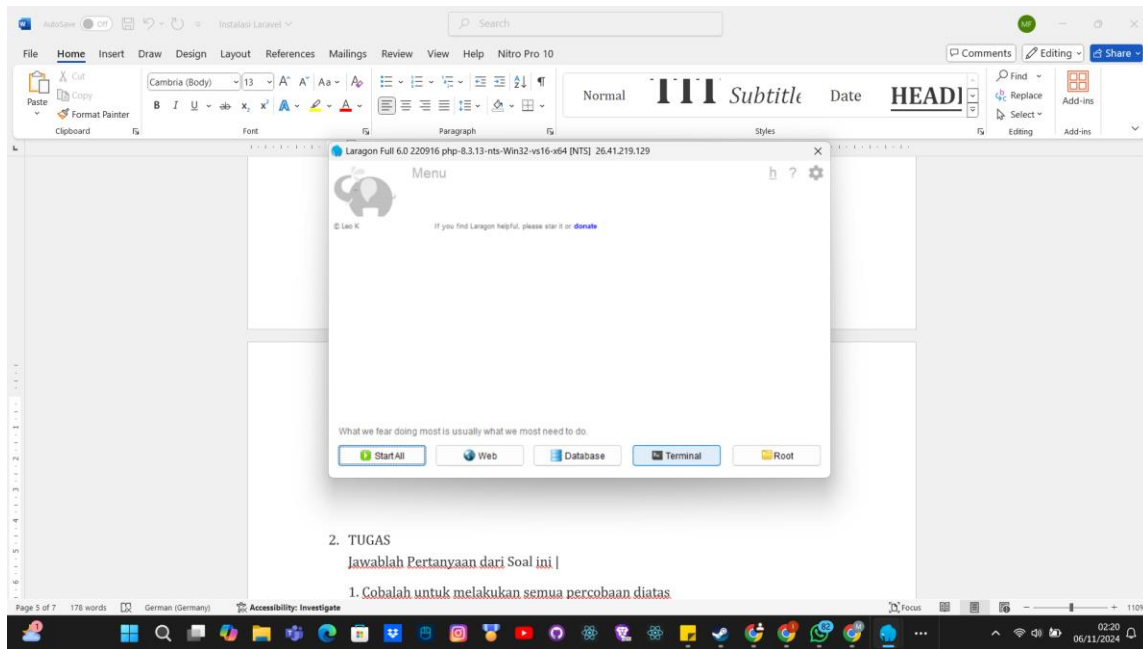
- d) Kemudian di cek apakah database sudah terbuat atau belum, jika sudah maka isinya sementara seperti ini



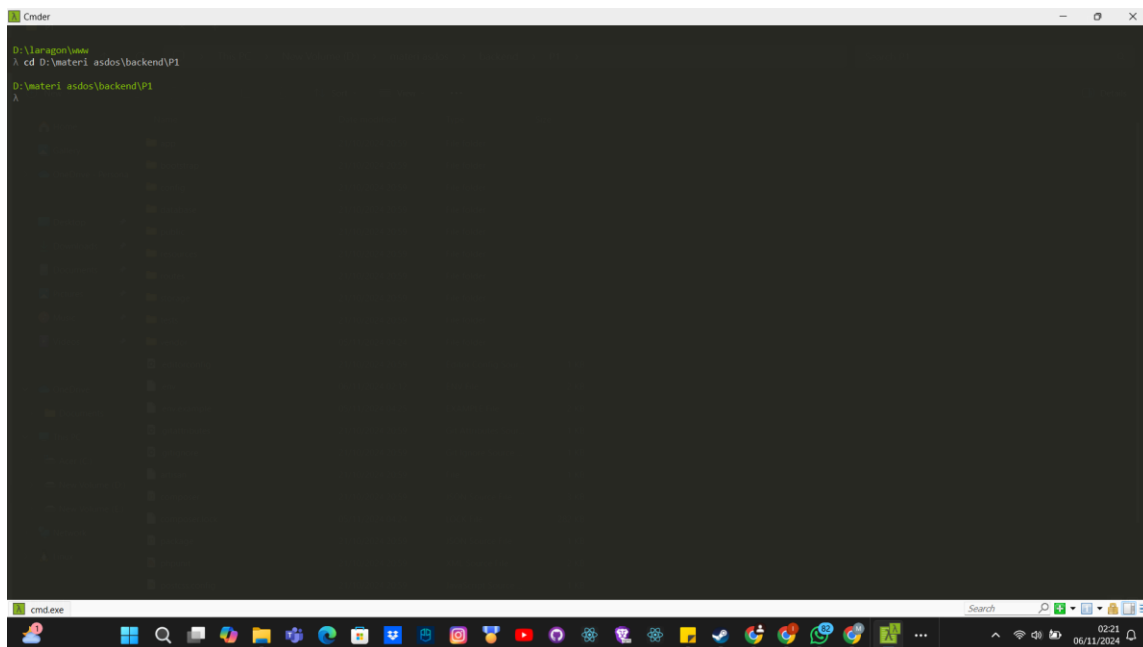
2. Setelah membuat database aturlah file .env nya, dengan konfigurasi seperti ini , yang diubah hanya nama dari databasenya saja.

```
23
24 DB_CONNECTION=mysql
25 DB_HOST=127.0.0.1
26 DB_PORT=3306
27 DB_DATABASE=percobaan1 #nama database yang dibuat
28 DB_USERNAME=root
29 DB_PASSWORD=
30
```

3. Kemudian, buka kembali ke laragon dan klik terminalnya



4. Setelah itu buatlah si terminal masuk ke folder project kalian



5. Kemudian untuk membuat tablenya dari si Laravel, kita harus membuat migration tabelnya, dengan cara membuat nya di terminalnya tadi dengan code :

Format :

```
php artisan make:migration create_students_table
```

contoh :

```
php artisan make:migration create_post_table
```

```
D:\materi asdos\backend\P1
λ php artisan make:migration create_post_table

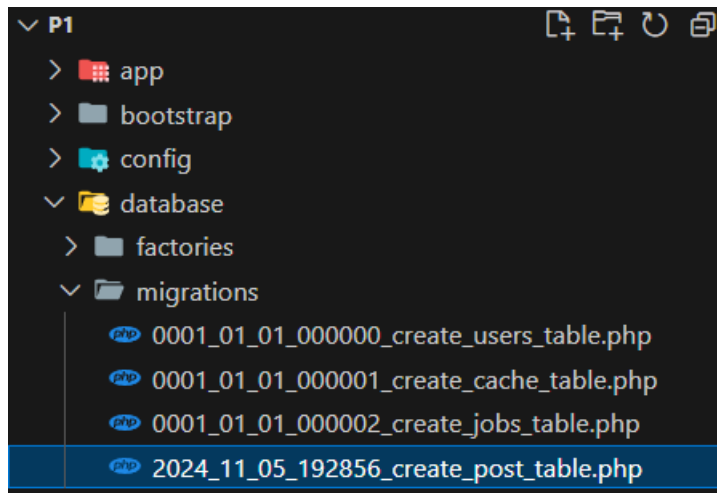
[INFO] Migration [D:\materi asdos\backend\P1\database\Migrations\2024_11_05_192856_create_post_table.php] created successfully.

D:\materi asdos\backend\P1
λ |
```

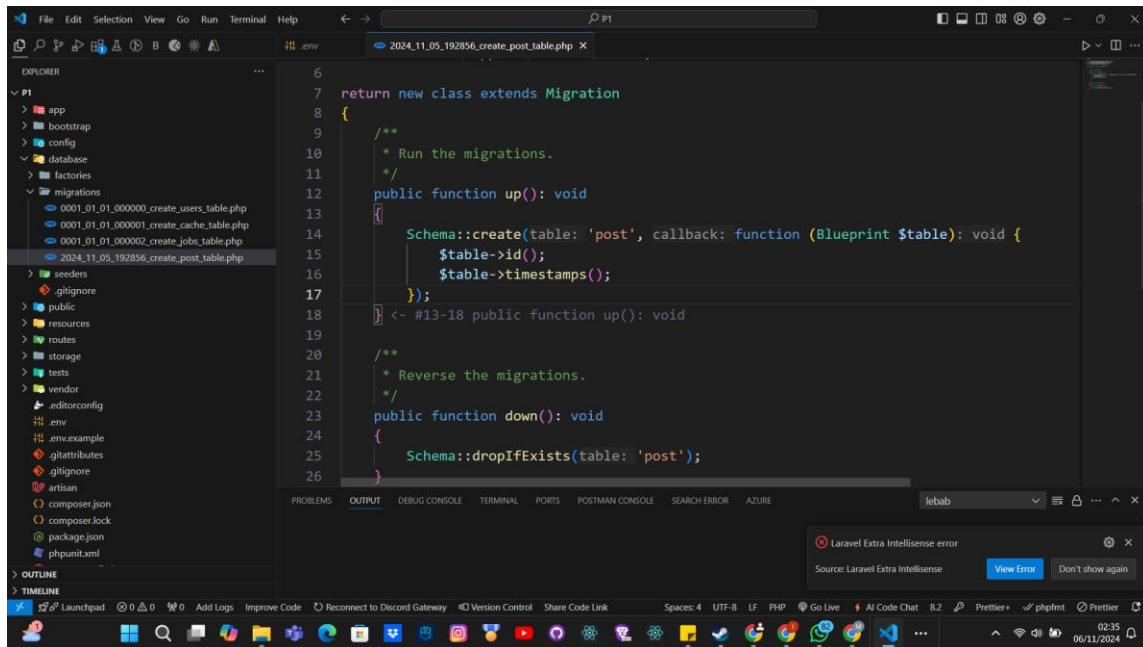
6. Kemudian setelah membuat migration tablenya, kemudian cek file migration apakah sudah terbuat atau belum letaknya di \database\migrations

```
[INFO] Migration [D:\materi asdos\backend\P1\database\Migrations\2024_11_05_192856_create_post_table.php]
```

Sesuai dengan informasi dari terminalnya tadi







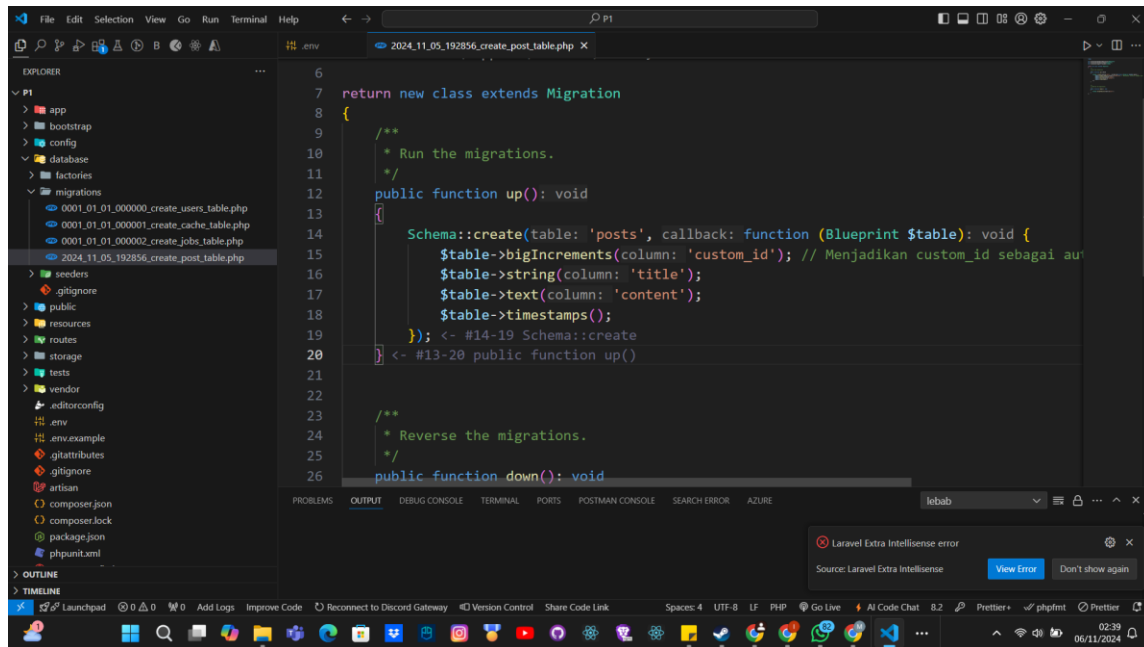
Sebelum diisi :



7. Kemudian kita isi bagian “public function up” nya agar bisa membuat isian tablenya dengan code ini :

```
public function up()
{
    Schema::create('posts', function (Blueprint $table) {
        $table->bigIncrements('custom_id'); // Menjadikan custom_id
        sebagai auto increment primary key
        $table->string('title'); // string = type data, title = variable
        $table->text('content');
        $table->timestamps();
    });
}
```

Sesudah diisi :

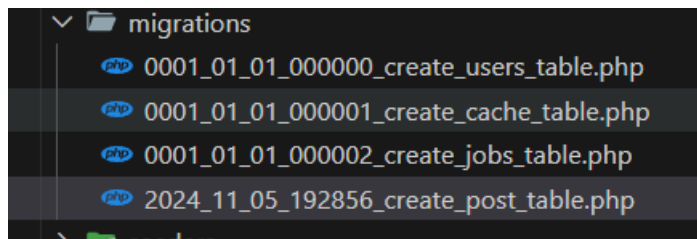


The screenshot shows the Visual Studio Code editor with a file named `2024_11_05_192856_create_post_table.php` open. The file contains a Laravel migration class that extends `Migration`. The `up()` method uses `Schema::create` to create a table named 'posts' with columns 'custom\_id', 'title', and 'content'. The `down()` method is also present. The Explorer sidebar on the left shows the project structure, including the 'migrations' folder. The bottom status bar indicates the file is using the 'larabab' theme.

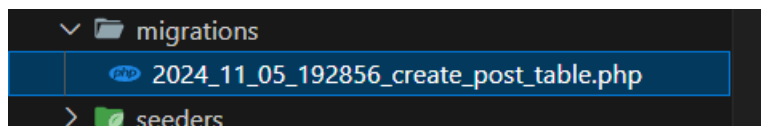
```
6
7 return new class extends Migration
8 {
9     /**
10      * Run the migrations.
11      */
12     public function up(): void
13     {
14         Schema::create(table: 'posts', callback: function (Blueprint $table): void {
15             $table->bigIncrements(column: 'custom_id'); // Menjadikan custom_id sebagai au
16             $table->string(column: 'title');
17             $table->text(column: 'content');
18             $table->timestamps();
19         }); <- #14-19 Schema::create
20     } <- #13-20 public function up()
21
22     /**
23      * Reverse the migrations.
24      */
25     public function down(): void
26     {
```

8. Kemudian hapuslah migration yang tidak terpakai, sehingga menyisakan yang terpakai :

Sebelum :



Sesudah :



9. Kemudian lakukan migrasi tablenya dengan buka terminalnya kembali dan ketik

“php artisan config:cache”

```
D:\materi asdos\backend\P1
λ php artisan config:cache

INFO Configuration cached successfully.
```

“php artisan session:table”

```
D:\materi asdos\backend\P1
λ php artisan session:table

INFO Migration created successfully.
```

“ php artisan migrate “

```
D:\materi asdos\backend\P1
λ php artisan migrate
```

```
D:\materi asdos\backend\P1
λ php artisan migrate

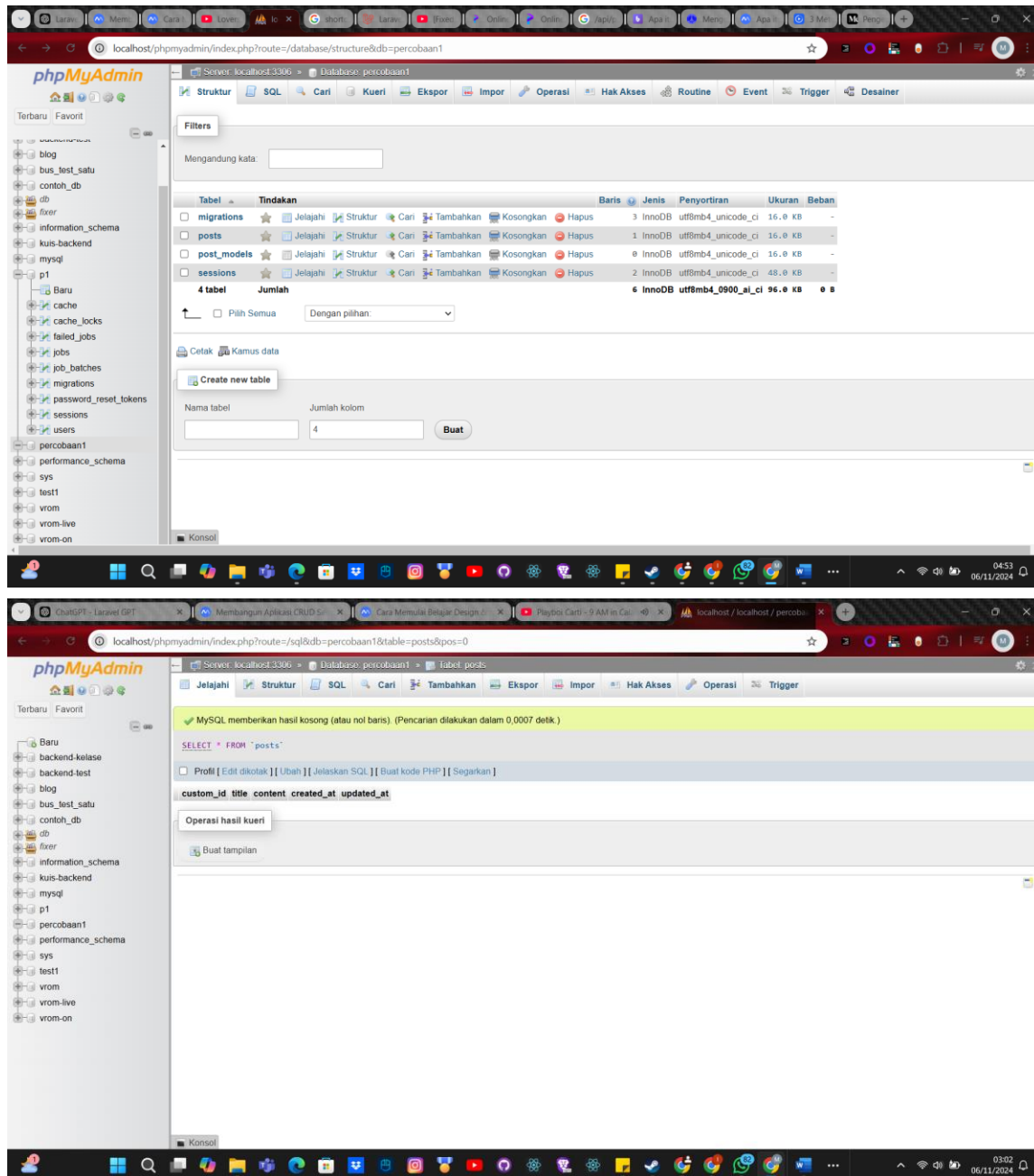
INFO Preparing database.
Creating migration table ..... 14.40ms DONE

INFO Running migrations.

2024_11_05_192856_create_post_table ..... 12.79ms DONE

D:\materi asdos\backend\P1
λ |
```

10. Kemudian cek di phpMyAdmin apakah table yang dibuat sudah terbuat atau belum,



Jika sudah seperti ini maka table tersebut sudah dibuat

11. Kemudian buatlah controller untuk si logika tablenya di terminal,

Format :

php artisan make:controller NamatableController

Contoh :

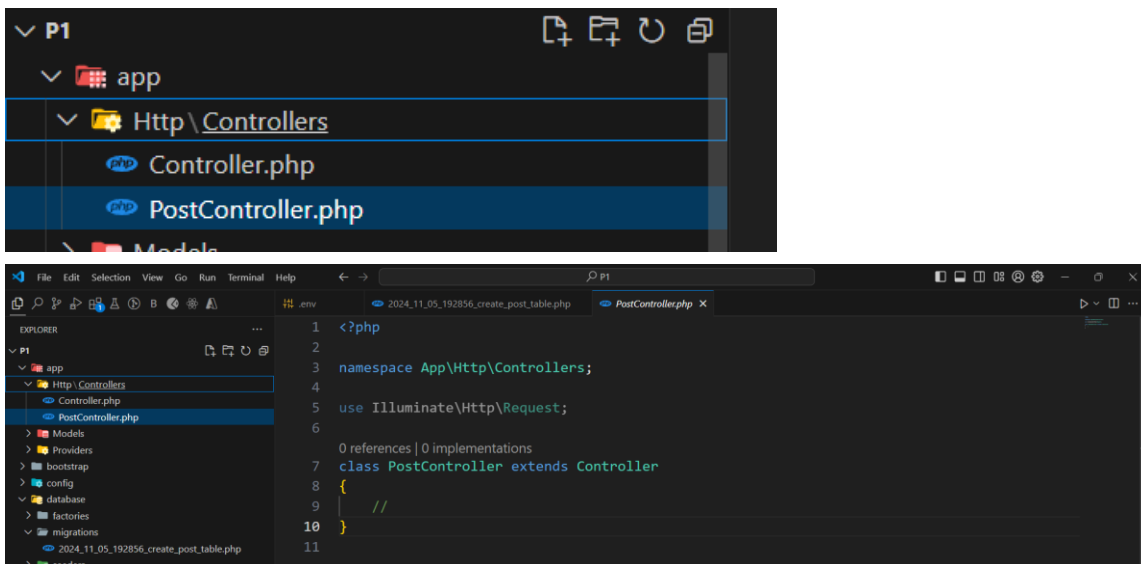
php artisan make:controller PostController

```
D:\materi asdos\backend\P1
λ php artisan make:controller PostController

INFO Controller [D:\materi asdos\backend\P1\app\Http\Controllers\PostController.php] created successfully.

D:\materi asdos\backend\P1
```

12. Kemudian cek di bagian \app\Http\Controllers , apakah controllernya sudah terbuat atau belum :



13. Kemudian setelah controllernya sudah dibuat, kemudian buatlah model dari tablenya, dengan cara buka Kembali terminalnya, dan ketik command ini :

Format :

php artisan make:model NamaModel -m

Contoh :

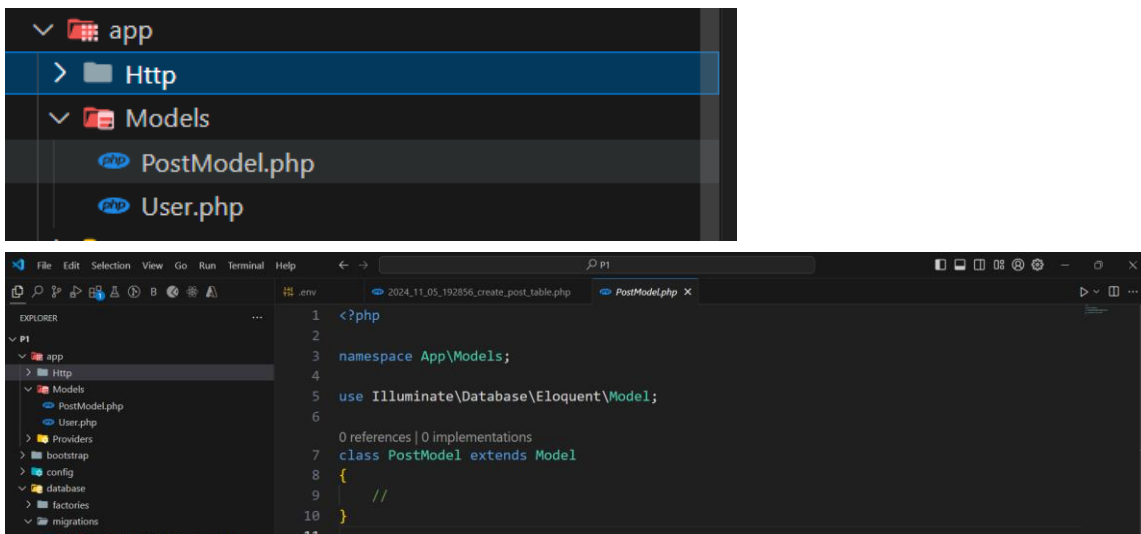
php artisan make:model PostModel -m

```
D:\materi asdos\backend\P1
λ php artisan make:model PostModel -m

[INFO] Model [D:\materi asdos\backend\P1\app\Models\PostModel.php] created successfully.
[INFO] Migration [D:\materi asdos\backend\P1\database\migrations\2024_11_05_201154_create_post_models_table.php] created successfully.

D:\materi asdos\backend\P1
λ
```

14. Kemudian cek di bagian `\app\Models`, apakah modelnya sudah terbuat atau belum :



15. Kemudian Isilah model tersebut dengan kode ini, sesuaikan dengan table yang dibuat :

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class PostModel extends Model
{
    use HasFactory;

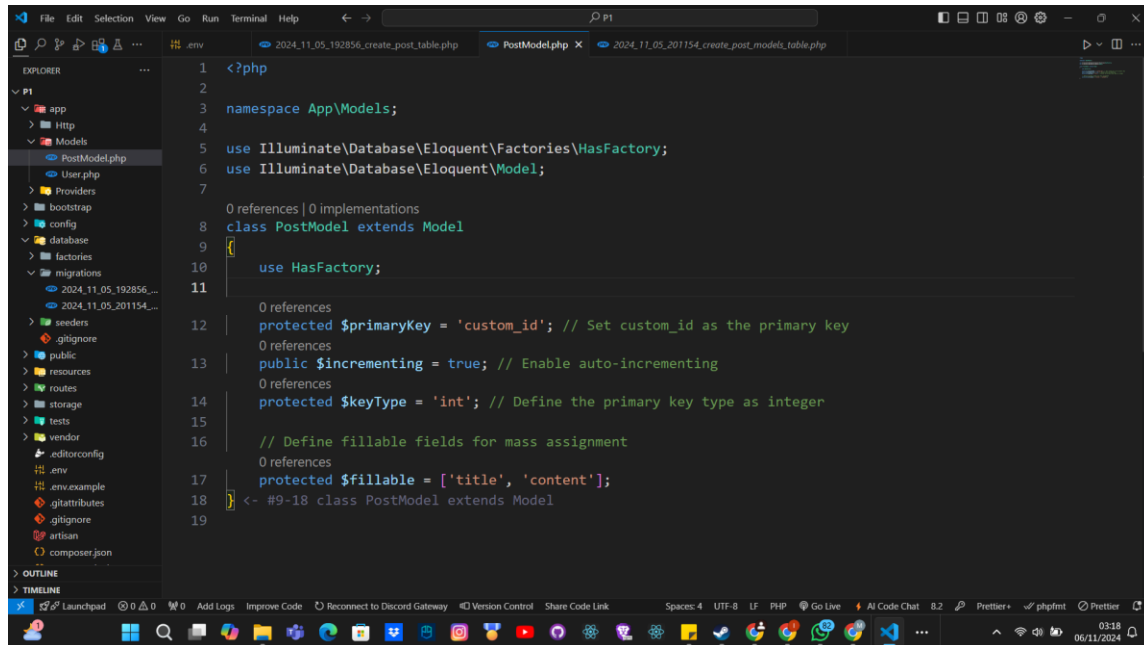
    protected $table = 'posts'; // Pastikan model merujuk ke tabel 'posts'
    protected $primaryKey = 'custom_id'; // Set custom_id as the primary key
    public $incrementing = true; // Enable auto-incrementing
}
```

```

protected $keyType = 'int'; // Define the primary key type as integer

// Define fillable fields for mass assignment
protected $fillable = ['title', 'content'];
}

```



16. Kemudian pergi Kembali ke controller, dan buatlah function untuk create, dengan kode ini :

```

<?php

namespace App\Http\Controllers;

use App\Models\PostModel;
use Illuminate\Http\Request;

class PostController extends Controller
{
    public function create(Request $request)
    {

```

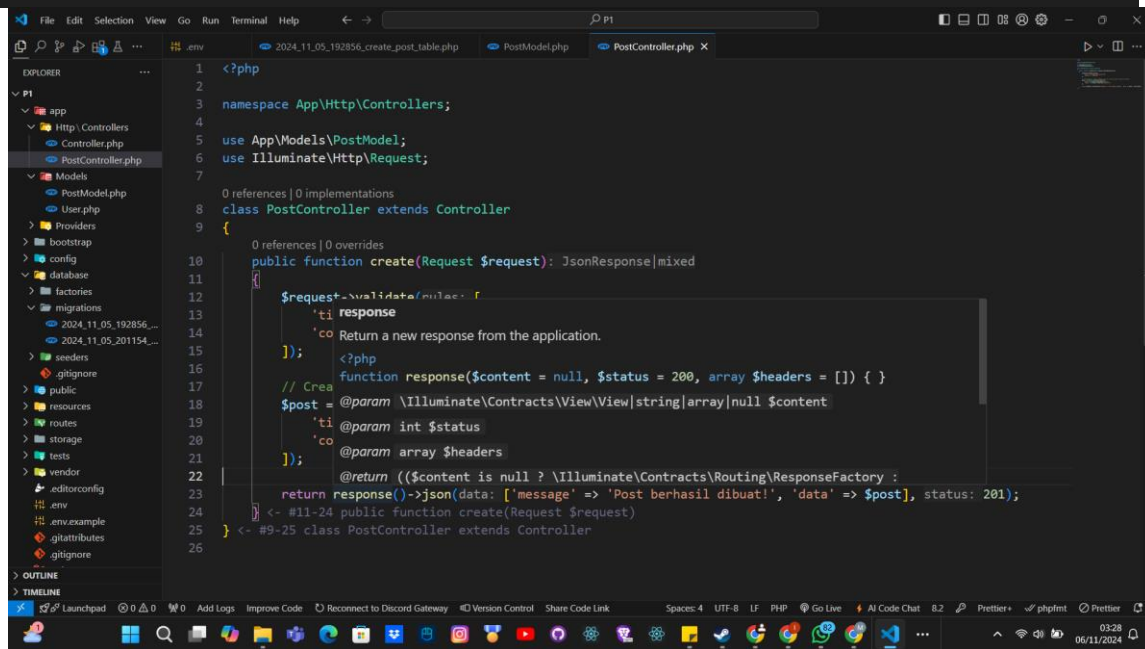
```

$request->validate([
    'title' => 'required|string|max:255',
    'content' => 'required',
]);

// Create the post without needing custom_id from the request (auto-incremented)
$post = PostModel::create([
    'title' => $request->input('title'),
    'content' => $request->input('content'),
]);

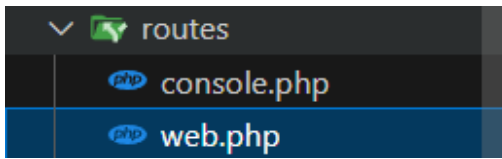
return response()->json(['message' => 'Post berhasil dibuat!', 'data' => $post], 201);
}
}

```





17. Kemudian setelah controller dibuat, kemudian buatlah routesnya, buka file routes di routes\web.php



18. Kemudian jika sudah dibuat, bisa dimasukkan code routesnya berdasarkan dari controller yang dibuat, contoh seperti ini :

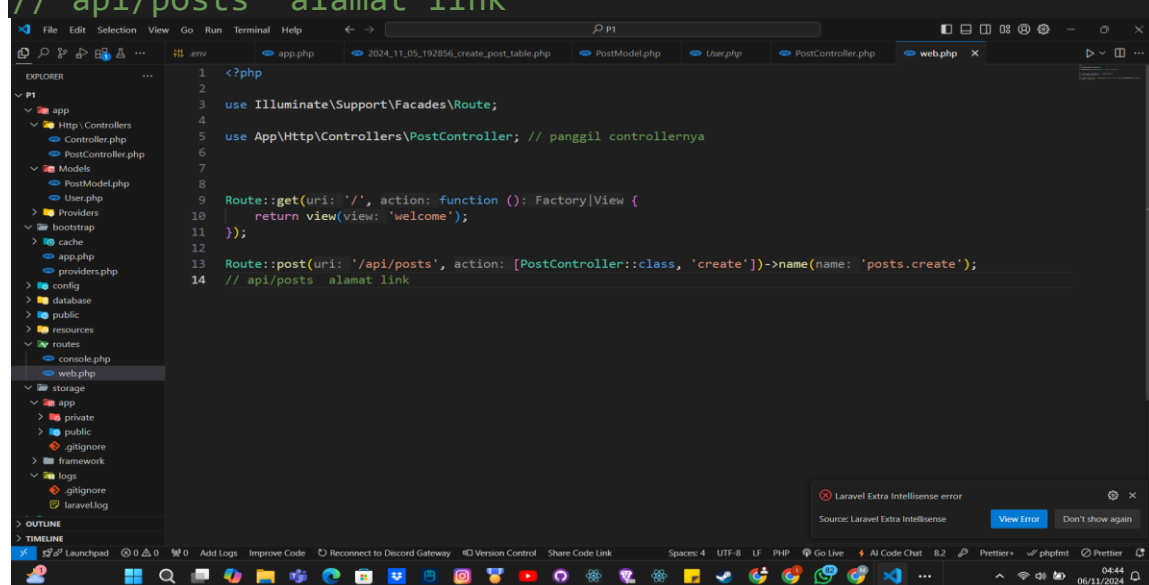
```
<?php

use Illuminate\Support\Facades\Route;

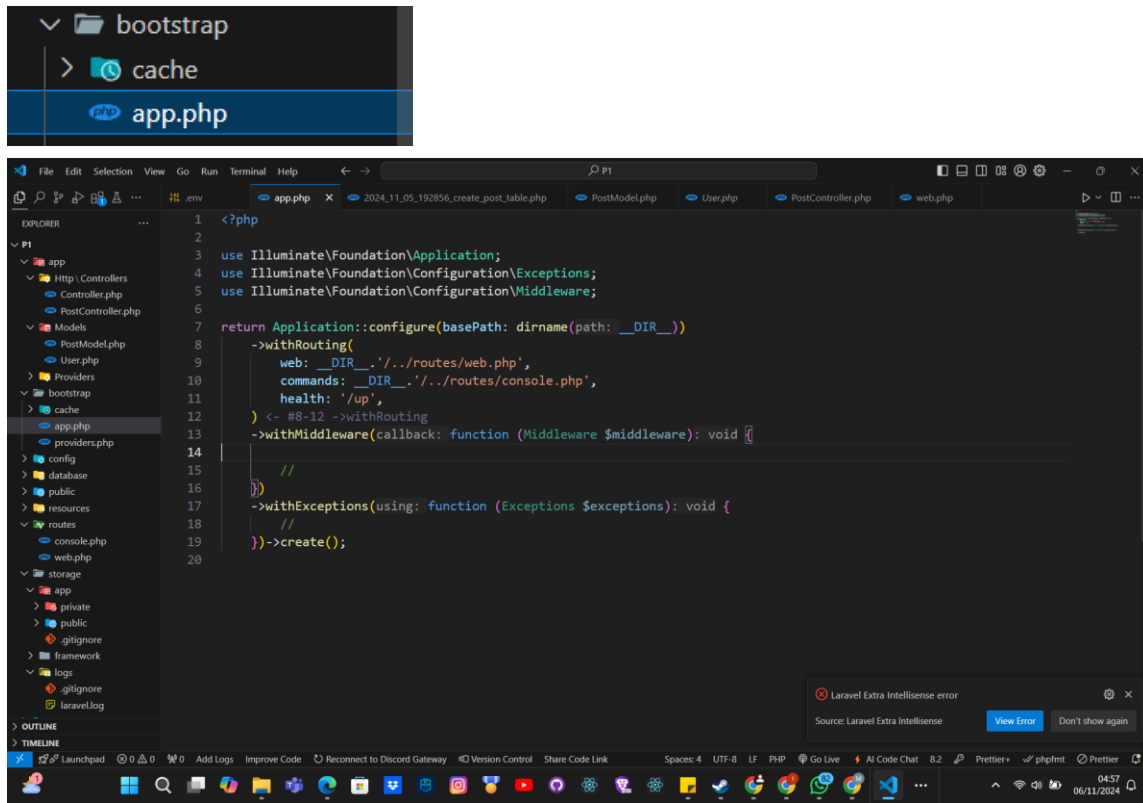
use App\Http\Controllers\PostController; // panggil
controllernya

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

Route::post('/api/posts', [PostController::class,
'create'])->name('posts.create');
// api/posts alamat link
```

A screenshot of a Visual Studio Code editor window. The main editor area displays the PHP code from the previous block, which defines routes for a Laravel application. The code includes using the Route facade, importing the PostController, and defining a GET route for the root and a POST route for '/api/posts'. The Explorer sidebar on the left shows the project structure, including the 'routes' directory. The bottom status bar indicates the current file is 'web.php'. There is a small error message at the bottom right: 'Laravel Extra Intellisense error'.

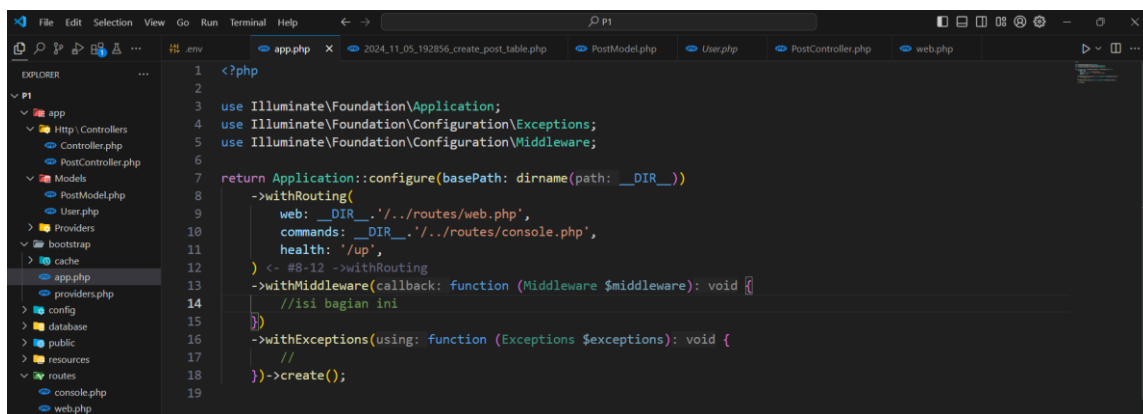
19. Setelah membuat routesnya, kemudian buka bootstrap\app.php



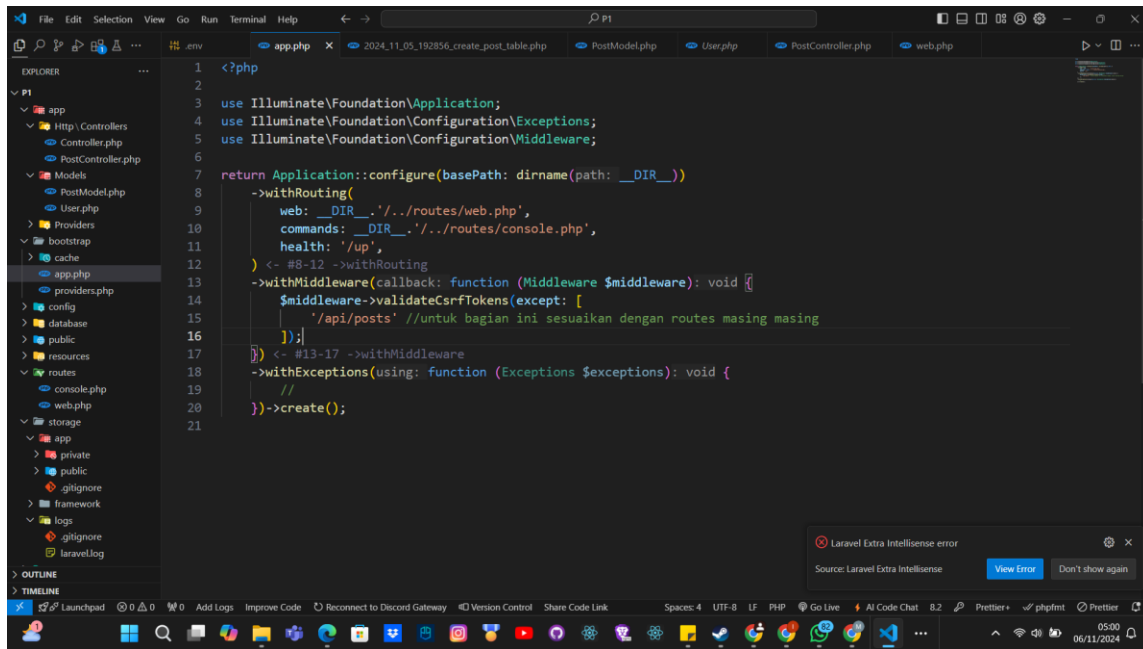
20. Setelah itu isillah bagian retyrn yang withMiddleware, dan isikanlah dengan code ini :

```
$middleware->validateCsrfTokens(except: [
    '/api/posts' //untuk bagian ini sesuaikan dengan routes masing masing
]);
```

Sebelum :



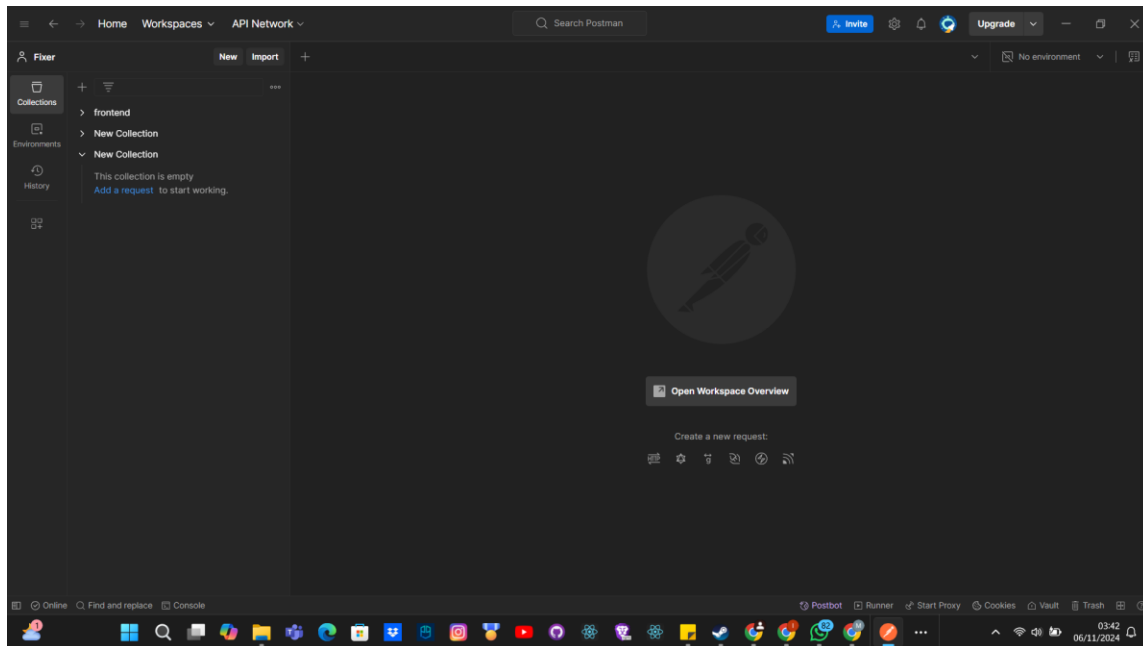
Sesudah :



21. Kemudian buka kembali terminal dan nyalakan project laravelnya dengan command :
- php artisan serve



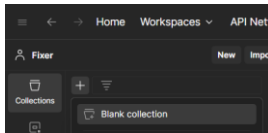
22. Kemudian buka postman, setelah itu buatlah collectionnya dengan nama project yang kalian buat.



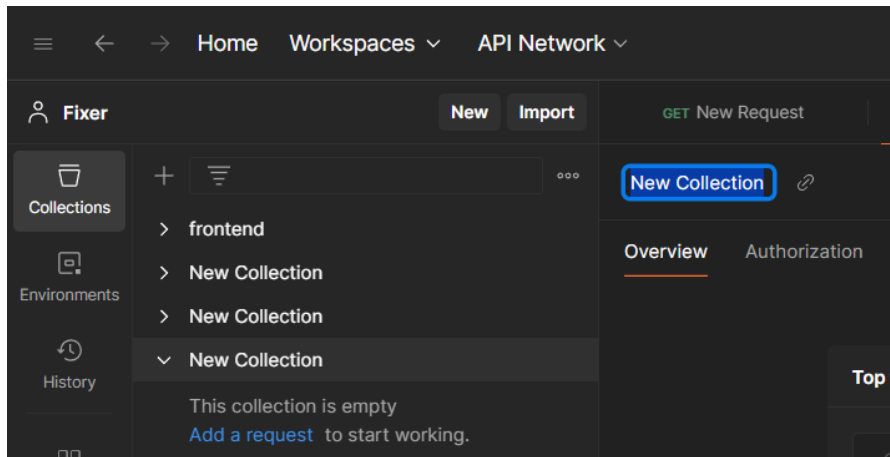
Klik tombol tambah



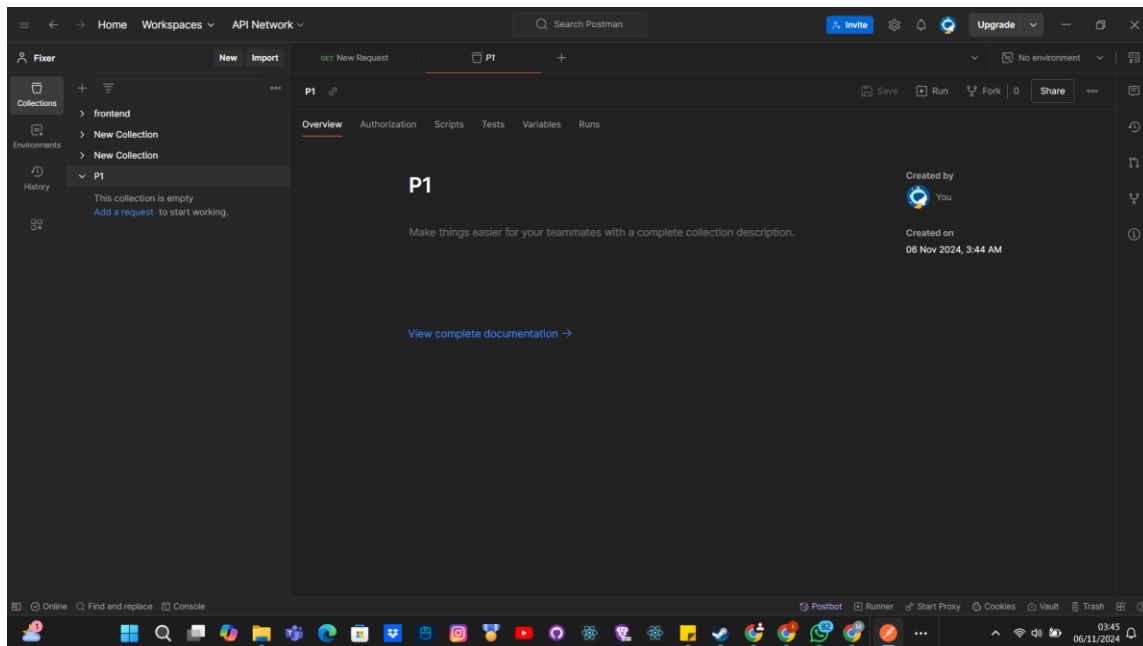
Kemudian pilih yang blank collection



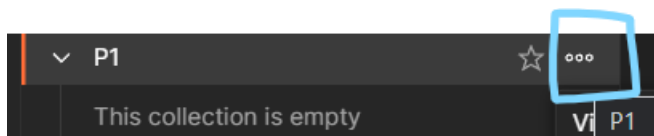
Setelah itu jangan lupa untuk melakukan rename namanya dengan project kalian, dengan klik new collection (yang ditandai biru)



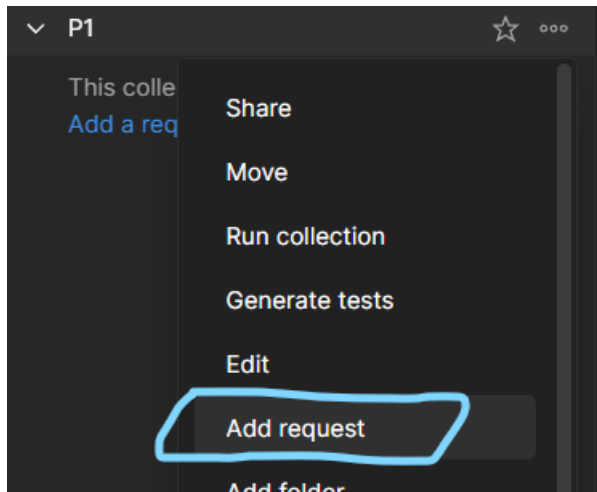
Contoh :



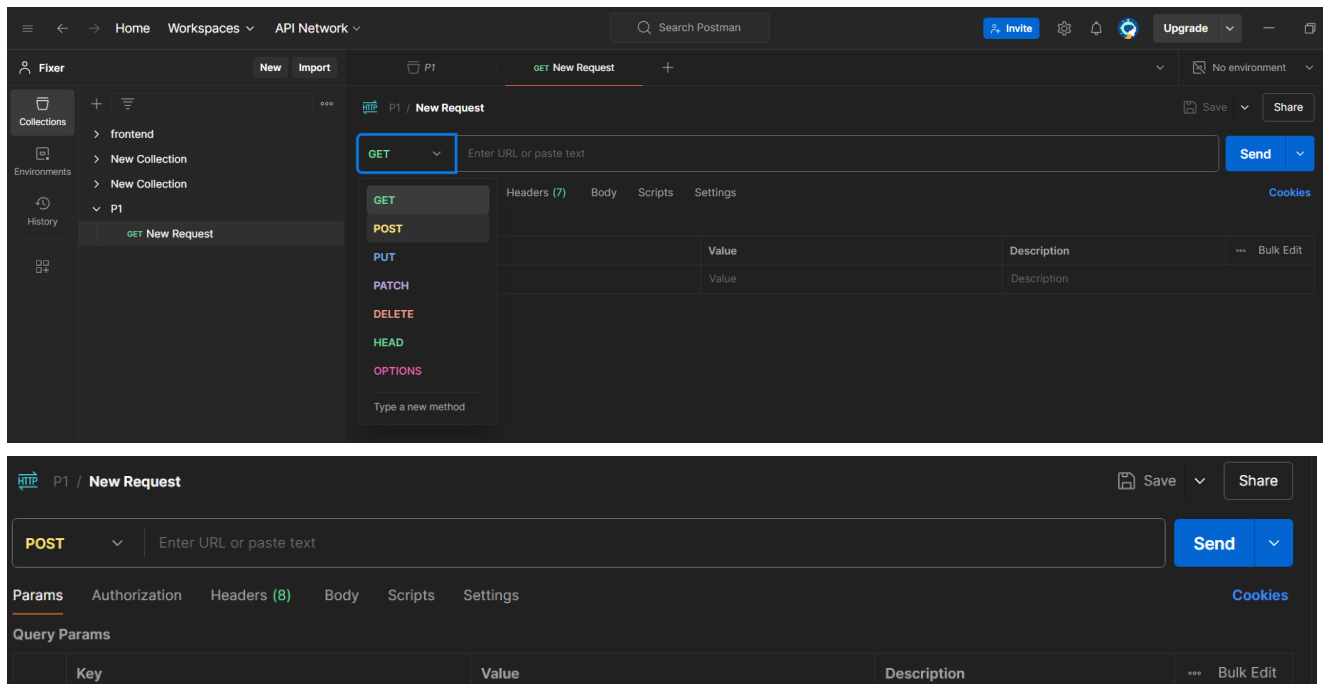
23. Setelah terbuat, klik titik tiga yang berada di sebelah nama project



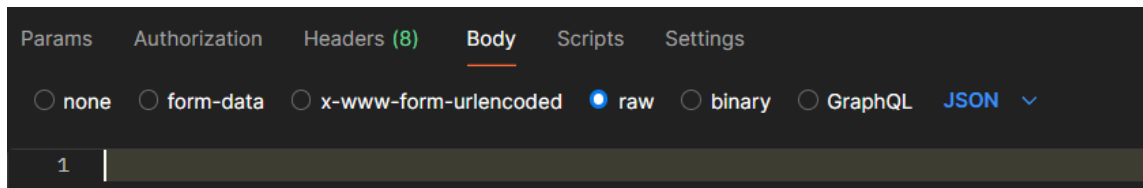
24. Kemudian klik add request



25. Setelah itu gantilah requestnya dari sebelumnya Get menjadi Post

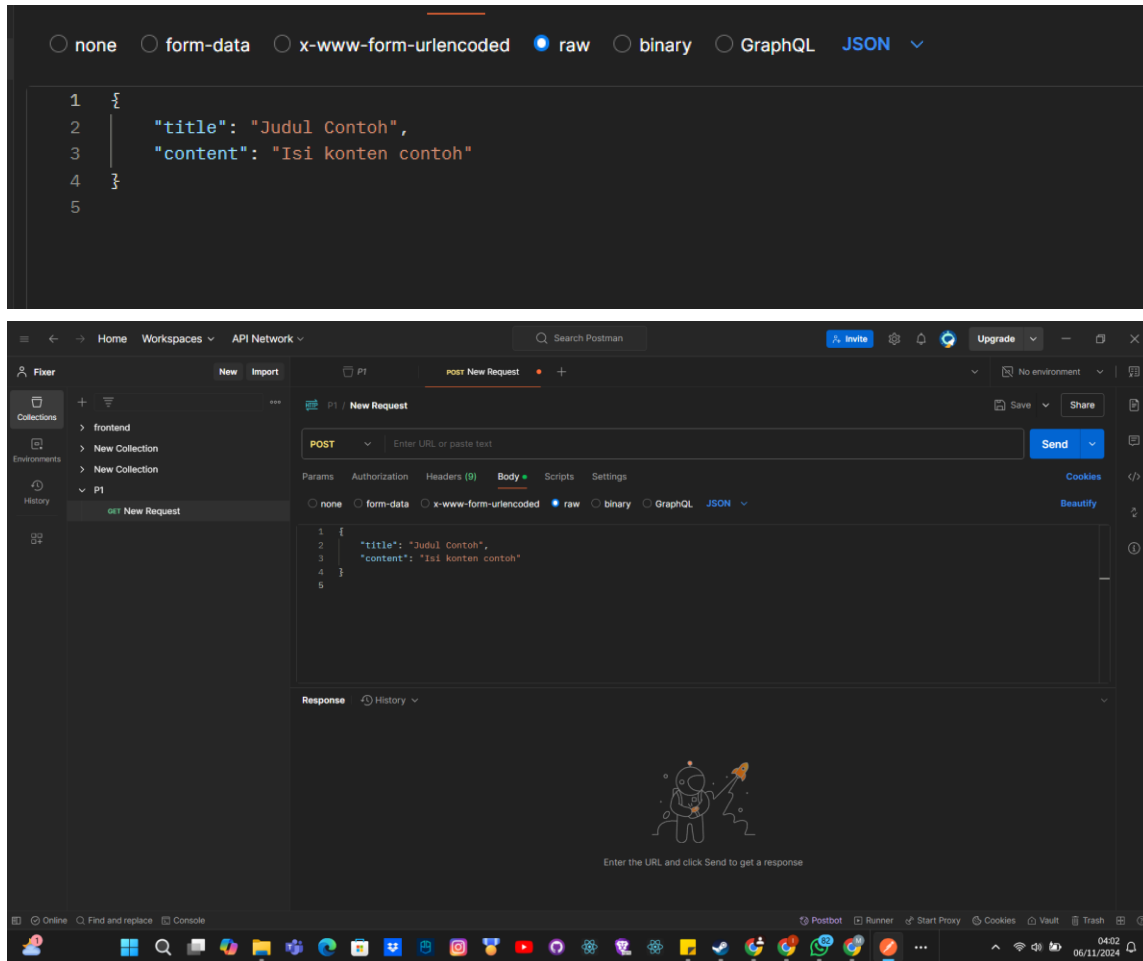


26. Setelah itu masukkan lah contoh inputannya dalam format json ini  
dengan cara klik dibawah nya url, pilih yang Body, kemudian formatnya  
pilih yang json



27. Setelah itu isilah dengan format data yang sesuai dengan controllernya :

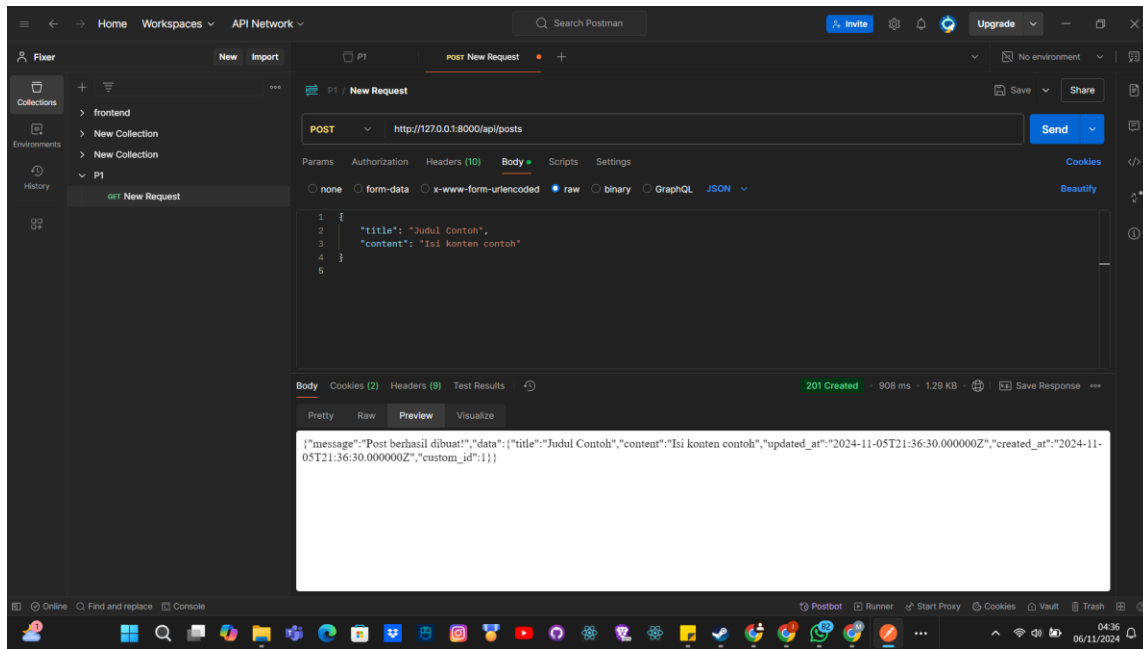
Contoh :



28. Setelah itu isilah url atasnya sesuai dengan routes yang dibuat :



29. Kemudian klik send dan hasilnya seperti ini





## 2. TUGAS

Jawablah Pertanyaan dari Soal ini |

1. Cobalah untuk melakukan semua percobaan diatas
2. Kemudian buatlah versi project kalian sendiri
3. minimal sudah 3 table yang bisa melakukan proses create

### PENGUMPULAN TUGAS DAN DEADLINE

- a. Pengumpulan Tugas di Google Drive
- b. Deadline = 07 November 2024 | 23.00 WIB

## DAFTAR PUSTAKA

<https://buildwithangga.com/tips/membangun-aplikasi-crud-sederhana-menggunakan-laravel-11>

<https://baraka.uma.ac.id/mengenal-controller-pada-laravel-pusat-kendali-aplikasi-web/#:~:text=Dalam%20konteks%20Laravel%2C%20controller%20adalah,atau%20ke%20tampilan%20yang%20tepat.>

<https://buildwithangga.com/tips/apa-itu-model-pada-framework-laravel>  
<https://www.php.net/downloads>

<https://parsinta.com/articles/3-metode-laravel-routing-yang-sering-digunakan-ztwbcp>

<https://medium.com/@kevinffa0107/pengertian-mvc-model-view-controller-pada-framework-laravel-20f261ccf233>