

Dokumentasi Request

KelasController adalah controller yang berada dalam namespace `App\Http\Controllers`. Controller ini menggunakan framework Laravel dan bertanggung jawab untuk mengelola data yang berkaitan dengan 'kelas'.

Metode index

Tujuan: Metode index digunakan untuk menampilkan halaman index dari 'kelas' dengan kemampuan pencarian.

Parameter:

Request \$request: Objek request dari Laravel yang digunakan untuk menerima data dari user.

Proses:

Mengambil parameter **search** dari query string yang dikirimkan oleh user.

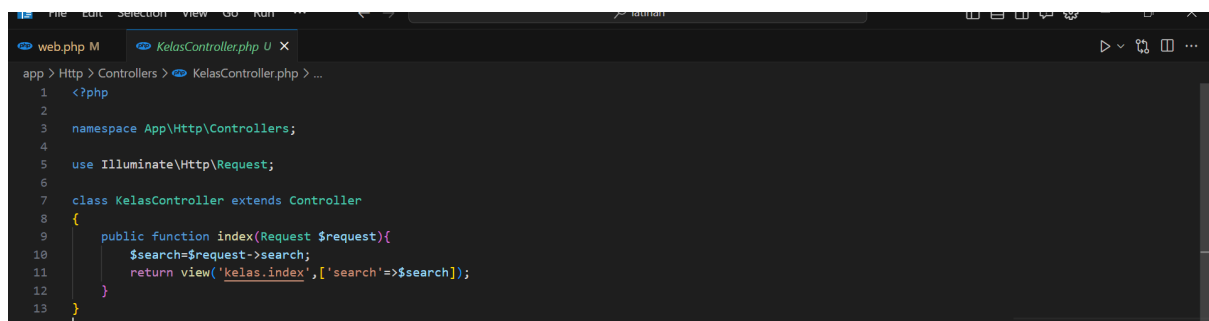
Parameter **search** ini digunakan untuk filter pencarian pada halaman 'kelas'.

Return:

Mengembalikan view `kelas.index` dengan data **search** yang telah diambil dari request.

View:

kelas.index: View yang akan menampilkan data atau informasi yang berkaitan dengan pencarian 'kelas'.

A screenshot of a code editor showing the implementation of the `index` method in the `KelasController` class. The code is written in PHP and includes the following lines:

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 class KelasController extends Controller
8 {
9     public function index(Request $request){
10         $search=$request->search;
11         return view('kelas.index',['search'=>$search]);
12     }
13 }
```

Membuat Model

Untuk membuat model di Laravel, Anda dapat mengikuti langkah-langkah berikut. Model ini akan berinteraksi dengan database dan menyediakan cara untuk mengelola data yang terkait dengan entitas tertentu dalam aplikasi Anda. Di bawah ini adalah dokumentasi langkah demi langkah untuk membuat model `Kelas`:

Langkah 1: Membuat Model

Untuk membuat model, Anda dapat menggunakan Artisan CLI yang disediakan oleh Laravel. Buka terminal atau command prompt dan jalankan perintah berikut di direktori root proyek Laravel Anda:

```
php artisan make:model Kelas
```

Perintah ini akan membuat file model baru di dalam direktori `app/Models` dengan nama `Kelas.php`

Langkah 2: Mengkonfigurasi Model

Setelah model dibuat, buka file `app/Models/Kelas.php` untuk mengkonfigurasinya. Berikut adalah contoh konfigurasi dasar:

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
class Kelas extends Model
{
    // Tentukan nama tabel jika berbeda dari nama model dalam bentuk plural
    protected $table = 'kelas';

    // Primary key tabel
    protected $primaryKey = 'id';

    // Daftar kolom yang dapat diisi
    protected $fillable = ['nama_kelas', 'tingkat', 'wali_kelas'];

    // Menonaktifkan timestamps jika tidak diperlukan
    public $timestamps = false;
}
```

Langkah 3: Menyesuaikan Model

\$table : Properti ini digunakan untuk mendefinisikan nama tabel yang model ini akan mewakili. Secara default, Laravel mengasumsikan tabel dengan nama plural dari nama model.

\$primaryKey : Properti ini digunakan untuk menentukan kolom yang digunakan sebagai primary key.

\$fillable: Array ini digunakan untuk menentukan kolom mana dalam tabel yang dapat diisi menggunakan mass assignment.

\$timestamps : Laravel secara default mengharapkan kolom `created_at` dan `updated_at` di tabel. Jika tabel Anda tidak memiliki kolom ini, setel properti ini ke `false`.

Menampilkan Data Kelas ke View `kelas.index`

Controller: `KelasController`

File: `app/Http/Controllers/KelasController.php`

Fungsi: `index`

Deskripsi:

Fungsi index digunakan untuk mengambil data kelas dari database dan menampilkannya pada view `kelas.index`.

Proses:

1. Mengambil nilai pencarian dari request jika ada:

```
$search = $request->search;
```

2. Mengambil semua data kelas dari model `Kelas`:

```
$dataKelas = Kelas::get();
```

3. Mengembalikan view `kelas.index` dengan data yang diperoleh:

```
return view('kelas.index', [
    'search' => $search,
    'dataKelas' => $dataKelas
]);
```

View:

Lokasi view: `resources/views/kelas/index.blade.php`

Data yang dikirim ke view:

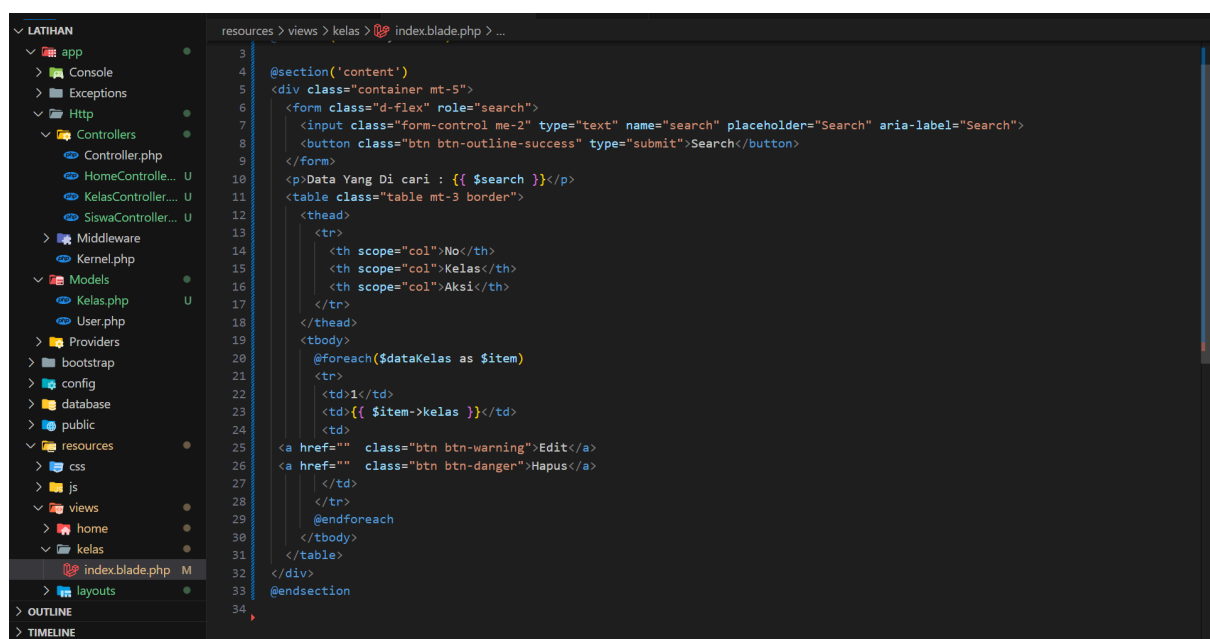
search: Kata kunci pencarian .

dataKelas: Koleksi data kelas yang diambil dari database.

Contoh Penggunaan di View:

Dalam file view `kelas.index`, Anda dapat menampilkan data kelas dengan melakukan iterasi pada `dataKelas` dan menggunakan `search` untuk menampilkan pencarian yang dilakukan pengguna.

Contoh:



Catatan:

Pastikan bahwa model `Kelas` dan view `kelas.index` sudah terdefinisi dengan benar untuk menghindari error saat mengakses data atau render view.