

Tugas Pertemuan 12
RestApi menggunakan framework yang terdiri dari

Nama : Irfan Mulyana Abdillah
Kelas : SE02-A
Nim : 18104013

Tugas Restful Api Framework Laravel

Disini saya akan membuat project sederhana tentang Restfull Api menggunakan laravel 7, sebelumnya hal yang harus dipersiapkan untuk membuat Rest Api adalah:

1. Xampp
2. Composer
3. Laravel
4. Kode editor
5. Postman ataupun aplikasi SOAP dan rest lainnya

Langkah pertama:

Instal composer jika belum, anda bisa download di situs resminya namun jika windows bisa langsung download file dengan ekstensi.exe dan instal seperti aplikasi pada umumnya

Langkah kedua:

Buat project laravel, sebelum itu pastikan sudah berada di direktori htdocs pada XAMPP jika sudah kalian bisa menjalankan perintah dibawah ini untuk mendownloadnya:

```
composer create-project --prefer-dist laravel/laravel:^7.0 nama_folder
```

Contoh:

```
composer create-project --prefer-dist laravel/laravel:^7.0 Restapi
```

Jika sudah, aktifkan xampp dan buatlah database dengan namanya terserah kalian. Selanjutnya buka project laravel yang kalian telah buat pada kode editor favorit kalian. Alihkan perhatian pada file nama .env dan buka kemudian ganti nilai dari

```
DB_DATABASE = 'isi sesuai nama database yang kalian buat'
```

Contoh:

```
DB_CONNECTION=mysql  
DB_HOST=127.0.0.1  
DB_PORT=3306  
DB_DATABASE=coba_api  
DB_USERNAME=root  
DB_PASSWORD=
```

Langkah ketiga:

Membuat model, controller berupa resource, serta controllernya langsung dengan cara seperti dibawah ini:

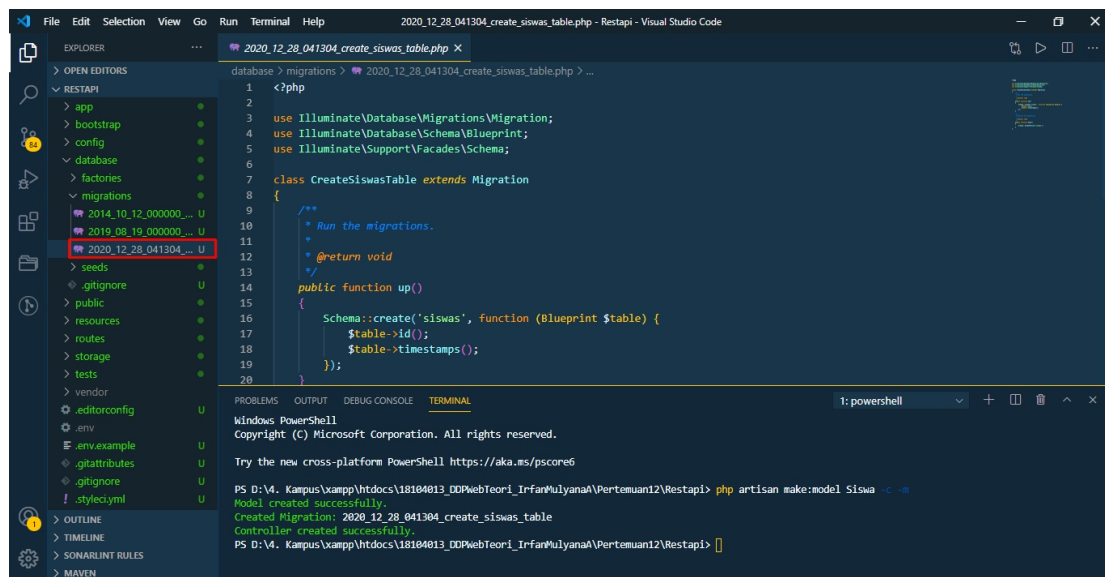
```
php artisan make:model Siswa -c -m
```

Artinya kita membuat model dengan nama siswa dengan menyertakan controller serta migration pada controllernya. Namun cara manual seperti berikut ini contohnya:

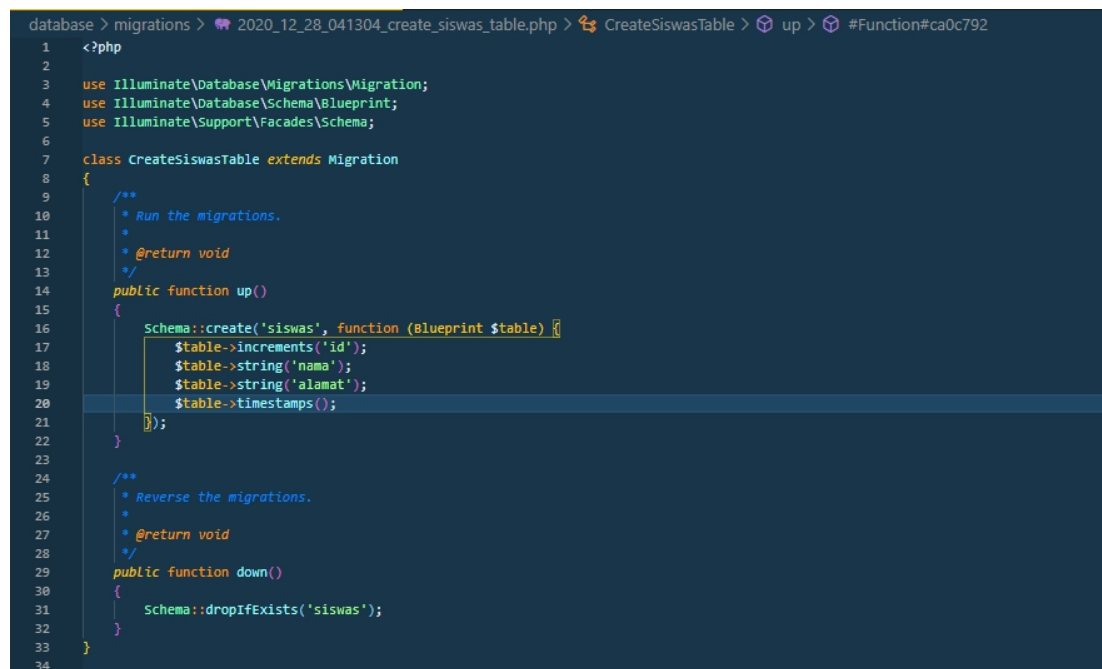
php artisan make:migration nama_file_migrasi

Contoh:

php artisan make:migration create_siswas_table



Gambar 1.1 migration create_siswas_table



Gambar 1.2 Membuat table

Gambar 1.2 disini kita akan membuat table id (primary key), nama (string), alamat (string) kemudian kita akan me migratennya, namun sebelum itu kita atur pada **app/providers/AppServiceProvider.php** tambahkan kode seperti dibawah ini:

```

app > Providers > AppServiceProvider.php > AppServiceProvider > boot
1  <?php
2
3  namespace App\Providers;
4
5  use Illuminate\Support\Facades\Schema;
6  use Illuminate\Support\ServiceProvider;
7
8  class AppServiceProvider extends ServiceProvider
9  {
10     /**
11      * Register any application services.
12      *
13      * @return void
14      */
15     public function register()
16     {
17         //
18     }
19
20     /**
21      * Bootstrap any application services.
22      *
23      * @return void
24      */
25     public function boot()
26     {
27         //
28         Schema::defaultStringLength(191);
29     }
30 }
31

```

Gambar 1.3 menambahkan code pada AppServiceProvider.php

Gambar 1.3 Artinya adalah kita memberikan panjang nilai default dari tipe data string sepanjang 191 karakter.

Langkah keempat:

Memberikan perintah migrate pada command prompt dari folder project yang kita buat agar table yang terbuat muncul di database. Seperti berikut ini contohnya:

Php artisan migrate

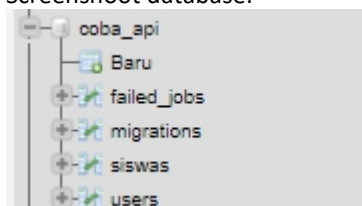
Screenshot:

```

PS D:\4. Kampus\xampp\htdocs\18104013_DDPWebTeori_IrfanMulyanaA\Pertemuan12\Restapi> php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (0.69 seconds)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (0.13 seconds)
Migrating: 2020_12_28_041304_create_siswas_table
Migrated: 2020_12_28_041304_create_siswas_table (0.26 seconds)

```

Screenshot database:



Langkah kelima:

Seharusnya dilangkah ini kita membuat controller dan model namun karna dilangkah ketiga kita menggunakan syntax ini maka controller, model, dan migrate pada controller sudah langsung dibuat. Namun agar lebih memahami berikut ini cara manualnya:

Membuat controller

php artisan make:controller nama_controller

Contoh:

```
php artisan make:controller SiswaController
```

Membuat model

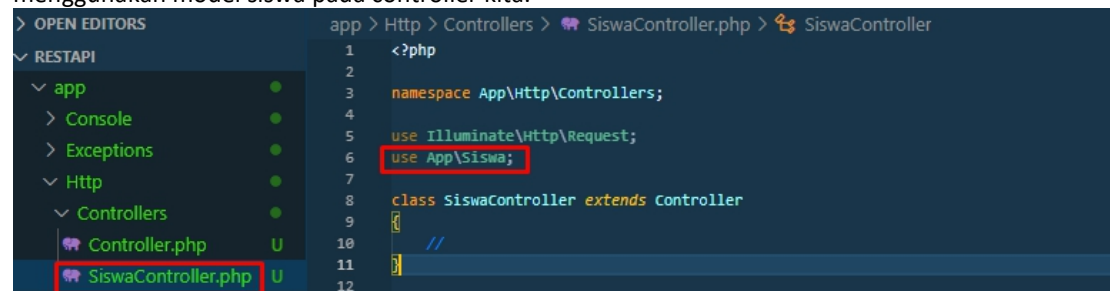
```
php artisan make:model nama_model
```

Contoh:

```
php artisan make:model Siswa
```

Langkah keenam:

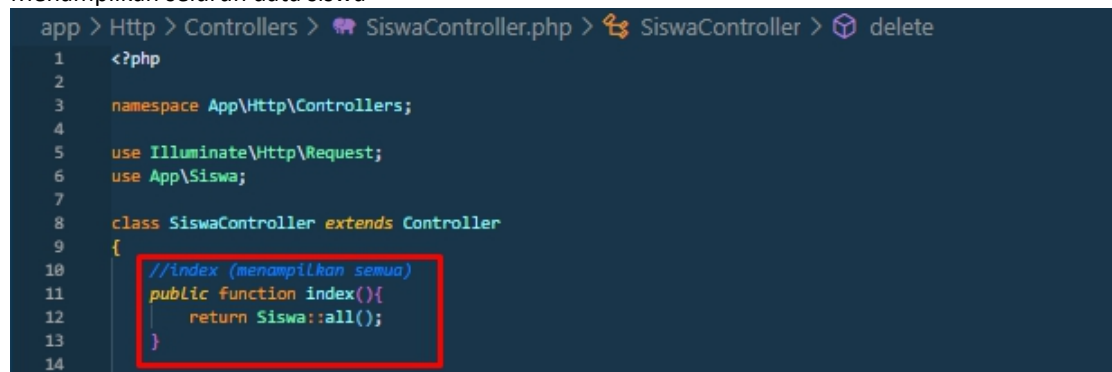
Mengisi controllernya untuk menambahkan app siswa kedalam controllernya karena kita akan menggunakan model siswa pada controller kita.



```
app > Http > Controllers > SiswaController.php > SiswaController
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use App\Siswa;
7
8  class SiswaController extends Controller
9  {
10     //
11 }
12
```

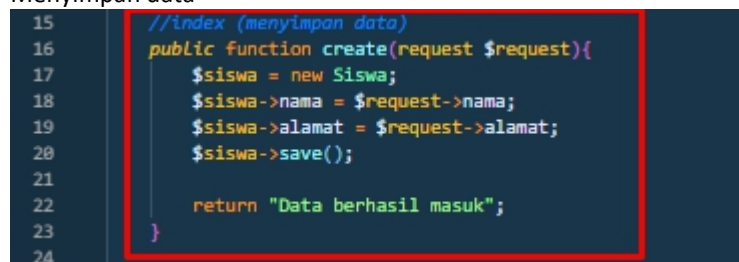
Kemudian tambahkan kode pada index seperti dibawah ini:

Menampilkan seluruh data siswa



```
app > Http > Controllers > SiswaController.php > SiswaController > delete
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use App\Siswa;
7
8  class SiswaController extends Controller
9  {
10     //index (menampilkan semua)
11     public function index(){
12         return Siswa::all();
13     }
14 }
```

Menyimpan data



```
15 //index (menyimpan data)
16 public function create(request $request){
17     $siswa = new Siswa;
18     $siswa->nama = $request->nama;
19     $siswa->alamat = $request->alamat;
20     $siswa->save();
21
22     return "Data berhasil masuk";
23 }
24
```

Mengupdate data

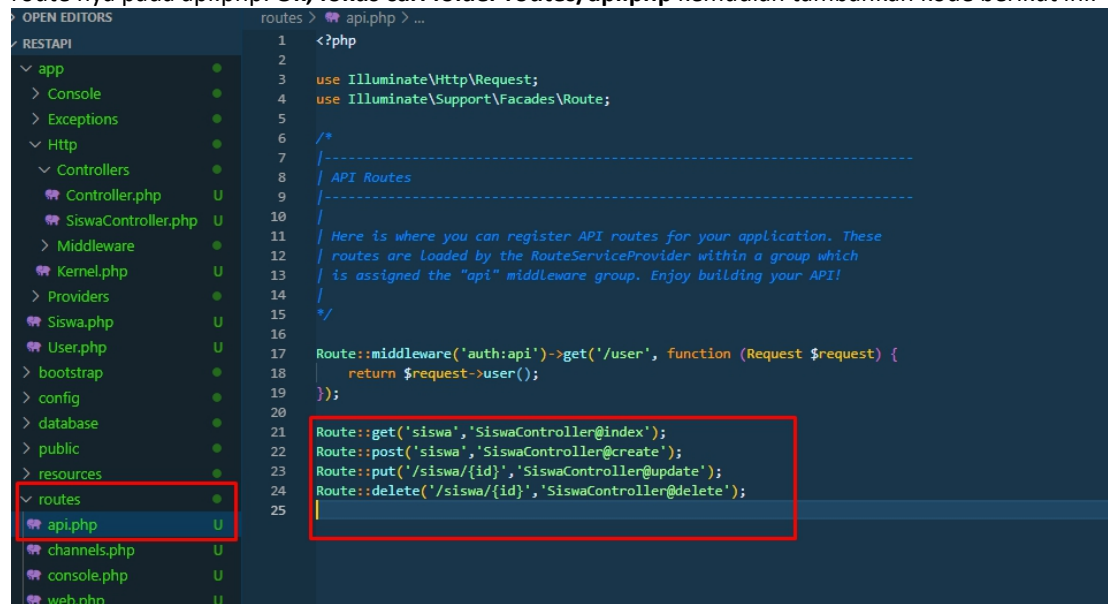
```
25 //index (update semua)
26 public function update(request $request, $id){
27     $nama = $request->nama;
28     $alamat = $request->alamat;
29
30     $siswa = Siswa::find($id);
31     $siswa->nama = $nama;
32     $siswa->alamat = $alamat;
33     $siswa->save();
34
35     return "Data berhasil di update";
36 }
37
```

Menghapus data

```
38 //index (menghapus data)
39 public function delete($id){
40     $siswa = Siswa::find($id);
41     $siswa->delete();
42
43     return "Data berhasil di hapus";
44 }
45
46
```

Langkah ketujuh:

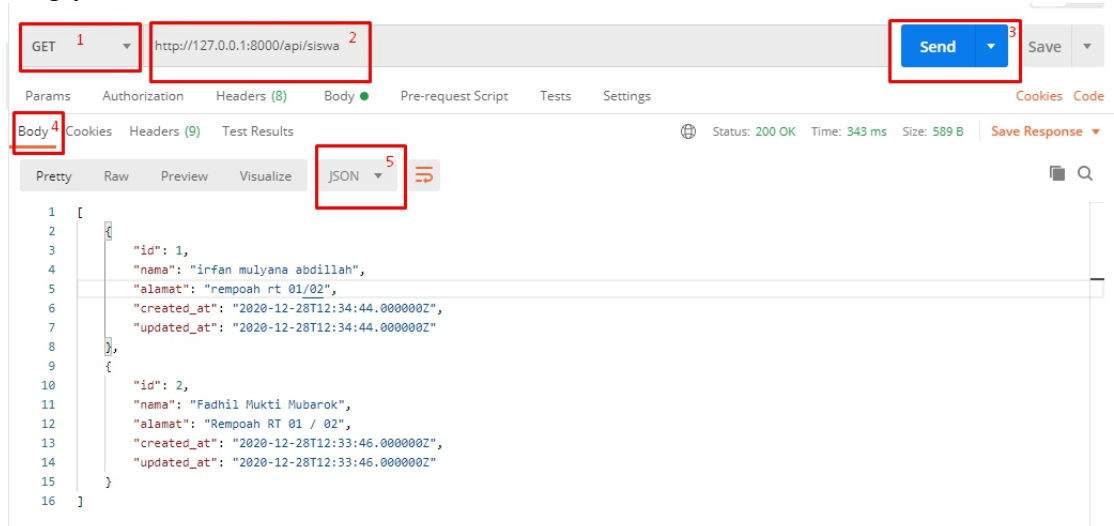
Membuat route API nya, perbedaannya jika kita membuat projek pada umumnya dengan apabila kita membuat API adalah apabila kita biasanya meletakkan route pada web.php disini kita meletakkan route nya pada api.php. **Ok, fokus cari folder routes/api.php** kemudian tambahkan kode berikut ini:



```
1 <?php
2
3 use Illuminate\Http\Request;
4 use Illuminate\Support\Facades\Route;
5
6 /*
7 |-----
8 | API Routes
9 |-----
10 |
11 | Here is where you can register API routes for your application. These
12 | routes are loaded by the RouteServiceProvider within a group which
13 | is assigned the "api" middleware group. Enjoy building your API!
14 |
15 */
16
17 Route::middleware('auth:api')->get('/user', function (Request $request) {
18     return $request->user();
19 });
20
21 Route::get('siswa', 'SiswaController@index');
22 Route::post('siswa', 'SiswaController@create');
23 Route::put('/siswa/{id}', 'SiswaController@update');
24 Route::delete('/siswa/{id}', 'SiswaController@delete');
```

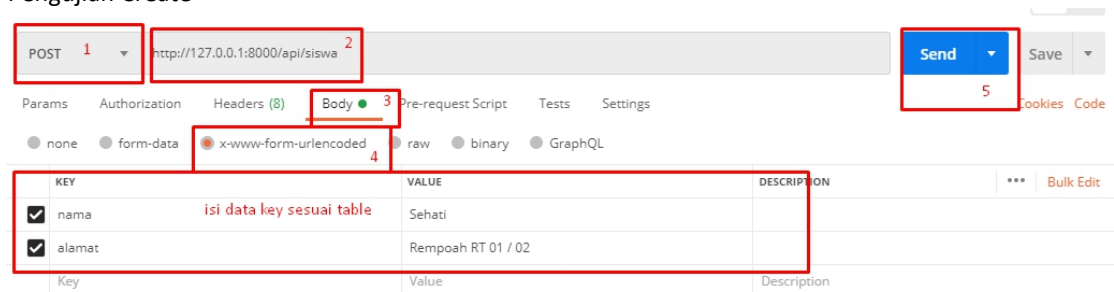
Langkah kedelapan (pengujian) :

1. Pengujian index

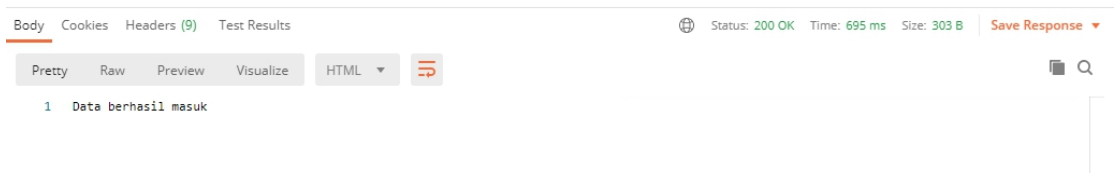


Jika kita ingin menguji index maka kita pilih GET dan sebelahny adalah alamat dari web kita <http://127.0.0.1:8000/api/siswa> karna kita menggunakan api siswa. Nah gambar diatas adalah tampilan untuk index, hasil dari datanya ada dibawahnya yaitu berupa JSON. Index tersebut bisa tampil karena kita sudah melakukan proses penginputan di database table siswas.

2. Pengujian Create



Pertama kita atur method menjadi post dan url. Lalu kita pilih body kemudian pilih `x-www-form-urlencoded` setelah itu isi key sesuai field database yang kita buat tadi di program create. Kemudian isi value jika sudah send untuk menguji data API untuk create berhasil atau tidak. Jika berhasil maka tampilannya akan seperti dibawah ini:



Kemudian kita cek lewat method GET data sudah masuk atau belum jika sudah maka akan tampil seperti dibawah ini:

GET <http://127.0.0.1:8000/api/siswa> Send Save

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies Code

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	***	Bulk Edit
<input checked="" type="checkbox"/> nama	Sehati			
<input checked="" type="checkbox"/> alamat	Rempoah RT 01 / 02			
Key	Value	Description		

Body Cookies Headers (9) Test Results Status: 200 OK Time: 259 ms Size: 731 B Save Response

Pretty Raw Preview Visualize JSON ≡

```
15 },
16 {
17   "id": 3,
18   "nama": "Sehati",
19   "alamat": "Rempoah RT 01 / 02",
20   "created_at": "2020-12-28T06:02:02.000000Z",
21   "updated_at": "2020-12-28T06:02:02.000000Z"
22 }
23 ]
```

3. Pengujian update

Hampir sama seperti sebelumnya namun method kita ubah jadi PUT kemudian url kita tambahkan id yang akan diupdate seperti dibawah ini:

PUT <http://127.0.0.1:8000/api/siswa/1> Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	**
<input checked="" type="checkbox"/> nama	Irfan Mulyana Abdillah		
<input checked="" type="checkbox"/> alamat	Rempoah RT 01 / 03		
Key	Value	Description	

Disini kita rubah alamatnya menjadi Rempoah RT 01 / 03 kemudian kita lakukan send jika data berhasil maka akan muncul notif seperti berikut ini:

Body Cookies Headers (9) Test Results Status: 200 OK Time: 611 ms Size: 307 B Save Res

Pretty Raw Preview Visualize HTML ≡

```
1 Data berhasil di update
```

Kemudian kita cek lewat cara index yang dijelaskan sebelumnya seperti dibawah ini agar kita tahu datanya sudah update atau belum:


```

1  [
2      {
3          "id": 1,
4          "nama": "Irfan Mulyana Abdillah",
5          "alamat": "Rempoah RT 01 / 03",
6          "created_at": "2020-12-28T12:34:44.000000Z",
7          "updated_at": "2020-12-28T06:16:46.000000Z"
8      },

```

4. Pengujian delete

Sebelum menguji ini kita rubah methodnya delete dan menambahkan url dengan id yang akan di delete. Misalnya seperti berikut ini kita akan menghapus id 3:

DELETE

http://127.0.0.1:8000/api/siswa/3

Send

Jika berhasil maka akan muncul notif seperti berikut ini:

Body Cookies Headers (9) Test Results

Pretty Raw Preview Visualize HTML

1 Data berhasil di hapus

Kemudian kita cek lewat method GET seperti cara yang dianjurkan dalam pengujian index

GET

http://127.0.0.1:8000/api/siswa

Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Body Cookies Headers (9) Test Results

Status: 200 OK Time: 253 ms Size: 591 B Sav

Pretty Raw Preview Visualize JSON

```

1  [
2      {
3          "id": 1,
4          "nama": "Irfan Mulyana Abdillah",
5          "alamat": "Rempoah RT 01 / 03",
6          "created_at": "2020-12-28T12:34:44.000000Z",
7          "updated_at": "2020-12-28T06:16:46.000000Z"
8      },
9      {
10         "id": 2,
11         "nama": "Fadhil Mukti Mubarak",
12         "alamat": "Rempoah RT 01 / 02",
13         "created_at": "2020-12-28T12:33:46.000000Z",
14         "updated_at": "2020-12-28T12:33:46.000000Z"
15     }
16 ]

```

Maka id 3 sudah tidak ada di database.