

# **JOBSHEET PRAKTIKUM WEB LANJUTAN**

Membuat RESTful API Laravel  
Mata Kuliah Pemrograman Web Lanjut



Oleh:

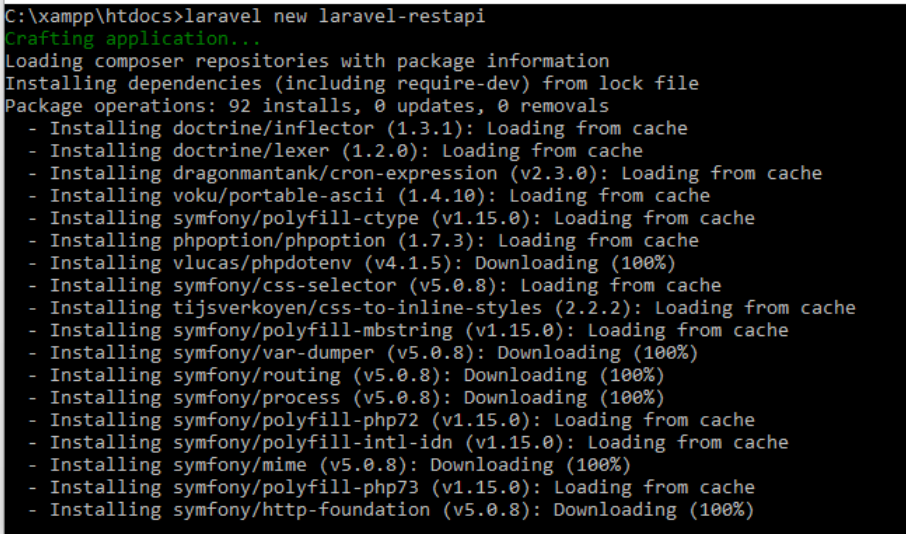
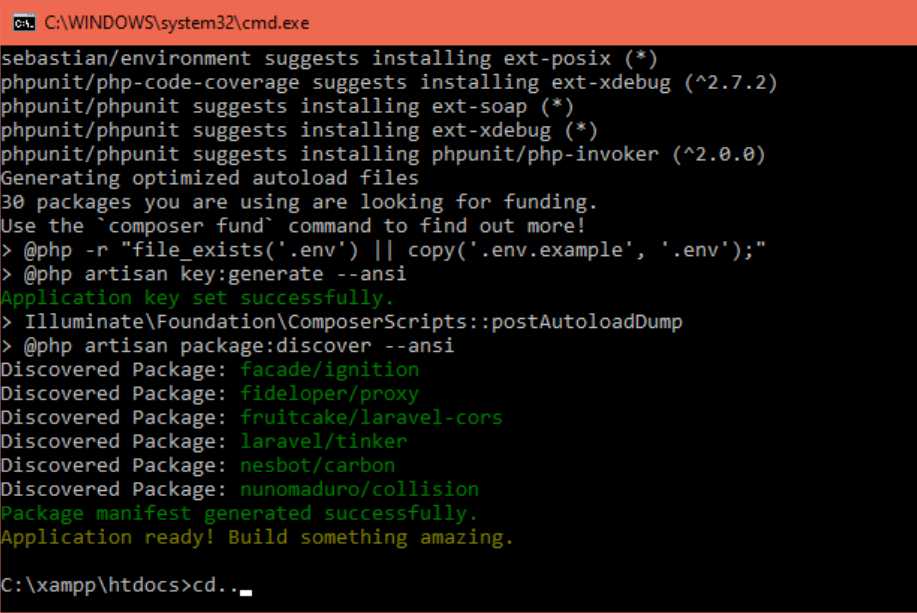
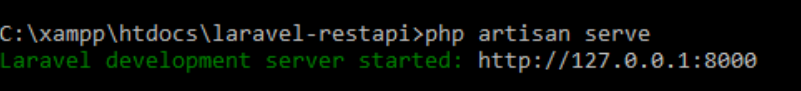
**Adristi Iftitah Yuniar**

**1841720015**

**Jurusan Teknologi Informasi**

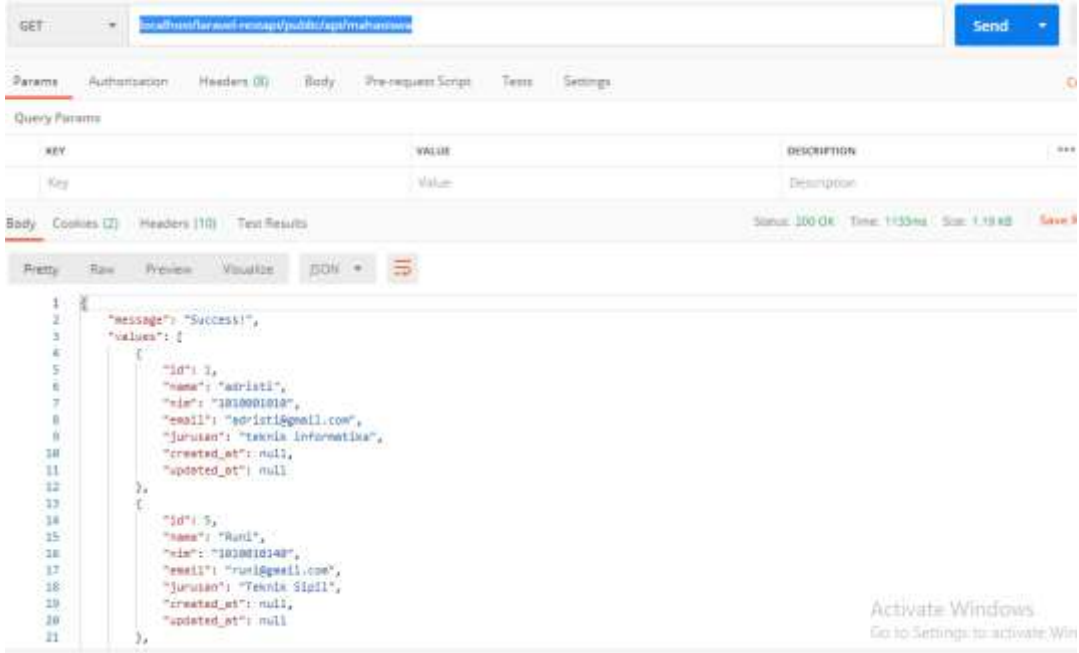
**Politeknik Negeri Malang**

**2020**

Langkah	Keterangan
1.	<p>Buat project baru dengan nama “laravel-restapi”. Buka command prompt, tuliskan perintah berikut.</p> <pre>cd C:\xampp\htdocs laravel new laravel-restapi</pre>  
2.	<p>Kita coba jalankan dulu project tersebut. Pada command prompt tulis perintah berikut.</p> <pre>cd C:\laravel-restapi php artisan serve</pre> <p>Akan tampil halaman default Laravel seperti di bawah ini.</p> 

	
<p><b>3.</b></p>	<p>Kemudian lakukan konfigurasi database pada file .env. Isikan nama database, username, dan password yang akan digunakan. Pada project ini, kita gunakan database dari latihan di minggu-minggu sebelumnya yaitu “latihan_laravel”</p> <pre> 8 9 DB_CONNECTION=mysql 0 DB_HOST=127.0.0.1 1 DB_PORT=3306 2 DB_DATABASE=latihan_laravel 3 DB_USERNAME=root 4 DB_PASSWORD= </pre>
<p><b>4.</b></p>	<p>Buat model dengan nama Mahasiswa, buat juga controllernya. Untuk membuat model dan controllernya sekaligus tuliskan perintah berikut pada command prompt (terlebih dahulu keluar dari php artisan serve dengan mengetik ctrl+C pada keyboard)</p> <p>php artisan make:model Mahasiswa -c</p> <p>Keterangan :</p> <p>-c merupakan perintah untuk menyertakan pembuatan controller</p> <p>Sehingga pada project laravel-restapi akan bertambah dua file yaitu model Mahasiswa.php serta controller MahasiswaController.php.</p> <pre> C:\xampp\htdocs\laravel-restapi&gt;php artisan make:model Mahasiswa -c Model created successfully. Controller created successfully. </pre> 
<p><b>5.</b></p>	<p>Selanjutnya ubah isi model Mahasiswa.php seperti berikut ini.</p>

	 <pre> 1 &lt;?php 2 3 namespace App; 4 5 use Illuminate\Database\Eloquent\Model; 6 7 class Mahasiswa extends Model 8 { 9     protected \$table = 'mahasiswa'; 10 } 11 </pre>
<p>6.</p>	<p>Kemudian kita akan memodifikasi isi dari MahasiswaController.php untuk dapat mengolah data pada tabel 'mahasiswa'. Pada controller ini, kita akan melakukan operasi untuk menampilkan, menambah, mengubah, dan menghapus data.</p> <p>Pertama, kita akan mengubah fungsi index agar saat fungsi index dipanggil, maka aplikasi akan menampilkan seluruh data dari tabel mahasiswa.</p>  <pre> 1 &lt;?php 2 3 namespace App\Http\Controllers; 4 5 use Illuminate\Http\Request; 6 use App\Mahasiswa; 7 8 class MahasiswaController extends Controller 9 { 10     public function index() 11     { 12         \$data = Mahasiswa::all(); 13 14         //cek data tidak kosong 15         if(count(\$data) &gt; 0){ 16             \$res['message'] = "Success!"; 17             \$res['values'] = \$data; 18             return response(\$res); 19         } 20         //jika data kosong 21         else{ 22             \$res['message'] = "Kosong!"; 23             return response(\$res); 24         } 25     } 26 27 } 28 29 </pre>
<p>7.</p>	<p>Tambahkan route untuk memanggil fungsi index pada file routes/api.php (Line 21).</p>  <pre> 17 Route::middleware('auth:api')-&gt;get('/user', function ( 18     Request \$request) { 19     return \$request-&gt;user(); 20 }); 21 Route::get('mahasiswa', 'MahasiswaController@index'); </pre>
<p>8.</p>	<p>Ketikkan perintah php artisan serve pada command prompt. Lalu kita coba menguji fungsi untuk menampilkan data menggunakan aplikasi Postman.</p> <p>Gunakan perintah GET, isikan url : <a href="http://localhost:8000/api/mahasiswa">http://localhost:8000/api/mahasiswa</a></p> <p>Berikut adalah tampilan dari aplikasi Postman.</p>

	 <p>GET: <code>localhost:3000/api/mahasiswa</code> <span>Send</span></p> <p>Params: Authorization Headers (0) Body Pre-request Script Tests Settings</p> <p>Query Params</p> <table><thead><tr><th>KEY</th><th>VALUE</th><th>DESCRIPTION</th></tr></thead><tbody><tr><td>Key</td><td>Value</td><td>Description</td></tr></tbody></table> <p>Body Cookies (2) Headers (10) Test Results <span>Status: 200 OK Time: 1155ms Size: 1.19 KB Save</span></p> <p>Pretty Raw Preview Viewable JSON <span>5</span></p> <pre>1 { 2   "message": "Success!", 3   "values": [ 4     { 5       "id": 2, 6       "nama": "Adristi", 7       "nim": "1810001018", 8       "email": "adristi@gmail.com", 9       "jurusan": "Teknik Informatika", 10      "created_at": null, 11      "updated_at": null 12    }, 13    { 14      "id": 5, 15      "nama": "Rudi", 16      "nim": "18100010148", 17      "email": "rudi@gmail.com", 18      "jurusan": "Teknik Sipil", 19      "created_at": null, 20      "updated_at": null 21    } 22  ] 23 }</pre> <p>Activate Windows Go to Settings to activate Win</p>	KEY	VALUE	DESCRIPTION	Key	Value	Description
KEY	VALUE	DESCRIPTION					
Key	Value	Description					
9.	<p>Selanjutnya kita akan menambahkan fungsi untuk melihat suatu data ketika dipilih ID tertentu. Buat fungsi baru yaitu <code>getId</code> pada <code>MahasiswaController.php</code>.</p> <pre>27 //fungsi untuk menampilkan data dari sebuah ID 28 public function getId(\$id) 29 { 30     \$data = Mahasiswa::where('id',\$id)-&gt;get(); 31 32     //cek jika data ditemukan 33     if (count(\$data) &gt; 0) { 34         \$res['message'] = "Success!"; 35         \$res['values'] = \$data; 36         return response(\$res); 37     } 38     //jika data kosong 39     else{ 40         \$res['message'] = "Gagal!"; 41         return response(\$res); 42     } 43 }</pre>						
10.	<p>Tambahkan route untuk memanggil fungsi <code>getId</code> pada <code>routes/api.php</code></p> <pre>21 Route::get('mahasiswa', 'MahasiswaController@index'); 22 Route::get('/mahasiswa/{id}', 'MahasiswaController@getId');</pre>						
11.	<p>Sekarang kita coba untuk menampilkan data berdasarkan ID mahasiswa yang dipilih menggunakan Postman. Gunakan perintah GET untuk menampilkan data. Di bawah ini adalah contoh untuk menampilkan data dengan ID=2, maka url diisi :</p>						

http://localhost:8000/api/mahasiswa/2

GET localhost/laravel-restapi/public/api/mahasiswa/5

Params Authorization Headers (8) Body Pre-request Script

Query Params

KEY	VALUE
Key	Value

Body Cookies (2) Headers (10) Test Results

Pretty Raw Preview Visualize JSON ↵

```
1 {
2   "message": "Success!",
3   "values": [
4     {
5       "id": 5,
6       "nama": "Runi",
7       "nim": "1010010140",
8       "email": "runi@gmail.com",
9       "jurusan": "Teknik Sipil",
10      "created_at": null,
11      "updated_at": null
12    }
13  ]
14 }
```

Lalu bila di cari berdasarkan id = 15 , akan gagal karena tidak ada data tersebut

GET localhost/laravel-restapi/public/api/mahasiswa/15

Params Authorization Headers (8) Body Pre-request Script

Query Params

KEY	VALUE
Key	Value

Body Cookies (2) Headers (10) Test Results

Pretty Raw Preview Visualize JSON ↵

```
1 {
2   "message": "Gagal!"
3 }
```

- 12.** Setelah dapat menampilkan data, kita akan membuat fungsi untuk menambahkan data baru ke database dengan nama create pada MahasiswaController.php.

```

45 //fungsi tambah data
46 public function create(Request $request)
47 {
48     $mhs = new Mahasiswa();
49     $mhs->nama = $request->nama;
50     $mhs->nim = $request->nim;
51     $mhs->email = $request->email;
52     $mhs->jurusan = $request->jurusan;
53
54     //jika data berhasil tersimpan
55     if ($mhs->save()) {
56         $res['message'] = "Data berhasil ditambah!";
57         $res['values'] = $mhs;
58         return response($res);
59     }
60 }

```

13. Tambahkan route untuk memanggil fungsi create pada routes/api.php

```

20
21 Route::get('mahasiswa', 'MahasiswaController@index');
22 Route::get('/mahasiswa/{id}', 'MahasiswaController@getId');
23 Route::post('/mahasiswa', 'MahasiswaController@create');

```

14. Kita coba untuk menambahkan data melalui Postman.

POST localhost/laravel-restapi/public/api/mahasiswa

Params Authorization Headers (10) Body Pre-request Script Tests Settings

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

	KEY	VALUE	DI
<input checked="" type="checkbox"/>	nama	Eka	
<input checked="" type="checkbox"/>	nim	1001001045	
<input checked="" type="checkbox"/>	email	Eka@gmail.com	
<input checked="" type="checkbox"/>	jurusan	Akutansi	
	Key	Value	D

Body Cookies (2) Headers (10) Test Results Statu

Pretty Raw Preview Visualize JSON

```

1 {
2   "message": "Data berhasil ditambah!",
3   "values": {
4     "nama": "Eka",
5     "nim": "1001001045",
6     "email": "Eka@gmail.com",
7     "jurusan": "Akutansi",
8     "updated_at": "2020-05-03T21:37:16.000000Z",
9     "created_at": "2020-05-03T21:37:16.000000Z",
10    "id": 13
11  }
12 }

```

Kemudian coba untuk tampilkan semua data, kita lihat apakah data baru sudah masuk ke database.

	<div data-bbox="379 197 1321 1160"> <div>GET localhost/laravel-restapi/public/api/mahasiswa</div> <div> <div>Pretty Raw Preview Visualize JSON</div> <pre> 35         "email": "ranmat@gmail.com", 36         "jurusan": "Teknik Listrik", 37         "created_at": "2020-04-13T05:29:45.000000Z", 38         "updated_at": "2020-04-13T06:10:28.000000Z" 39     }, 40     { 41         "id": 9, 42         "nama": "Kevin", 43         "nim": "1001001060", 44         "email": "kevin@gmail.com", 45         "jurusan": "Teknik Elektro", 46         "created_at": null, 47         "updated_at": null 48     }, 49     { 50         "id": 12, 51         "nama": "claudyo", 52         "nim": "100100165", 53         "email": "claudyo@gmail.com", 54         "jurusan": "Teknik Informatika", 55         "created_at": null, 56         "updated_at": null 57     }, 58     { 59         "id": 13, 60         "nama": "Eka", 61         "nim": "1001001045", 62         "email": "Eka@gmail.com", 63         "jurusan": "Akutansi", 64         "created_at": "2020-05-03T21:37:16.000000Z", 65         "updated_at": "2020-05-03T21:37:16.000000Z" 66     } 67 ] 68 </pre> </div> </div>
15.	<p>Selanjutnya kita akan menambahkan fungsi untuk mengubah data dari database. Buat fungsi update pada MahasiswaController.php.</p> <div data-bbox="379 1272 1177 1921"> <pre> 62 //fungsi untuk mengubah data 63 public function update(Request \$request, \$id) 64 { 65     \$nama = \$request-&gt;nama; 66     \$nim = \$request-&gt;nim; 67     \$email = \$request-&gt;email; 68     \$jurusan = \$request-&gt;jurusan; 69 70     \$mhs = Mahasiswa::find(\$id); 71     \$mhs-&gt;nama = \$nama; 72     \$mhs-&gt;nim = \$nim; 73     \$mhs-&gt;email = \$email; 74     \$mhs-&gt;jurusan = \$jurusan; 75 76     if (\$mhs-&gt;save()) { 77         \$res['message'] = "Data berhasil diubah!"; 78         \$res['values'] = \$mhs; 79         return response(\$res); 80     }else{ 81         \$res['message'] = "Gagal!"; 82         return response(\$res); 83     } 84 } </pre> </div>
16.	<p>Tambahkan route untuk memanggil fungsi update pada routes/api.php</p>



```
21 Route::get('mahasiswa', 'MahasiswaController@index');
22 Route::get('/mahasiswa/{id}', 'MahasiswaController@getId');
23 Route::post('/mahasiswa', 'MahasiswaController@create');
24 Route::put('/mahasiswa/update/{id}', 'MahasiswaController@update');
```

17.

Sekarang kita coba untuk mengubah data berdasarkan ID mahasiswa yang dipilih menggunakan Postman. Gunakan perintah PUT untuk mengubah data. Berikut adalah contoh untuk mengubah data dengan ID=2, maka url diisi : <http://localhost:8000/api/mahasiswa/update/2>. Pilih tab Body dan pilih radio button xwww-form-urlencoded. Isikan nama kolom pada database pada KEY, untuk isian data yang diubah tuliskan pada VALUE.

Sebelum

```
{
  "values": [
    {
      "id": 5,
      "nama": "Runi",
      "nim": "1010010140",
      "email": "runi@gmail.com",
      "jurusan": "Teknik Sipil",
      "created_at": null,
      "updated_at": null
    }
  ]
}
```

Sesudah

PUT localhost/laravel-restapi/public/api/mahasiswa/update/5

Params Authorization Headers (10) **Body** Pre-request Script Tests

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

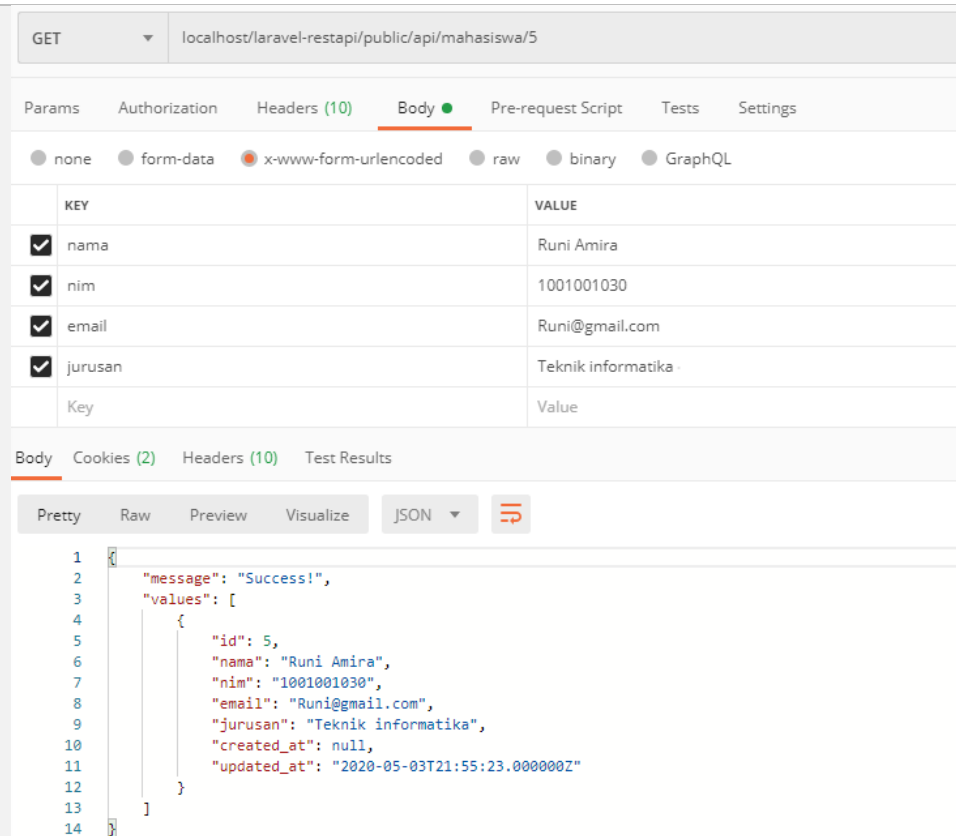
	KEY	VALUE
<input checked="" type="checkbox"/>	nama	Runi Amira
<input checked="" type="checkbox"/>	nim	1001001030
<input checked="" type="checkbox"/>	email	Runi@gmail.com
<input checked="" type="checkbox"/>	jurusan	Teknik informatika
	Key	Value

Body Cookies (2) Headers (10) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Data berhasil diubah!",
3   "values": {
4     "id": 5,
5     "nama": "Runi Amira",
6     "nim": "1001001030",
7     "email": "Runi@gmail.com",
8     "jurusan": "Teknik informatika",
9     "created_at": null,
10    "updated_at": "2020-05-03T21:55:23.000000Z"
11  }
12 }
```

Kemudian coba untuk menampilkan data dengan ID=2 untuk melihat apakah data sudah ter-update.



18. Terakhir kita akan membuat fungsi untuk menghapus data dengan nama delete di MahasiswaController.php.

```

86 //fungsi untuk menghapus data
87 public function delete($id)
88 {
89     $mhs = Mahasiswa::where('id',$id);
90
91     if ($mhs->delete()) {
92         $res['message'] = "Data berhasil dihapus!";
93         return response($res);
94     }else{
95         $res['message'] = "Gagal!";
96         return response($res);
97     }
98 }

```

19. Tambahkan route untuk memanggil fungsi delete pada routes/api.php

```

17 Route::middleware('auth:api')->get('/user', function (Request $
18     request) {
19     return $request->user();
20 });
21
22 Route::get('mahasiswa', 'MahasiswaController@index');
23 Route::get('/mahasiswa/{id}', 'MahasiswaController@getId');
24 Route::post('/mahasiswa', 'MahasiswaController@create');
25 Route::put('/mahasiswa/update/{id}', 'MahasiswaController@update');
26 Route::delete('/mahasiswa/{id}', 'MahasiswaController@delete');

```

20. Buka Postman untuk mencoba menghapus data dari database. Gunakan perintah DELETE untuk mengubah data.  
Berikut adalah contoh untuk menghapus data dengan ID=10, maka url diisi :

<http://localhost:8000/api/mahasiswa /10>

DELETE

localhost/laravel-restapi/public/api/mahasiswa/6

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Headers 

8 hidden

KEY	VALUE
Key	Value

Body

Cookies (2)

Headers (10)

Test Results

Pretty

Raw

Preview

Visualize

JSON

1

2

3

{

"message": "Data berhasil dihapus!"

}