

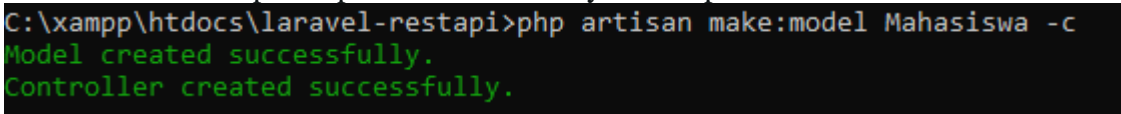
**PEMROGRAMAN WEB LANJUT**  
**JOBSHEET 13**

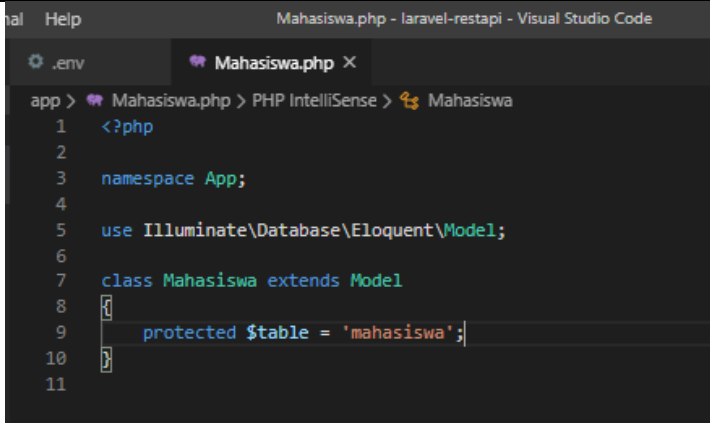
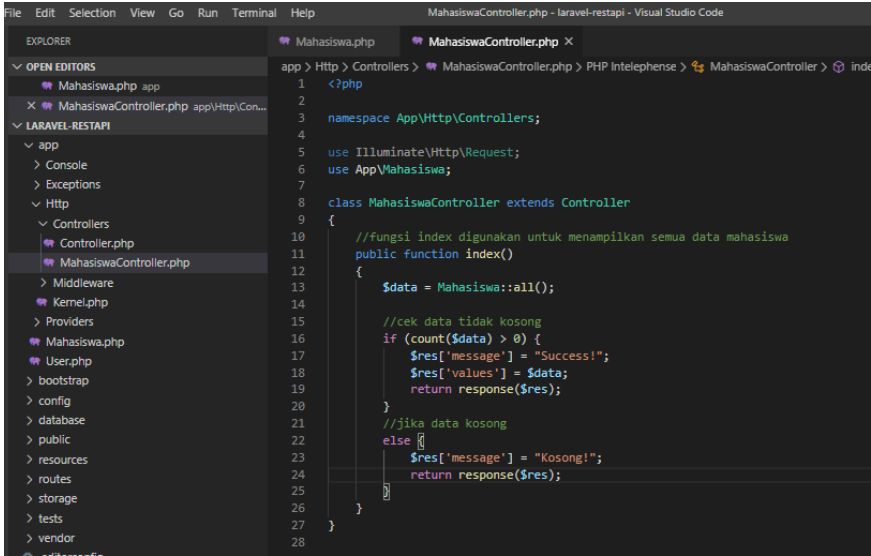


**Oleh:**  
**SUGENG PRASTIYO**  
**NIM. 1841720113**

**PROGRAM STUDI TEKNIK INFORMATIKA**  
**JURUSAN TEKNOLOGI INFORMASI**  
**POLITEKNIK NEGERI MALANG**  
**2020**

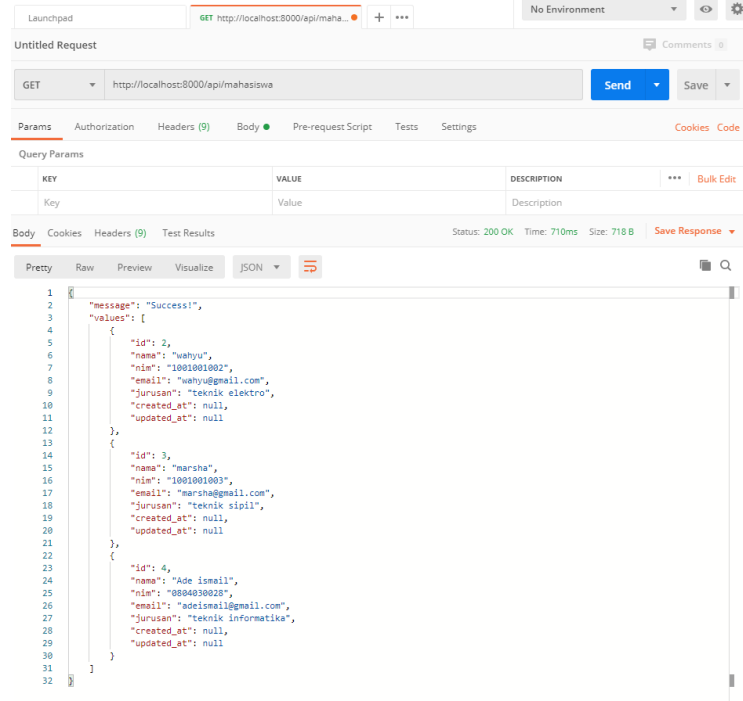
No	<p style="text-align: center;"><b>Uraian Jawaban</b> <b>Membuat RESTfull API di Laravel</b></p>
1	<p>Buat project baru dengan nama “laravel-restapi”. Buka command prompt, tuliskan perintah berikut. cd C:\xampp\htdocs laravel new laravel-restapi</p> <pre> C:\xampp\htdocs&gt;laravel new laravel-restapi Crafting application... Loading composer repositories with package information Installing dependencies (including require-dev) from lock file Package operations: 92 installs, 0 updates, 0 removals   - Installing doctrine/inflector (1.3.1): Loading from cache   - Installing doctrine/lexer (1.2.0): Loading from cache   - Installing dragonmantank/cron-expression (v2.3.0): Loading from cache   - Installing voku/portable-ascii (1.4.10): Loading from cache   - Installing symfony/polyfill-ctype (v1.15.0): Loading from cache   - Installing phpoption/phpoption (1.7.3): Loading from cache   - Installing vlucas/phpdotenv (v4.1.5): Downloading (100%)   - Installing symfony/css-selector (v5.0.8): Loading from cache   - Installing tijsverkoyen/css-to-inline-styles (2.2.2): Loading from cache   - Installing symfony/polyfill-mbstring (v1.15.0): Loading from cache   - Installing symfony/var-dumper (v5.0.8): Downloading (100%)   - Installing symfony/routing (v5.0.8): Downloading (100%)   - Installing symfony/process (v5.0.8): Downloading (100%)   - Installing symfony/polyfill-php72 (v1.15.0): Loading from cache   - Installing symfony/polyfill-intl-idn (v1.15.0): Loading from cache   - Installing symfony/mime (v5.0.8): Downloading (100%)   - Installing symfony/polyfill-php73 (v1.15.0): Loading from cache   - Installing symfony/http-foundation (v5.0.8): Downloading (100%)   - Installing psr/event-dispatcher (1.0.0): Loading from cache   - Installing symfony/event-dispatcher-contracts (v2.0.1): Loading from cache   - Installing symfony/event-dispatcher (v5.0.8): Loading from cache   - Installing psr/log (1.1.3): Loading from cache   - Installing symfony/error-handler (v5.0.8): Loading from cache </pre>
2	<p>Kita coba jalankan dulu project tersebut. Pada command prompt tulis perintah berikut. cd C:\laravel-restapi php artisan serve Akan tampil halaman default Laravel seperti di bawah ini.</p> 
3	<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> 1 APP_NAME=Laravel 2 APP_ENV=local 3 APP_KEY=base64:JxJD8Lyau3MH50VTou6vwPPmVUYj+ceT18xMODfKFYI= 4 APP_DEBUG=true 5 APP_URL=http://localhost 6 7 LOG_CHANNEL=stack 8 9 DB_CONNECTION=mysql 10 DB_HOST=localhost 11 DB_PORT=3306 12 DB_DATABASE=latihan_laravel 13 DB_USERNAME=root 14 DB_PASSWORD= 15 </pre> <p>Db Host di sesuaikan penggunaan localhost</p>
4	<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> <ul style="list-style-type: none"> <li>-c merupakan perintah untuk menyertakan pembuatan controller</li> </ul>  <pre> C:\xampp\htdocs\laravel-restapi&gt;php artisan make:model Mahasiswa -c Model created successfully. Controller created successfully. </pre> <p>Sehingga pada project laravel-restapi akan bertambah dua file yaitu model Mahasiswa.php serta controller MahasiswaController.php</p> 
5	<p>Selanjutnya ubah isi model Mahasiswa.php seperti berikut ini.</p>

	 <p>Keterangan:</p> <ul style="list-style-type: none"> <li>Model ini akan mengelola tabel “mahasiswa” yang terdapat pada database latihan_laravel</li> </ul>
6	<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. Pertama, kita akan mengubah fungsi index agar saat fungsi index dipanggil, maka aplikasi akan menampilkan seluruh data dari tabel mahasiswa.</p>  <p>Keterangan:</p> <ul style="list-style-type: none"> <li>Tambahkan line 6 agar model Mahasiswa dapat digunakan pada MahasiswaController <ul style="list-style-type: none"> <li>Line 15-19 digunakan untuk memeriksa apakah data &gt; 0 atau data tidak kosong</li> <li>Variabel \$res['message'] digunakan untuk menampilkan pesan apakah ada data atau tidak ada data di tabel mahasiswa</li> <li>Variabel \$array['values'] akan menyimpan semua baris data pada tabel mahasiswa</li> </ul> </li> </ul>
7	<p>Tambahkan route untuk memanggil fungsi index pada file routes/api.php (Line 21).</p>

```
api.php - laravel-restapi - Visual Studio Code
Mahasiswa.php MahasiswaController.php api.php x
routes > api.php > ...
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('mahasiswa', 'MahasiswaController@index');
```

8 Ketikkan perintah php artisan serve pada command prompt. Lalu kita coba menguji fungsi untuk menampilkan data menggunakan aplikasi Postman. Gunakan perintah GET, isikan url : <http://localhost:8000/api/mahasiswa> Berikut adalah tampilan dari aplikasi Postman.



Semua data pada tabel mahasiswa akan tampil, ditampilkan juga pesan sukses.

9 Selanjutnya kita akan menambahkan fungsi untuk melihat suatu data ketika dipilih ID tertentu. Buat fungsi baru yaitu getId pada MahasiswaController.php.

```

28 public function getId($id)
29 {
30     $data = Mahasiswa::where('id', $id)->get();
31
32     //cek jika data ditemukan
33     if (count($data) > 0) {
34         $res['message'] = "Success!";
35         $res['values'] = $data;
36         return response($res);
37     }
38     //jika data tidak ditemukan
39     else {
40         $res['message'] = "Gagal!";
41         return response($res);
42     }
43 }

```

Keterangan:

- Fungsi getId menerima parameter \$id yang menunjukkan ID mahasiswa yang dipilih
- Line 30 merupakan pemanggilan model untuk membaca data berdasarkan ID

10 Tambahkan route untuk memanggil fungsi getId pada routes/api.php

```

22
23 Route::get('/mahasiswa/{id}', 'MahasiswaController@getId');
24

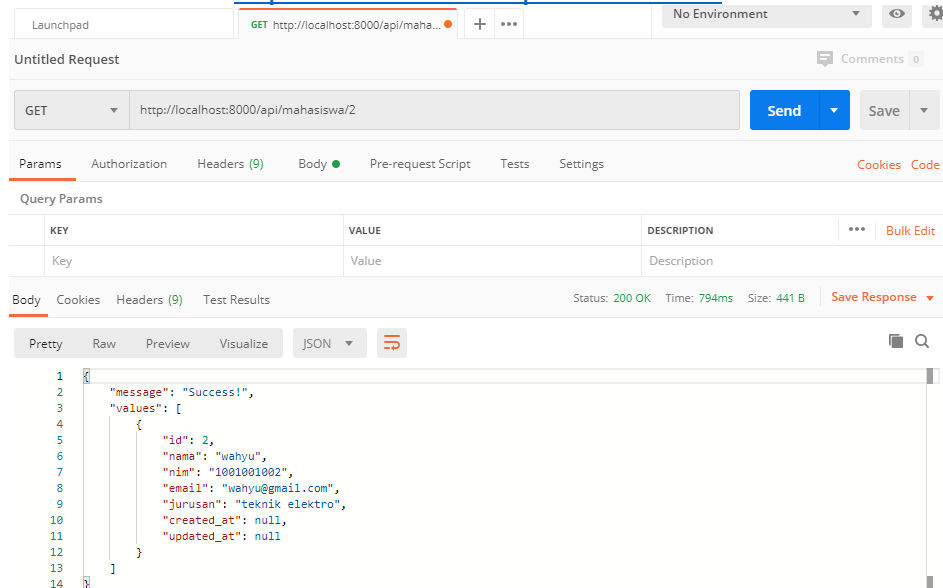
```

Karena kita ingin menampilkan data, maka perintah yang dipakai adalah 'get'

11 Sekarang kita coba untuk menampilkan data berdasarkan ID mahasiswa yang dipilih menggunakan Postman. Gunakan perintah GET untuk menampilkan data.

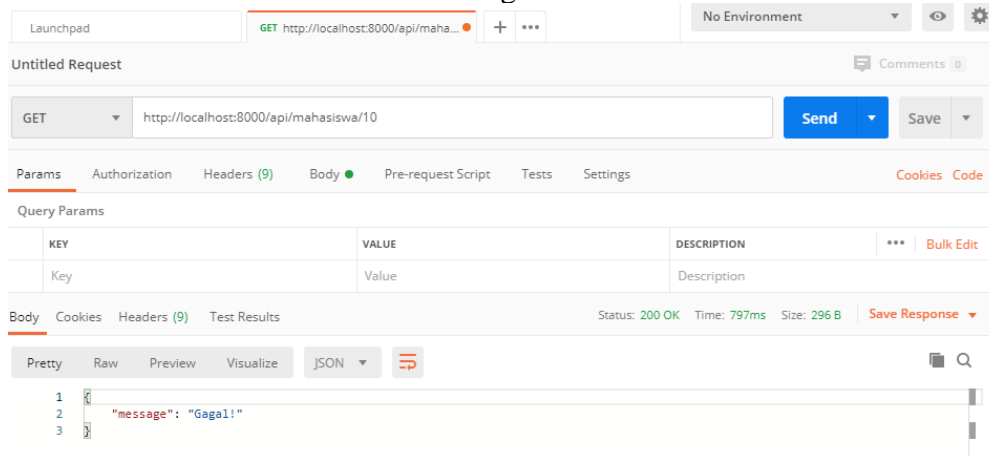
Di bawah ini adalah contoh untuk menampilkan data dengan ID=2, maka url diisi :

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

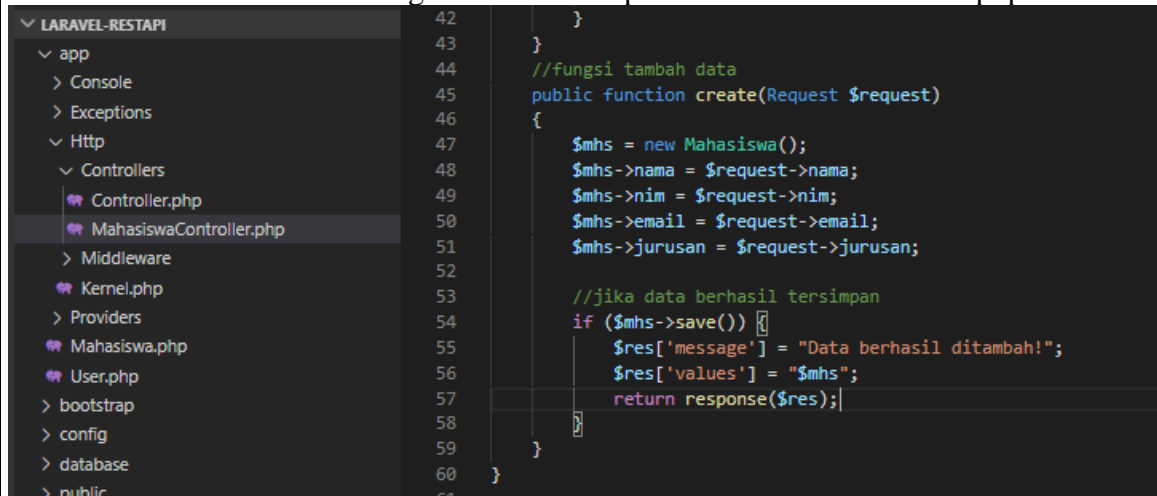


Ini adalah hasil dari code program diatas yang menyatakan mencari mahasiswa dengan parameter idnya

Ketika mencoba menampilkan ID=10 akan muncul pesan “Gagal”, karena tidak ada data mahasiswa dengan ID tersebut.



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



Keterangan:

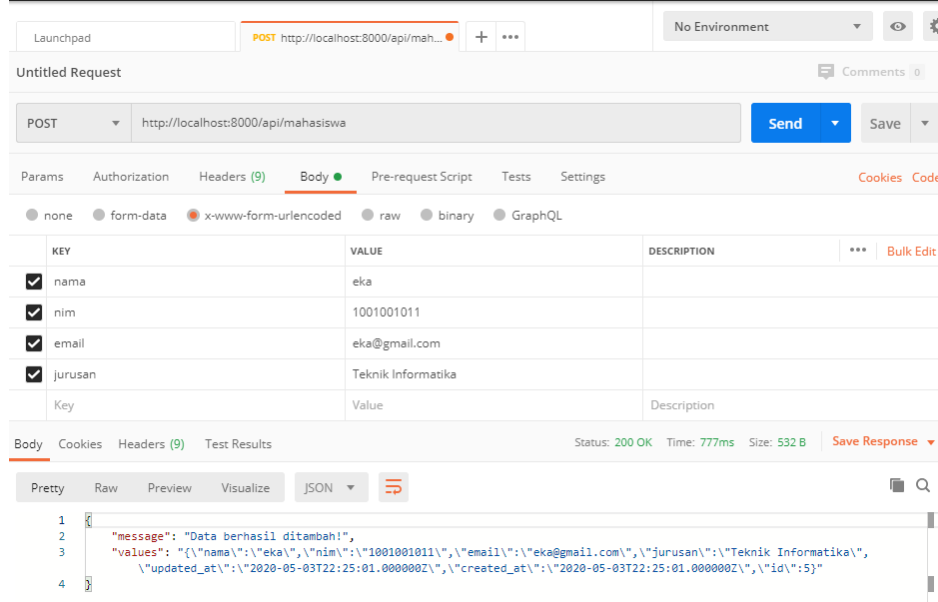
- Fungsi create menerima parameter Request yang menampung isian data mahasiswa yang akan ditambahkan ke database.
- Line 55-59 : \$mhs->save() digunakan untuk menyimpan data ke database, apabila save() berhasil dijalankan maka akan ditampilkan pesan berhasil serta data yang ditambah.

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

```
24  
25 Route::post('/mahasiswa', 'MahasiswaController@create');  
26
```

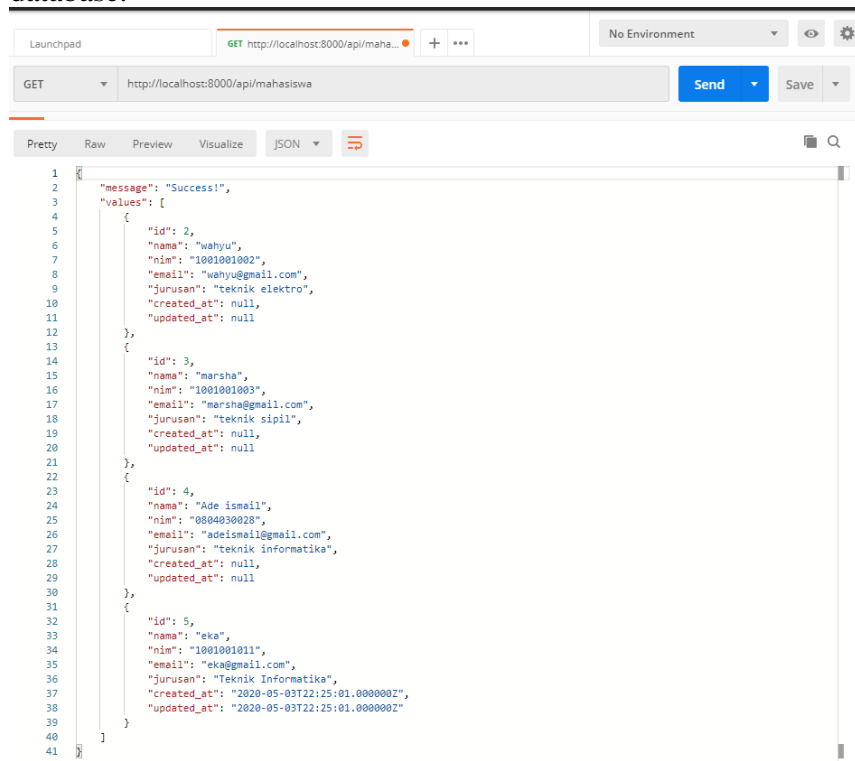
Karena kita ingin menambah data, maka perintah yang dipakai adalah 'post'.

14 Kita coba untuk menambahkan data melalui Postman.



- Isikan url : http://localhost:8000/api/mahasiswa. Karena kita ingin mengirim data ke database, maka perintah yang dipakai adalah 'POST'.
- Pilih tab Body dan pilih radio button x-www-form-urlencoded. Isikan nama kolom pada database pada KEY, untuk isian datanya tuliskan pada VALUE.

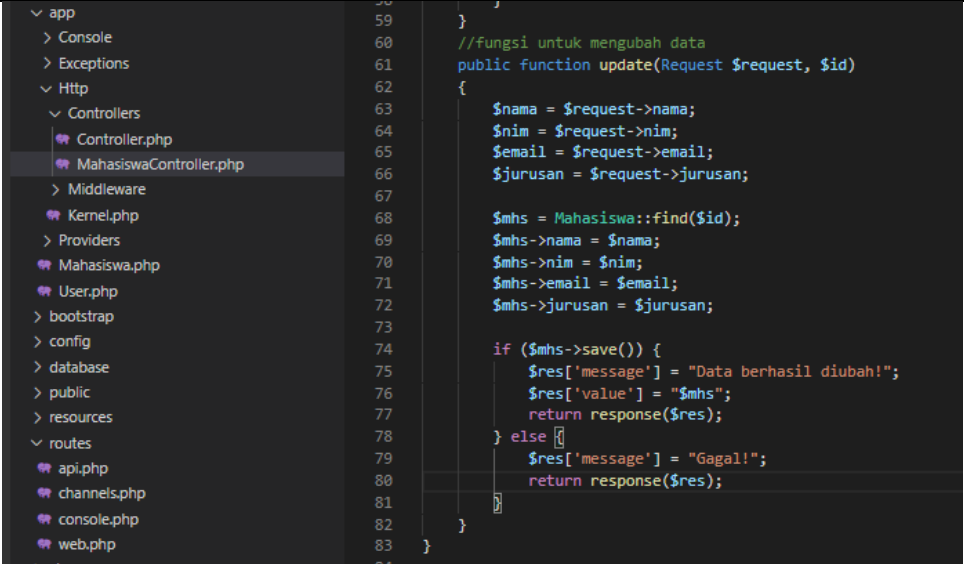
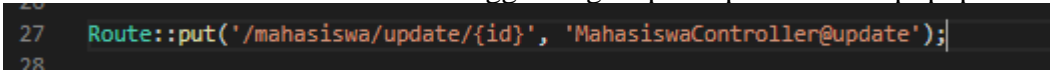
Kemudian coba untuk tampilkan semua data, kita lihat apakah data baru sudah masuk ke database.

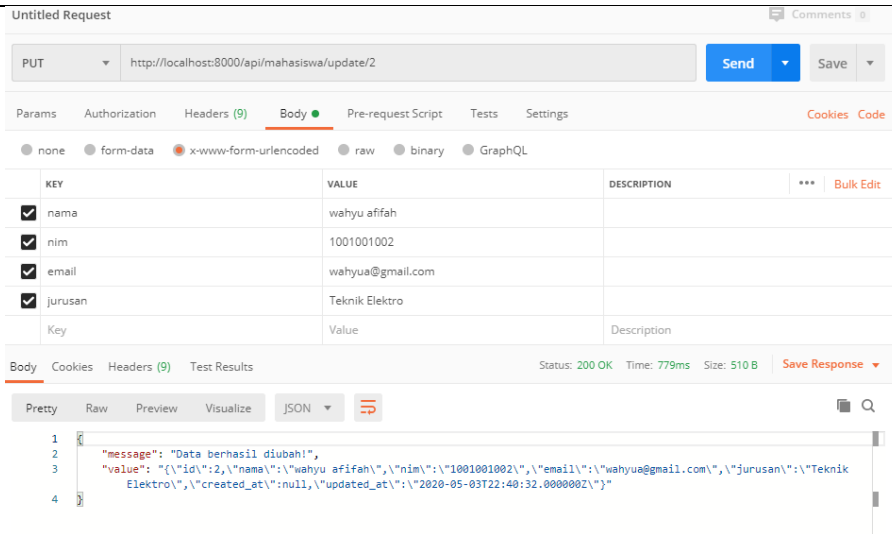


15

Selanjutnya kita akan menambahkan fungsi untuk mengubah data dari database. Buat fungsi update pada MahasiswaController.php.

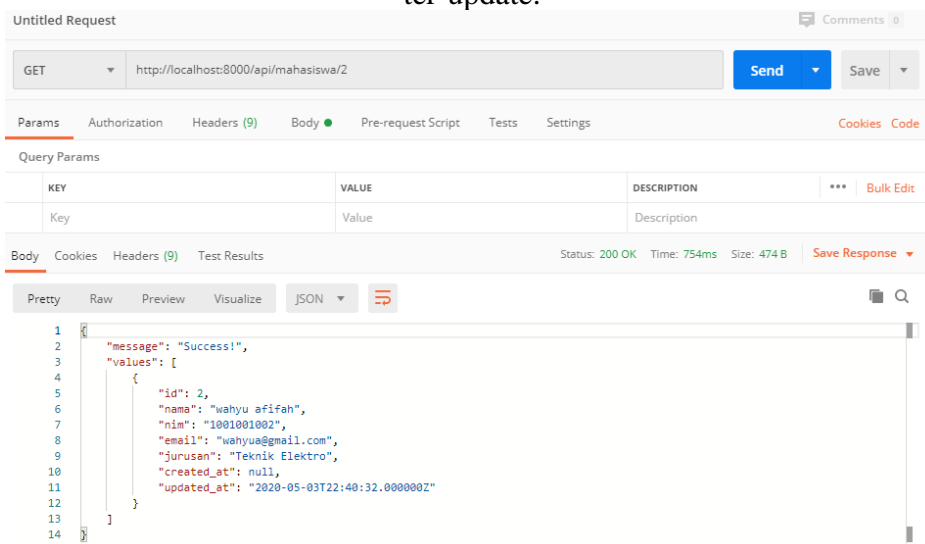


	 <pre> 59 } 60 //fungsi untuk mengubah data 61 public function update(Request \$request, \$id) 62 { 63     \$nama = \$request-&gt;nama; 64     \$nim = \$request-&gt;nim; 65     \$email = \$request-&gt;email; 66     \$jurusan = \$request-&gt;jurusan; 67 68     \$mhs = Mahasiswa::find(\$id); 69     \$mhs-&gt;nama = \$nama; 70     \$mhs-&gt;nim = \$nim; 71     \$mhs-&gt;email = \$email; 72     \$mhs-&gt;jurusan = \$jurusan; 73 74     if (\$mhs-&gt;save()) { 75         \$res['message'] = "Data berhasil diubah!"; 76         \$res['value'] = "\$mhs"; 77         return response(\$res); 78     } else { 79         \$res['message'] = "Gagal!"; 80         return response(\$res); 81     } 82 } 83 </pre> <p>Keterangan:</p> <ul style="list-style-type: none"> <li>• Fungsi update menerima parameter Request yang menampung isian data mahasiswa yang akan diubah dan parameter id yang menunjukkan ID yang dipilih.</li> <li>• Line 70 : Mahasiswa::find(\$id) digunakan untuk pencarian data pada tabel mahasiswa berdasarkan \$id.</li> <li>• Line 76-80 : \$mhs-&gt;save() digunakan untuk menyimpan perubahan data ke database, apabila save() berhasil dijalankan maka akan ditampilkan pesan berhasil serta data yang diubah.</li> </ul>
16	<p>Tambahkan route untuk memanggil fungsi update pada routes/api.php</p>  <pre> 27 Route::put('/mahasiswa/update/{id}', 'MahasiswaController@update'); 28 </pre> <p>Karena kita ingin memasukkan perubahan data, maka perintah yang dipakai adalah 'put'</p>
17	<p>Sekarang kita coba untuk mengubah data berdasarkan ID mahasiswa yang dipilih menggunakan Postman. Gunakan perintah PUT untuk mengubah data.</p> <p>Berikut adalah contoh untuk mengubah data dengan ID=2, maka url diisi :  <a href="http://localhost:8000/api/mahasiswa/update/2">http://localhost:8000/api/mahasiswa/update/2</a>. 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.</p>



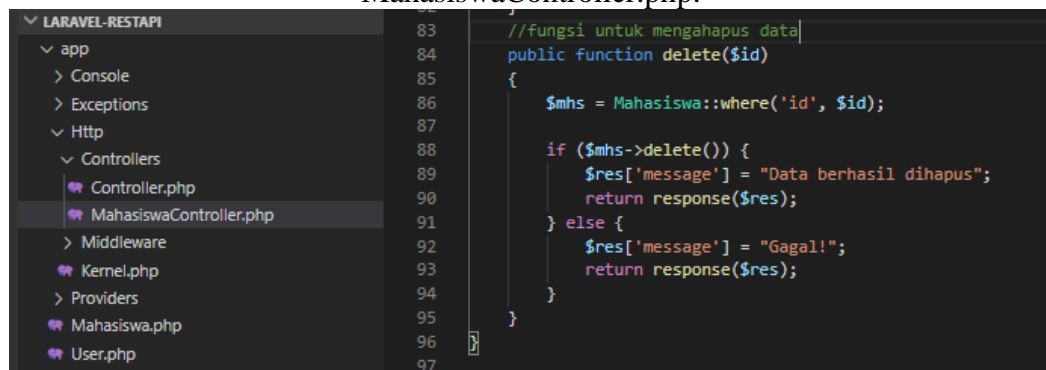
Akan muncul pesan berhasil serta perubahan data dari ID=2.

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

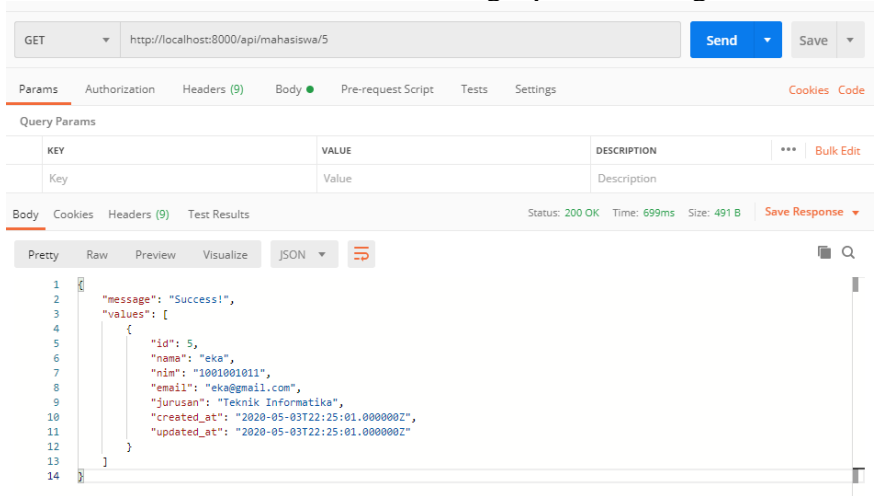
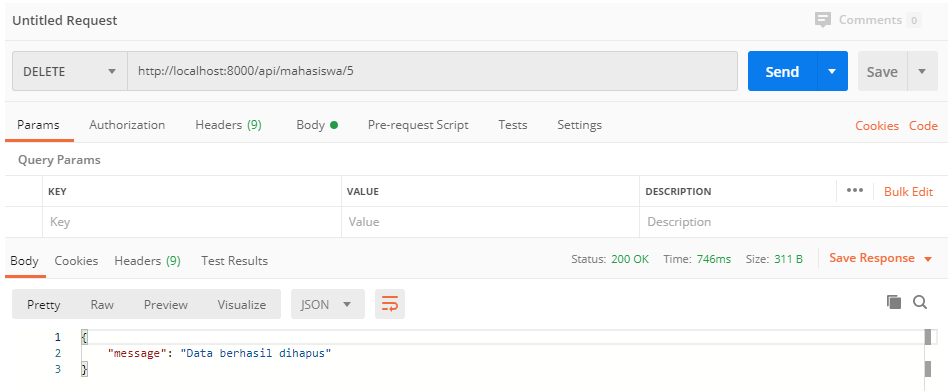


18

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



Keterangan:

	<ul style="list-style-type: none"> <li>• Fungsi delete menerima parameter id yang menunjukkan ID yang dipilih.</li> <li>• Line 92-99 : \$mhs-&gt;delete() digunakan untuk menghapus data dari database, apabila delete() berhasil dijalankan maka akan ditampilkan pesan berhasil.</li> </ul>
19	<p>Tambahkan route untuk memanggil fungsi delete pada api.php</p> <pre>28 29 Route::delete('/mahasiswa/{id}', 'MahasiswaController@delete'); 30</pre> <p>Jika kita ingin menghapus data maka perintah di dalam route adalah delete</p>
20	<p>Buka Postman untuk mencoba menghapus data dari database. Gunakan perintah DELETE untuk mengubah data.</p> <p>Berikut adalah contoh untuk menghapus data dengan ID=5 :</p>  <p>Ini adalah hasil dari get yang ada di ID 5 belum terhapus</p>  <p>Ini adalah hasil dari delete yang menghapus data id nomor 5, dan ketika berhasil akan memunculkan pesan sebagai berikut</p>

Launchpad GET http://localhost:8000/api/maha... + ... No Environment

GET http://localhost:8000/api/mahasiswa Send Save

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (9) Test Results Status: 200 OK Time: 712ms Size: 751 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Success!",
3   "values": [
4     {
5       "id": 2,
6       "nama": "wahyu afifah",
7       "nim": "1001001002",
8       "email": "wahyua@gmail.com",
9       "jurusan": "Teknik Elektro",
10      "created_at": null,
11      "updated_at": "2020-05-03T22:40:32.000000Z"
12    },
13    {
14      "id": 3,
15      "nama": "marsha",
16      "nim": "1001001003",
17      "email": "marsha@gmail.com",
18      "jurusan": "teknik sipil",
19      "created_at": null,
20      "updated_at": null
21    },
22    {
23      "id": 4,
24      "nama": "Ade ismail",
25      "nim": "0804030028",
26      "email": "adeismail@gmail.com",
27      "jurusan": "teknik informatika",
28      "created_at": null,
29      "updated_at": null
30    }
31  ]
32 }
```

Ini adalah hasil dari perintah delete diatas, ID nomor 5 sudah hilang/dihapus