

LAPORAN PEMROGRAMAN WEB LANJUT

Restful API dengan Laravel

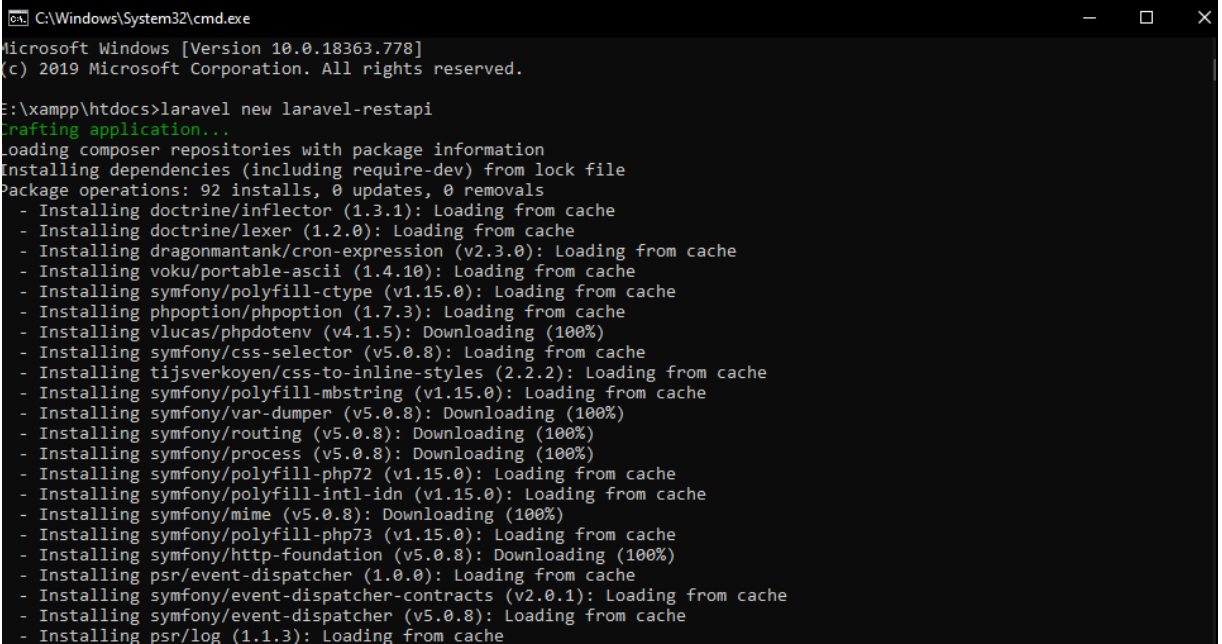
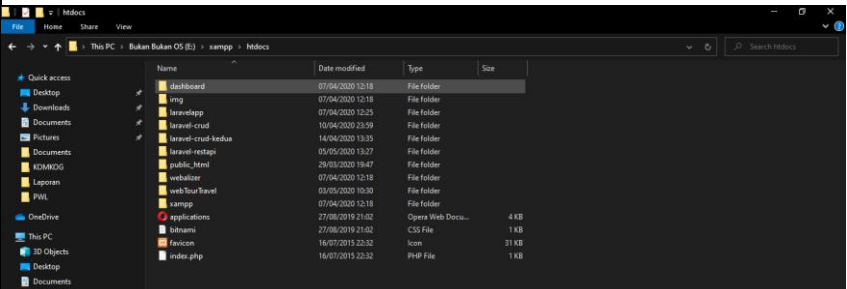



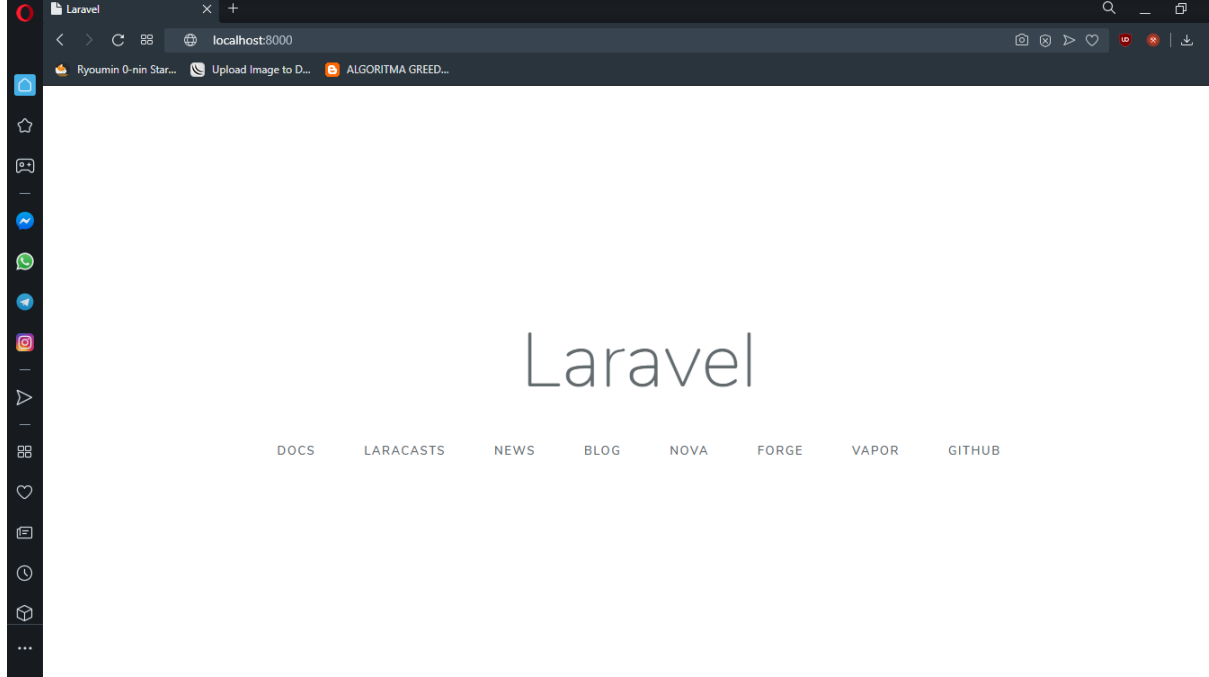
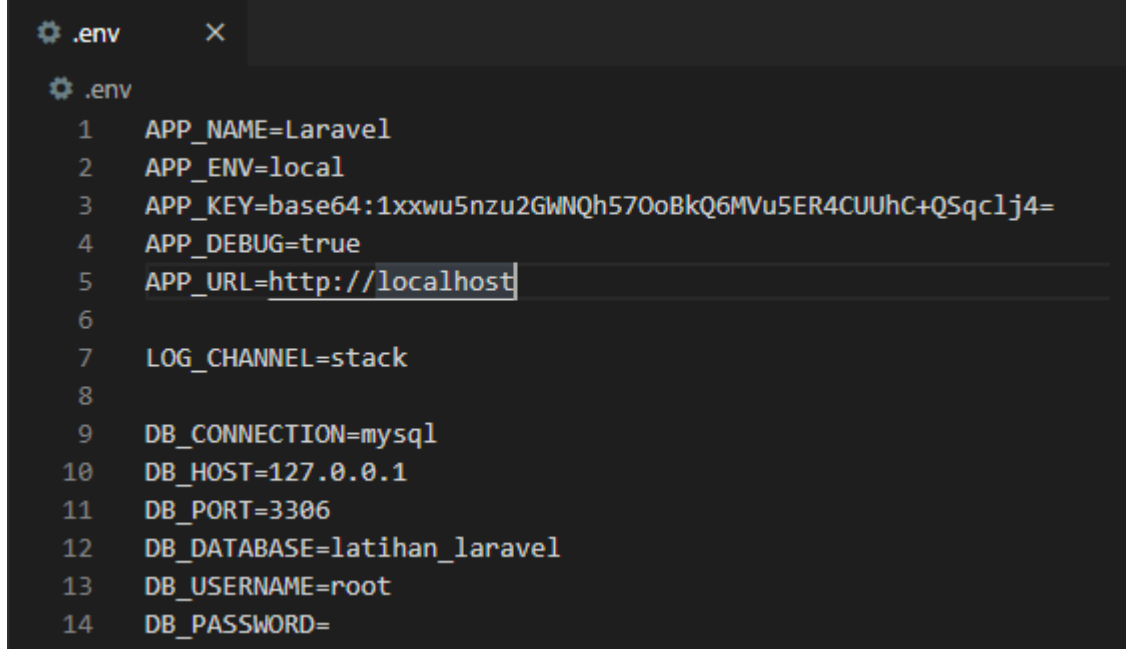
Oleh :

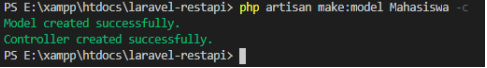
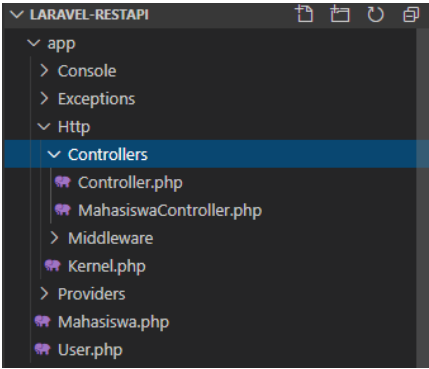
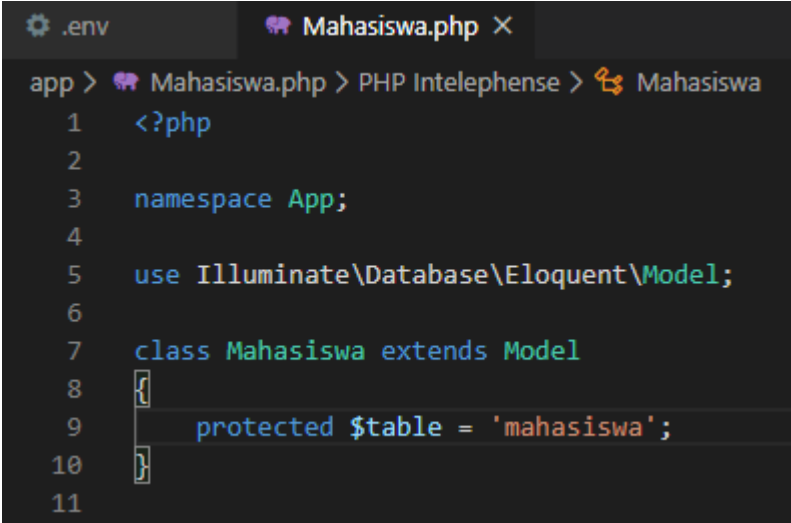
MIRZA ZARQANI RAYHAN / NIM : 1841720205


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

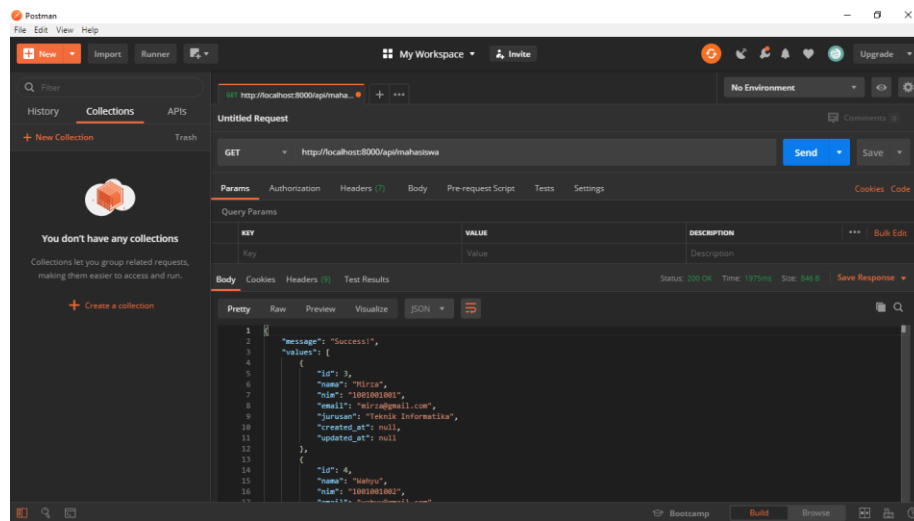
Praktikum: Membuat RESTful API di Laravel

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> <p>Cd C:\laravel-restapi php artisan serve</p>  <p>Akan tampil halaman default Laravel seperti di bawah ini.</p> 
3	<p>Kemudian lakukan konfigurasi <i>database</i> 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> 

4	<p>Buat model dengan nama Mahasiswa, buat juga controllernya. Untuk membuat model dan controllernya sekaligus tuliskan perintah berikut pada <i>command prompt</i> (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"> -c merupakan perintah untuk menyertakan pembuatan <i>controller</i>  <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"> Model ini akan mengelola tabel “mahasiswa” yang terdapat pada database latihan_laravel
	<p>Kemudian kita akan memodifikasi isi dari MahasiswaController.php untuk dapat mengolah data pada tabel ‘mahasiswa’. Pada <i>controller</i> 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>

6	
	<p>Keterangan:</p> <ul style="list-style-type: none"> • Tambahkan line 6 agar model Mahasiswa dapat digunakan pada MahasiswaController • Line 15-19 digunakan untuk memeriksa apakah data>0 atau data tidak kosong • Variabel \$res[message] digunakan untuk menampilkan pesan apakah ada data atau tidak ada data di tabel mahasiswa • Variabel \$array[values] akan menyimpan semua baris data pada tabel mahasiswa
7	<p>Tambahkan route untuk memanggil fungsi index pada file routes/api.php (Line 21).</p> <pre> 21 Route::get('mahasiswa', 'MahasiswaController@index'); </pre> <p>Karena kita ingin menampilkan data, maka perintah yang dipakai adalah 'get'.</p>
8	<p>Ketikkan perintah php artisan serve pada <i>command prompt</i>. 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.</p> 



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

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

Keterangan:

- Fungsi getId menerima parameter \$id yang menunjukkan ID mahasiswa yang dipilih
- Line 28 merupakan pemanggilan model untuk membaca data berdasarkan ID

9

```
env  Mahasiswa.php  MahasiswaController.php X  api.php
http > Controllers > MahasiswaController.php > PHP Intelephense > MahasiswaCon

22
23     public function getId($id){
24         $data = Mahasiswa::where('id', $id)->get();
25
26         if (count($data) > 0) {
27             $res['message'] = "Success!";
28             $res['values'] = $data;
29             return response($res);
30         } else {
31             $res['message'] = "Failed!";
32             return response($res);
33         }
34     }
35 }
36
```

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

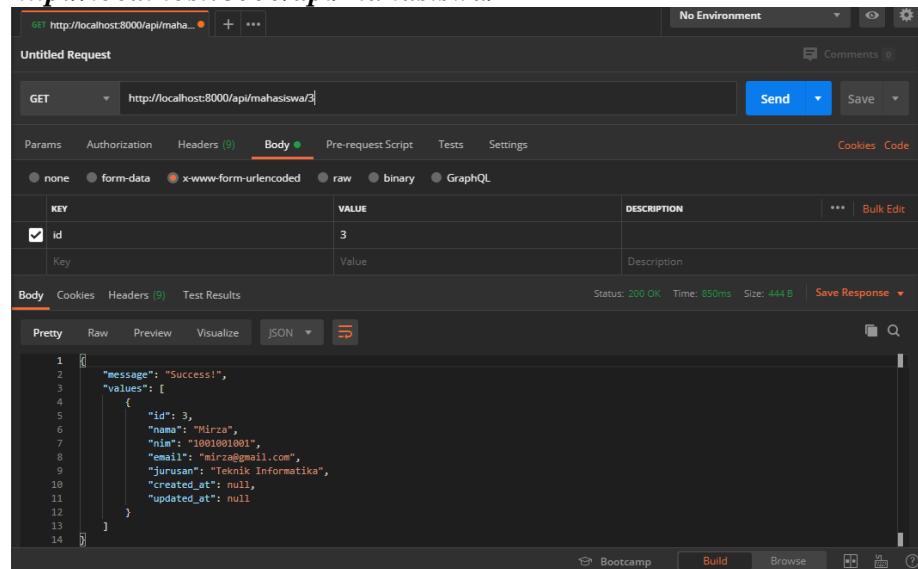
10

```
api.php X
laravel-restapi > routes > api.php
23     Route::get('/mahasiswa/{id}', 'MahasiswaController@getId');
```

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

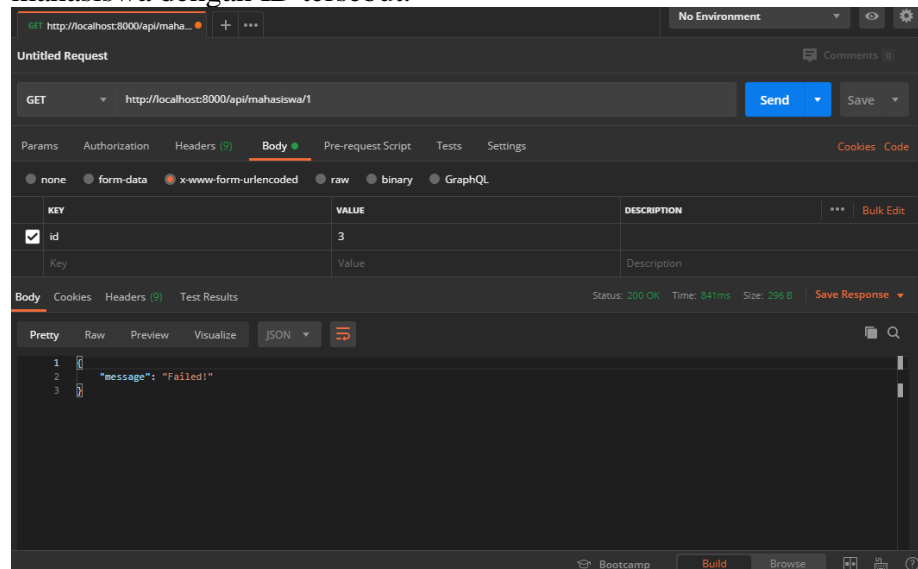
Sekarang kita coba untuk menampilkan data berdasarkan ID mahasiswa yang dipilih menggunakan Postman. Gunakan perintah **GET** untuk menampilkan data.

Di bawah ini adalah contoh untuk menampilkan data dengan ID=1, maka url diisi :
http://localhost:8000/api/mahasiswa/1



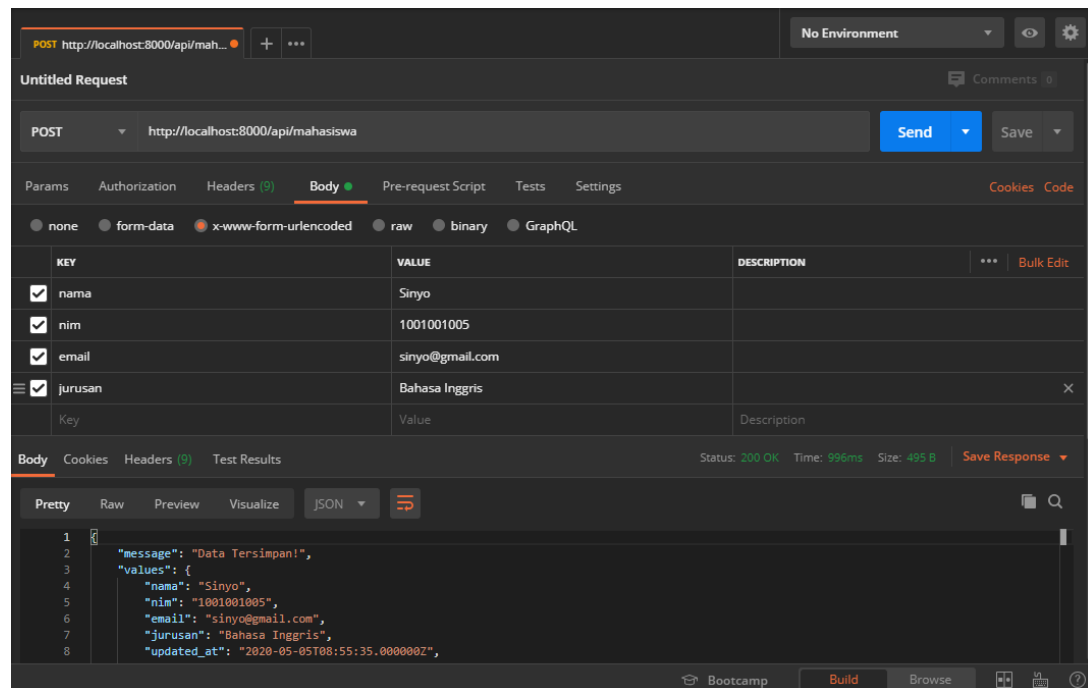
11

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



12	<p>Setelah dapat menampilkan data, kita akan membuat fungsi untuk menambahkan data baru ke database dengan nama create pada MahasiswaController.php.</p> <pre> 36 public function create(Request \$request){ 37 \$mhs = new Mahasiswa(); 38 \$mhs->nama = \$request->nama; 39 \$mhs->nim = \$request->nim; 40 \$mhs->email = \$request->email; 41 \$mhs->jurusan = \$request->jurusan; 42 43 if(\$mhs->save()){ 44 \$res['message'] = "Data Tersimpan!"; 45 \$res['values'] = \$mhs; 46 return response(\$res); 47 } else { 48 \$res['message'] = "Failed!"; 49 return response(\$res); 50 } 51 } </pre> <p>Keterangan:</p> <ul style="list-style-type: none"> • Fungsi create menerima parameter Request yang menampung isian data mahasiswa yang akan ditambahkan ke database. • Line 51-55 : <code>\$mhs->save()</code> digunakan untuk menyimpan data ke database, apabila <code>save()</code> berhasil dijalankan maka akan ditampilkan pesan berhasil serta data yang ditambah.
13	<p>Tambahkan <i>route</i> untuk memanggil fungsi create pada routes/api.php</p> <pre>Route::post('/mahasiswa', 'MahasiswaController@create');</pre> <p>Karena kita ingin menambah data, maka perintah yang dipakai adalah 'post'.</p>

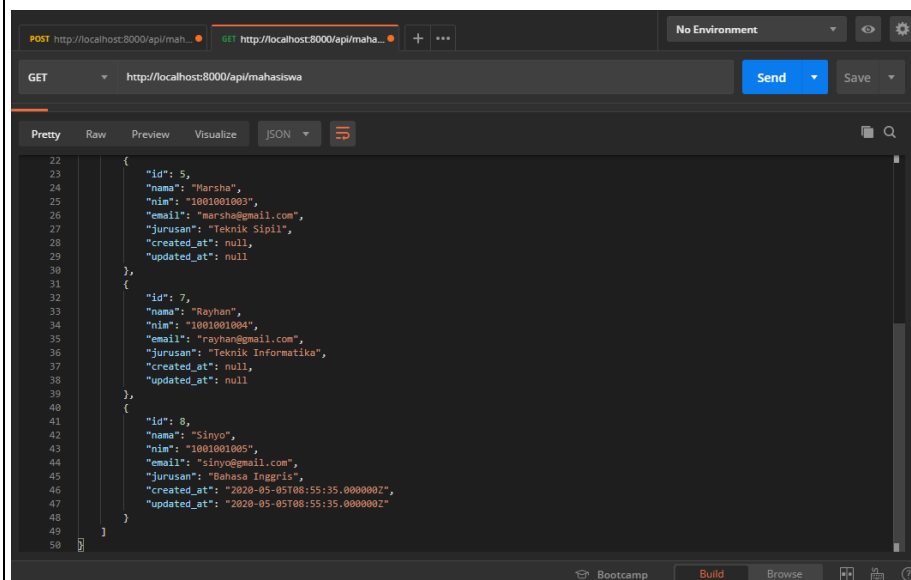
Kita coba untuk menambahkan data melalui Postman.



14

- 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**.

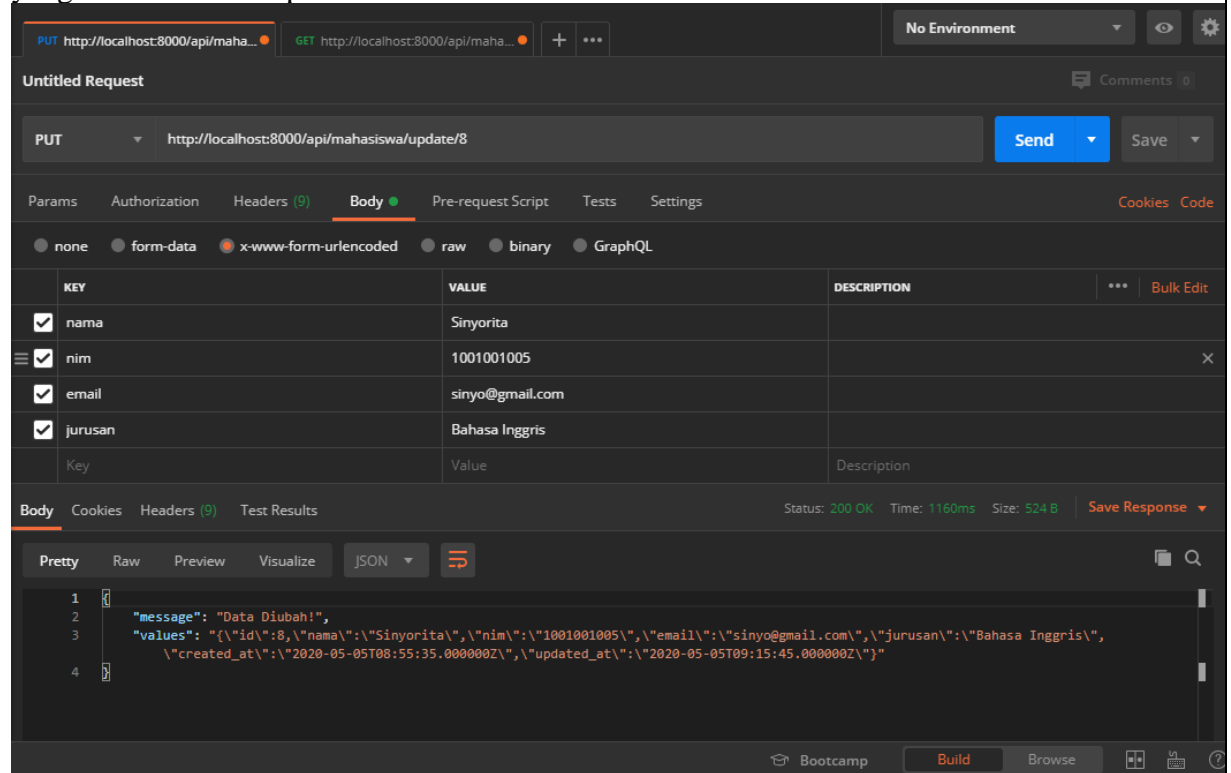
Kemudian coba untuk tampilkan semua data, kita lihat apakah data baru sudah masuk ke database.



15	<p>Selanjutnya kita akan menambahkan fungsi untuk mengubah data dari database. Buat fungsi update pada MahasiswaController.php.</p> <pre> 53 public function update(Request \$request, \$id){ 54 \$nama = \$request->nama; 55 \$nim = \$request->nim; 56 \$email = \$request->email; 57 \$jurusan = \$request->jurusan; 58 59 \$mhs = Mahasiswa::find(\$id); 60 \$mhs->nama = \$nama; 61 \$mhs->nim = \$nim; 62 \$mhs->email = \$email; 63 \$mhs->jurusan = \$jurusan; 64 65 if(\$mhs->save()){ 66 \$res['message'] = "Data Diubah!"; 67 \$res['values'] = \$mhs; 68 return response(\$res); 69 } else { 70 \$res['message'] = "Failed!"; 71 return response(\$res); 72 } 73 } </pre> <p>Keterangan:</p> <ul style="list-style-type: none"> • Fungsi update menerima parameter Request yang menampung isian data mahasiswa yang akan diubah dan parameter id yang menunjukkan ID yang dipilih. • Line 64 : <code>Mahasiswa::find(\$id)</code> digunakan untuk pencarian data pada tabel mahasiswa berdasarkan \$id. • Line 70-74 : <code>\$mhs->save()</code> digunakan untuk menyimpan perubahan data ke database, apabila <code>save()</code> berhasil dijalankan maka akan ditampilkan pesan berhasil serta data yang diubah.
16	<p>Tambahkan <i>route</i> untuk memanggil fungsi update pada routes/api.php</p> <pre> 24 Route::put('/mahasiswa/update/{id}', 'MahasiswaController@update'); </pre> <p>Karena kita ingin memasukkan perubahan data, maka perintah yang dipakai adalah 'put'.</p>

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=9, maka url diisi : ***http://localhost:8000/api/mahasiswa/update/9***. Pilih tab **Body** dan pilih radio button **x-www-form-urlencoded**. Isikan nama kolom pada database pada **KEY**, untuk isian data yang diubah tuliskan pada **VALUE**.



Akan muncul pesan berhasil serta perubahan data dari ID=9.

Kemudian coba untuk menampilkan data dengan ID=9 untuk melihat apakah data sudah ter-update.

PUT http://localhost:8000/api/maha...

GET http://localhost:8000/api/maha...

+ ...

No Environment

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

Send Save

PrettyRawPreviewVisualizeJSON

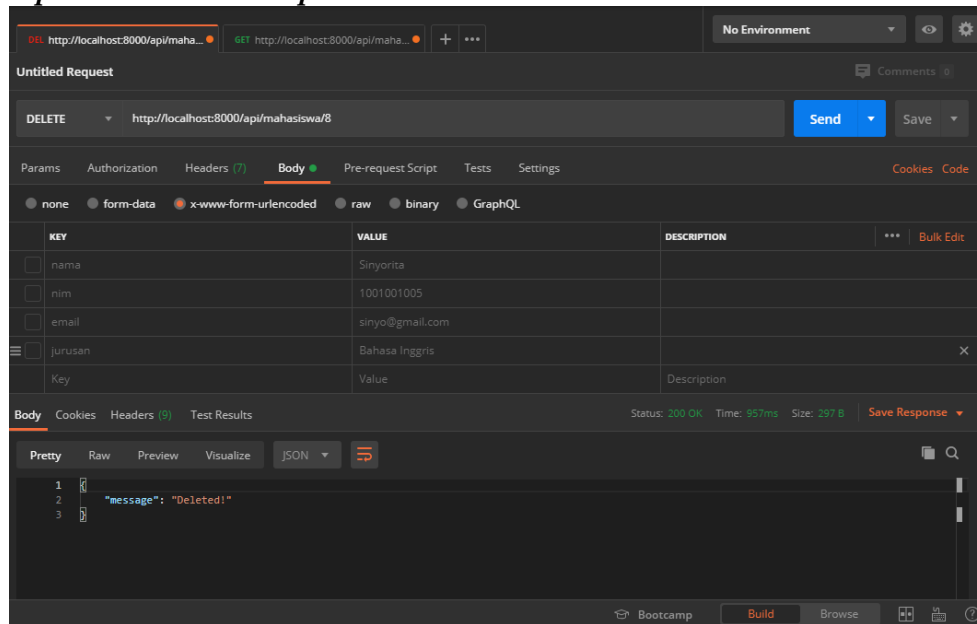
```
22  {
23    "id": 5,
24    "nama": "Marsha",
25    "nim": "1001001003",
26    "email": "marsha@gmail.com",
27    "jurusan": "Teknik Sipil",
28    "created_at": null,
29    "updated_at": null
30  },
31  {
32    "id": 7,
33    "nama": "Rayhan",
34    "nim": "1001001004",
35    "email": "rayhan@gmail.com",
36    "jurusan": "Teknik Informatika",
37    "created_at": null,
38    "updated_at": null
39  },
40  {
41    "id": 8,
42    "nama": "Sinyorita",
43    "nim": "1001001005",
44    "email": "sinyo@gmail.com",
45    "jurusan": "Bahasa Inggris",
46    "created_at": "2020-05-05T08:55:35.000000Z",
47    "updated_at": "2020-05-05T09:15:45.000000Z"
48  }
49 ]
50 }
```

BootcampBuildBrowse

18	<p>Terakhir kita akan membuat fungsi untuk menghapus data dengan nama delete di MahasiswaController.php.</p> <pre data-bbox="337 199 1141 667"> 79 public function delete(\$id) 80 { 81 \$mhs = Mahasiswa::where('id', \$id); 82 83 if (\$mhs->delete()) { 84 \$res['message'] = "Deleted!"; 85 return response(\$res); 86 } else { 87 \$res['message'] = "Failed!"; 88 return response(\$res); 89 } 90 } </pre> <p>Keterangan:</p> <ul style="list-style-type: none"> • Fungsi delete menerima parameter id yang menunjukkan ID yang dipilih. • Line 83-89 : <code>\$mhs->delete()</code> digunakan untuk menghapus data dari database, apabila <code>delete()</code> berhasil dijalankan maka akan ditampilkan pesan berhasil.
19	<p>Tambahkan <i>route</i> untuk memanggil fungsi delete pada routes/api.php</p> <pre data-bbox="337 997 1510 1060"> 25 Route::delete('/mahasiswa/{id}', 'MahasiswaController@delete'); </pre> <p>Karena kita ingin menghapus data, maka perintah yang dipakai adalah 'delete'.</p>

Buka Postman untuk mencoba menghapus data dari database. Gunakan perintah **DELETE** untuk mengubah data.

Berikut adalah contoh untuk menghapus data dengan ID=2, maka url diisi :
`http://localhost:8000/api/mahasiswa /2`



Muncul pesan berhasil ketika data terhapus dari database.