

# **LAPORAN JOOBSHEET 14**

**(Membuat RESTful API Laravel)**

**Disusun sebagai**

**MATA KULIAH :**

**Pemrograman Web Lanjut**



**TI-2B**

**OLEH**

**WILDAN DAWAM BASH (1841720166)**

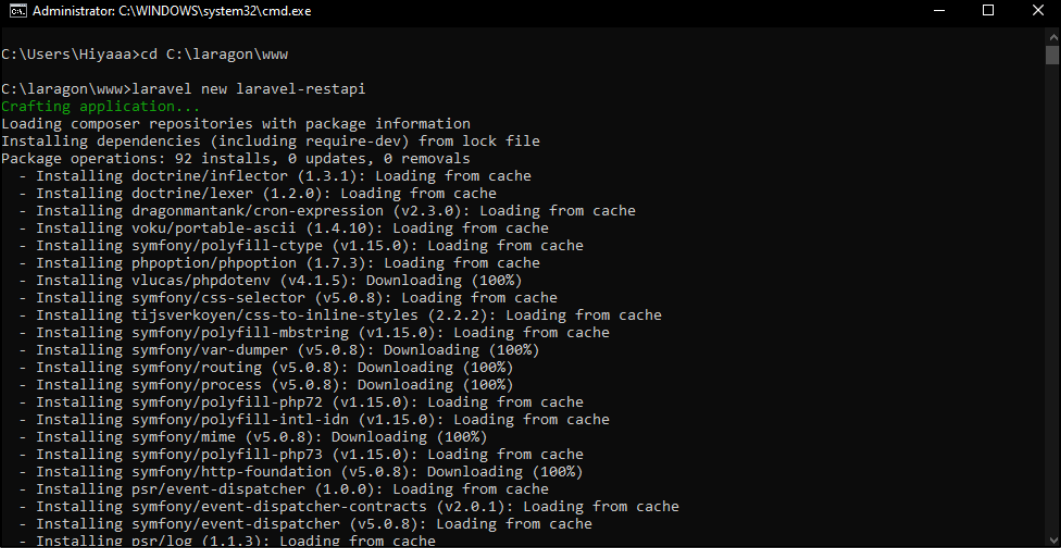
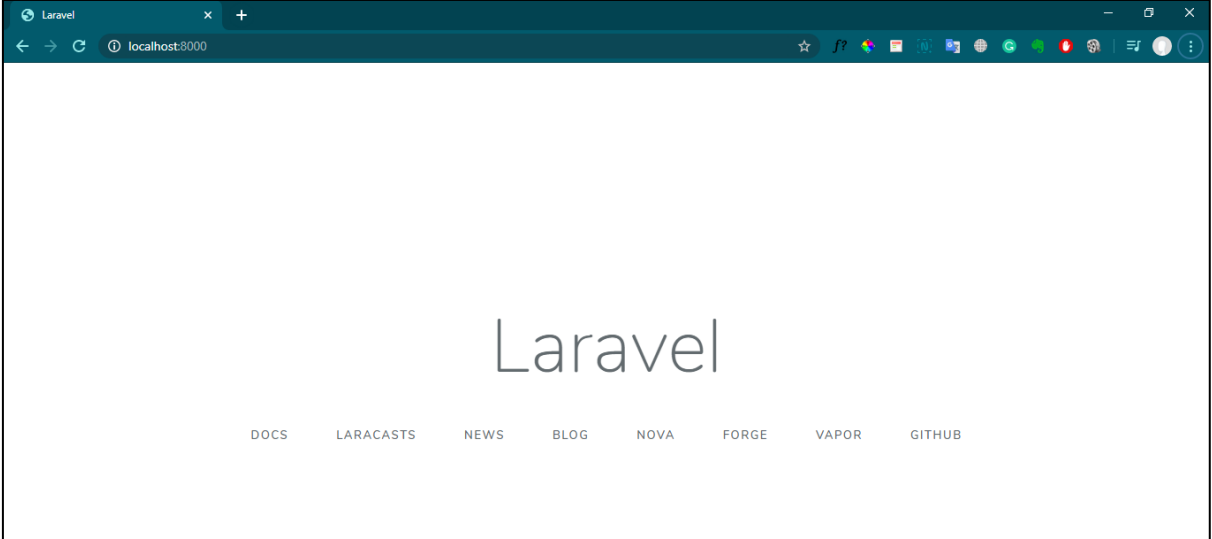
**PROGAM STUDI D-IV TEKNIK INFORMATIKA**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**2020**

## Praktikum

No	Keterangan
1	<p>Buat project baru dengan nama “<b>laravel-restapi</b>”. Buka command prompt, tuliskan perintah berikut.</p> <pre>cd C:\laragon\www laravel new laravel-restapi</pre>  <p>Disini saya menggunakan laragon jadi menyesuaikan.</p>
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>
3	<p>Lakukan konfigurasi database pada file <b>.env</b> isikan nama database, username, dan password yang akan digunakan. Pada project ini kita akan menggunakan database dari minggu-minggu sebelumnya yaitu “<b>latihan_laravel</b>”</p>

```

.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:fXCq9Vg5/PnCJQ9MaxF8p/F7gPKlqjge3oxAhZI5aSE=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=latihan_laravel
13 DB_USERNAME=root
14 DB_PASSWORD=mysql
15

```

4. 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)

**php artisan make:model Mahasiswa -c**

*-c merupakan perintah untuk menyertakan pembuatan controller*

```

Administrator: C:\WINDOWS\system32\cmd.exe
C:\laragon\www\laravel-restapi>php artisan make:model Mahasiswa -c
Model created successfully.
Controller created successfully.
C:\laragon\www\laravel-restapi>

```

Sehingga pada project laravel-restapi akan bertambah dua file yaitu **model Mahasiswa.php** serta controller **MahasiswaController.php**.

```

LARAVEL-RESTAPI
├── app
│   ├── Console
│   ├── Exceptions
│   └── Http
│       ├── Controllers
│       │   ├── Controller.php
│       │   └── MahasiswaController.php
│       ├── Middleware
│       ├── Kernel.php
│       ├── Providers
│       │   ├── Mahasiswa.php
│       │   └── User.php
│       └── bootstrap

```

5. Selanjutnya ubah isi model Mahasiswa.php seperti berikut ini.

```

Mahasiswa.php
app > Mahasiswa.php > ...
1 <?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

```

- 6 Modifikasi 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.

```
Mahasiswa.php  MahasiswaController.php X
app > Http > Controllers > MahasiswaController.php > MahasiswaController > index
1  <?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         $data = Mahasiswa::all();
12
13         if(count($data) > 0){
14             $res['message'] = 'Success!';
15             $res['values'] = $data;
16             return response($res);
17         }else{
18             $res['message'] = 'Kosong!';
19             return response($res);
20         }
21     }
22 }
23
```

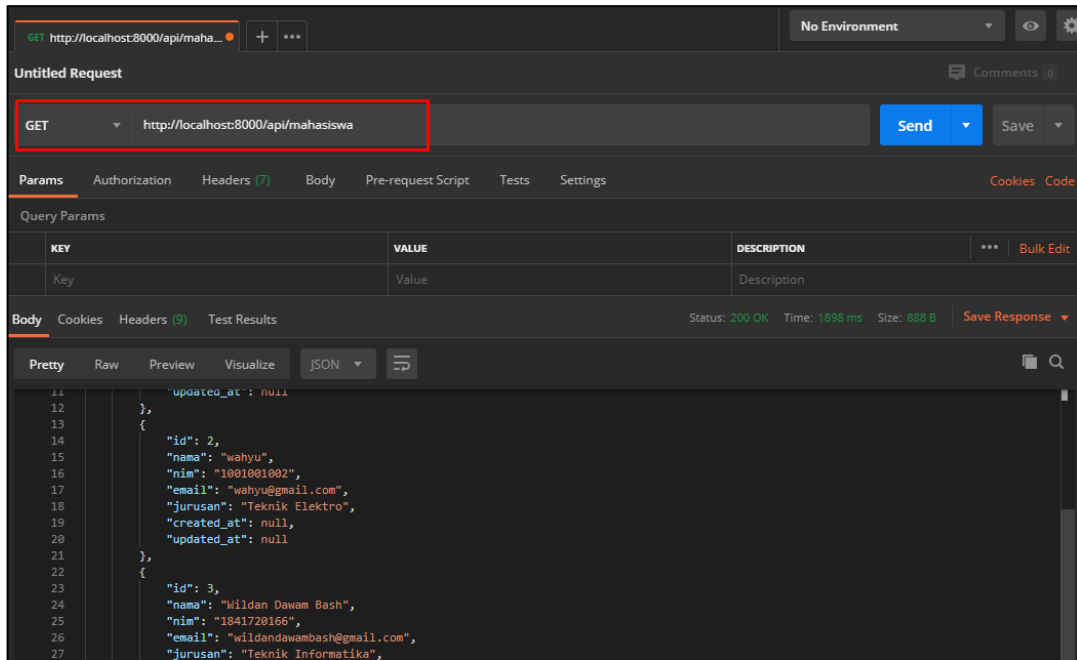
- 7 Tambahkan route untuk memanggil fungsi index pada file **routes/api.php** (Line 21).

```
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.



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

```

Mahasiswa.php MahasiswaController.php X api.php
app > Http > Controllers > MahasiswaController.php > ...

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         $res['message'] = 'Gagal!';
32         return response($res);
33     }
34 }
35
36

```

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

```

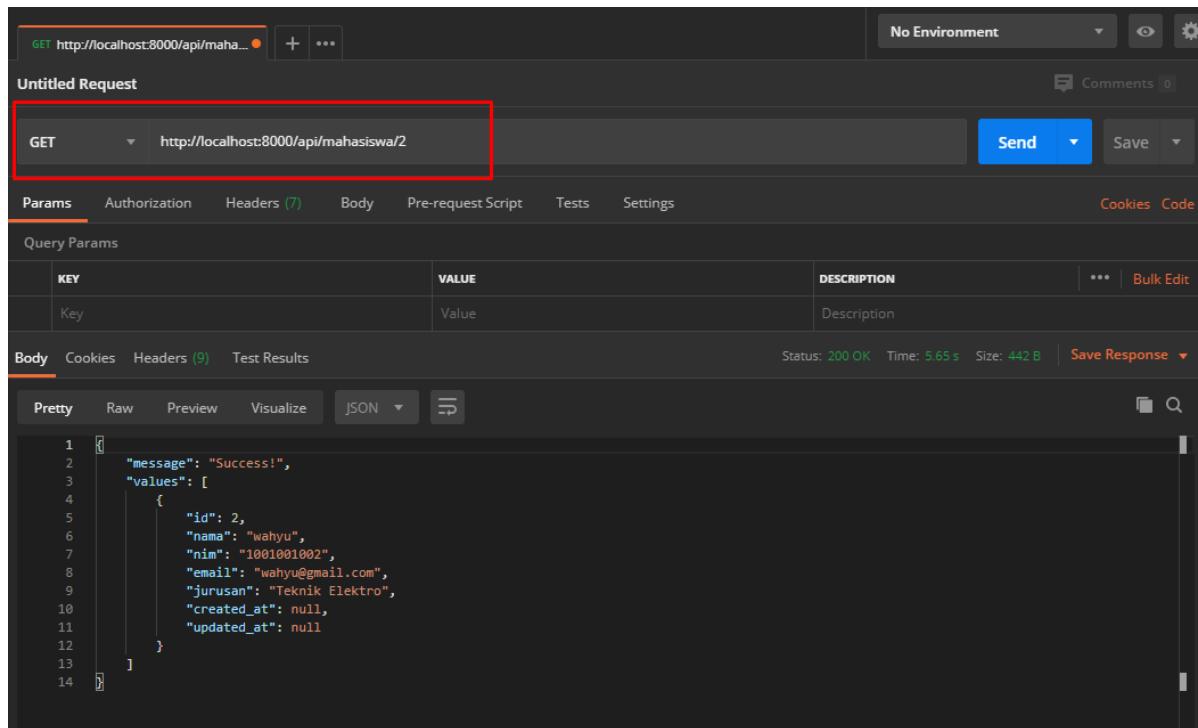
22
23 Route::get('/mahasiswa/{id}', 'MahasiswaController@getId');

```

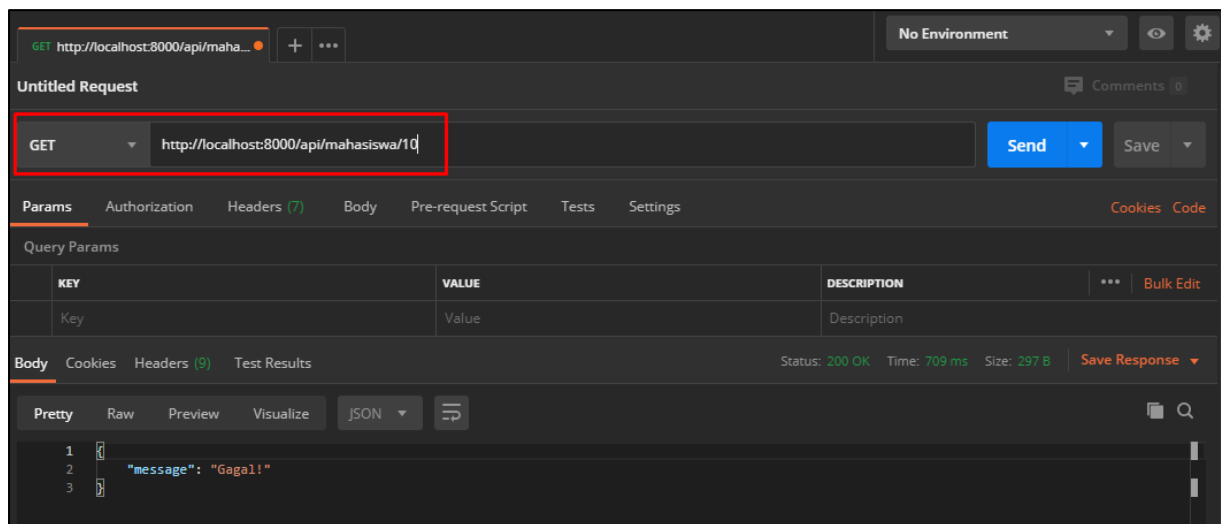
Karena kita akan mengambil data maka menggunakan perintah get

- 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***



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**.

```

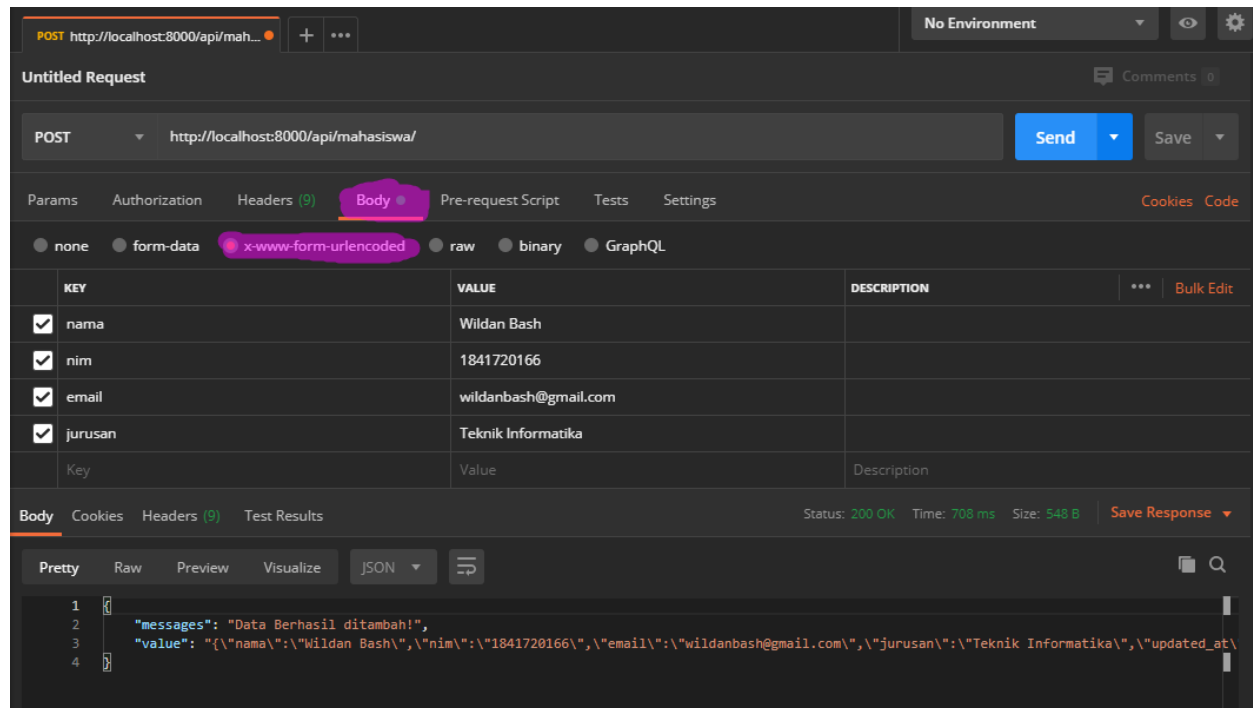
Mahasiswa.php  MahasiswaController.php  api.php
app > Http > Controllers > MahasiswaController.php > ...
35
36 public function create(Request $request){
37     $mhs = new Mahasiswa();
38     $mhs->nama = $request->nama;
39     $mhs->nim = $request->nim;
40     $mhs->email = $request->email;
41     $mhs->jurusan = $request->jurusan;
42
43     if($mhs->save()){
44         $res['messages'] = "Data Berhasil ditambah!";
45         $res['value'] = "$mhs";
46         return response($res);
47     }
48
49 }
50

```

- 13 Tambahkan *route* untuk memanggil fungsi create pada **routes/api.php**. Karena kita ingin menambah data, maka perintah yang dipakai adalah 'post' .

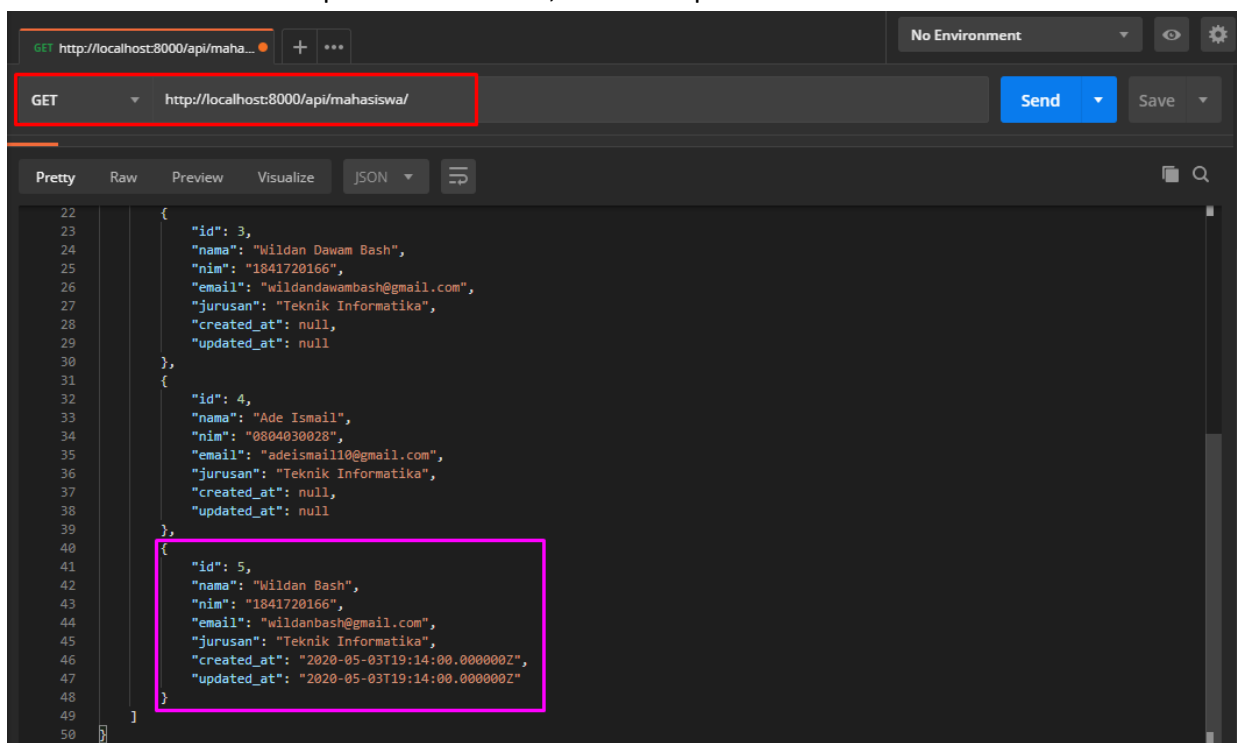
```
24  
25 Route::post('/mahasiswa', 'MahasiswaController@create');
```

- 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**.

```
app > Http > Controllers > MahasiswaController.php > ...
49
50 public function update(Request $request, $id){
51     $nama = $request->nama;
52     $nim = $request->nim;
53     $email = $request->email;
54     $jurusan = $request->jurusan;
55
56     $mhs = Mahasiswa::find($id);
57     $mhs->nama = $nama;
58     $mhs->nim = $nim;
59     $mhs->email = $email;
60     $mhs->jurusan = $jurusan;
61
62     if($mhs->save()){
63         $res['message'] = "Data berhasil diubah!";
64         $res['value'] = "$mhs";
65         return response($res);
66     }else{
67         $res['message'] = "Gagal!";
68         return response($res);
69     }
70 }
71 }
```

- 16 Tambahkan *route* untuk memanggil fungsi update pada **routes/api.php**. Karena kita ingin memasukkan perubahan data, maka perintah yang dipakai adalah 'put'.

```
26
27 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.

The screenshot shows the Postman interface for a PUT request. The URL is `http://localhost:8000/api/mahasiswa/update/2`. The request body is set to `x-www-form-urlencoded` and contains the following data:

KEY	VALUE	DESCRIPTION
nama	Wahyu afifah	
nim	1001001002	
email	wahyua@gmail.com	
jurusan	Teknik Elektro	
Key	Value	Description

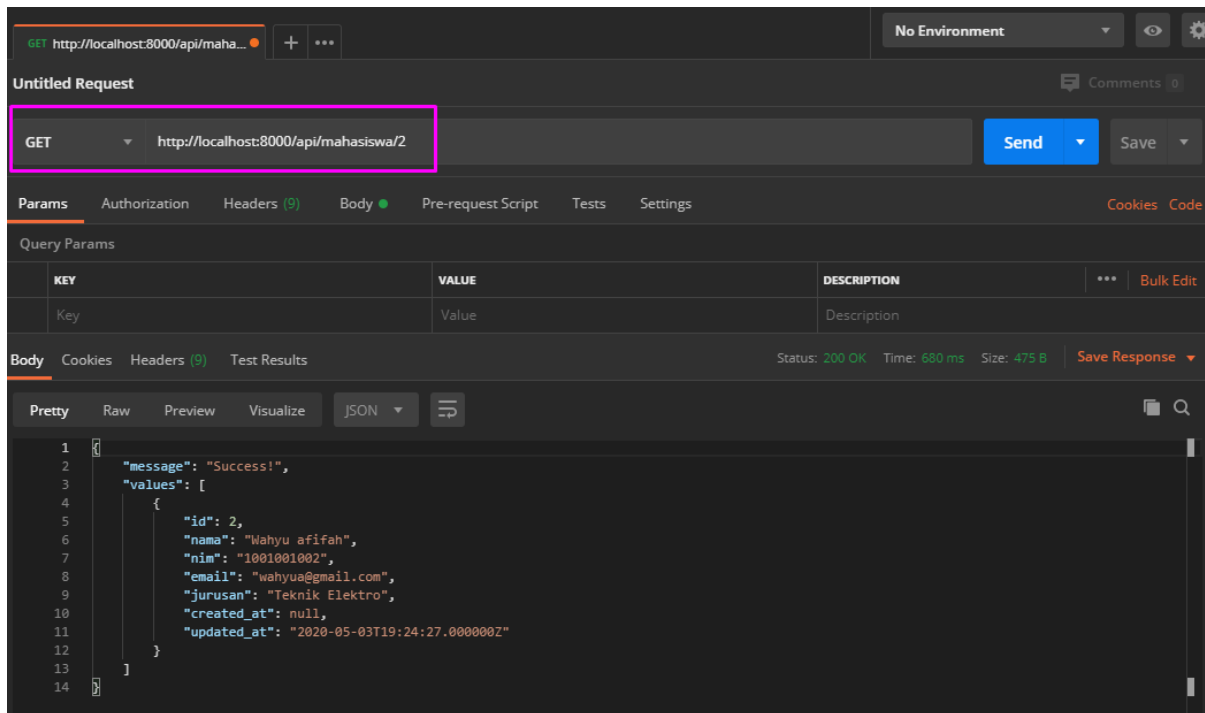
The response status is `200 OK` with a time of `3.27 s` and a size of `511 B`. The response body is shown in JSON format:

```
{
  "message": "Data berhasil diubah!",
  "value": "{\n  \"id\": 2,\n  \"nama\": \"Wahyu afifah\",\n  \"nim\": \"1001001002\",\n  \"email\": \"wahyua@gmail.com\",\n  \"jurusan\": \"Teknik Elektro\",\n  \"created_at\": \"2024-01-15 15:30:00\"\n}"
}
```

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 **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=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**.

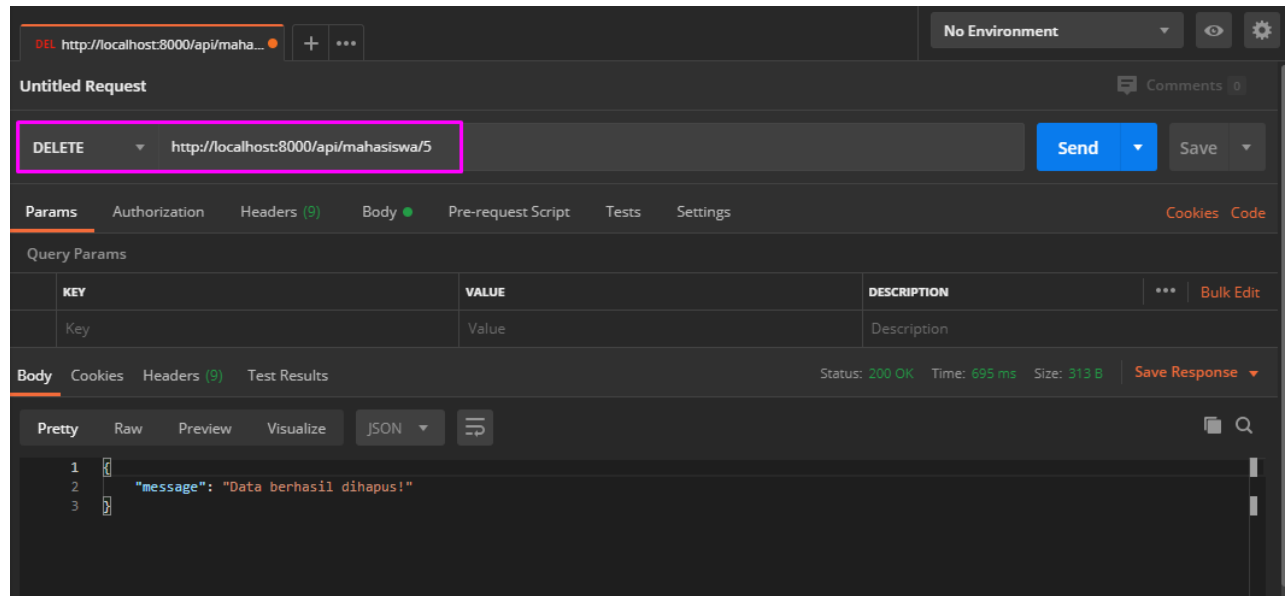
```
app > Http > Controllers > MahasiswaController.php > ...
71
72 public function delete($id){
73     $mhs = Mahasiswa::where('id', $id);
74
75     if($mhs->delete()){
76         $res['message'] = "Data berhasil dihapus!";
77         return response($res);
78     }else{
79         $res['message'] = "Gagal!";
80         return response($res);
81     }
82 }
83 }
```

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

```
28
29 Route::delete('/mahasiswa/{id}', 'MahasiswaController@delete');
```

Karena kita ingin menghapus data, maka perintah yang dipakai adalah 'delete'.

- 20 Buka Postman untuk mencoba menghapus data dari database. Gunakan perintah **DELETE** untuk mengubah data.  
Contoh kita akan menghapus data dengan id = 5, sesuai dengan yang ada dalam database, id 5 disini merupakan data yang ditambahkan dalam tahap sebelumnya  
***http://localhost:8000/api/mahasiswa /5***



Muncul pesan berhasil ketika data terhapus dari database