

**LAPORAN TUGAS  
PEMROGRAMAN WEB LANJUT**

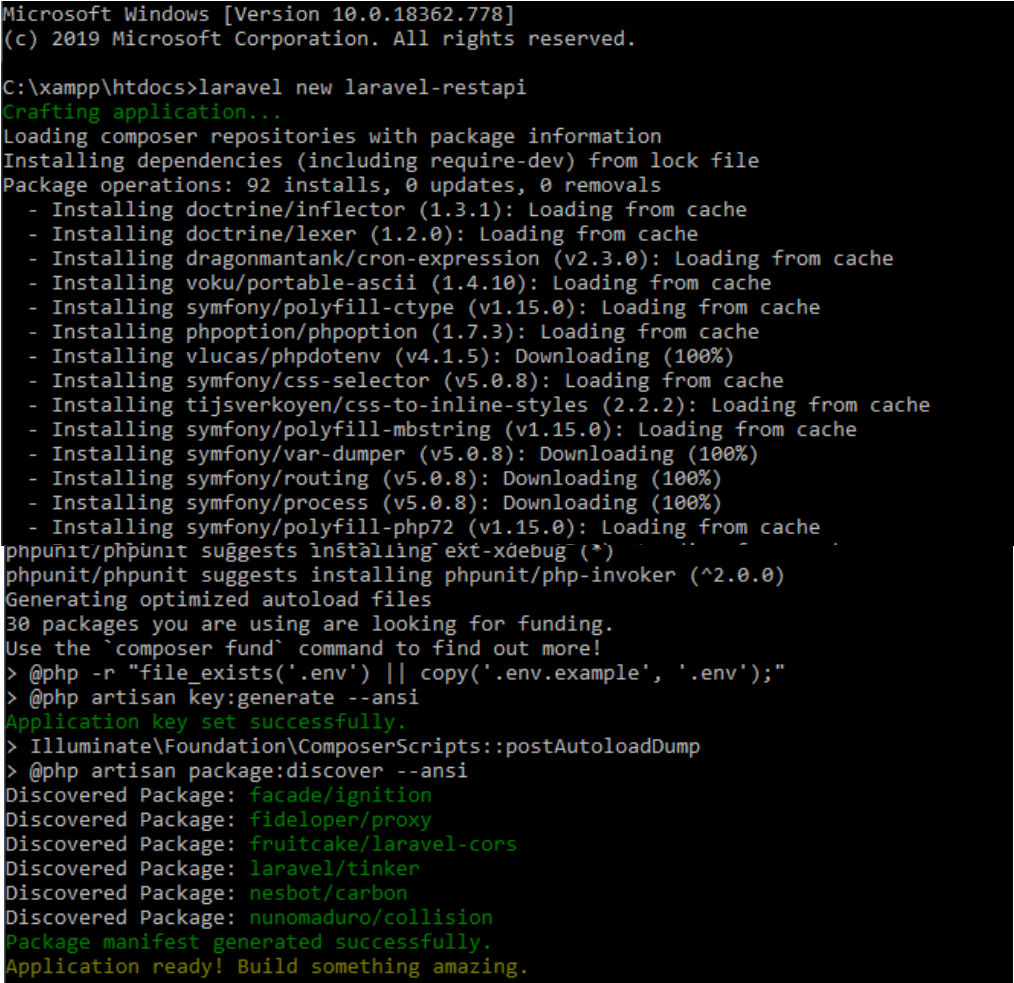
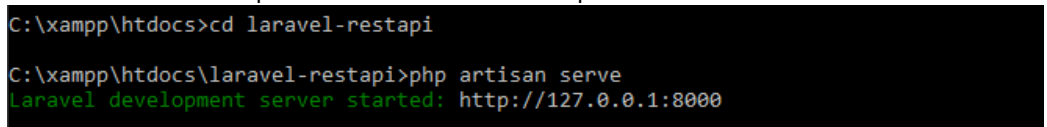
**JOBSHEET 14**

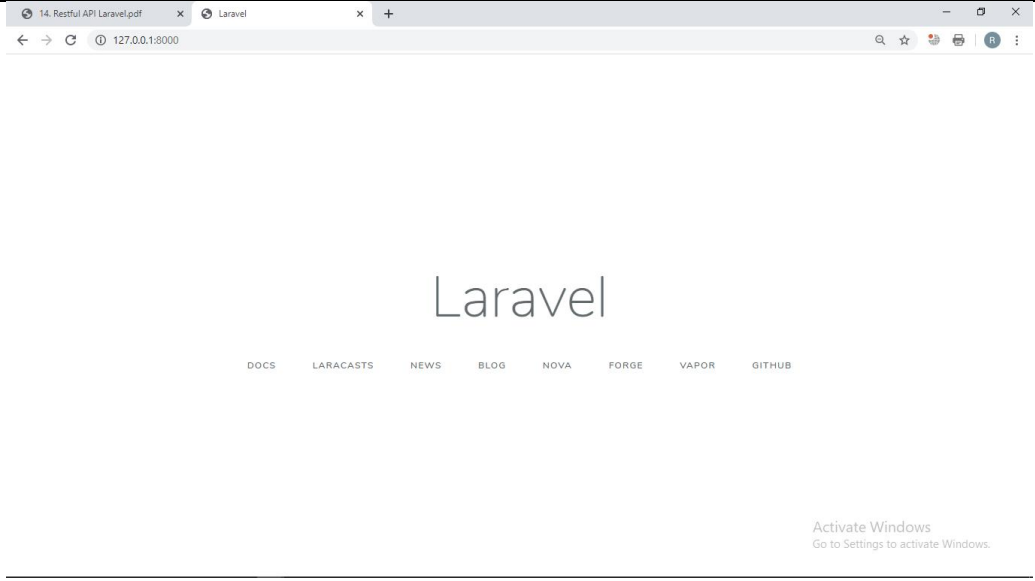
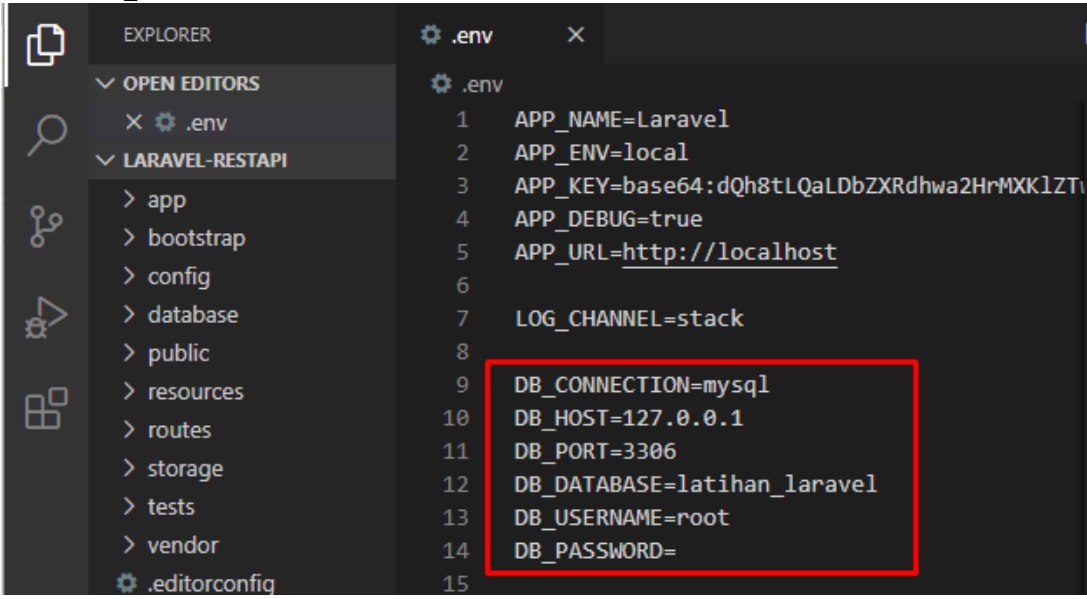
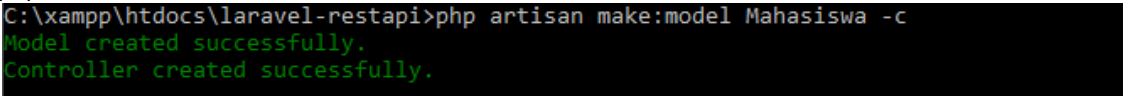


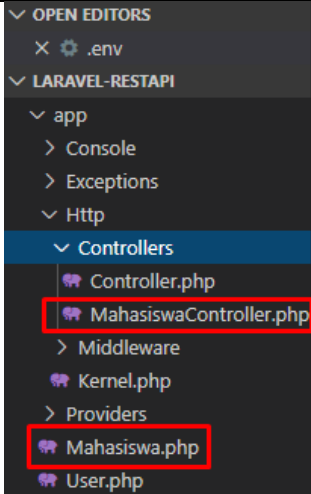
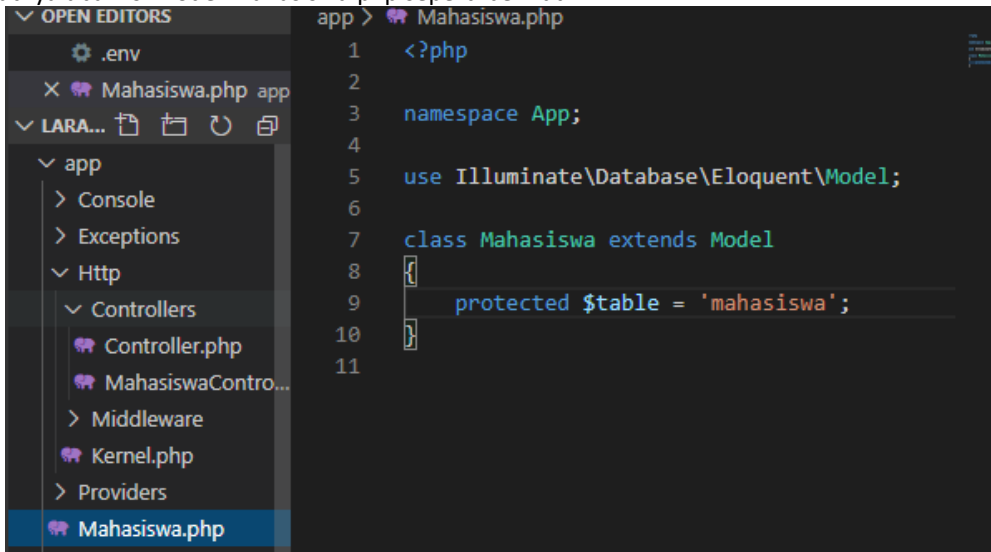
**Oleh :**

**Rahmad Alfian Maskuri / 1841720048**

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

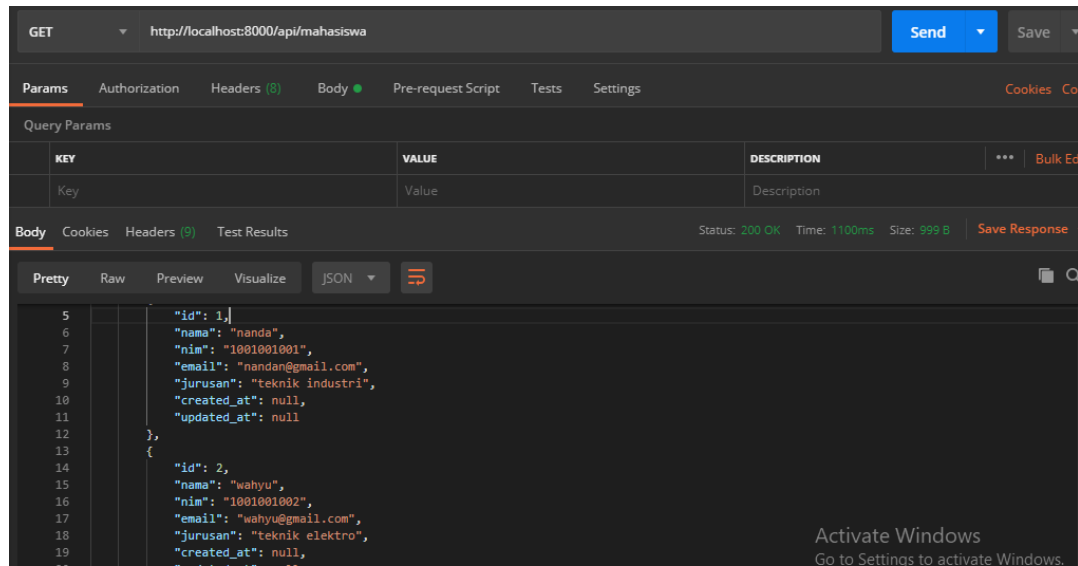
No	Langkah Percobaan
1	<p>Buat project baru dengan nama "laravel-restapi". Buka command prompt, tuliskan perintah berikut.</p> <pre>cd C:\xampp\htdocs laravel new laravel-restapi</pre>  <p>The screenshot shows the terminal output for creating a new Laravel project. It starts with the command 'laravel new laravel-restapi' in the directory 'C:\xampp\htdocs'. The output shows the installation of dependencies and the creation of the project files. The final output is 'Application ready! Build something amazing.'</p>
2	<p>Kita coba jalankan dulu project tersebut. Pada command prompt tulis perintah berikut. cd C:\laravel-restapi</p> <p>php artisan serve Akan tampil halaman default Laravel seperti di bawah ini.</p>  <p>The screenshot shows the terminal output for running the Laravel development server. It starts with the command 'php artisan serve' in the directory 'C:\xampp\htdocs\laravel-restapi'. The output shows the server starting and listening on port 8000. The final output is 'Laravel development server started: http://127.0.0.1:8000'.</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> 
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> <pre>php artisan make:model Mahasiswa -c</pre> 

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

	<pre> .env  Mahasiswa.php  MahasiswaController.php X app &gt; Http &gt; Controllers &gt; MahasiswaController.php 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         \$data = Mahasiswa::all(); 12 13         if(count(\$data) &gt; 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 </pre>
7	<p>Tambahkan route untuk memanggil fungsi index pada file routes/api.php (Line 21).</p> <pre> routes &gt; api.php 1  &lt;?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 applicatio 12   routes are loaded by the RouteServiceProvider within a group 13   is assigned the "api" middleware group. Enjoy building your # 14   15 */ 16 17 Route::middleware('auth:api')-&gt;get('/user', function (Request \$ 18       return \$request-&gt;user(); 19   }); 20 21 Route::get('mahasiswa', 'MahasiswaController@index'); </pre>
8	<p>Ketikkan perintah php artisan serve pada command prompt. Lalu kita coba menguji fungsi untuk menampilkan data menggunakan aplikasi Postman. Gunakan perintah GET, isikan url : <a href="http://localhost:8000/api/mahasiswa">http://localhost:8000/api/mahasiswa</a> Berikut adalah tampilan dari aplikasi Postman.</p>

```
C:\xampp\htdocs\laravel-restapi>php artisan serve
Laravel development server started: http://127.0.0.1:8000
```



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

```
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 }
37
```

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

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

- 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>

GET <http://localhost:8000/api/mahasiswa/2> Send Save

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (9) Test Results Status: 200 OK Time: 842ms Size: 442 B Save Response

Pretty Raw Preview Visualize JSON ⌵

```

1 {
2   "message": "Success!",
3   "values": [
4     {
5       "id": 2,
6       "nama": "wahyu",
7       "nim": "1001001002",
8       "email": "wahyu@gmail.com",
9       "jurusan": "teknik elektro",
10      "created_at": null,
11      "updated_at": null
12    }
13  ]
14 }

```

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

GET <http://localhost:8000/api/mahasiswa/10> Send Save

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (9) Test Results Status: 200 OK Time: 839ms Size: 297 B Save Response

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

```

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 berhasil ditambah";
45         $res['value'] = "$mhs";
46         return response($res);
47     }
48 }
49

```

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

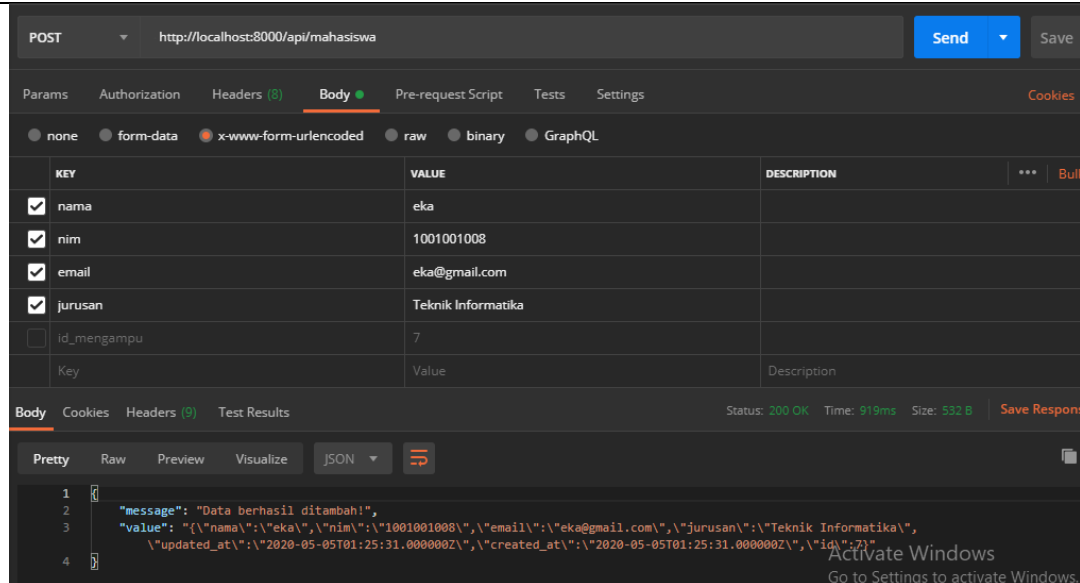
```

23 Route::post('/mahasiswa', 'MahasiswaController@create');

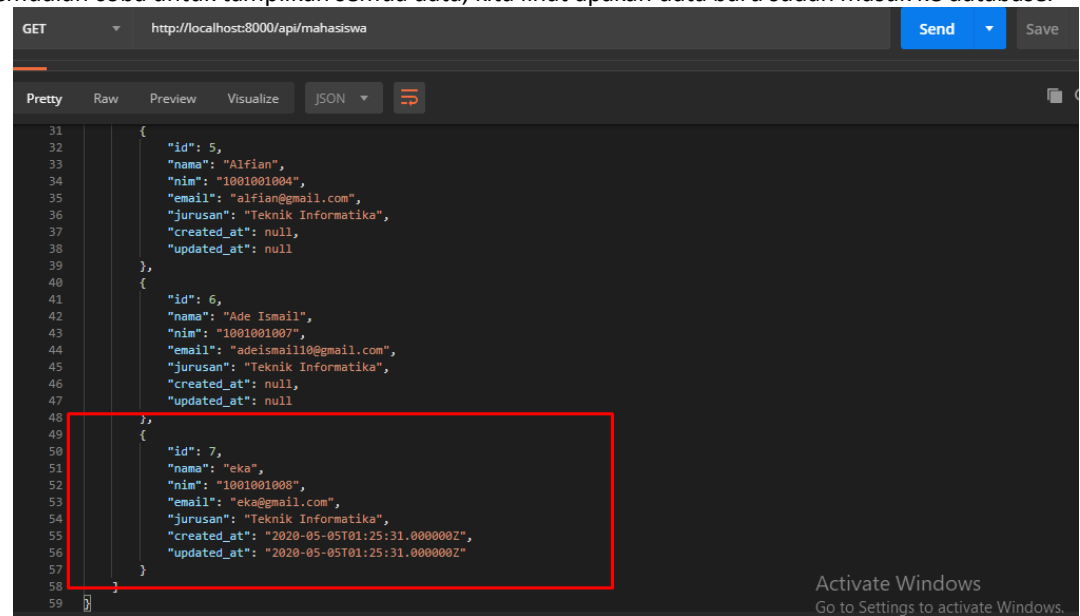
```

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

- 14 Kita coba untuk menambahkan data melalui Postman.



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.



```

52     public function update(Request $request, $id){
53         $nama = $request->nama;
54         $nim = $request->nim;
55         $email = $request->email;
56         $jurusan = $request->jurusan;
57
58         $mhs = Mahasiswa::find($id);
59         $mhs->nama = $nama;
60         $mhs->nim = $nim;
61         $mhs->email = $email;
62         $mhs->jurusan = $jurusan;
63
64         if($mhs->save()){
65             $res['message'] = "Data berhasil diubah";
66             $res['value'] = "$mhs";
67             return response($res);
68         }else{
69             $res['message'] = "Gagal!";
70             return response($res);
71         }
72     }
73 }
74

```

- 16 Tambahkan route untuk memanggil fungsi update pada routes/api.php

```

24 Route::put('/mahasiswa/update/{id}', 'MahasiswaController@update');

```

Karena kita ingin memasukkan perubahan data, maka perintah yang dipakai adalah 'put'.

- 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 x-www-form-urlencoded. Isikan nama kolom pada database pada KEY, untuk isian data yang diubah tuliskan pada VALUE.

PUT <http://localhost:8000/api/mahasiswa/update/2> Send Save

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

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

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> nama	wahyu afifah	
<input checked="" type="checkbox"/> nim	1001001002	
<input checked="" type="checkbox"/> email	wahyu@gmail.com	
<input checked="" type="checkbox"/> jurusan	Teknik Elektro	
<input type="checkbox"/> id_mengampu	7	
Key	Value	Description

**Body** Cookies Headers (9) Test Results Status: 200 OK Time: 1117ms Size: 509 B Save Response

Pretty Raw Preview Visualize JSON

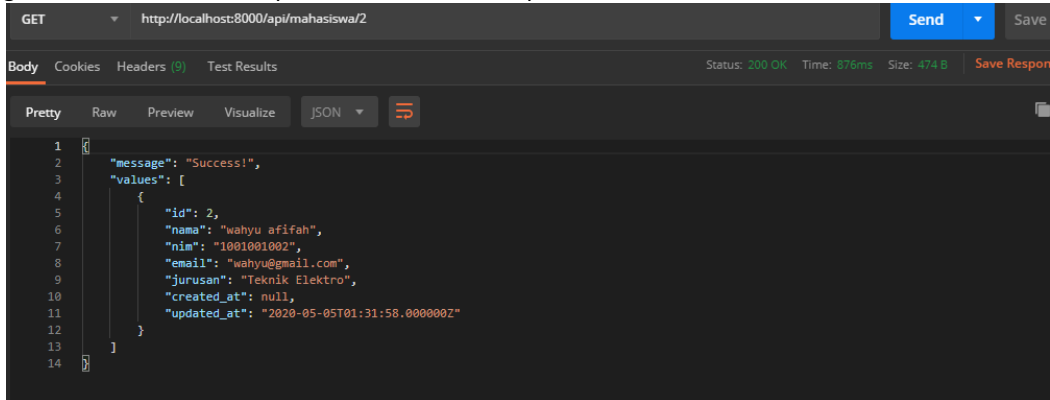
```

1 {
2   "message": "Data berhasil diubah",
3   "value": "{\n  \"id\":2,\n  \"nama\": \"wahyu afifah\",\n  \"nim\": \"1001001002\",\n  \"email\": \"wahyu@gmail.com\",\n  \"jurusan\": \"Teknik Elektro\",\n  \"created_at\": null,\n  \"updated_at\": \"2020-05-05T01:31:58.000000Z\"\n}"
4 }

```

Activate Windows  
Go to Settings to activate Windows.

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.

```
74 public function delete($id){
75     $mhs = Mahasiswa::where('id',$id);
76
77     if($mhs->delete()){
78         $res['message'] = "Data berhasil dihapus";
79         return response($res);
80     }else{
81         $res['message'] = "Gagal!";
82         return response($res);
83     }
84 }
85 }
```

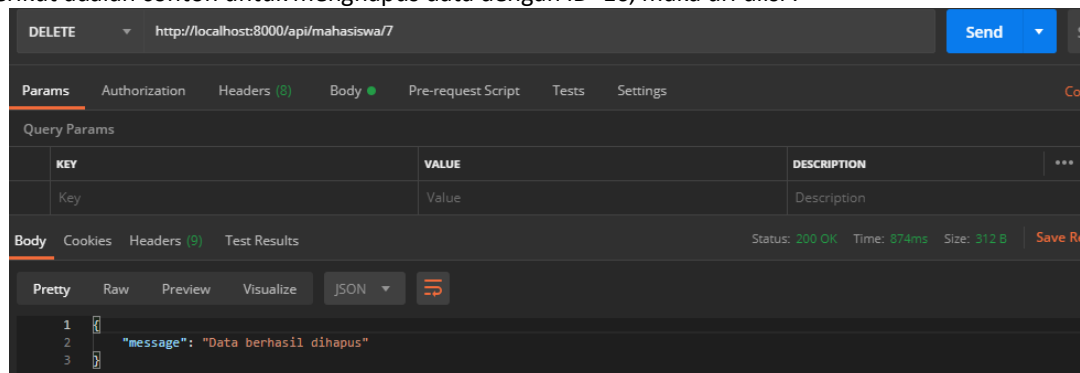
- 19 Tambahkan route untuk memanggil fungsi delete pada routes/api.php

```
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.

Berikut adalah contoh untuk menghapus data dengan ID=10, maka url diisi :



Muncul pesan berhasil ketika data terhapus dari database.