

**LAPORAN PRAKTIKUM
PEMROGRAMAN WEB LANJUT**

**Jobsheet-8
Implementasi Restful API**



Disusun oleh:
Livia Yurike Khuril Maula
D4 TI – 2A / 16
NIM : 1841720025

**POLITEKNIK NEGERI MALANG
JURUSAN TEKNOLOGI INFORMASI
PROGRAM STUDI D4 TEKNIK INFORMATIKA**

MARET 2020



Topik

Implementasi Restful API

Tujuan

Mahasiswa diharapkan dapat:

1. Memahami proses membaca data menggunakan Library Rest Server di Codeigniter.
2. Memahami proses menambah data menggunakan Library Rest Server di Codeigniter.
3. Memahami proses mengubah data menggunakan Library Rest Server di Codeigniter.
4. Memahami proses menghapus data menggunakan Library Rest Server di Codeigniter.

Praktikum – Bagian 1: Instalasi library restful codeigniter

Langkah	Keterangan
1	Silahkan unduh aplikasi Postman terlebih dahulu.
2	Pada praktikum kali ini kita akan menggunakan source codeigniter yang baru dan Library Rest Server. Untuk versi codeigniter yang digunakan adalah versi 3.1.9. sedangkan untuk Library Rest Server dapat di unduh pada link berikut :
3	<p>Extract codeigniter yang sudah di unduh dan letakkan pada folder htdocs. Beri nama Code-kedelapan dan coba akses di http://localhost/Code-kedelapan/. Pastikan tampilan seperti gambar dibawah ini:</p> 
4	<p>Selanjutnya lakukan konfigurasi dasar pada config.php seperti berikut:</p> <pre> 24 25 */ 26 \$config['base_url'] = ''; 27 28 /* </pre>

```

37  */
38  $config['index_page'] = 'index.php';
39
40  /*

```

Menjadi seperti berikut :

```

25  */
26  $config['base_url'] = 'http://localhost/rest_server/';
27
28  /*

```

```

37  */
38  $config['index_page'] = '';
39
40  /*

```

Dan file **autoload.php**

```

60  */
61  $autoload['libraries'] = array();
62
63  /*

```

```

91  */
92  $autoload['helper'] = array();
93
94  /*

```

Menjadi

```

60  */
61  $autoload['libraries'] = array('database');
62
63  /*

```

```

91  */
92  $autoload['helper'] = array('url');
93
94  /*

```

Dan database.php seperti berikut:

```

76  $db['default'] = array(
77      'dsn' => '',
78      'hostname' => 'localhost',
79      'username' => 'root',
80      'password' => 'mysql',
81      'database' => 'rest_server',
82      'dbdriver' => 'mysqli',

```

5

Selanjutnya buatlah database **rest_server** pada phpmyadmin dan masukkan data dummy. Untuk data dummy dapat dibuat sebagai berikut:

```

CREATE TABLE `mahasiswa` (
  `id` int(11) NOT NULL,
  `nrp` char(9) NOT NULL,
  `nama` varchar(250) NOT NULL,

```

```
`email` varchar(250) DEFAULT NULL,
`jurusan` varchar(64) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
INSERT INTO `mahasiswa` (`id`, `nrp`, `nama`, `email`, `jurusan`) VALUES
(1, '043040001', 'Doddy Ferdiansyah', 'doy@gmail.com', 'Teknik Mesin'),
(2, '023040123', 'Erik', 'erik@gmail.com', 'Teknik Industri'),
(3, '043040321', 'Rommy Fauzi', 'rommy@gmail.com', 'Teknik Planologi'),
(4, '033040023', 'Fajar Darmawan', 'fajar@yahoo.com', 'Teknik Informatika'),
(5, '113040321', 'Ferry Mulyanto', 'ferry@yahoo.com', 'Manajemen');
```

```
ALTER TABLE `mahasiswa`
ADD PRIMARY KEY (`id`);
```

```
ALTER TABLE `mahasiswa`
MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=33;
```

Hasil :

The screenshot shows the phpMyAdmin interface for a database named 'rest_server' on localhost. The 'mahasiswa' table is selected, and its structure is displayed. The table has 5 columns: id, nrp, nama, email, and jurusan. The 'id' column is the primary key and is set to AUTO_INCREMENT. The 'nrp' column is a character string of length 9. The 'nama', 'email', and 'jurusan' columns are variable-length character strings of lengths 250, 250, and 64 respectively, all using the latin1_swedish_ci collation.

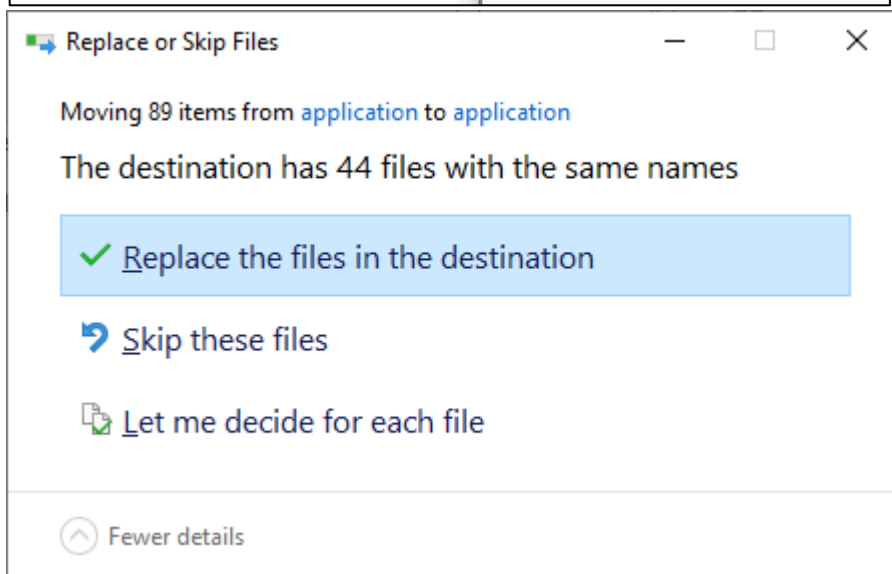
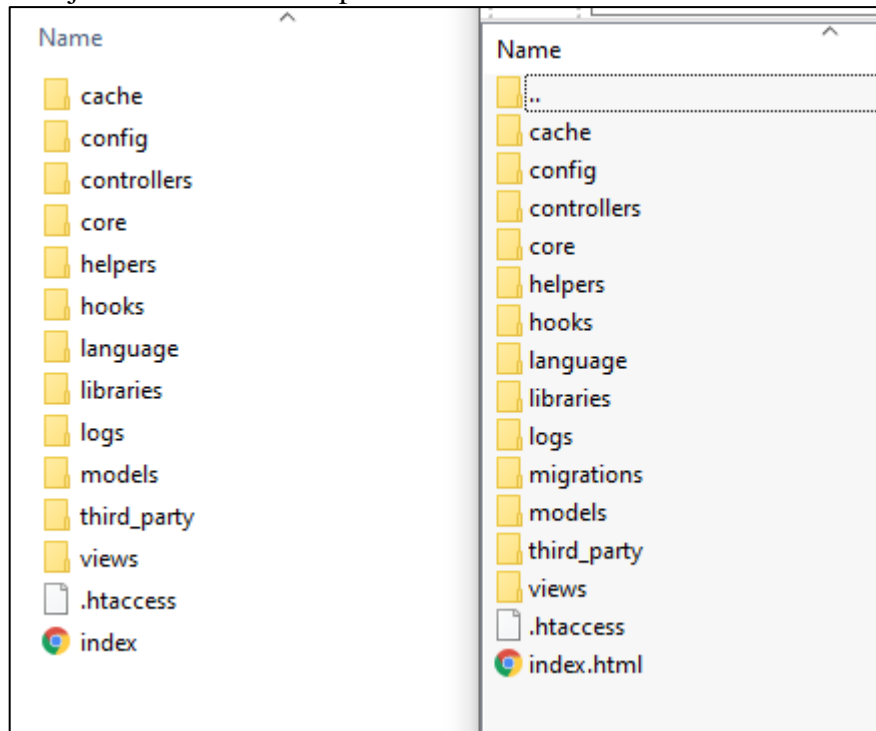
#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra	Tindakan
1	id	int(11)		Tidak	Tidak ada		AUTO_INCREMENT		Ubah Hapus Lainnya
2	nrp	char(9)	latin1_swedish_ci	Tidak	Tidak ada				Ubah Hapus Lainnya
3	nama	varchar(250)	latin1_swedish_ci	Tidak	Tidak ada				Ubah Hapus Lainnya
4	email	varchar(250)	latin1_swedish_ci	Ya	NULL				Ubah Hapus Lainnya
5	jurusan	varchar(64)	latin1_swedish_ci	Tidak	Tidak ada				Ubah Hapus Lainnya

Below the structure view, the data is displayed in a table with 7 columns: id, nrp, nama, email, jurusan, and two action buttons (Ubah and Salin). The data rows are as follows:

id	nrp	nama	email	jurusan
1	043040001	Doddy Ferdiansyah	doy@gmail.com	Teknik Mesin
2	023040123	Erik	erik@gmail.com	Teknik Industri
3	043040321	Romy Fauzi	rommy@gmail.com	Teknik Planologi
4	033040023	Fajar Darmawan	fajar@yahoo.com	Teknik Informatika
5	113040321	Ferry Mulyanto	ferry@yahoo.com	Manajemen

6

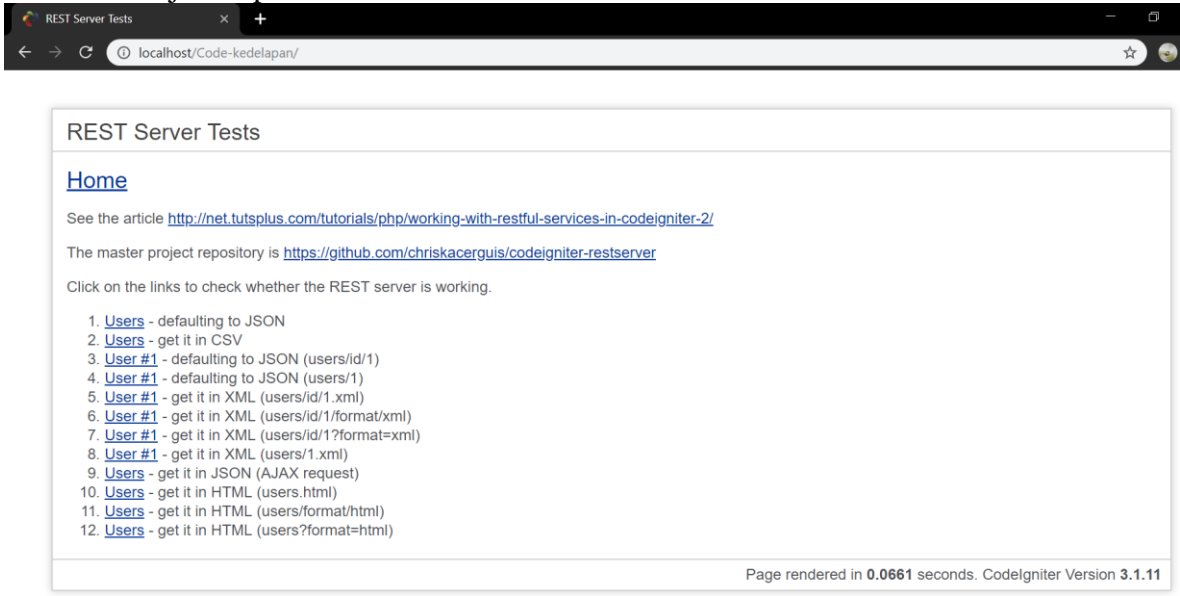
Selanjutnya extract file Library Rest Server, buka folder **application** dan duplikasi file dan folder tersebut kedalam folder **application** pada project **Code-kedelapan**. Klik **Replace** files jika ada notifikasi duplikat.



Setelah proses duplikasi file selesai, pastikan file-file yang kita sudah konfigurasi di awal tetap terkonfigurasi.

This PC > Windows-SSD (C:) > Program Files > Ampms > www > Code-kedelapan

Name	Date modified	Type	Size
application	23/03/2020 20:32	File folder	
system	19/09/2019 19:08	File folder	
user_guide	19/09/2019 19:08	File folder	
.editorconfig	19/09/2019 19:08	EDITORCONFIG File	1 KB
.gitignore	19/09/2019 19:08	Text Document	1 KB
composer.json	19/09/2019 19:08	JSON File	1 KB
contributing.md	19/09/2019 19:08	MD File	7 KB
index	19/09/2019 19:08	PHP File	11 KB
license	19/09/2019 19:08	Text Document	2 KB
readme.rst	19/09/2019 19:08	RST File	3 KB

7	<p>Selanjutnya buka project http://localhost/Code-kedelapan/. Jika berhasil, tampilan akan berubah menjadi seperti ini:</p> 
8	Tahap instalasi Library Rest Server selesai

Praktikum – Bagian 2: Membuat Aplikasi CRUD Restful Codeigniter

Langkah	Keterangan
1	<p>Buatlah file Mahasiswa.php pada controller/api. Selanjutnya tuliskan kode berikut:</p> <pre> application > controllers > api > Mahasiswa.php 1 2 3 defined('BASEPATH') or exit('No direct script access allowed'); 4 5 require APPPATH . '/libraries/REST_Controller.php'; 6 require APPPATH . '/libraries/Format.php'; 7 8 use Restserver\Libraries\REST_Controller; 9 10 class Mahasiswa extends REST_Controller 11 { 12 13 function __construct() 14 { 15 parent::__construct(); 16 \$this->load->model('Mahasiswa_model', 'mahasiswa'); 17 } 18 19 // GET 20 function index_get() 21 { 22 \$id = \$this->get('id'); 23 if (\$id === null) { 24 \$mahasiswa = \$this->mahasiswa->getMahasiswa(); 25 } else { 26 \$mahasiswa = \$this->mahasiswa->getMahasiswa(\$id); 27 } 28 if (\$mahasiswa) { 29 \$this->response([30 'status' => true, 31 'data' => \$mahasiswa 32], REST_Controller::HTTP_OK); 33 } else { 34 \$this->response([35 'status' => false, 36 'message' => 'id not found' 37], REST_Controller::HTTP_NOT_FOUND); 38 } 39 } 40 </pre>

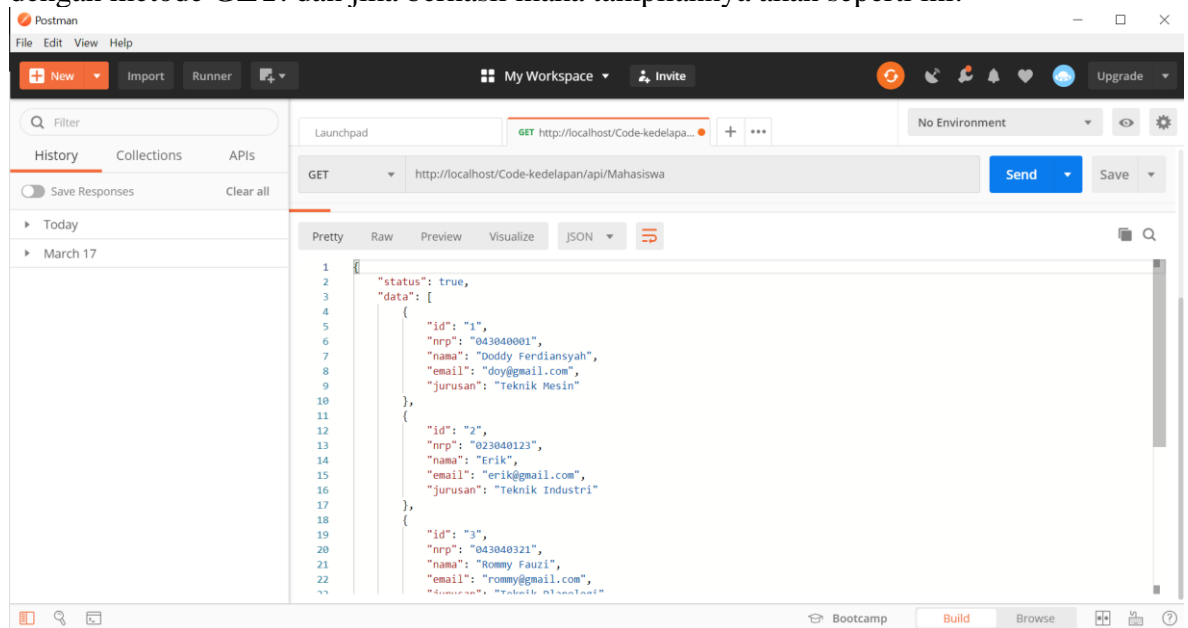
2

Selanjutnya Buatlah file **Mahasiswa_model.php** pada model. Selanjutnya tuliskan kode berikut:

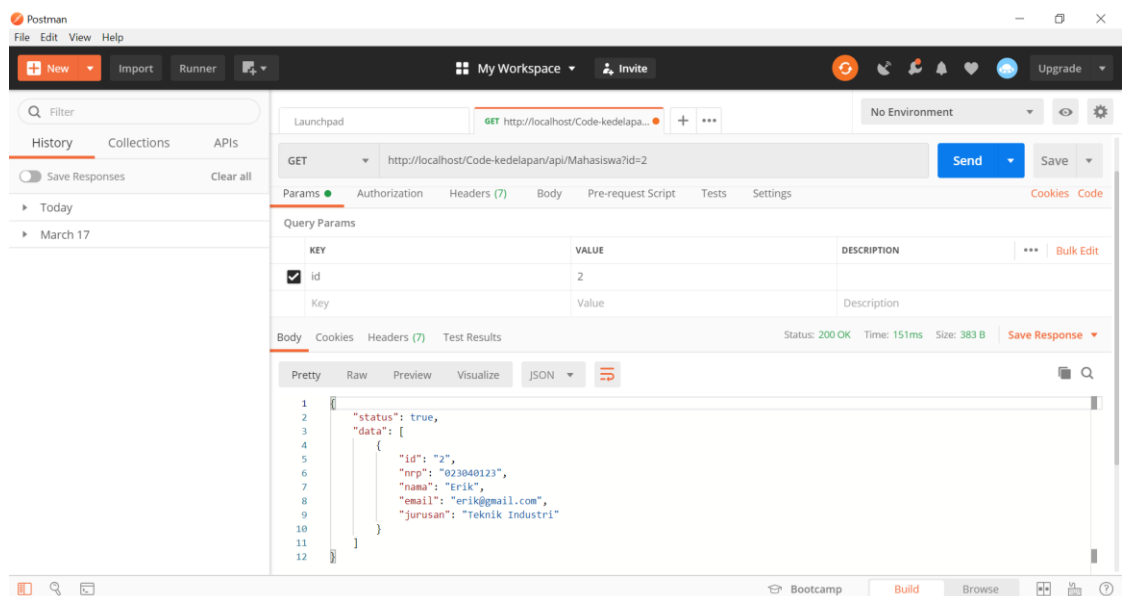
```
application > models > Mahasiswa_model.php
1  <?php
2
3  class Mahasiswa_model extends CI_Model
4  {
5      public function getMahasiswa($id = null)
6      {
7          if($id === null) {
8              return $this->db->get('mahasiswa')->result_array();
9          } else {
10             return $this->db->get_where('mahasiswa', ['id'=> $id])->result_array();
11         }
12     }
13 }
14
15 /* End of file Mahasiswa_model.php */
16
```

3

Buka aplikasi postman, masukkan alamat <http://localhost/Code-kedelapan/api/mahasiswa> dengan metode **GET**. dan jika berhasil maka tampilannya akan seperti ini:



Kita juga dapat melakukan pencarian berdasarkan id, hasilnya akan seperti ini:



4

Untuk fungsi lain **Create**, **Update** dan **Delete**. Lanjutkan kode berikut pada file **mahasiswa.php**:

```

application > controllers > api > 🐛 Mahasiswa.php
41 // DELETE
42 function index_delete()
43 {
44     $id = $this->delete('id');
45     if ($id === null) {
46         $this->response([
47             'status' => false,
48             'message' => 'provide an id!'
49         ], REST_Controller::HTTP_BAD_REQUEST);
50     } else {
51         if ($this->mahasiswa->deleteMahasiswa($id) > 0) {
52             //ok
53             $this->response([
54                 'status' => true,
55                 'id' => $id,
56                 'message' => 'deleted.'
57             ], REST_Controller::HTTP_OK);
58         } else {
59             //id not found
60             $this->response([
61                 'status' => false,
62                 'message' => 'provide an id!'
63             ], REST_Controller::HTTP_BAD_REQUEST);
64         }
65     }
66 }
67 }
68

```

```

application > controllers > api > 🐛 Mahasiswa.php
69 // POST
70 public function index_post()
71 {
72     $data = [
73         'nrp' => $this->post('nrp'),
74         'nama' => $this->post('nama'),
75         'email' => $this->post('email'),
76         'jurusan' => $this->post('jurusan'),
77     ];
78
79     if ($this->mahasiswa->createMahasiswa($data) > 0) {
80         $this->response([
81             'status' => true,
82             'message' => 'new mahasiswa has been created'
83         ], REST_Controller::HTTP_CREATED);
84     } else {
85         //id not found
86         $this->response([
87             'status' => false,
88             'message' => 'failed to create new data!'
89         ], REST_Controller::HTTP_BAD_REQUEST);
90     }
91 }
92

```



```

application > controllers > api > Mahasiswa.php
95     public function index_put()
96     {
97         $id = $this->put('id');
98         $data = [
99             'nrp' => $this->put('nrp'),
100            'nama' => $this->put('nama'),
101            'email' => $this->put('email'),
102            'jurusan' => $this->put('jurusan'),
103        ];
104
105        if ($this->mahasiswa->updateMahasiswa($data, $id) > 0) {
106            $this->response([
107                'status' => true,
108                'message' => 'data mahasiswa has been updated'
109            ], REST_Controller::HTTP_OK);
110        } else {
111            //id not found
112            $this->response([
113                'status' => false,
114                'message' => 'failed to update data!'
115            ], REST_Controller::HTTP_BAD_REQUEST);
116        }
117    }
118 }
119
120
121 /* End of file Controllername.php */
122

```

5

Dan lanjutkan kode pada file **Mahasiswa_model.php**:

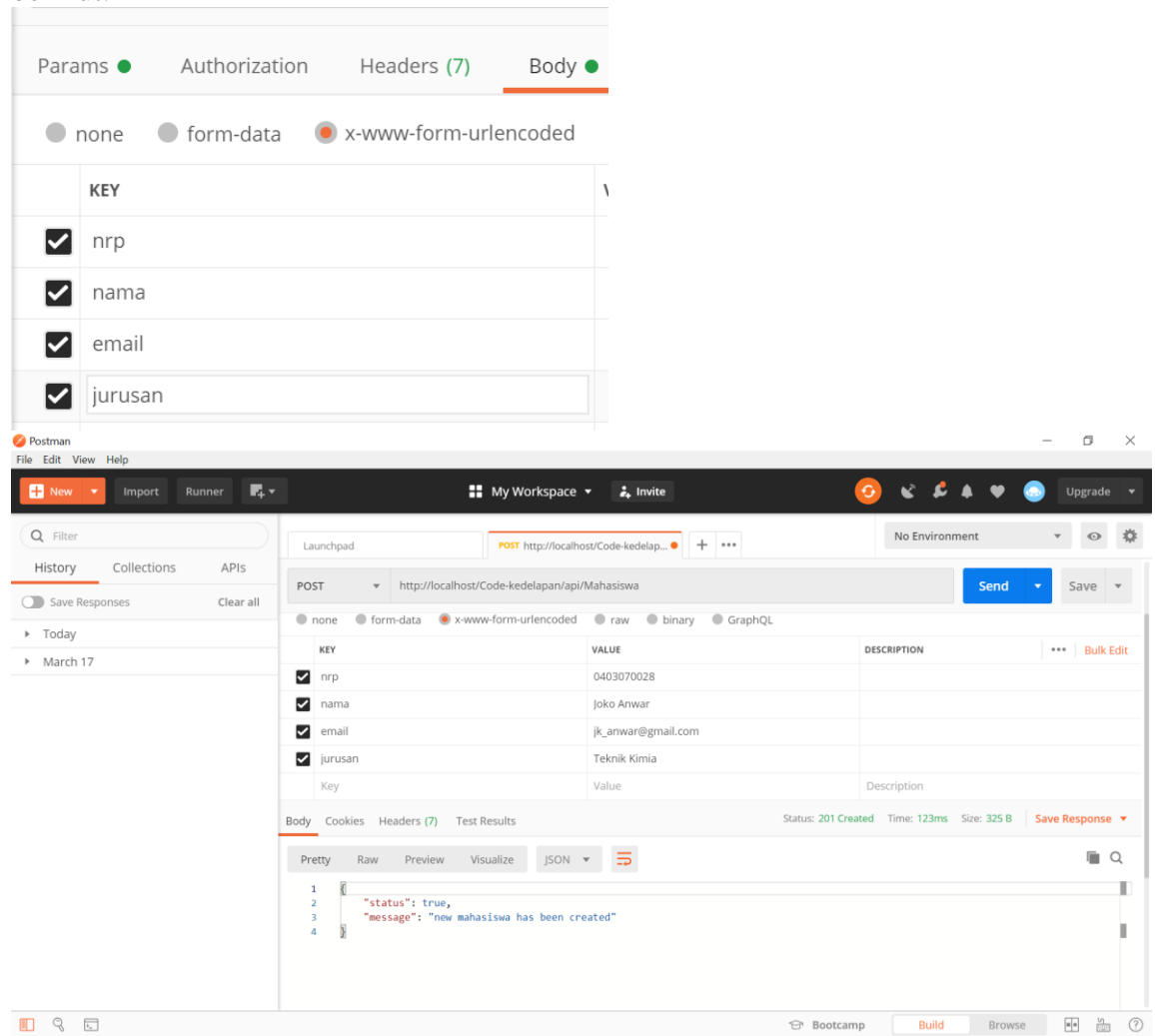
```

application > models > Mahasiswa_model.php
1  <?php
2
3  class Mahasiswa_model extends CI_Model
4  {
5      public function getMahasiswa($id = null)
6      {
7          if($id === null) {
8              return $this->db->get('mahasiswa')->result_array();
9          } else {
10             return $this->db->get_where('mahasiswa', ['id' => $id])->result_array();
11         }
12     }
13
14     public function deleteMahasiswa($id)
15     {
16         $this->db->delete('mahasiswa', ['id' => $id]);
17         return $this->db->affected_rows();
18     }
19
20     public function createMahasiswa($data)
21     {
22         $this->db->insert('mahasiswa', $data);
23         return $this->db->affected_rows();
24     }
25
26     public function updateMahasiswa($data, $id)
27     {
28         $this->db->update('mahasiswa', $data, ['id' => $id]);
29         return $this->db->affected_rows();
30     }
31 }
32
33 /* End of file Mahasiswa_model.php */
34

```

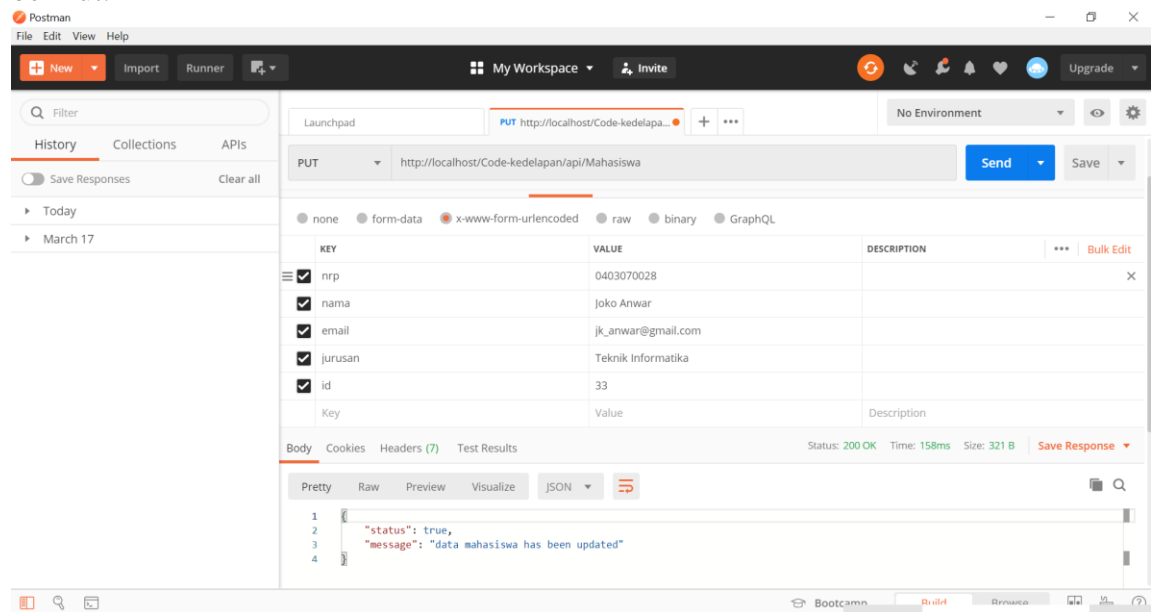
6

Selanjutnya lakukan operasi **Create** data pada aplikasi postman, untuk operasi **Create**, **Update** dan **Delete**, arahkan parameter ke **Body** dan **x-www-form-urlencoded** seperti berikut:



7

Lakukan juga operasi **Update** data pada aplikasi postman, masukkan parameter id seperti berikut:



8

Lakukan juga operasi **Delete** data pada aplikasi postman, parameter yang dimasukkan cukup id saja, seperti berikut:

