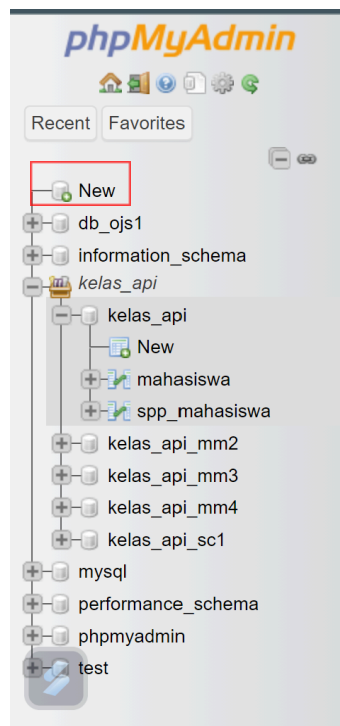


Konfigurasi Database

Sebelum membuat sebuah rest API langkah 1 adalah dengan membuat database dan melakukan export data yang ada di LMS dengan format `".sql"`.

Langkah 1. Create Database

Untuk membuat database dapat dengan menggunakan terminal dengan mengetikan `"CREATE DATABASE nama_database"` atau dengan klik **"New"** seperti gambar 1 kemudian ketikan **"nama Database"** kemudian klik **"Create"**



Gambar 1. Struktur Database

Langkah 2. Import Database

Setelah anda membuat database kemudian masuk ke database pada localhost dengan klik nama_database yang ada di panel kiri kemudian pilih tab **"Import"**. Pada **"File to import:"** cari file `.sql` untuk di import, pada **"Format"** pilih `sql`, kemudian klik tombol **"import"**. Hasil Import kemudian akan

ada 2 tabel yaitu **mahasiswa** dan **spp_mahasiswa** seperti pada gambar 2.

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> mahasiswa	★ Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> spp_mahasiswa	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
2 tables	Sum	12	InnoDB	utf8mb4_general_ci	32.0 KiB	0 B

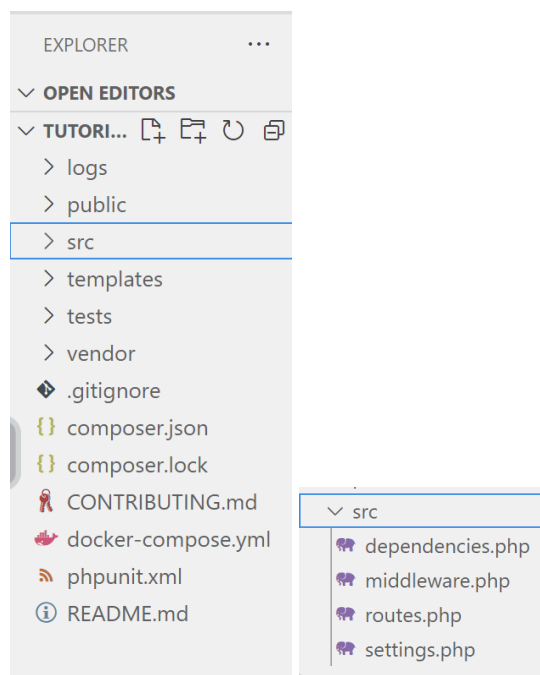
Gambar 2. Hasil Export

Konfigurasi Rest API

Rest API dibuat menggunakan bahasa pemrograman php dengan menggunakan framework Slim. Framework Slim sudah install menggunakan composer dan hasil install dapat di download pada laman <https://github.com/zarkuz/restapislim>.

Langkah 1 Konfigurasi db

Untuk konfigurasi database pertama adalah dengan masuk pada folder **src->settings.php** seperti pada gambar 3.



Gambar 3. Lokasi "settings.php"

Kemudian setelah masuk ubah host, user, pass, db, dan driver sesuai dengan konfigurasi pada server(contoh menggunakan XAMPP) yang anda gunakan. Posisi script Konfigurasi Database dapat dilihat seperti pada gambar 4.



```
1 <?php
2 return [
3     'settings' => [
4         'displayErrorDetails' => true, // set to false in production
5         'addContentLengthHeader' => false, // Allow the web server to send the content-length header
6
7         // Renderer settings
8         'renderer' => [
9             'template_path' => __DIR__ . '/../templates/',
10        ],
11
12        // Monolog settings
13        'logger' => [
14            'name' => 'slim-app',
15            'path' => isset($_ENV['docker']) ? 'php://stdout' : __DIR__ . '/../logs/app.log',
16            'level' => \Monolog\Logger::DEBUG,
17        ],
18
19        'db' => [
20            'host' => 'localhost',
21            'user' => 'root',
22            'pass' => '',
23            'dbname' => 'kelas_api',
24            'driver' => 'mysql'
25        ],
26    ],
27 ];
28
```

Gambar 4. Konfigurasi Database

Langkah 2 Konfigurasi API

Setelah semua setting selesai maka bisa dicek apakah framework sudah dapat berjalan dengan membuka pada browser dengan link "**Link_disini/public**" hingga tampil seperti gambar 5.



Gambar 5. Tampilan Framework Slim

Setelah framework sudah berjalan kemudian dapat masuk ke konfigurasi Rest API dengan masuk pada **src->routes.php** kemudian edit pada bagian yang ada seperti pada gambar 6 yang diberi kotak merah



```
src > routes.php > ...
1  <?php
2
3  use Slim\App;
4  use Slim\Http\Request;
5  use Slim\Http\Response;
6
7  return function (App $app) {
8      $container = $app->getContainer();
9
10     $app->get('/[{name}]', function (Request $request, Response $response, array $args) use ($container) {
11         // Sample log message
12         $container->get('logger')->info("Slim-Skeleton '/' route");
13
14         // Render index view
15         return $container->get('renderer')->render($response, 'index.phtml', $args);
16     });
17
18
19     //kode disini
20
21     //kode disini
22
23
24
25 };
26
```

Gambar 6. Lokasi Script Setting Rest API

Skenario 1 Melihat data

Skenario 1 Melihat data dengan method GET dibuat untuk melihat data dengan data berasal dari table mahasiswa join dengan spp_mahasiswa. Kemudian Script yang ditambahkan adalah sebagai berikut:

```
$app->get('/mahasiswa/pembayaran', function(Request $request, Response $response){
    //disini dimasukan script sql
    $sql = "SELECT mahasiswa.nim, nama,
        CASE
            WHEN jk='L' THEN 'Laki-Laki'
            WHEN jk='P' THEN 'Perempuan'
        END as jenis_kelamin
        , umur, ifnull(semester,0) as semester, ifnull(jumlah,0) as jumlah_spp
        FROM mahasiswa LEFT JOIN spp_mahasiswa
        ON mahasiswa.nim = spp_mahasiswa.nim WHERE mahasiswa.nim<>0";

    $jk = $request->getQueryParam("jk");
    $umur = $request->getQueryParam("umur");

    if($jk<>null){
        $sql = $sql." AND jk='$jk'";
    }
});
```

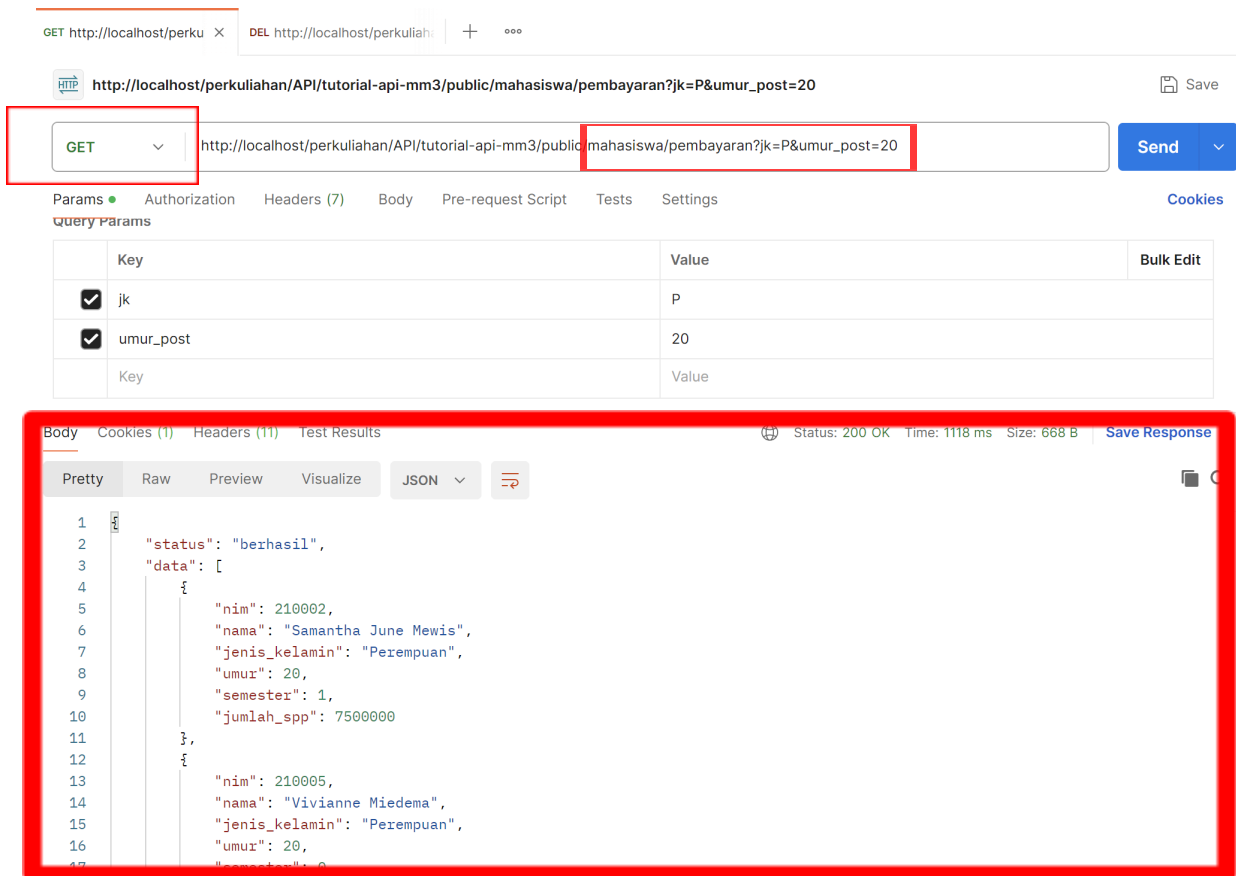
```

if ($umur <> null) {
    $sql = $sql." AND umur=$umur";
};

$stmt = $this->db->prepare($sql);
$stmt->execute();
$result = $stmt->fetchAll();
return $response->withJson(["status"=>"success", "data"=>$result],200);
});

```

Setelah menambahkan untuk mengecek dapat menggunakan postman dengan link sama seperti sebelumnya kemudian ditambahkan **"/mahasiswa/pembayaran"** sesuai dengan yang dideklarasikan pada script. Untuk menjalankan dilakukan dengan method GET dengan memasukkan link diatas dengan parameter yang dapat digunakan adalah umur dan jk. Hasil method GET dapat dilihat pada gambar 7.



Gambar 7. Hasil method GET

Skenario 2 Menambahkan data dengan method POST

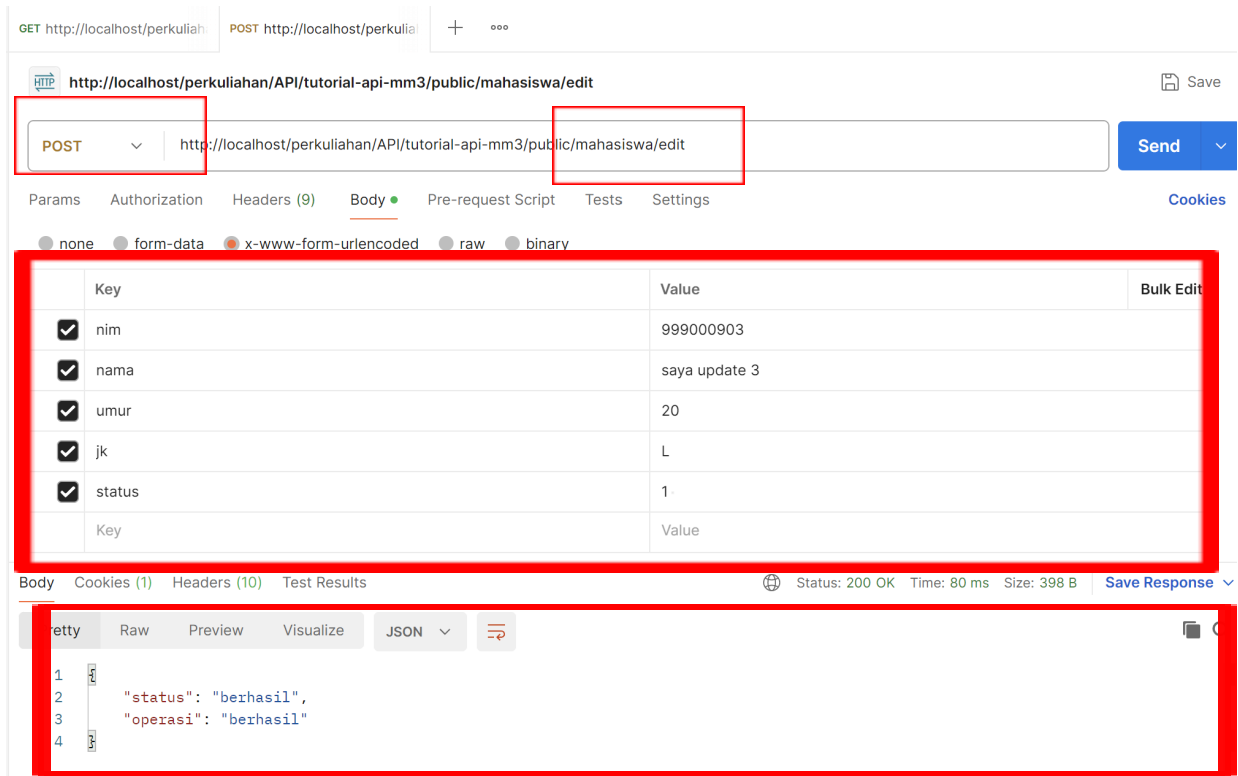
Skenario 2 Menambahkan data dengan method POST dibuat untuk menambahkan/insert data pada tabel mahasiswa. Kemudian Script yang ditambahkan adalah sebagai berikut:

```
$app->post('/mahasiswa/edit', function(Request $request, Response $response){
    $add_mahasiswa = $request->getParsedBody();
    $nim = $add_mahasiswa['nim'];
    $jk = $add_mahasiswa['jk'];
    $nama = $add_mahasiswa['nama'];
    $umur = $add_mahasiswa['umur'];

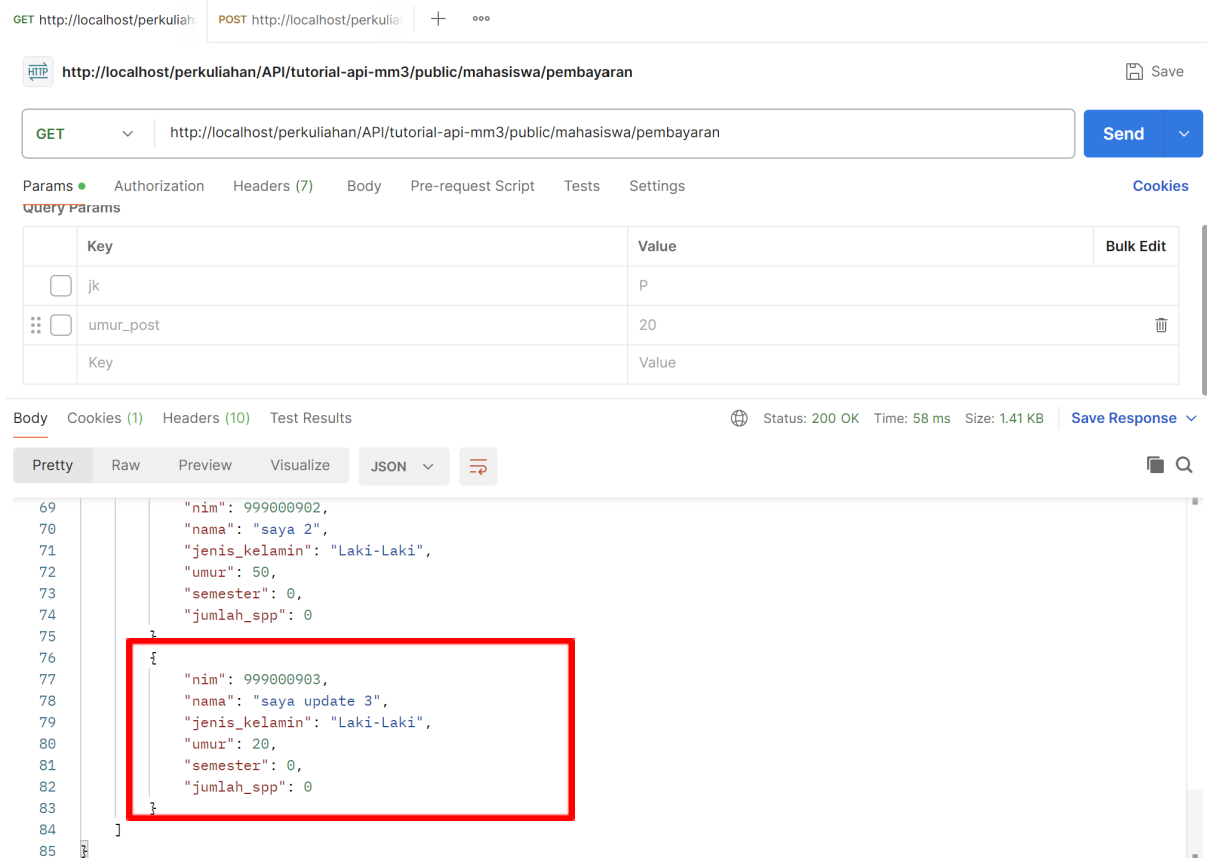
    //disini dimasukan script sql
    $sql = "INSERT INTO mahasiswa(nim, nama, jk, umur) VALUES
        ($nim, '$nama', '$jk', $umur)";

    $stmt = $this->db->prepare($sql);
    $stmt->execute();
    return $response->withJson(["status"=>"success", "data"=>"1"],200);
});
```

Setelah menambahkan untuk mengecek dapat menggunakan postman dengan link sama seperti sebelumnya kemudian ditambahkan **"/mahasiswa/edit"** sesuai dengan yang dideklarasikan pada script. Untuk menjalankan dilakukan dengan method POST dengan memasukan link diatas dengan menambahkan data yang diinput pada Body -> x-www-form-urlencoded. Hasil method POST dapat dilihat pada gambar 8 kemudian dapat di kroscek kembali pada data untuk melihat dapat apakah sudah bertambah atau belum pada gambar 9.



Gambar 8. Hasil Method Post



Gambar 9. Hasil Method GET (Melihat Hasil Penambahan Data)

Skenario 3 Mengubah data dengan method PUT

Skenario 3 Mengubah data dengan method PUT dibuat untuk mengubah/update data pada tabel mahasiswa. Kemudian Script yang ditambahkan adalah sebagai berikut:

```
$app->put('/mahasiswa/edit', function(Request $request, Response $response){
    $add_mahasiswa = $request->getParsedBody();
    $nim = $add_mahasiswa['nim'];
    $jk = $add_mahasiswa['jk'];
    $nama = $add_mahasiswa['nama'];
    $umur = $add_mahasiswa['umur'];

    $textupdate= "";
    if($nama<>null){
        $textupdate = $textupdate.", nama='$nama'";
    };
    if($jk<>null){
        $textupdate = $textupdate.", jk='$jk'";
    };
    if($umur<>null){
        $textupdate = $textupdate.", umur=$umur";
    };
    //disini dimasukan script sql
    $sql = "UPDATE mahasiswa SET nim=$nim".$textupdate." WHERE nim=$nim";

    $stmt = $this->db->prepare($sql);
    $stmt->execute();
    return $response->withJson(["status"=>"success", "data"=>"1"],200);
});
```

Setelah menambahkan untuk mengecek dapat menggunakan postman dengan link sama seperti sebelumnya kemudian ditambahkan **"/mahasiswa/edit"** sesuai dengan yang dideklarasikan pada script. Untuk menjalankan dilakukan dengan method PUT dengan memasukan link diatas dengan menambahkan data yang diinput pada Body -> x-www-form-urlencoded dengan acuan data nim merupakan data nim yang akan di update. Hasil method PUT dapat dilihat pada gambar 10 kemudian dapat di kroscek kembali pada data untuk melihat dapat apakah sudah berubah atau belum pada gambar 11.

GET http://localhost/perkuliah PUT http://localhost/perkuliah + ...

HTTP http://localhost/perkuliah/API/tutorial-api-mm3/public/mahasiswa/edit Save

PUT http://localhost/perkuliah/API/tutorial-api-mm3/public/mahasiswa/edit Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary

Key	Value	Bulk Edit
<input checked="" type="checkbox"/> nim	999000903	
<input checked="" type="checkbox"/> nama	nama berubah	
<input checked="" type="checkbox"/> umur	30	
<input checked="" type="checkbox"/> jk	P	
<input checked="" type="checkbox"/> status	1	
Key	Value	

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 93 ms Size: 398 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "status": "berhasil",
3   "operasi": "berhasil"
4 }
```

Gambar 10. Hasil Method PUT

GET http://localhost/perkuliah PUT http://localhost/perkuliah + ...

HTTP http://localhost/perkuliah/API/tutorial-api-mm3/public/mahasiswa/pembayaran Save

GET http://localhost/perkuliah/API/tutorial-api-mm3/public/mahasiswa/pembayaran Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Bulk Edit
<input type="checkbox"/> jk	P	
<input type="checkbox"/> umur_post	20	
Key	Value	

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 94 ms Size: 1.41 KB Save Response

Pretty Raw Preview Visualize JSON

```
69 {
70   "nim": 999000902,
71   "nama": "saya 2",
72   "jenis_kelamin": "Laki-Laki",
73   "umur": 50,
74   "semester": 0,
75   "jumlah_spp": 0
76 },
77 {
78   "nim": 999000903,
79   "nama": "nama berubah",
80   "jenis_kelamin": "Perempuan",
81   "umur": 30,
82   "semester": 0,
83   "jumlah_spp": 0
84 }
85 ]
```

Gambar 11. Hasil Method GET (Melihat Hasil Perubahan Data)

Skenario 4 Menghapus data dengan method DELETE

Skenario 4 Menghapus data dengan method DELETE dibuat untuk menghapus/delete data pada tabel mahasiswa. Kemudian Script yang ditambahkan adalah sebagai berikut:

```
$app->delete('/mahasiswa/edit', function(Request $request, Response $response){
    $add_mahasiswa = $request->getParsedBody();
    $nim = $add_mahasiswa['nim'];

    //disini dimasukan script sql
    $sql = "DELETE FROM mahasiswa WHERE nim=$nim";

    $stmt = $this->db->prepare($sql);
    $stmt->execute();
    return $response->withJson(["status"=>"success", "data"=>"1"],200);
});
```

Setelah menambahkan untuk mengecek dapat menggunakan postman dengan link sama seperti sebelumnya kemudian ditambahkan **"/mahasiswa/edit"** sesuai dengan yang dideklarasikan pada script. Untuk pengujian dilakukan dengan method DELETE dengan memasukan link diatas dengan menambahkan data yang diinput pada Body -> x-www-form-urlencoded dengan acuan data nim untuk data yang akan dihapus. Hasil pengujian dapat dilihat pada gambar 12 kemudian dapat di kroscek kembali pada data untuk melihat dapat apakah sudah dihapus atau belum pada gambar 13.

GET http://localhost/perkuliah: DEL http://localhost/perkuliah: + ...

HTTP http://localhost/perkuliah/API/tutorial-api-mm3/public/mahasiswa/edit Save

DELETE http://localhost/perkuliah/API/tutorial-api-mm3/public/mahasiswa/edit Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary

Key	Value	Bulk Edit
<input checked="" type="checkbox"/> nim	999000903	
<input type="checkbox"/> nama	nama berubah	
<input type="checkbox"/> umur	30	
<input type="checkbox"/> jk	P	
<input type="checkbox"/> status	1	
Key	Value	

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 80 ms Size: 398 B Save Response

Pretty Raw Preview Visualize JSON

```
1
2  "status": "berhasil",
3  "operasi": "berhasil"
4
```

Gambar 12. Hasil Method DELETE

GET http://localhost/perkuliah: DEL http://localhost/perkuliah: + ...

HTTP http://localhost/perkuliah/API/tutorial-api-mm3/public/mahasiswa/pembayaran Save

GET http://localhost/perkuliah/API/tutorial-api-mm3/public/mahasiswa/pembayaran Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Bulk Edit
<input type="checkbox"/> jk	P	
<input type="checkbox"/> umur_post	20	
Key	Value	

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 109 ms Size: 1.3 KB Save Response

Pretty Raw Preview Visualize JSON

```
61  {
62    "nim": 999000901,
63    "nama": "saya",
64    "jenis_kelamin": "Laki-Laki",
65    "umur": 20,
66    "semester": 0,
67    "jumlah_spp": 0
68  },
69  {
70    "nim": 999000902,
71    "nama": "saya 2",
72    "jenis_kelamin": "Laki-Laki",
73    "umur": 50,
74    "semester": 0,
75    "jumlah_spp": 0
76  }
77 ]
```

Gambar 13. Hasil Method DELETE (Melihat Hasil Penghapusan Data)