

Jobsheet-10: Membuat CRUD menggunakan Laravel

Mata Kuliah Pemrograman Web Lanjut



Oleh :

ALAN PERDHANA TIMOR / NIM : 1841720187

KELAS TI 2A

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

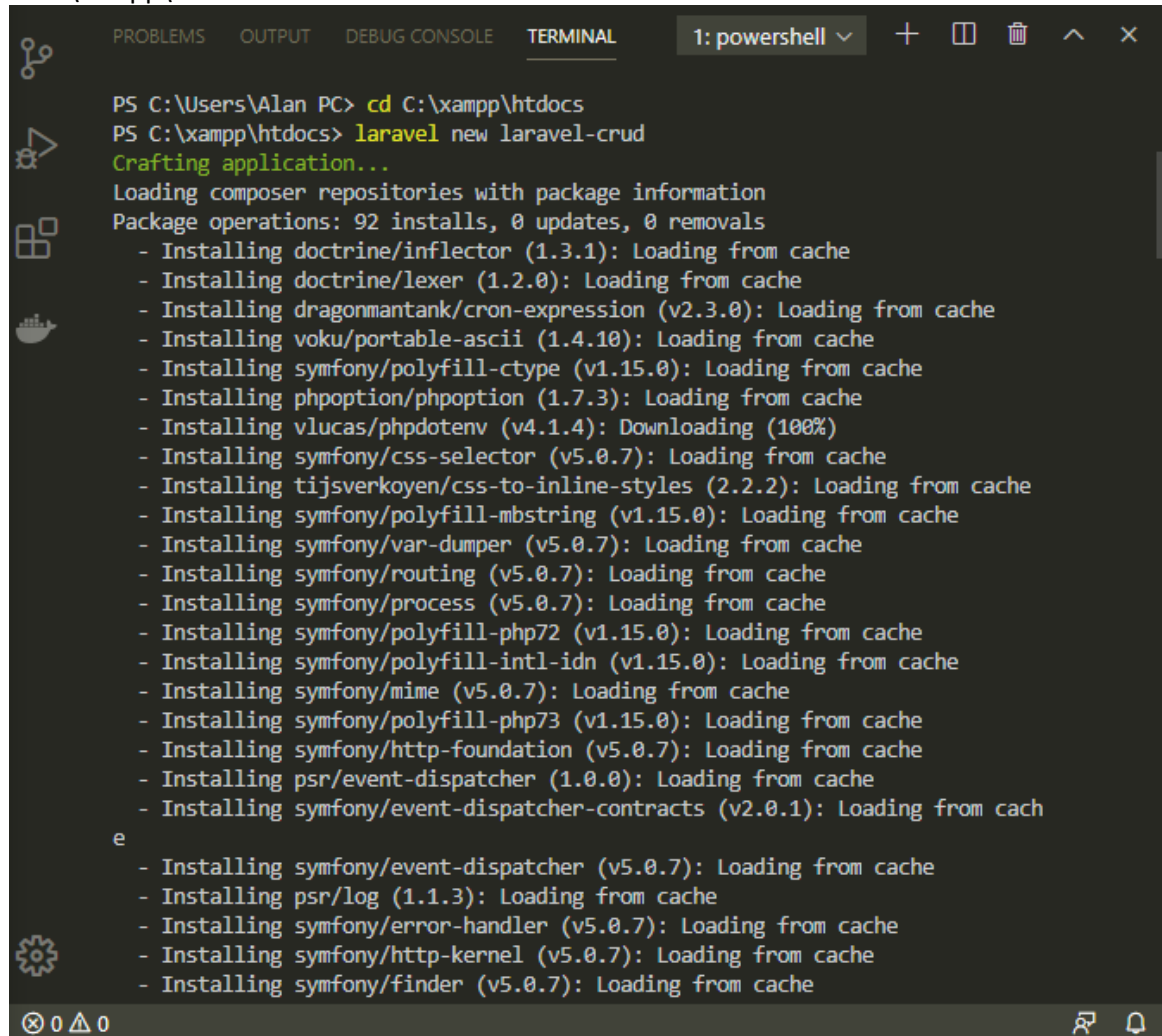
POLITEKNIK NEGERI MALANG

2020

Praktikum – Bagian 1: Membuat CRUD di Laravel menggunakan Query Builder

1. Buatlah project Laravel baru dengan nama laravel-crud. Buka command prompt, tuliskan perintah berikut.

```
cd C:\xampp\htdocs laravel new laravel-crud
```



```
PS C:\Users\Alan PC> cd C:\xampp\htdocs
PS C:\xampp\htdocs> laravel new laravel-crud
Crafting application...
Loading composer repositories with package information
Package operations: 92 installs, 0 updates, 0 removals
- Installing doctrine/inflector (1.3.1): Loading from cache
- Installing doctrine/lexer (1.2.0): Loading from cache
- Installing dragonmantank/cron-expression (v2.3.0): Loading from cache
- Installing voku/portable-ascii (1.4.10): Loading from cache
- Installing symfony/polyfill-ctype (v1.15.0): Loading from cache
- Installing phoption/phoption (1.7.3): Loading from cache
- Installing vlucas/phpdotenv (v4.1.4): Downloading (100%)
- Installing symfony/css-selector (v5.0.7): Loading from cache
- Installing tijsverkoyen/css-to-inline-styles (2.2.2): Loading from cache
- Installing symfony/polyfill-mbstring (v1.15.0): Loading from cache
- Installing symfony/var-dumper (v5.0.7): Loading from cache
- Installing symfony/routing (v5.0.7): Loading from cache
- Installing symfony/process (v5.0.7): Loading from cache
- Installing symfony/polyfill-php72 (v1.15.0): Loading from cache
- Installing symfony/polyfill-intl-idn (v1.15.0): Loading from cache
- Installing symfony/mime (v5.0.7): Loading from cache
- Installing symfony/polyfill-php73 (v1.15.0): Loading from cache
- Installing symfony/http-foundation (v5.0.7): Loading from cache
- Installing psr/event-dispatcher (1.0.0): Loading from cache
- Installing symfony/event-dispatcher-contracts (v2.0.1): Loading from cache
- Installing symfony/event-dispatcher (v5.0.7): Loading from cache
- Installing psr/log (1.1.3): Loading from cache
- Installing symfony/error-handler (v5.0.7): Loading from cache
- Installing symfony/http-kernel (v5.0.7): Loading from cache
- Installing symfony/finder (v5.0.7): Loading from cache
```

Selanjutnya kita lakukan konfigurasi database di Laravel. Untuk melakukan konfigurasi database, bukalah file .env pada project laravel-crud. Ubah seperti di bawah ini.

```
.env
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:1LBhLgroZ20s4/1Uzm4o1C7ngBw0K2yYS5DTfAG3I3Q=
APP_DEBUG=true
APP_URL=http://localhost

LOG_CHANNEL=stack

DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=latihan_laravel
DB_USERNAME=root
DB_PASSWORD=
```

Jalankan xampp, selanjutnya buat tabel dengan nama mahasiswa di mysql pada database latihan_laravel.

Server: 127.0.0.1 » Database: latihan_laravel » Table: mahasiswa

Table structure

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	nama	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
3	nim	varchar(10)	latin1_swedish_ci		No	None			Change Drop More
4	email	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
5	jurusan	varchar(20)	latin1_swedish_ci		No	None			Change Drop More

Check all With selected: [Browse](#) [Change](#) [Drop](#) [Primary](#) [Unique](#) [Index](#) [Fulltext](#) [Add to central columns](#) [Remove from central columns](#)

Isilah beberapa data pada tabel mahasiswa tersebut.

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

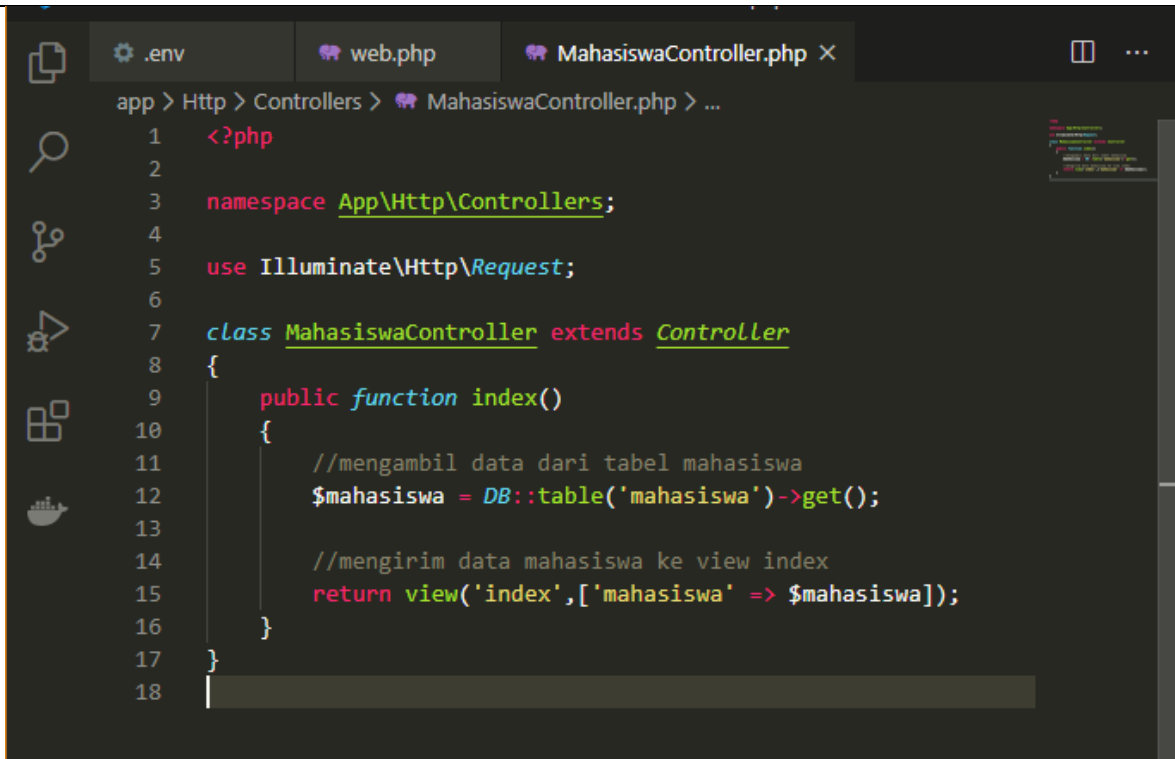
+ Options

				id	nama	nim	email	jurusan
<input type="checkbox"/>	Edit	Copy	Delete	1	Ayuna	1001001001	ayuna@hololive.com	Teknik Informatika
<input type="checkbox"/>	Edit	Copy	Delete	2	Wahyu	1001001002	Wahyu@email.com	Teknik Elektro

Check all With selected: [Edit](#) [Copy](#) [Delete](#) [Export](#)

Setelah kita memiliki beberapa data pada tabel mahasiswa, kita akan mencoba untuk menampilkan data tersebut ketika project dijalankan. Pertama, buatlah route pada routes/web.php sehingga ketika pertama kali project dijalankan akan terbuka halaman yang menampilkan data.

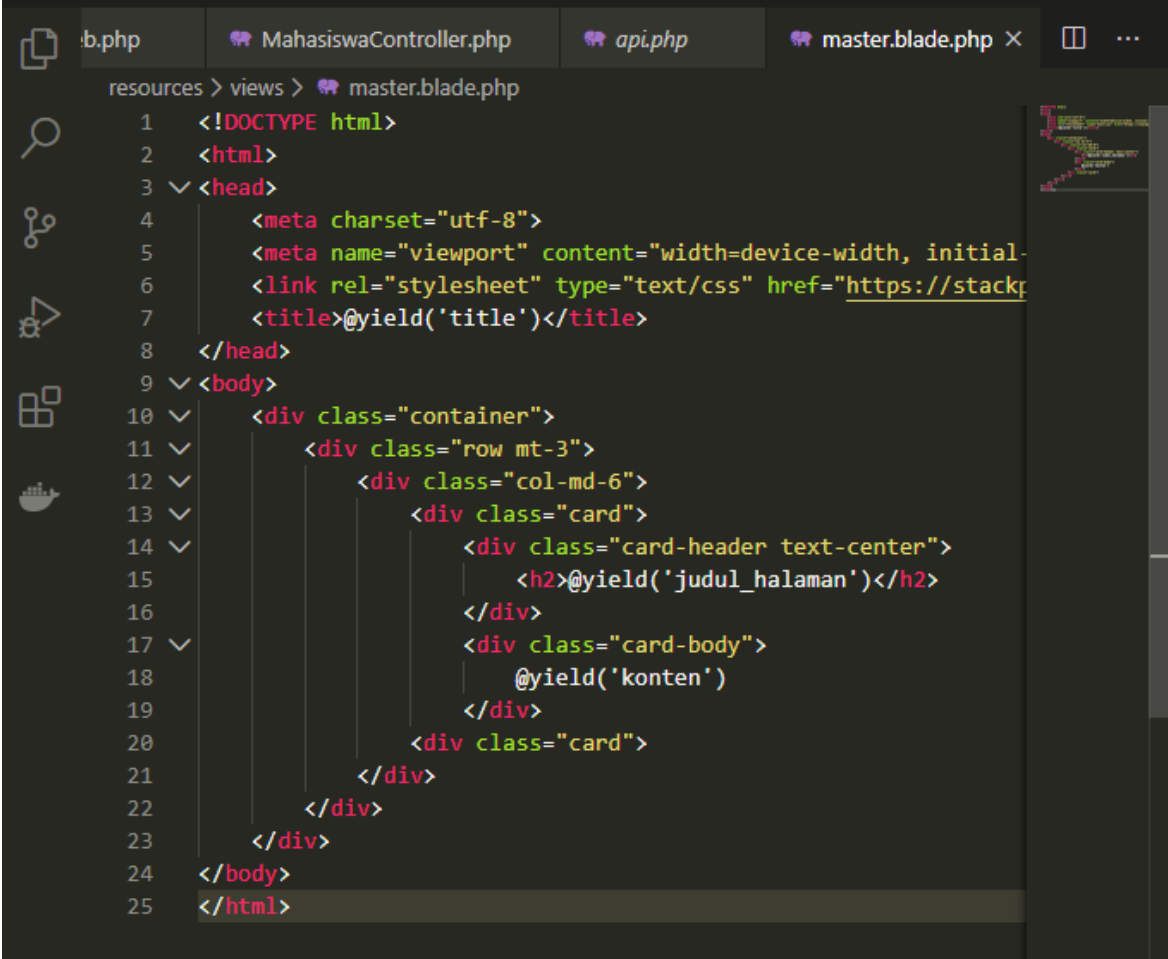
	<pre> routes > web.php 1 <?php 2 3 use Illuminate\Support\Facades\Route; 4 5 /* 6 ----- 7 Web Routes 8 ----- 9 10 Here is where you can register web routes for your application 11 routes are loaded by the RouteServiceProvider within a group 12 contains the "web" middleware group. Now create something gre 13 14 */ 15 16 > /*Route::get('/', function () { ... 17 18 });*/ 19 Route::get('/', 'MahasiswaController@index'); 20 </pre>
	<p>Buat controller baru menggunakan command prompt yaitu MahasiswaController menggunakan php artisan</p> <p>cd laravel-crud php artisan make:controller MahasiswaController</p> <pre> PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: powershell v + [] [X] ^ Windows PowerShell Copyright (C) Microsoft Corporation. All rights reserved. Try the new cross-platform PowerShell https://aka.ms/pscore6 PS C:\xampp\htdocs\laravel-crud> php artisan make:controller MahasiswaController Controller created successfully. PS C:\xampp\htdocs\laravel-crud> </pre>
	<p>Buat method index pada MahasiswaController.php pada folder app/Http/Controllers</p>



```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 class MahasiswaController extends Controller
8 {
9     public function index()
10     {
11         //mengambil data dari tabel mahasiswa
12         $mahasiswa = DB::table('mahasiswa')->get();
13
14         //mengirim data mahasiswa ke view index
15         return view('index',['mahasiswa' => $mahasiswa]);
16     }
17 }
18
```

Keterangan: - Tambahkan 'use Illuminate\Support\Facades\DB;' (line 6) agar query builder dapat digunakan - Line 13 untuk mengambil data dari tabel mahasiswa menggunakan query builder laravel dan akan disimpan di variabel \$mahasiswa - Line 16 : data akan dikirim ke blade view bernama index

Selanjutnya kita akan membuat view untuk menampilkan data mahasiswa dengan nama index.blade.php. Tetapi agar pembuatan view selanjutnya menjadi lebih mudah, terlebih dahulu kita akan membuat template blade (seperti file header-footer pada pembahasan CI) dengan nama master.blade.php pada folder resources/views.



```
resources > views > master.blade.php
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8">
5     <meta name="viewport" content="width=device-width, initial-
6     <link rel="stylesheet" type="text/css" href="https://stackp
7     <title>@yield('title')</title>
8 </head>
9 <body>
10 <div class="container">
11 <div class="row mt-3">
12 <div class="col-md-6">
13 <div class="card">
14 <div class="card-header text-center">
15 <h2>@yield('judul_halaman')</h2>
16 </div>
17 <div class="card-body">
18 @yield('konten')
19 </div>
20 <div class="card">
21 </div>
22 </div>
23 </div>
24 </body>
25 </html>
```

Keterangan: - Fungsi @yield pada line 6(title), 15(judul_halaman), dan 19(konten) berfungsi sebagai penanda bagian-bagian pada master blade. Nantinya bagian @yield ini akan diisi sesuai dengan halaman view yang menerapkan master.blade.php

Sekarang buat file index.blade.php yang menerapkan template master.blade.php dan akan digunakan untuk menampilkan data.

```
resources > views > index.blade.php
1  @extends('master')
2
3  @section('title','Home')
4
5  @section('judul_halaman','Data Mahasiswa')
6
7  @section('konten')
8  <a href="/mahasiswa/tambah" class="btn btn-primary">Tambah Data
9  <br/>
10 <br/>
11 <table class="table table-bordered table-hover table-striped">
12 <thead>
13 <tr>
14 <th>Nama</th>
15 <th>NIM</th>
16 </tr>
17 </thead>
18 <tbody>
19 @foreach($mahasiswa as $mhs)
20 <tr>
21 <td>{{ $mhs->nama }}</td>
22 <td>{{ $mhs->nim }}</td>
23 </tr>
24 @endforeach
25 </tbody>
26 </table>
27 @endsection
```

Keterangan: - Line 1 : @extends menunjukkan bahwa file index.blade.php menerapkan blade lain yaitu master.blade.php

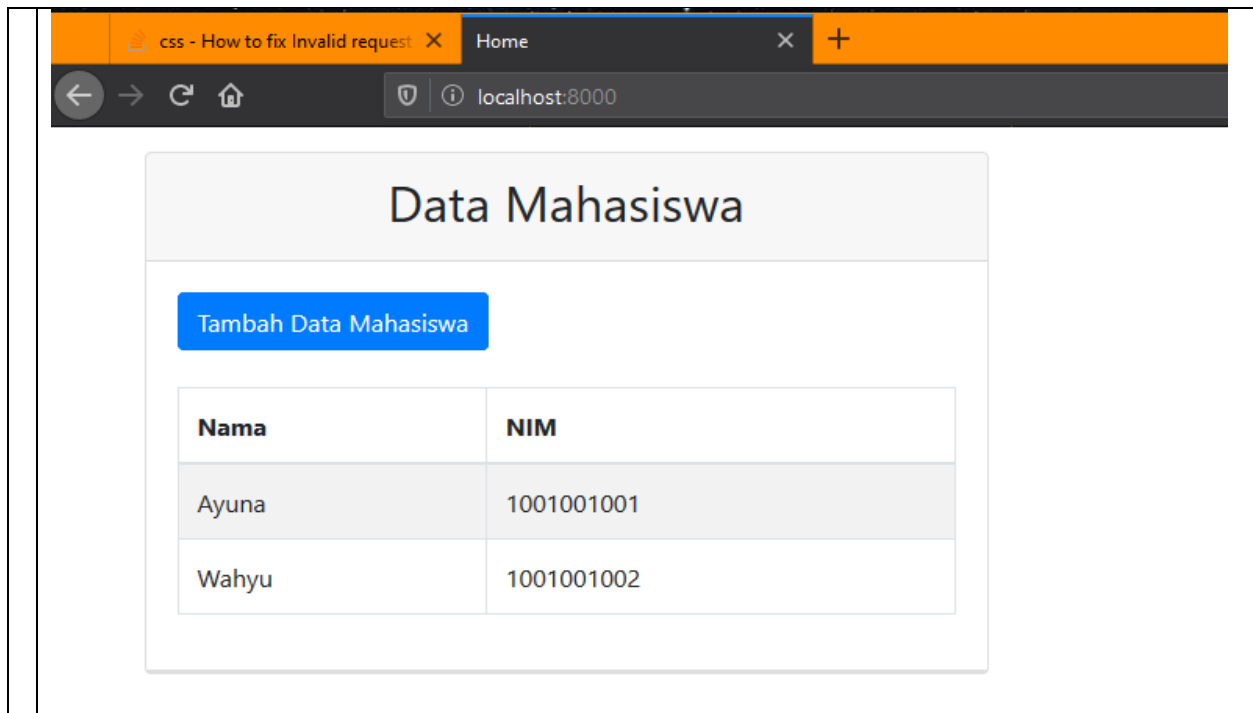
- Line 4 : @section('title', 'Home') berarti bahwa file index mengisi @yield('title') pada master dengan 'Home'

- Line 7 : @section('judul_halaman', 'Data Mahasiswa') berarti bahwa file index mengisi @yield('judul_halaman') pada master dengan 'Home'

- Line 10-30 : mengisi yield(@konten), karena terdapat banyak baris pada diawali dengan @section('konten') dan diakhiri dengan @endsection

- Line 24-25 menampilkan data mahasiswa dengan kolom nama dan nim

Jalankan command prompt, tuliskan perintah untuk menjalankan project laravel-crud php artisan serve Buka browser dan ketikkan localhost:8000, maka akan tampil sebagai berikut



Buatlah route baru pada routes/web.php dengan nama /mahasiswa/tambah yang akan menjalankan fungsi tambah pada MahasiswaController

```
20 Route::get('/mahasiswa/tambah', 'MahasiswaController@tambah');  
21
```

Buat method tambah pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view tambah

```
17     }  
18     public function index()  
19     {  
20         return view('tambah');  
21     }  
22 }  
23
```

Kemudian buatlah view tambah.blade.php yang berisi form untuk memasukkan data baru pada folder resources/views. View tambah juga mengaplikasikan master.blade.php


```
resources > views > tambah.blade.php
1  @extends('master')
2
3  @section('title','Tambah Data')
4
5  @section('judul_halaman','Tambah Data Mahasiswa')
6
7  @section('konten')
8      <a href="/mahasiswa" class="btn btn-danger">Kembali</a>
9      <br/>
10     <br/>
11     <form action="/mahasiswa/simpan" method="post">
12         {{ csrf_field() }}
13         <div class="form-group">
14             <label for="namamhs">Nama</label>
15             <input type="text" name="namamhs" class="form-control">
16         </div>
17         <div class="form-group">
18             <label for="nimhs">NIM</label>
19             <input type="number" name="nimhs" class="form-control">
20         </div>
21         <div class="form-group">
22             <label for="emailmhs">E-mail</label>
23             <input type="email" name="emailmhs" class="form-control">
24         </div>
25         <div class="form-group">
26             <label for="jurusanmhs">Jurusan</label>
27             <input type="text" name="jurusanmhs" class="form-control">
28         </div>
29         <button type="submit" name="tambah" class="btn btn-primary">Simpan</button>
30     </form>
31 @endsection
```

Keterangan: - Line 13-32 merupakan form untuk memasukkan data mahasiswa berupa nama, nim, email, dan jurusan

- Line 13 terdapat action="/mahasiswa/simpan" yang menunjukkan routes /mahasiswa/simpan dimana data pada form tersebut akan dikirimkan ke fungsi simpan pada controller MahasiswaController

- Line 14 terdapat {{ csrf_field() }} yang digunakan untuk menerapkan fitur laravel yaitu csrf protection. csrf protection adalah fitur keamanan untuk mencegah penginputan dari luar aplikasi, csrf akan men-generate kode token otomatis yang dibuat dalam bentuk form hidden.

Ketika tombol simpan ditekan, akan dipanggil routes /mahasiswa/simpan. Oleh karena itu, kita buat terlebih dahulu route tersebut.

```
21 Route::post('/mahasiswa/simpan', 'MahasiswaController@simpan');
22
```

Keterangan: - Pada route ini menggunakan metode post karena data mahasiswa dari form akan dikirim ke method simpan di MahasiswaController.php

Buat method simpan pada MahasiswaController untuk menyimpan data ke database.

```

21     }
22     public function simpan(Request $request)
23     {
24         //insert data ke table mahasiswa
25         DB::table('mahasiswa')->insert([
26             'nama' => $request->namamhs,
27             'nim' => $request->nimmhs,
28             'email' => $request->emailmhs,
29             'jurusan' => $request->jurusanmhs
30         ]);
31         return redirect('/mahasiswa');
32     }
33 }
34

```




Keterangan: - Variabel \$request untuk menerima data yang akan ditambahkan ke database - Line 28-33 merupakan query builder untuk insert data ke tabel mahasiswa

Kembali coba jalankan localhost:8000 dan pilih tombol 'Tambah Data Mahasiswa', maka akan ditampilkan halaman tambah yang berisi form untuk memasukkan data baru. Kita coba isikan data pada form tersebut.

Invalid request

Tambah Data

+

 localhost:8000/mahasiswa/tambah

Tambah Data Mahasiswa

Kembali

Nama

NIM

E-mail

Jurusan




Tambah Data

Setelah kita klik tombol tambah data, maka data baru akan ditampilkan pada halaman index.

Invalid request X

Home X

+

  localhost:8000

Data Mahasiswa

Tambah Data Mahasiswa

Nama	NIM
Ayuna	1001001001
Wahyu	1001001002
Moonaroh	1001001003
Moonaroh	1001001003

Kita dapat mencoba menambahkan beberapa data lagi agar jumlah data lebih banyak.

Buatlah tombol untuk melihat detail data (Line 28) pada file index.blade.php di folder resources/views

```

9 <br/>
10 <br/>
11 <table class="table table-bordered table-hover table-striped">
12 <thead>
13 <tr>
14 <th>Nama</th>
15 <th>NIM</th>
16 </tr>
17 </thead>
18 <tbody>
19 @foreach($mahasiswa as $mhs)
20 <tr>
21 <td>{{ $mhs->nama }}</td>
22 <td>{{ $mhs->nim }}</td>
23 <td>
24 <a href="/mahasiswa/detail/{{ $mhs->id }}" class="btn btn-primary">Detail</a>
25 </td>
26 </tr>
27 @endforeach
28 </tbody>
29 </table>
30 @endsection

```

Buatlah route baru pada routes/web.php dengan nama /mahasiswa/detail yang akan menjalankan fungsi detail pada MahasiswaController

```

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

```

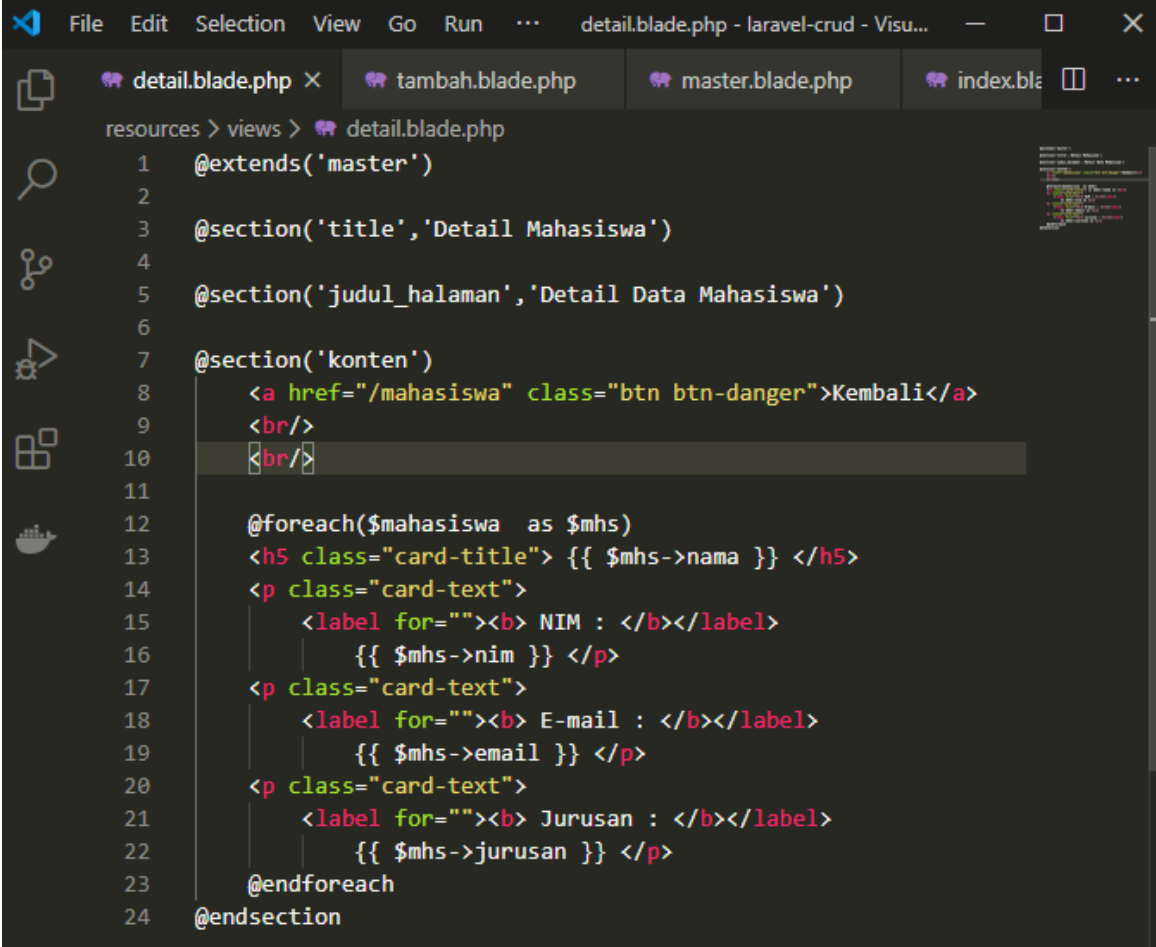
Buat method detail pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan data mahasiswa pada view detail.blade.php

```

32 }
33 public function detail($id)
34 {
35     $mahasiswa = DB::table('mahasiswa')->where('id',$id)->get();
36     return view('detail',['mahasiswa' => $mahasiswa]);
37 }
38 }
39

```

Kemudian buatlah view detail.blade.php yang menampilkan detail data mahasiswa pada folder resources/views. View detail juga mengaplikasikan master.blade.php.



```
1 @extends('master')
2
3 @section('title','Detail Mahasiswa')
4
5 @section('judul_halaman','Detail Data Mahasiswa')
6
7 @section('konten')
8     <a href="/mahasiswa" class="btn btn-danger">Kembali</a>
9     <br/>
10    <br/>
11
12    @foreach($mahasiswa as $mhs)
13        <h5 class="card-title"> {{ $mhs->nama }} </h5>
14        <p class="card-text">
15            <label for=""><b> NIM : </b></label>
16            {{ $mhs->nim }} </p>
17        <p class="card-text">
18            <label for=""><b> E-mail : </b></label>
19            {{ $mhs->email }} </p>
20        <p class="card-text">
21            <label for=""><b> Jurusan : </b></label>
22            {{ $mhs->jurusan }} </p>
23    @endforeach
24 @endsection
```

Keterangan: - Line 16-27 digunakan untuk menampilkan data mahasiswa berupa nama, nim, email, dan jurusan

Jalankan localhost:8000 dan pilih tombol 'Detail' di suatu data yang ingin kita lihat detailnya.

fix Invalid request X

Home X

+

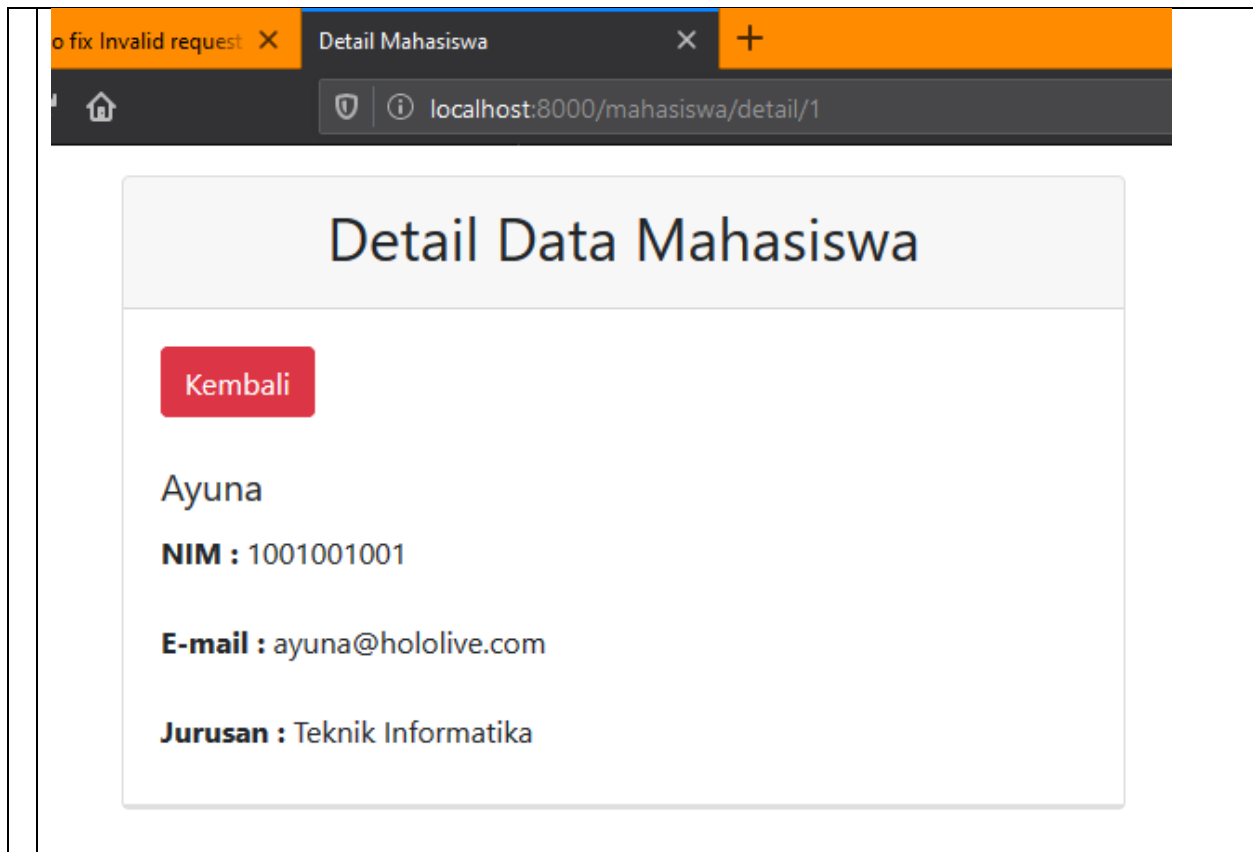


localhost:8000

Data Mahasiswa

Tambah Data Mahasiswa

Nama	NIM	
Ayuna	1001001001	Detail
Wahyu	1001001002	Detail
Moonaroh	1001001003	Detail



Buatlah tombol untuk mengubah data (Line 29) pada file index.blade.php di folder resources/views]

```

23         <td>
24             <a href="/mahasiswa/detail/{{ $mhs->id }}" class="btn btn-primary">Detail</a>
25             <a href="/mahasiswa/edit/{{ $mhs->id }}" class="btn btn-primary">Edit</a>
26         </td>
27     </tr>
28 @endforeach

```

Buatlah route baru pada routes/web.php dengan nama /mahasiswa/edit yang akan menjalankan fungsi edit pada MahasiswaController

```

22 Route::get('/mahasiswa/detail/{id}', 'MahasiswaController@detail');
23 Route::get('/mahasiswa/edit/{id}', 'MahasiswaController@edit');
24

```

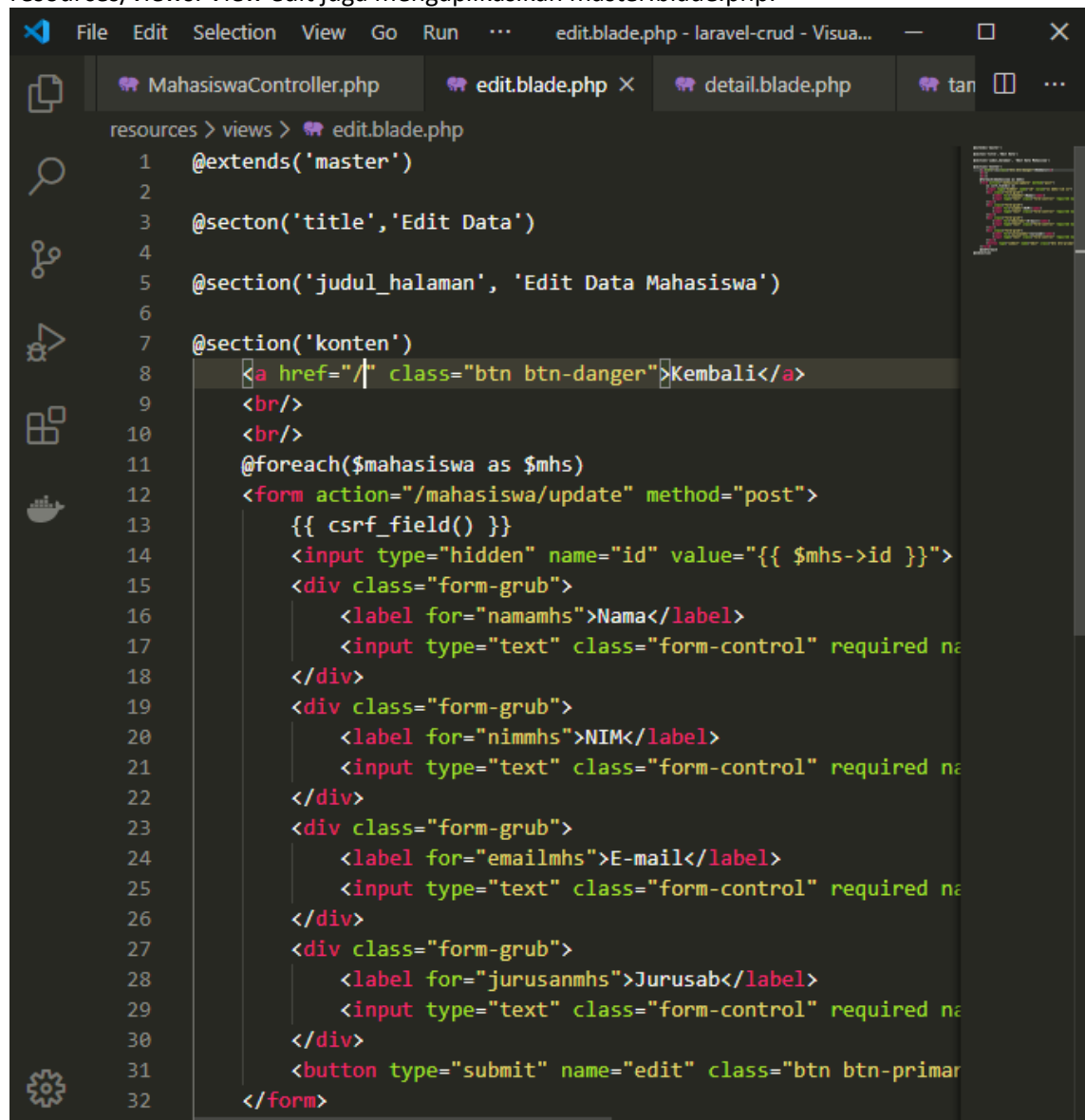
Buat method edit pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view edit.


```

    }
    public function edit($id)
    {
        $mahasiswa = DB::table('mahasiswa')->where('id',$id)->first();
        return view('edit',['mahasiswa' => $mahasiswa]);
    }

```

Kemudian buatlah view edit.blade.php yang berisi form untuk mengubah data pada folder resources/views. View edit juga mengaplikasikan master.blade.php.



```

resources > views > edit.blade.php
1  @extends('master')
2
3  @section('title','Edit Data')
4
5  @section('judul_halaman', 'Edit Data Mahasiswa')
6
7  @section('konten')
8      <a href="/" class="btn btn-danger">Kembali</a>
9      <br/>
10     <br/>
11     @foreach($mahasiswa as $mhs)
12         <form action="/mahasiswa/update" method="post">
13             {{ csrf_field() }}
14             <input type="hidden" name="id" value="{{ $mhs->id }}">
15             <div class="form-grub">
16                 <label for="namamhs">Nama</label>
17                 <input type="text" class="form-control" required name="nama">
18             </div>
19             <div class="form-grub">
20                 <label for="nimmhs">NIM</label>
21                 <input type="text" class="form-control" required name="nim">
22             </div>
23             <div class="form-grub">
24                 <label for="emailmhs">E-mail</label>
25                 <input type="text" class="form-control" required name="email">
26             </div>
27             <div class="form-grub">
28                 <label for="jurusanmhs">Jurusan</label>
29                 <input type="text" class="form-control" required name="jurusan">
30             </div>
31             <button type="submit" name="edit" class="btn btn-primary">Edit</button>
32         </form>

```

Keterangan:

- Line 14-36 merupakan form untuk memasukkan data mahasiswa berupa nama, nim, email, dan jurusan
- Line 15 terdapat action="/mahasiswa/update" yang menunjukkan routes /mahasiswa/update dimana data pada form tersebut akan dikirimkan ke fungsi update pada controller MahasiswaController

Ketika tombol simpan ditekan, akan dipanggil routes /mahasiswa/update. Oleh karena itu, kita buat terlebih dahulu route tersebut

```
25 Route::post('/mahasiswa/update', 'MahasiswaController@update');  
26
```

Keterangan:

- Pada route ini menggunakan metode post karena data mahasiswa dari form akan dikirim ke method update di MahasiswaController.php

Buat method update pada MahasiswaController untuk menyimpan data yang diubah ke database.

```
43 public function update(Request $request)  
44 {  
45     DB::table('mahasiswa')->where('id',$request->id)->update(  
46         'nama' => $request->namamhs,  
47         'nim' => $request->nimmhs,  
48         'email' => $request->emailmhs,  
49         'jurusan' => $request->jurusanmhs  
50     ]);  
51     return redirect('/mahasiswa');  
52 }  
53  
54
```

Keterangan: - Variabel \$request untuk menerima data yang akan ditambahkan ke database - Line 58-63 merupakan query builder untuk update data ke tabel mahasiswa

Jalankan localhost:8000 dan pilih tombol 'Edit', maka akan ditampilkan halaman edit yang berisi form untuk mengubah data baru.

invalid request ×

Home ×

+

localhost:8000

Data Mahasiswa

Tambah Data Mahasiswa

Nama	NIM	
Ayuna	1001001001	Detail Edit
Wahyu	1001001002	Detail Edit
Moonaroh	1001001003	Detail Edit

Klik edit di data pertama, kita akan mengubah namanya.

Invalid request

localhost:8000/mahasiswa/edit/1

localhost:8000/mahasiswa/edit/1

Edit Data)

Edit Data Mahasiswa

Kembali

Nama

lofifteen

NIM

1001001001

E-mail

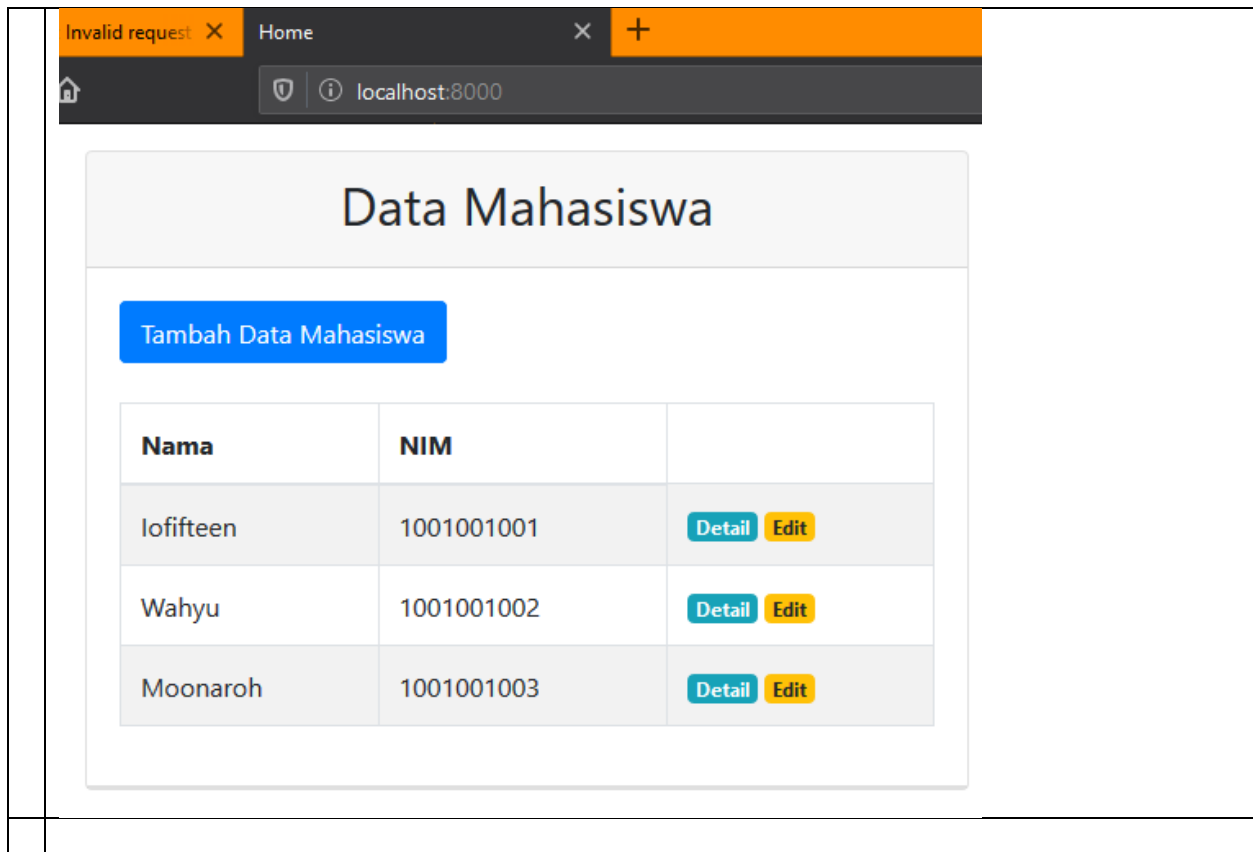
lofifteen@hololive.com

Jurusab

Teknik Informatika

Simpan Data

Setelah kita klik simpan Data, maka data pertama akan berubah.



Buatlah tombol untuk menghapus data (Line 30) pada file index.blade.php di folder resources/views

```

25 <a href="/mahasiswa/edit/{{ $mhs->id }}" class="badge badge-warning">Edit</a>
26 <a href="/mahasiswa/hapus/{{ $mhs->id }}" class="badge badge-danger">Hapus</a>
27

```

Buatlah route baru pada routes/web.php dengan nama /mahasiswa/hapus yang akan menjalankan fungsi hapus pada MahasiswaController

```

26 Route::get('/mahasiswa/hapus/{id}', 'MahasiswaController@hapus');
27

```

Buat method hapus pada MahasiswaController.php di folder app/Http/Controllers yang akan menjalankan fungsi hapus.

```

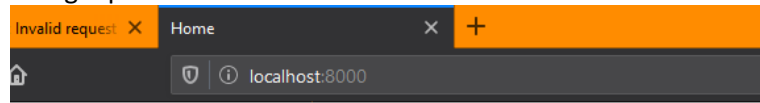
}
public function hapus($id)
{
    //menghapus data mahasiswa berdasarkan id yang dipilih
    DB::table('mahasiswa')->where('id', $id)->delete();

    return redirect('/');
}
}

```

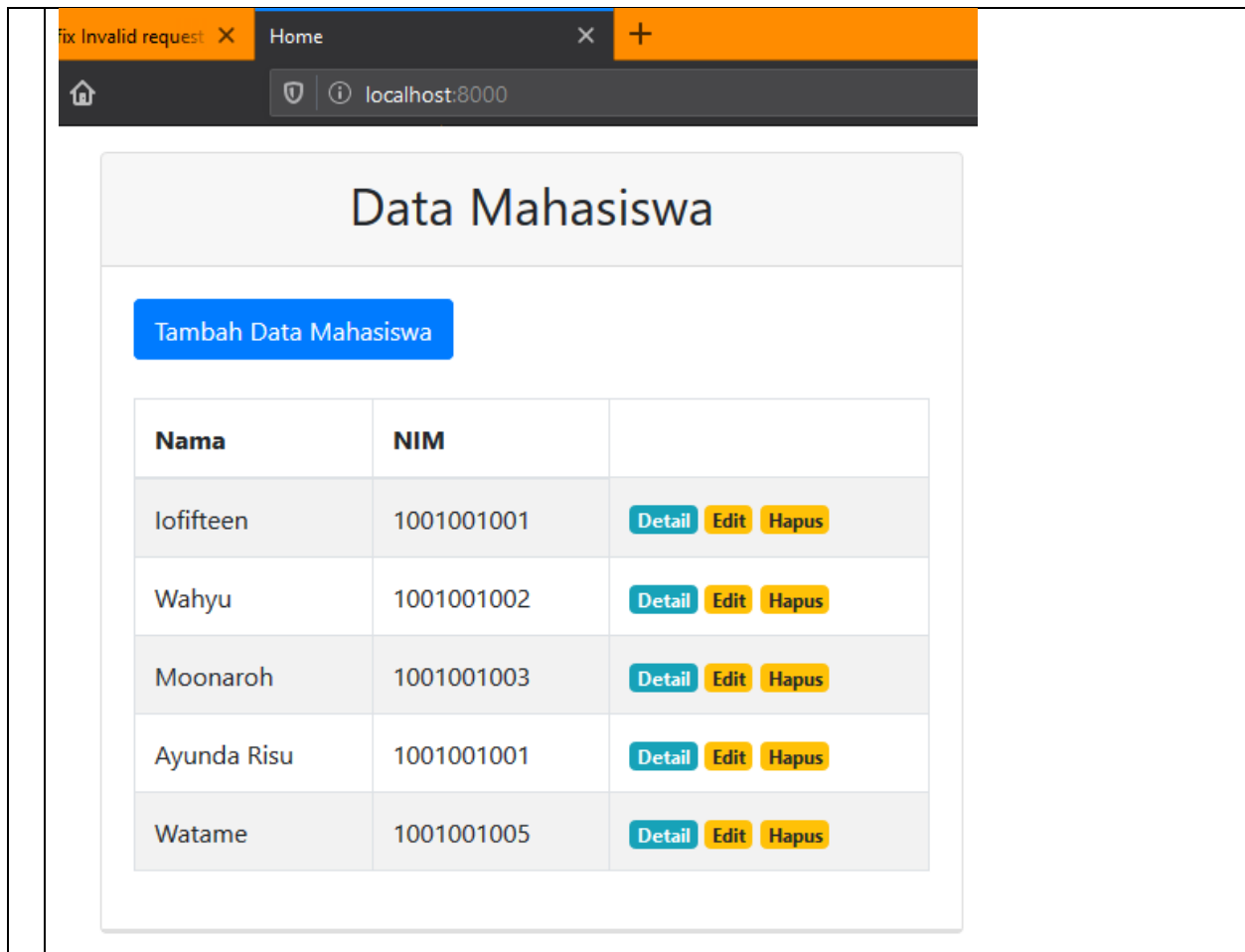
Keterangan : - Line 67 : query builder untuk menghapus data mahasiswa berdasarkan id yang dipilih

Jalankan localhost:8000 dan pilih tombol 'Hapus', maka data yang terpilih akan dihapus. Contoh: menghapus data terakhir.



Data Mahasiswa		
Tambah Data Mahasiswa		
Nama	NIM	
Iofifteen	1001001001	Detail Edit Hapus
Wahyu	1001001002	Detail Edit Hapus
Moonaroh	1001001003	Detail Edit Hapus
Ayunda Risu	1001001001	Detail Edit Hapus
Watame	1001001005	Detail Edit Hapus
Udin	1001001006	Detail Edit Hapus

Setelah menghapus data terakhir



Praktikum – Bagian 2: Membuat CRUD di Laravel menggunakan Eloquent

Buatlah project Laravel baru dengan nama laravel-crud-kedua. Buka command prompt, tuliskan perintah berikut. `cd C:\xampp\htdocs laravel new laravel-crud-kedua`

	<div data-bbox="267 199 1380 241"> <div>PROBLEMS</div> <div>OUTPUT</div> <div>DEBUG CONSOLE</div> <div>TERMINAL</div> <div>6: powershell ▾</div> <div>+</div> <div>□</div> <div>🗑</div> <div>^</div> <div>×</div> </div> <div data-bbox="267 273 1380 1249"> <pre> PS C:\xampp\htdocs> laravel new laravel-crud-kedua Crafting application... Loading composer repositories with package information Installing dependencies (including require-dev) from lock file Package operations: 92 installs, 0 updates, 0 removals - Installing doctrine/inflector (1.3.1): Loading from cache - Installing doctrine/lexer (1.2.0): Loading from cache - Installing dragonmantank/cron-expression (v2.3.0): Loading from cache - Installing voku/portable-ascii (1.4.10): Loading from cache - Installing symfony/polyfill-ctype (v1.15.0): Loading from cache - Installing phpoption/phpooption (1.7.3): Loading from cache - Installing vlucas/phpdotenv (v4.1.4): Loading from cache - Installing symfony/css-selector (v5.0.7): Loading from cache - Installing tijsverkoyen/css-to-inline-styles (2.2.2): Loading from cache - Installing symfony/polyfill-mbstring (v1.15.0): Loading from cache - Installing symfony/var-dumper (v5.0.7): Loading from cache - Installing symfony/routing (v5.0.7): Loading from cache - Installing symfony/process (v5.0.7): Loading from cache - Installing symfony/polyfill-php72 (v1.15.0): Loading from cache - Installing symfony/polyfill-intl-idn (v1.15.0): Loading from cache - Installing symfony/mime (v5.0.7): Loading from cache - Installing symfony/polyfill-php73 (v1.15.0): Loading from cache - Installing symfony/http-foundation (v5.0.7): Loading from cache - Installing psr/event-dispatcher (1.0.0): Loading from cache - Installing symfony/event-dispatcher-contracts (v2.0.1): Loading from cache e - Installing symfony/event-dispatcher (v5.0.7): Loading from cache - Installing psr/log (1.1.3): Loading from cache - Installing symfony/error-handler (v5.0.7): Loading from cache - Installing symfony/http-kernel (v5.0.7): Loading from cache - Installing symfony/finder (v5.0.7): Loading from cache </pre> </div>
	<p>Selanjutnya kita lakukan konfigurasi database di Laravel. Untuk melakukan konfigurasi database, bukalah file .env pada project laravel-crud. Ubah seperti di bawah ini.</p>


```

.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:d6LbqwPzUYmXv3Qn7CYnCQ8m
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=mysql
10 DB_HOST=localhost
11 DB_PORT=3306
12 DB_DATABASE=latihan_laravel
13 DB_USERNAME=root
14 DB_PASSWORD=

```

Pada project ini kita gunakan database dan tabel yang sebelumnya digunakan pada Praktikum Bagian 1. Tambahkan kolom created_at dan updated_at yang bertipe TIMESTAMP dan default nilainya NULL

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	nama	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
3	nim	varchar(10)	latin1_swedish_ci		No	None			Change Drop More
4	email	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
5	jurusan	varchar(20)	latin1_swedish_ci		No	None			Change Drop More
6	created_at	timestamp			Yes	NULL			Change Drop More
7	updated_at	timestamp			Yes	NULL			Change Drop More

Setelah kita memiliki beberapa data pada tabel mahasiswa, kita akan mencoba untuk menampilkan data tersebut ketika project dijalankan. Pertama, buatlah route pada routes/web.php sehingga ketika pertama kali project dijalankan akan terbuka halaman yang menampilkan data.

```

19 Route::get('/', 'MahasiswaController@index');
20 Route::get('/mahasiswa', 'MahasiswaController@index');
21
22

```

Buat model menggunakan command prompt dengan nama Mahasiswa menggunakan php artisan cd laravel-crud-kedua php artisan make:model Mahasiswa

	<pre>PS C:\xampp\htdocs\laravel-crud-kedua> php artisan make:model Mahasiswa Model created successfully. PS C:\xampp\htdocs\laravel-crud-kedua> </pre>	
	<p>Ubah model Mahasiswa.php pada folder App menjadi seperti berikut.</p>  <p>Keterangan: - Model Mahasiswa akan menangani tabel mahasiswa</p>	
	<p>Buat controller baru yaitu MahasiswaController menggunakan php artisan php artisan make:controller MahasiswaController</p> 	
	<p>Buat method index pada MahasiswaController.php pada folder app/Http/Controllers</p>	

app > Http > Controllers > 🐞 MahasiswaController.php > ...

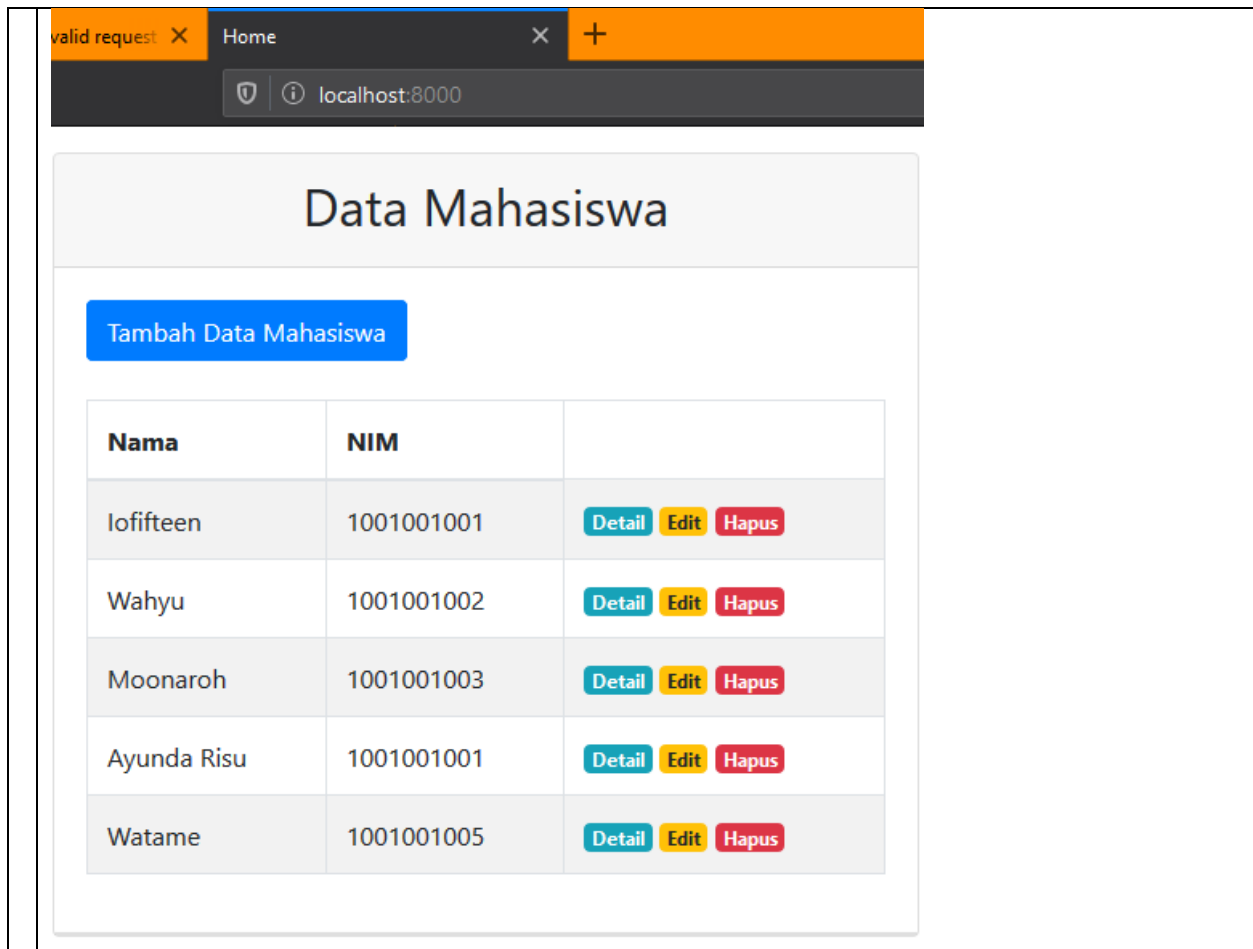
```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use App\Mahasiswa;
7
8  class MahasiswaController extends Controller
9  {
10     public function index()
11     {
12         $mahasiswa = Mahasiswa::all();
13         return view('index', ['mahasiswa' => $mahasiswa]);
14     }
15 }
16
```

Keterangan:

- Tambahkan 'use App\Mahasiswa' (line 6) untuk menggunakan model Mahasiswa
- Line 12 untuk mengambil semua data dari model/tabel Mahasiswa dan akan disimpan di variabel \$mahasiswa - Line 16 : data akan dikirim ke blade view bernama index

Selanjutnya kita akan membuat view untuk menampilkan data mahasiswa dengan nama index.blade.php. Tetapi agar pembuatan view selanjutnya menjadi lebih mudah, terlebih dahulu kita akan membuat template blade (seperti pada Bagian 1). Pada bagian view kita buat seperti bagian 1 (copy-paste dari bagian 1)

Jalankan command prompt, tuliskan perintah untuk menjalankan project laravel-crud php artisan serve Buka browser dan ketikkan localhost:8000, maka akan tampil sebagai berikut



Buatlah route baru pada routes/web.php dengan nama /mahasiswa/tambah yang akan menjalankan fungsi tambah pada MahasiswaController ketika tombol Tambah Data Mahasiswa ditekan.

```
21 Route::get('/mahasiswa/tambah', 'MahasiswaController@tambah');  
22
```

Buat method tambah pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view tambah.

```
}  
public function tambah()  
{  
    return view('tambah');  
}
```

Kemudian buatlah view tambah.blade.php yang berisi form untuk memasukkan data baru pada folder resources/views.

Kita lakukan copy paste dari tambah.blade.php di Bagian 1

Ketika tombol simpan ditekan, akan dipanggil routes /mahasiswa/simpan. Oleh karena itu, kita buat terlebih dahulu route tersebut.

```
Route::post('/mahasiswa/simpan', 'MahasiswaController@simpan');
```

Keterangan: - Pada route ini menggunakan metode post karena data mahasiswa dari form akan dikirim ke method simpan di MahasiswaController.php

Buat method simpan pada MahasiswaController untuk menyimpan data ke database.

```
public function simpan(Request $request)
{
    //insert data ke table mahasiswa
    Mahasiswa::create([
        'nama' => $request->namamhs,
        'nim' => $request->nimmhs,
        'email' => $request->emailmhs,
        'jurusan' => $request->jurusanmhs
    ]);
    return redirect('/');
}
```

Keterangan: - Variabel \$request untuk menerima data yang akan ditambahkan ke database
- Line 23-28 merupakan fitur eloquent menggunakan fungsi create() untuk insert data ke tabel mahasiswa




Karena kita menggunakan fungsi create pada Controller, maka butuh ditambahkan code pada Line 10 yang disebut Mass Assignment pada model Mahasiswa.php. Mass Assignment digunakan untuk memfilter kolom mana yang boleh dan tidak boleh diinput.

```
class Mahasiswa extends Model
{
    protected $table = "mahasiswa";
    protected $fillable = ['nama', 'nim', 'email', 'jurusan'];
}
```

Kembali coba jalankan localhost:8000 dan pilih tombol 'Tambah Data Mahasiswa', maka akan ditampilkan halaman tambah yang berisi form untuk memasukkan data baru. Kita coba isikan data pada form tersebut.

o fix Invalid request ✕

Tambah Data ✕ +

   localhost:8000/mahasiswa/tambah

Tambah Data Mahasiswa

Kembali

Nama

Yunyun

NIM

1001001007

E-mail

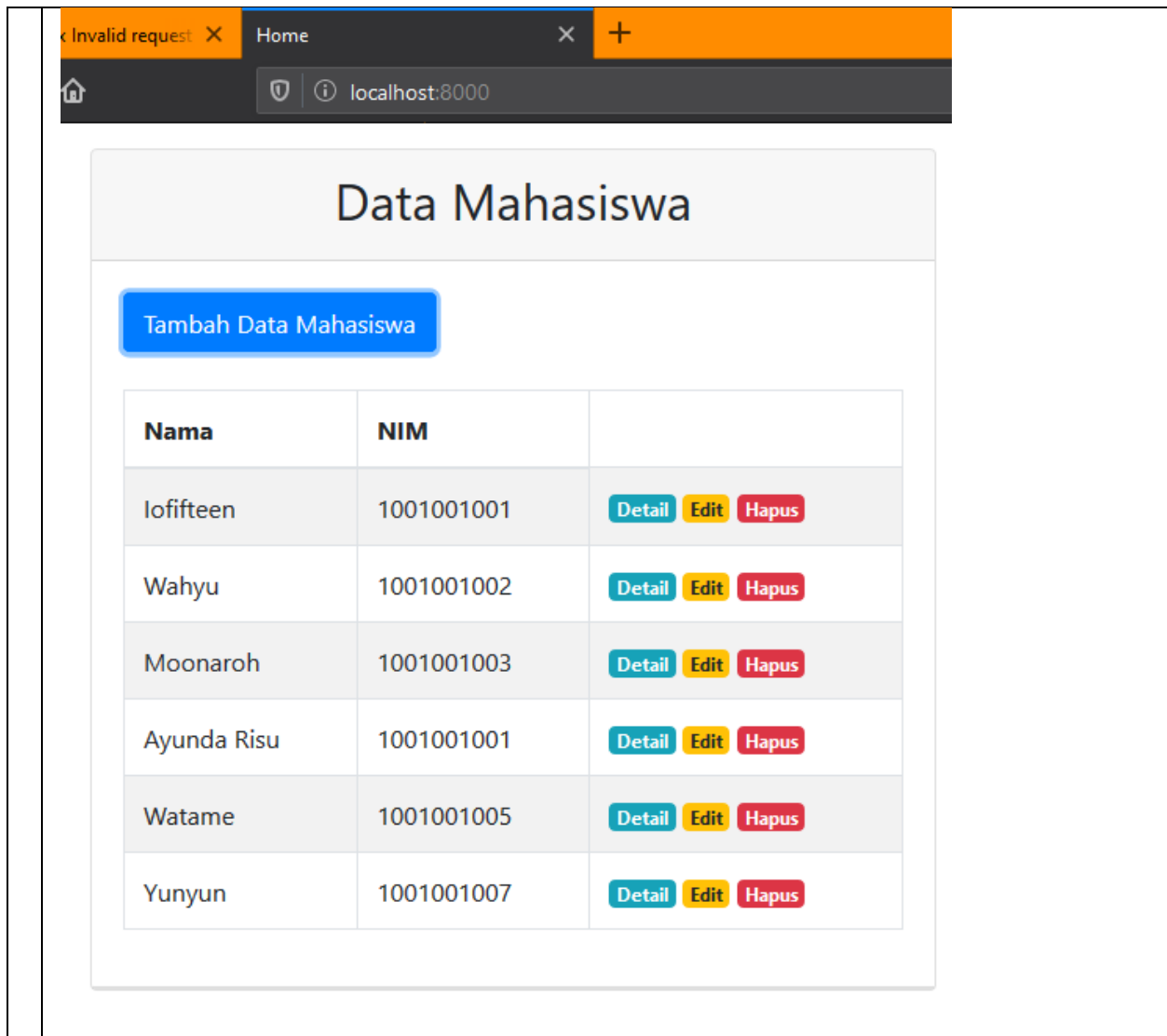
duar@email.com

Jurusan

Akuntansi

Tambah Data

Setelah kita klik tombol tambah data, maka hasilnya sebagai berikut.



Buatlah route baru pada routes/web.php dengan nama /mahasiswa/detail yang akan menjalankan fungsi detail pada MahasiswaController

```
Route::get('/mahasiswa/detail/{id}', 'MahasiswaController@detail');
```

Buat method detail pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan data mahasiswa pada view detail.blade.php.

```
}  
public function detail($id)  
{  
    $mahasiswa = Mahasiswa::find($id);  
    return view('detail', ['mahasiswa' => $mahasiswa]);  
}
```

Kemudian buatlah view detail.blade.php yang menampilkan detail data mahasiswa pada folder resources/views. View detail juga mengaplikasikan master.blade.php.

```
resources > views > detail.blade.php
1  @extends('master')
2
3  @section('title','Detail Mahasiswa')
4
5  @section('judul_halaman','Detail Data Mahasiswa')
6
7  @section('konten')
8      <a href="/" class="btn btn-danger">Kembali</a>
9      <br/>
10     <br/>
11
12     @foreach($mahasiswa as $mhs)
13         <h5 class="card-title"> {{ $mhs->nama }} </h5>
14         <p class="card-text">
15             <label for=""><b> NIM : </b></label>
16             {{ $mhs->nim }} </p>
17         <p class="card-text">
18             <label for=""><b> E-mail : </b></label>
19             {{ $mhs->email }} </p>
20         <p class="card-text">
21             <label for=""><b> Jurusan : </b></label>
22             {{ $mhs->jurusan }} </p>
23     @endforeach
24 @endsection
```

Jalankan localhost:8000 dan pilih tombol 'Detail' di suatu data yang ingin kita lihat detailnya.

	<div> <div>Invalid request</div> <div>Detail Mahasiswa</div> <div>+</div> </div> <div> <div>localhost:8000/mahasiswa/detail/1</div> </div> <div> <h2>Detail Data Mahasiswa</h2> <div> <div>Kembali</div> <div>lofifteen</div> <div>NIM : 1001001001</div> <div>E-mail : lofifteen@hololive.com</div> <div>Jurusan : Teknik Informatika</div> </div> </div>

	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/edit yang akan menjalankan fungsi edit pada MahasiswaController</p> <pre>Route::get('/mahasiswa/edit/{id}', 'MahasiswaController@edit');</pre>
	<p>Buat method edit pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view edit.</p> <pre>public function edit(\$id) { \$mahasiswa = Mahasiswa::find(\$id); return view('edit', ['mahasiswa' => \$mahasiswa]); }</pre> <p>Keterangan : - Line 41 : fungsi eloquent untuk mengambil data mahasiswa berdasarkan id yang dipilih</p>
	<p>Kemudian buatlah view edit.blade.php yang berisi form untuk mengubah data pada folder resources/views. View edit juga mengaplikasikan master.blade.php.</p>

```

6
7 @section('konten')
8     <a href="/" class="btn btn-danger">Kembali</a>
9     <br/>
10    <br/>
11
12    <form action="/mahasiswa/update/{{ $mahasiswa->id }}" method="POST"
13        {{ csrf_field() }}
14        <input type="hidden" name="id" value="{{ $mahasiswa->id }}" />
15        <div class="form-grub">
16            <label for="namamhs">Nama</label>
17            <input type="text" class="form-control" required name="nama" />
18        </div>
19        <div class="form-grub">
20            <label for="nimmhs">NIM</label>
21            <input type="text" class="form-control" required name="nim" />
22        </div>
23        <div class="form-grub">
24            <label for="emailmhs">E-mail</label>
25            <input type="text" class="form-control" required name="email" />
26        </div>
27        <div class="form-grub">
28            <label for="jurusanmhs">Jurusan</label>
29            <input type="text" class="form-control" required name="jurusan" />
30        </div>
31        <button type="submit" name="edit" class="btn btn-primary">Simpan</button>
32    </form>
33
34 @endsection

```

Keterangan: - Line 11-31 merupakan form untuk memasukkan data mahasiswa berupa nama, nim, email, dan jurusan

- Line 11 terdapat action="/mahasiswa/update/{{ \$mahasiswa->id }}" yang menunjukkan routes /mahasiswa/update/{id} dimana data pada form tersebut akan dikirimkan ke fungsi update pada controller MahasiswaController

Ketika tombol simpan ditekan, akan dipanggil routes /mahasiswa/update/{id}. Oleh karena itu, kita buat terlebih dahulu route tersebut.

```

23 Route::get('/mahasiswa/detail/{id}', 'MahasiswaController@detail');
24 Route::post('/mahasiswa/update/{id}', 'MahasiswaController@update');
25 Route::get('/mahasiswa/edit/{id}', 'MahasiswaController@edit');
26

```

Keterangan: - Pada route ini menggunakan metode post karena data mahasiswa dari form akan dikirim ke method update di MahasiswaController.php

Buat method update pada MahasiswaController untuk menyimpan data yang diubah ke database.

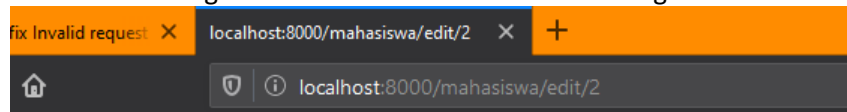
```

public function update(Request $request)
{
    $mahasiswa = Mahasiswa::find($id);
    $mahasiswa->nama = $request->namamhs;
    $mahasiswa->nim = $request->nimmhs;
    $mahasiswa->email = $request->emailmhs;
    $mahasiswa->jurusan = $request->jurusanmhs;
    $mahasiswa->save();
    return redirect('/');
}

```

Keterangan: - Variabel \$request untuk menerima data yang akan ditambahkan ke database - Line 48-52 merupakan fungsi eloquent untuk update data ke tabel mahasiswa

Jalankan localhost:8000 dan pilih tombol 'Edit', maka akan ditampilkan halaman edit yang berisi form untuk mengubah data baru. Cobalah untuk mengedit data.



, 'Edit Data')

Edit Data Mahasiswa

Kembali

Nama

NIM

E-mail

Jurusan

Simpan Data

--	--

	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/hapus yang akan menjalankan fungsi hapus pada MahasiswaController</p> <pre>26 ::get('/mahasiswa/hapus/{id}', 'MahasiswaController@hapus');</pre>
	<p>Buat method hapus pada MahasiswaController.php di folder app/Http/Controllers yang akan menjalankan fungsi hapus.</p> <pre>public function hapus(\$id) { \$mahasiswa = Mahasiswa::find(\$id); \$mahasiswa->delete(); return redirect('/'); }</pre> <p>Keterangan : - Line 59 :fungsi eloquent untuk menghapus data mahasiswa berdasarkan id yang dipilih</p>
	<p>Jalankan localhost:8000 dan pilih tombol 'Hapus', maka data yang terpilih akan dihapus.</p>

×

+

localhost:8000

Data Mahasiswa

Tambah Data Mahasiswa

Nama	NIM	
Iofifteen	1001001001	<div>DetailEditHapus</div>
Moonaroh	1001001003	<div>DetailEditHapus</div>
Ayunda Risu	1001001001	<div>DetailEditHapus</div>
Watame	1001001005	<div>DetailEditHapus</div>
Yunyun	1001001007	<div>DetailEditHapus</div>