

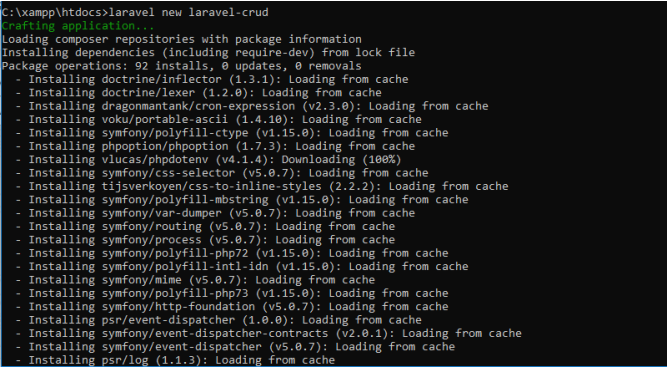
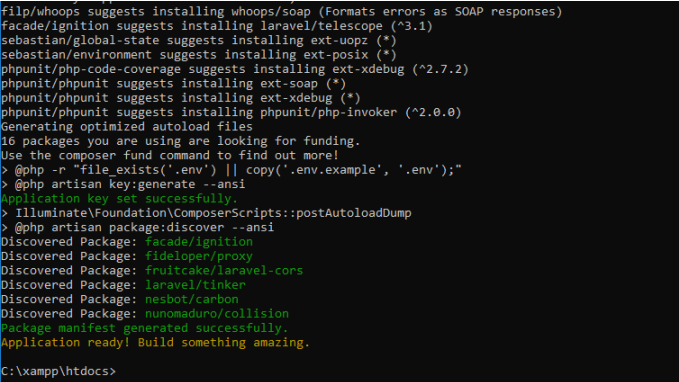


NAMA : QORINDA YULVARISMA
KELAS : TI-2A
NIM : 1841720084

LAPORAN PRAKTIKUM PEMROGRAMAN WEB LANJUT

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

a. Konfigurasi Database dan Pembuatan Tabel di MySQL

Langkah	Keterangan
1	<p>Buatlah project Laravel baru dengan nama laravel-crud. Buka command prompt, tuliskan perintah berikut.</p> <pre>cd C:\xampp\htdocs laravel new laravel-crud</pre>   <pre>C:\xampp\htdocs></pre>
2	<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> <p>Keterangan:</p>

```
.env
.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:VuIb0zDvhqo9J/cpFcxAGcdADwtISbY206mIHvvDp1U=
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=
15
```

- Nama database yang akan digunakan adalah latihan_laravel dengan username root.

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

Struktur tabel

Tampilan hubungan

#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra	Tindakan
<input type="checkbox"/>	1	id	int(11)		Tidak	Tidak ada		AUTO_INCREMENT	Ubah Hapus Lainnya
<input type="checkbox"/>	2	nama	varchar(100)	latin1_swedish_ci	Tidak	Tidak ada			Ubah Hapus Lainnya
<input type="checkbox"/>	3	nim	varchar(10)	latin1_swedish_ci	Tidak	Tidak ada			Ubah Hapus Lainnya
<input type="checkbox"/>	4	email	varchar(50)	latin1_swedish_ci	Tidak	Tidak ada			Ubah Hapus Lainnya
<input type="checkbox"/>	5	jurusan	varchar(20)	latin1_swedish_ci	Tidak	Tidak ada			Ubah Hapus Lainnya

4 Isilah beberapa data pada tabel mahasiswa tersebut.

+ Opsi

				id	nama	nim	email	jurusan			
<input type="checkbox"/>		Ubah		Salin		Hapus	1	nanda	1001001001	nandan@gmail.com	teknik informatika
<input type="checkbox"/>		Ubah		Salin		Hapus	2	wahyu	1001001002	wahyu@gmail.com	teknik elektro

b. Menampilkan Data dari Database

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

	 <pre> web.php x 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. These 11 routes are loaded by the RouteServiceProvider within a group which 12 contains the "web" middleware group. Now create something great! 13 14 */ 15 16 // Route::get('/', function () { 17 // return view('welcome'); 18 // }); 19 20 Route::get('/', 'mahasiswaController@index');</pre> <ul style="list-style-type: none"> - Ketika route ('/') diakses, akan dijalankan method index pada controller bernama MahasiswaController.
2	<p>Buat controller baru menggunakan command prompt yaitu MahasiswaController menggunakan php artisan</p> <p>cd laravel-crud</p> <p>php artisan make:controller MahasiswaController</p>  <pre> C:\xampp\htdocs>cd laravel-crud C:\xampp\htdocs\laravel-crud>php artisan make:controller MahasiswaController Controller created successfully.</pre>
3	<p>Buat method index pada MahasiswaController.php pada folder app/Http/Controllers</p> <p>Keterangan:</p>  <pre> MahasiswaController.php x app > Http > Controllers > MahasiswaController.php 1 <?php 2 3 namespace App\Http\Controllers; 4 5 use Illuminate\Http\Request; 6 use Illuminate\Support\Facades\DB; 7 8 class MahasiswaController extends Controller 9 { 10 public function index(){ 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</pre> <ul style="list-style-type: none"> - 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

4

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**.

```

master.blade.php X
resources > views > master.blade.php
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" rel="stylesheet" >
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                         <!-- bagian judul halaman -->
16                         <h2> @yield('judul_halaman')</h2>
17                     </div>
18                     <div class="card-body">
19                         <!-- bagian konten blog -->
20                         @yield('konten')
21                     </div>
22                     <div class="card">
23                         </div>
24                 </div>
25             </div>
26 </body>
27 </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

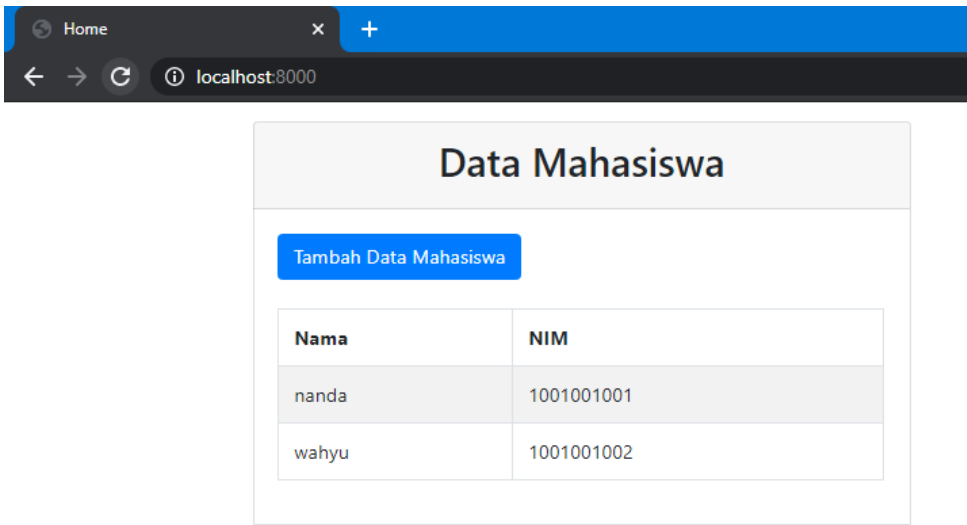
5

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

```

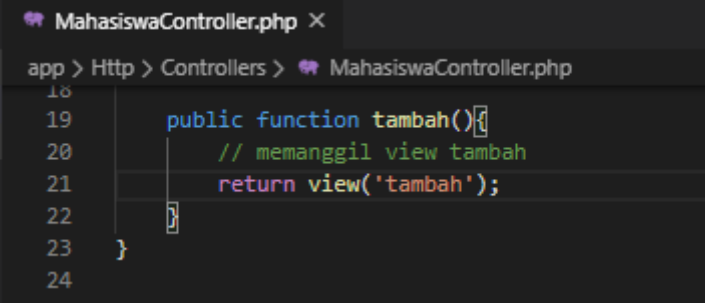
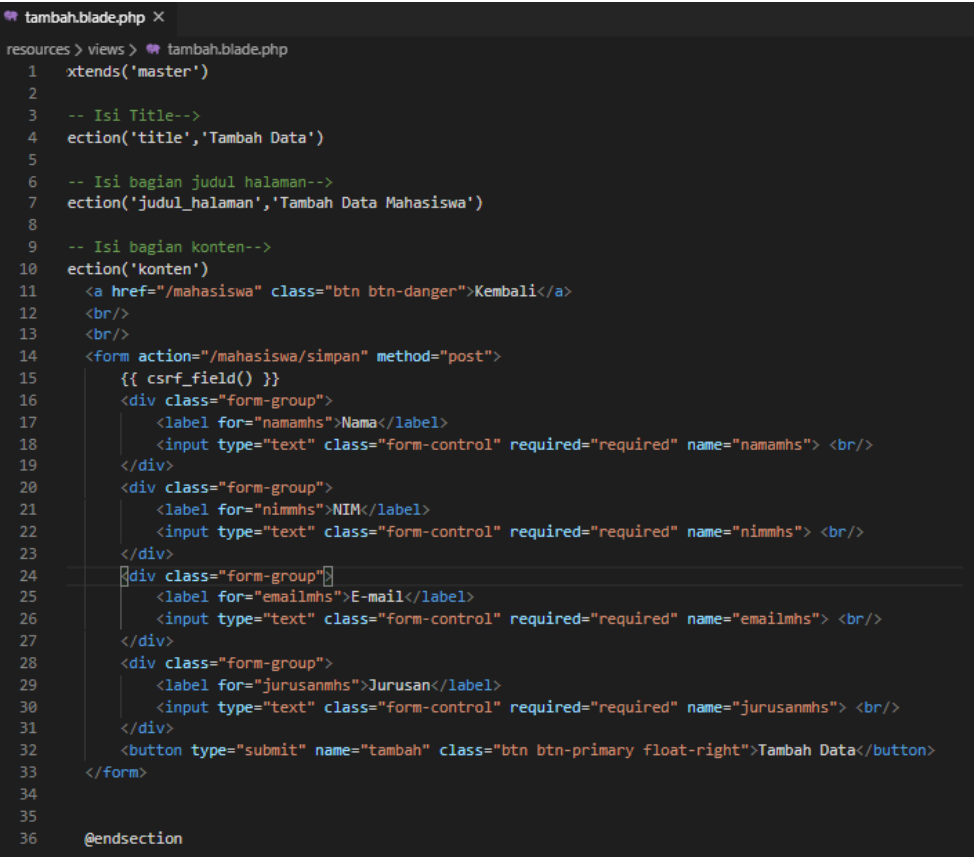
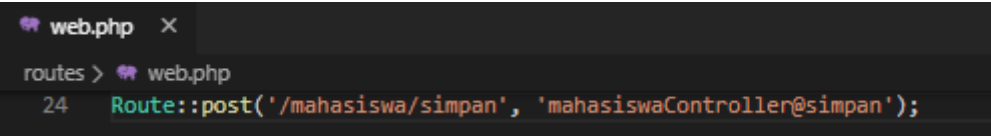
index.blade.php X
resources > views > index.blade.php
1 @extends('master')
2
3 <!-- Isi Title -->
4 @section('title','Home')
5
6 <!-- Isi bagian judul halaman -->
7 @section('judul_halaman','Data Mahasiswa')
8
9 <!-- Isi bagian konten -->
10 @section('konten')
11     <a href="/mahasiswa/tambah" class="btn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


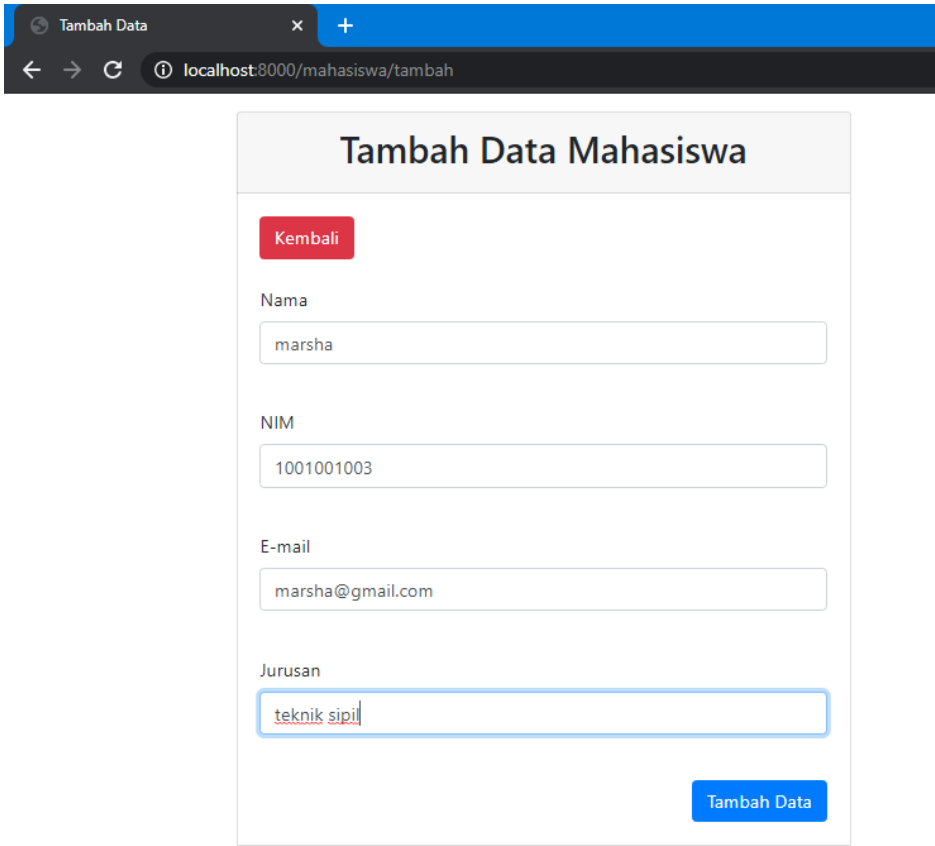
```

	<p>Keterangan:</p> <ul style="list-style-type: none"> - 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
6	<p>Jalankan command prompt, tuliskan perintah untuk menjalankan project laravel-crud php artisan serve</p> <pre>C:\xampp\htdocs\laravel-crud>php artisan serve Laravel development server started: http://127.0.0.1:8000</pre> <p>Buka browser dan ketikkan localhost:8000, maka akan tampil sebagai berikut</p> 

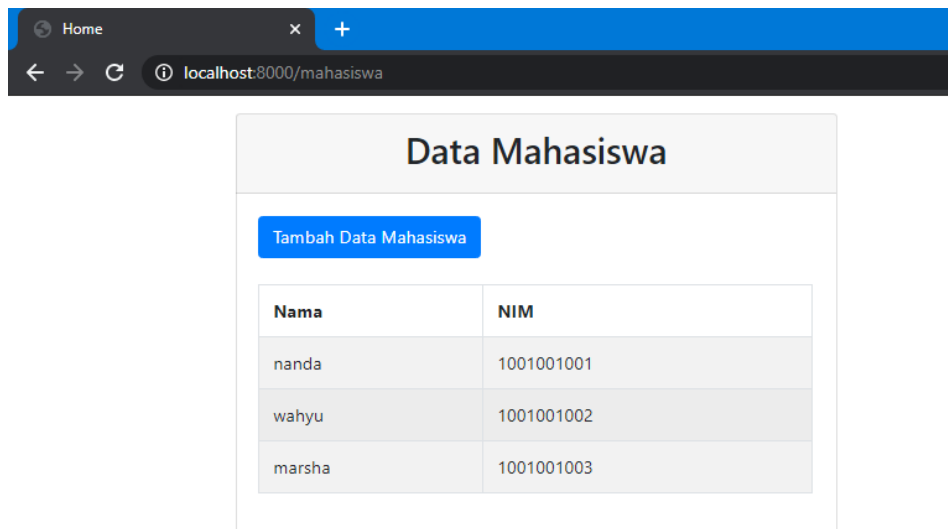
c. Memasukkan Data (Create) ke Database

Langkah	Keterangan
1	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/tambah yang akan menjalankan fungsi tambah pada MahasiswaController</p> <pre>web.php x routes > web.php 21 22 Route::get('/mahasiswa/tambah', 'mahasiswaController@tambah');</pre>
2	<p>Buat method tambah pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view tambah.</p>

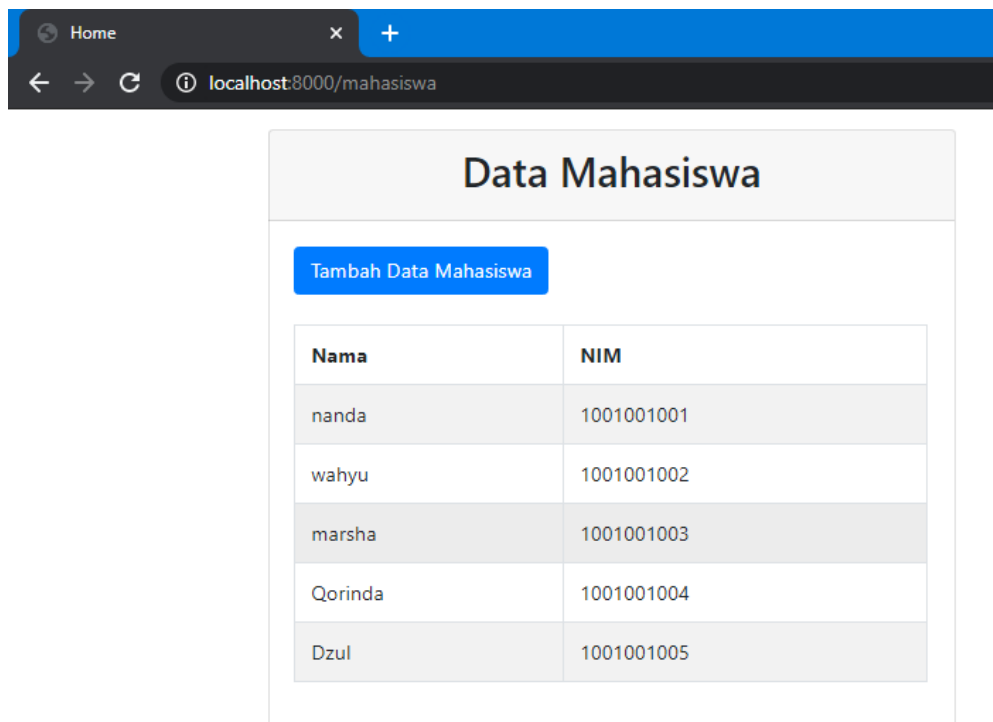
	 <pre> MahasiswaController.php X app > Http > Controllers > MahasiswaController.php 18 19 public function tambah() { 20 // memanggil view tambah 21 return view('tambah'); 22 } 23 } 24 </pre>
3	<p>Kemudian buatlah view tambah.blade.php yang berisi form untuk memasukkan data baru pada folder resources/views. View tambah juga mengaplikasikan master.blade.php.</p>  <pre> tambah.blade.php X resources > views > tambah.blade.php 1 extends('master') 2 3 -- Isi Title-- 4 section('title','Tambah Data') 5 6 -- Isi bagian judul halaman-- 7 section('judul_halaman','Tambah Data Mahasiswa') 8 9 -- Isi bagian konten-- 10 section('konten') 11 Kembali 12
 13
 14 <form action="/mahasiswa/simpan" method="post"> 15 {{ csrf_field() }} 16 <div class="form-group"> 17 <label for="namamhs">Nama</label> 18 <input type="text" class="form-control" required="required" name="namamhs">
 19 </div> 20 <div class="form-group"> 21 <label for="nimmhs">NIM</label> 22 <input type="text" class="form-control" required="required" name="nimmhs">
 23 </div> 24 <div class="form-group"> 25 <label for="emailmhs">E-mail</label> 26 <input type="text" class="form-control" required="required" name="emailmhs">
 27 </div> 28 <div class="form-group"> 29 <label for="jurusanmhs">Jurusan</label> 30 <input type="text" class="form-control" required="required" name="jurusanmhs">
 31 </div> 32 <button type="submit" name="tambah" class="btn btn-primary float-right">Tambah Data</button> 33 </form> 34 35 36 @endsection </pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - 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.
4	<p>Ketika tombol simpan ditekan, akan dipanggil routes /mahasiswa/simpan. Oleh karena itu, kita buat terlebih dahulu route tersebut.</p>  <pre> web.php X routes > web.php 24 Route::post('/mahasiswa/simpan', 'mahasiswaController@simpan'); </pre>

	<p>Keterangan:</p> <ul style="list-style-type: none"> - Pada route ini menggunakan metode post karena data mahasiswa dari form akan dikirim ke method simpan di MahasiswaController.php
5	<p>Buat method simpan pada MahasiswaController untuk menyimpan data ke database.</p>  <pre> 24 25 public function simpan(Request \$request){ 26 // insert data ke table mahasiswa 27 DB::table('mahasiswa')->insert([28 'nama' => \$request->namamhs, 29 'nim' => \$request->nimmhs, 30 'email' => \$request->emailmhs, 31 'jurusan' => \$request->jurusanmhs 32]); 33 return redirect('/mahasiswa'); 34 } 35 } 36 </pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - Variabel \$request untuk menerima data yang akan ditambahkan ke database - Line 28-33 merupakan query builder untuk insert data ke tabel mahasiswa
6	<p>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.</p> 

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



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



d. Melihat Detail Data dari Database

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

	 <pre> index.blade.php X resources > views > index.blade.php 1 @extends('master') 2 3 <!-- Isi Title--> 4 @section('title','Home') 5 6 <!-- Isi bagian judul halaman--> 7 @section('judul_halaman','Data Mahasiswa') 8 9 <!-- Isi bagian konten--> 10 @section('konten') 11 Tambah Data Mahasiswa 12
 13
 14 <table class="table table-bordered table-hover table-striped"> 15 <thead> 16 <tr> 17 <th>Nama</th> 18 <th>NIM</th> 19 <th></th> 20 </tr> 21 </thead> 22 <tbody>@foreach(\$mahasiswa as \$mhs) 23 <tr> 24 <td>{{ \$mhs->nama }}</td> 25 <td>{{ \$mhs->nim }}</td> 26 <td> 27 id }}" class="badge badge-info">Detail 28 </td> 29 </tr> 30 @endforeach 31 </tbody> 32 </table> 33 @endsection </pre>
2	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/detail yang akan menjalankan fungsi detail pada MahasiswaController</p>  <pre> web.php X routes > web.php 26 Route::get('/mahasiswa/detail/{id}', 'mahasiswaController@detail'); </pre>
3	<p>Buat method detail pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan data mahasiswa pada view detail.blade.php.</p>  <pre> MahasiswaController.php X app > Http > Controllers > MahasiswaController.php 36 37 public function detail(\$id){ 38 //mengambil data mahasiswa berdasarkan id yang dipilih 39 \$mahasiswa = DB::table('mahasiswa')->where('id',\$id)->get(); 40 //kirim data mahasiswa yang diambil ke view edit.blade.php 41 return view('detail',['mahasiswa' => \$mahasiswa]); 42 } 43 44 </pre>
4	<p>Kemudian buatlah view detail.blade.php yang menampilkan detail data mahasiswa pada folder resources/views. View detail juga mengaplikasikan master.blade.php.</p>

```

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

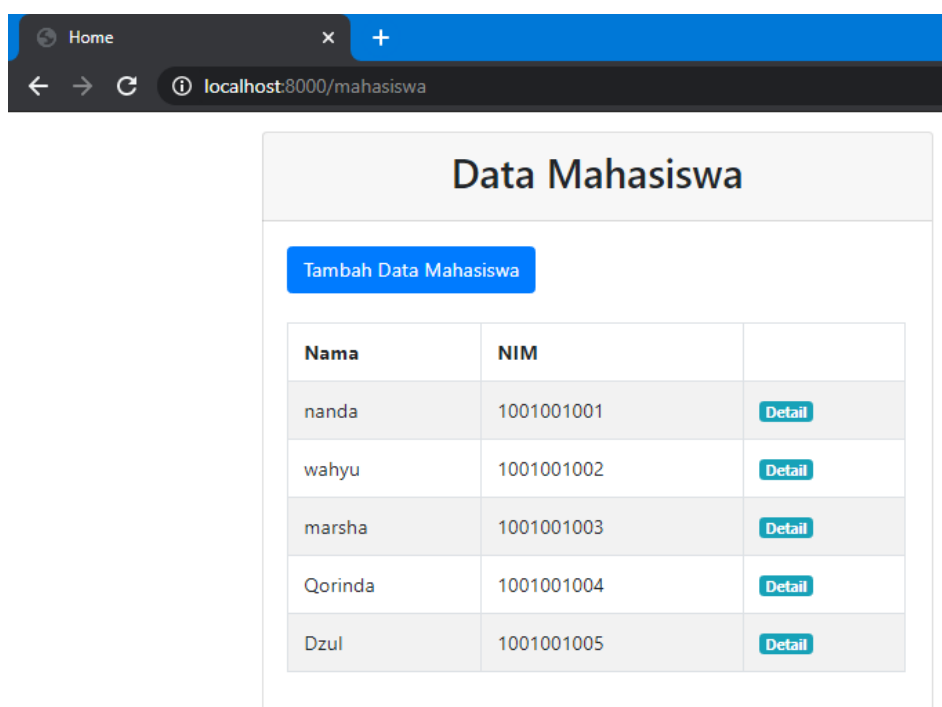
```

Keterangan:

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

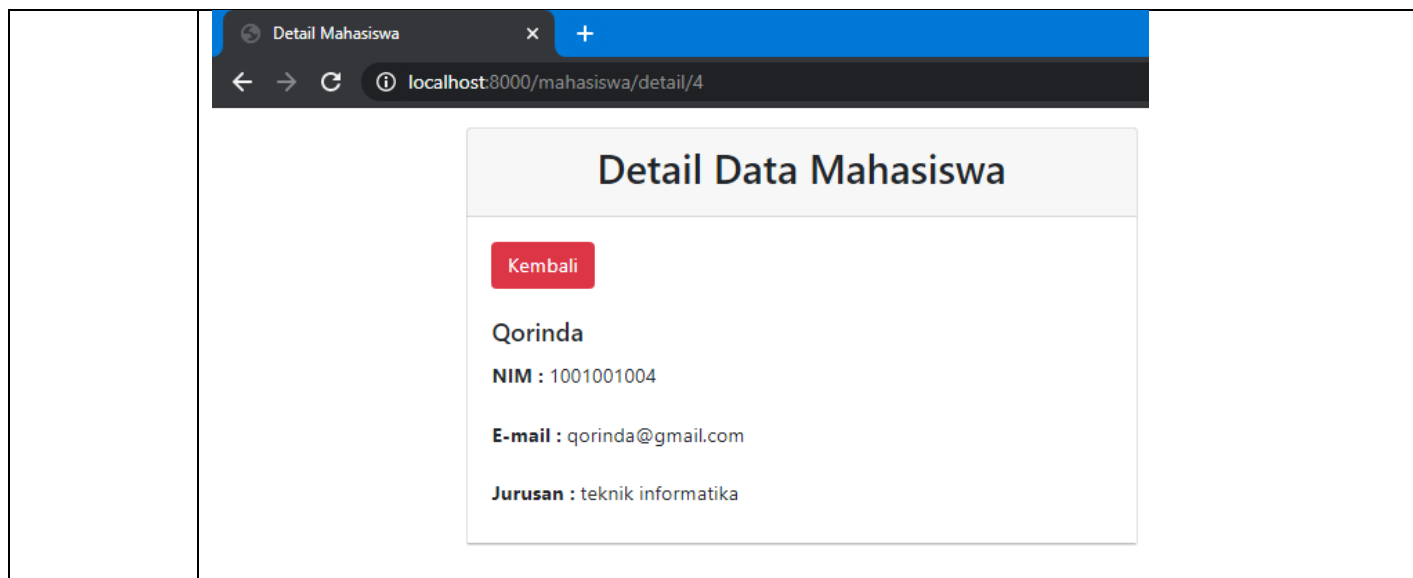
5

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



The screenshot shows a web application interface. At the top, there's a blue navigation bar with a 'Home' link and a search icon. Below this, the main content area has a light gray background. A blue button labeled 'Tambah Data Mahasiswa' is positioned at the top left of the content area. Below the button is a table with the following data:

Nama	NIM	
nanda	1001001001	Detail
wahyu	1001001002	Detail
marsha	1001001003	Detail
Qorinda	1001001004	Detail
Dzul	1001001005	Detail



e. Mengubah Data (Update) dari Database

Langkah	Keterangan
1	<p>Buatlah tombol untuk mengubah data (Line 29) pada file index.blade.php di folder resources/views</p> <pre> index.blade.php x resources > views > index.blade.php 1 @extends('master') 2 3 <!-- Isi Title--> 4 @section('title','Home') 5 6 <!-- Isi bagian judul halaman--> 7 @section('judul_halaman','Data Mahasiswa') 8 9 <!-- Isi bagian konten--> 10 @section('konten') 11 Tambah Data Mahasiswa 12
 13
 14 <table class="table table-bordered table-hover table-striped"> 15 <thead> 16 <tr> 17 <th>Nama</th> 18 <th>NIM</th> 19 <th></th> 20 </tr> 21 </thead> 22 <tbody>@foreach(\$mahasiswa as \$mhs) 23 <tr> 24 <td>{{ \$mhs->nama }}</td> 25 <td>{{ \$mhs->nim }}</td> 26 <td> 27 id }}" class="badge badge-info">Detail 28 id }}" class="badge badge-warning">Edit 29 </td> 30 </tr> 31 @endforeach 32 </tbody> 33 </table> 34 @endsection </pre>
2	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/edit yang akan menjalankan fungsi edit pada MahasiswaController</p> <pre> web.php x routes > web.php 28 Route::get('/mahasiswa/edit/{id}', 'mahasiswaController@edit'); </pre>

3

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

```

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

44
45     public function edit($id){
46         //mengambil data mahasiswa berdasarkan id yang dipilih
47         $mahasiswa = DB::table('mahasiswa')->where('id',$id)->get();
48         //kirim data mahasiswa yang diambil ke view edit.blade.php
49         return view('edit',['mahasiswa' => $mahasiswa]);
50     }
51 }
52

```

Keterangan :

- Line 49 : query builder untuk mengambil data mahasiswa berdasarkan id yang dipilih

4

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

```

edit.blade.php X
resources > views > edit.blade.php

1  @extends('master')
2
3  <!-- Isi Title-->
4  @section('title','Edit Data')
5
6  <!-- Isi bagian judul halaman-->
7  @section('judul_halaman','Edit Data Mahasiswa')
8
9  <!-- Isi bagian konten-->
10 @section('konten')
11 <a href="/mahasiswa" class="btn btn-danger">Kembali</a>
12 <br/>
13 <br/>
14
15 @foreach($mahasiswa as $mhs)
16 <form action="/mahasiswa/update" method="post">
17     {{ csrf_field() }}
18     <input type="hidden" name="id" value="{{ $mhs->id }}"><br/>
19     <div class="form-group">
20         <label for="namamhs">Nama</label>
21         <input type="text" name="namamhs" class="form-control" required="required" value="{{ $mhs->nama }}"> <br/>
22     </div>
23     <div class="form-group">
24         <label for="nimhs">NIM</label>
25         <input type="number" name="nimhs" class="form-control" required="required" value="{{ $mhs->nim }}"> <br/>
26     </div>
27     <div class="form-group">
28         <label for="emailhs">E-mail</label>
29         <input type="email" name="emailhs" class="form-control" required="required" value="{{ $mhs->email }}"> <br/>
30     </div>
31     <div class="form-group">
32         <label for="jurusanhs">Jurusan</label>
33         <input type="text" name="jurusanhs" class="form-control" required="required" value="{{ $mhs->jurusan }}"> <br/>
34     </div>
35     <button type="submit" name="tambah" class="btn btn-primary float-right">Simpan Data</button>
36 </form>
37 @endforeach
38 @endsection

```

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`

5

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

```

web.php X
routes > web.php

30 Route::post('/mahasiswa/update', 'mahasiswaController@update');

```

Keterangan:

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

6

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

```
MahasiswaController.php X
app > Http > Controllers > MahasiswaController.php
52
53     public function update(Request $request){
54         //update data mahasiswa
55         DB::table('mahasiswa')->where('id',$request->id)->update([
56             'nama' => $request->namamhs,
57             'nim' => $request->nimmhs,
58             'email' => $request->emailmhs,
59             'jurusan' => $request->jurusanmhs
60         ]);
61         return redirect('/mahasiswa');
62     }
63 }
64
```

Keterangan:

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

7

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

Home

localhost:8000

Data Mahasiswa

Tambah Data Mahasiswa

Nama	NIM	
nanda	1001001001	Detail Edit
wahyu	1001001002	Detail Edit
marsha	1001001003	Detail Edit
Qorinda	1001001004	Detail Edit
Dzul	1001001005	Detail Edit

Klik edit di data pertama, kita akan mengubah namanya.

Edit Data

localhost:8000/mahasiswa/edit/5

Edit Data Mahasiswa

Kembali

Nama

Dzul

NIM

1001001005

E-mail

dzul@gmail.com

Jurusan

teknik mesin

Simpan Data

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

Home

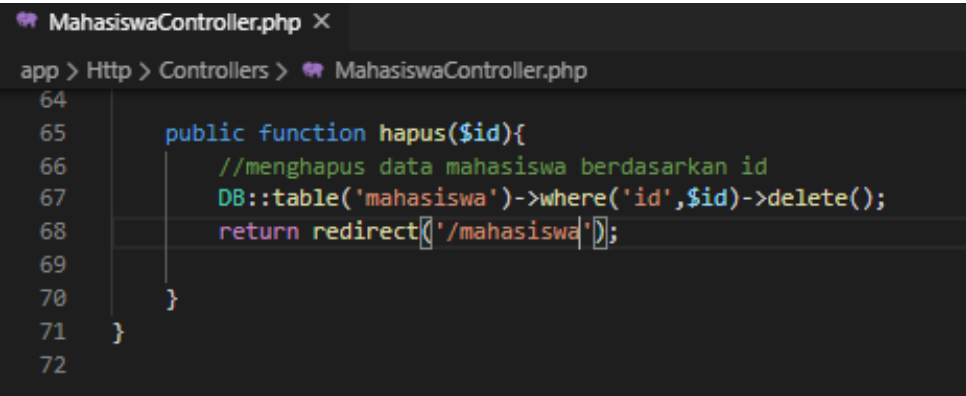
localhost:8000/mahasiswa

Data Mahasiswa

Tambah Data Mahasiswa

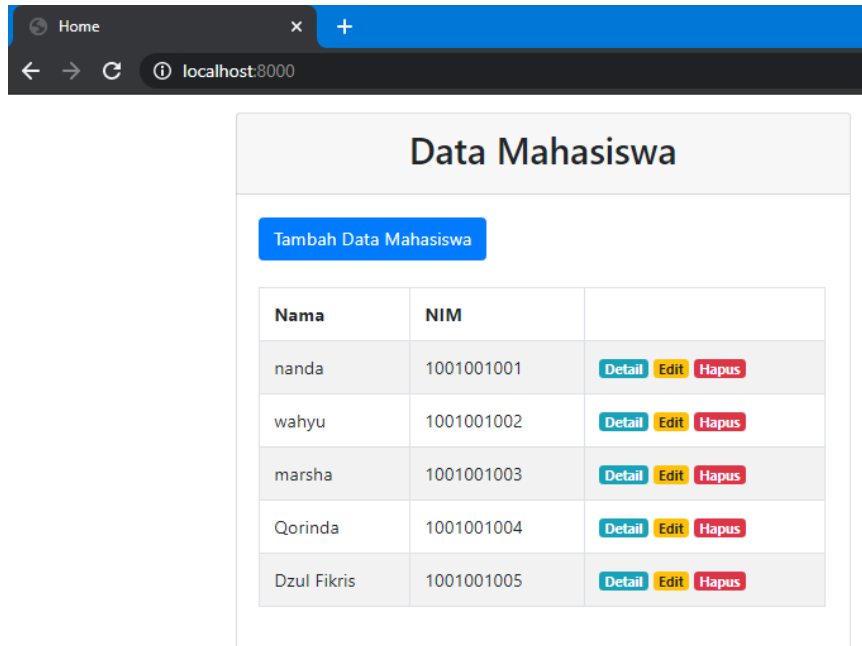
Nama	NIM	
nanda	1001001001	<div>Detail Edit</div>
wahyu	1001001002	<div>Detail Edit</div>
marsha	1001001003	<div>Detail Edit</div>
Qorinda	1001001004	<div>Detail Edit</div>
Dzul Fikris	1001001005	<div>Detail Edit</div>

f. Menghapus Data (Delete) dari Database

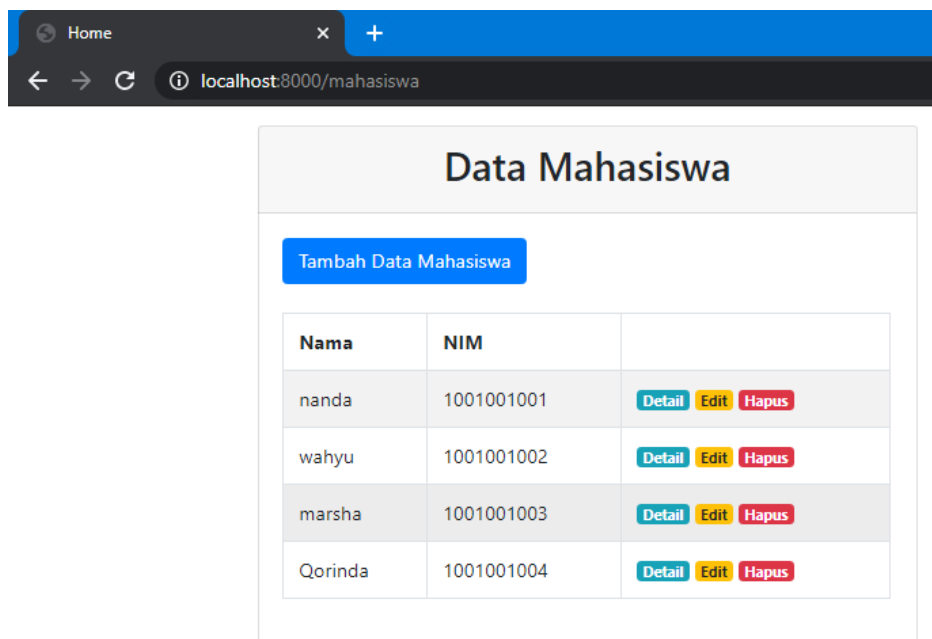
Langkah	Keterangan
1	<p>Buatlah tombol untuk menghapus data (Line 30) pada file index.blade.php di folder resources/views</p>  <pre> index.blade.php X resources > views > index.blade.php 1 @extends('master') 2 3 <!-- Isi Title--> 4 @section('title','Home') 5 6 <!-- Isi bagian judul halaman--> 7 @section('judul_halaman','Data Mahasiswa') 8 9 <!-- Isi bagian konten--> 10 @section('konten') 11 Tambah Data Mahasiswa 12
 13
 14 <table class="table table-bordered table-hover table-striped"> 15 <thead> 16 <tr> 17 <th>Nama</th> 18 <th>NIM</th> 19 <th></th> 20 </tr> 21 </thead> 22 <tbody>@foreach(\$mahasiswa as \$mhs) 23 <tr> 24 <td>{{ \$mhs->nama }}</td> 25 <td>{{ \$mhs->nim }}</td> 26 <td> 27 id }}" class="badge badge-info">Detail 28 id }}" class="badge badge-warning">Edit 29 id }}" class="badge badge-danger">Hapus 30 </td> 31 </tr> 32 </tbody> 33 @endforeach 34 </table> 35 @endsection </pre>
2	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/hapus yang akan menjalankan fungsi hapus pada MahasiswaController</p>  <pre> web.php X routes > web.php 34 Route::get('/mahasiswa/hapus/{id}', 'mahasiswaController@hapus'); </pre>
3	<p>Buat method hapus pada MahasiswaController.php di folder app/Http/Controllers yang akan menjalankan fungsi hapus.</p>  <pre> MahasiswaController.php X app > Http > Controllers > MahasiswaController.php 64 65 public function hapus(\$id){ 66 //menghapus data mahasiswa berdasarkan id 67 DB::table('mahasiswa')->where('id',\$id)->delete(); 68 return redirect('/mahasiswa'); 69 } 70 71 } 72 </pre> <p>Keterangan :</p> <ul style="list-style-type: none"> - Line 67 : query builder untuk menghapus data mahasiswa berdasarkan id yang dipilih

4

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



Hasil setelah di hapus



Praktikum – Bagian 2: Membuat CRUD di Laravel menggunakan Eloquent

a. Konfigurasi Database

Langkah	Keterangan
1	<p>Buatlah project Laravel baru dengan nama laravel-crud-kedua. Buka command prompt, tuliskan perintah berikut.</p> <pre>cd C:\xampp\htdocs</pre> <pre>laravel new laravel-crud-kedua</pre>


```

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

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/infectoer (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 (1.4.0): Loading from cache
- Installing symfony/polyfill-ctype (v1.15.0): Loading from cache
- Installing phpoption/phpoption (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
- Installing symfony/event-dispatcher (v5.0.7): Loading from cache
- Installing psr/log (1.1.3): Loading from cache

psr/psr suggests installing ext-pdo-sqlite (The doc command requires SQLite to work.)
psr/psr suggests installing ext-posix (If you have PCNTL, you'll want the POSIX extension as well.)
psr/psr suggests installing hoar/console (A pure PHP readline implementation. You'll want this if your PHP install does
n't already support readline or libedit.)
file/whoops suggests installing whoops/soap (Formats errors as SOAP responses)
facade/ignition suggests installing laravel/telescope (^3.1)
sebastian/global-state suggests installing ext-uopz (*)
sebastian/environment suggests installing ext-posix (*)
phpunit/php-code-coverage suggests installing ext-xdebug (^2.7.2)
phpunit/phpunit suggests installing ext-soap (*)
phpunit/phpunit suggests installing ext-xdebug (*)
phpunit/phpunit suggests installing phpunit/php-invoker (^2.0.0)
Generating optimized autoload files
16 packages you are using are looking for funding.
Use the composer fund command to find out more!
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
> @php artisan key:generate --ansi
Application key set successfully.
> illuminate/Foundation/ComposerScripts:postAutoloadDump
> @php artisan package:discover --ansi
Discovered Package: facade/ignition
Discovered Package: fideloper/proxy
Discovered Package: fruitcake/laravel-cors
Discovered Package: laravel/tinker
Discovered Package: nesbot/carbon
Discovered Package: nunomaduro/collision
Package manifests generated successfully.
Application ready! Build something amazing.

C:\xampp\htdocs>

```

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

```

.env
.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:ZR1UibwZ1bED0VLM14BxEgytIZf3+NYjpYwWSHbU
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=
15

```

Keterangan:

- Nama database yang akan digunakan adalah latihan_laravel dengan username root.

3 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

#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra	Tindakan
1	id	int(11)		Tidak	Tidak ada			AUTO_INCREMENT	Ubah Hapus Lainnya
2	nama	varchar(100)	latin1_swedish_ci	Tidak	Tidak ada				Ubah Hapus Lainnya
3	nim	varchar(10)	latin1_swedish_ci	Tidak	Tidak ada				Ubah Hapus Lainnya
4	email	varchar(50)	latin1_swedish_ci	Tidak	Tidak ada				Ubah Hapus Lainnya
5	jurusan	varchar(20)	latin1_swedish_ci	Tidak	Tidak ada				Ubah Hapus Lainnya
6	created_at	timestamp		Ya	NULL				Ubah Hapus Lainnya
7	updated_at	timestamp		Ya	NULL				Ubah Hapus Lainnya

b. Menampilkan Data dari Database

Langkah	Keterangan
1	<p>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.</p>  <pre>web.php x routes > web.php 19 20 Route::get('/', 'MahasiswaController@index'); 21 22 Route::get('/mahasiswa', 'MahasiswaController@index'); 23</pre>
2	<p>Buat model menggunakan command prompt dengan nama Mahasiswa menggunakan php artisan</p> <p>cd laravel-crud-kedua</p> <p>php artisan make:model Mahasiswa</p>  <pre>C:\xampp\htdocs\laravel-crud-kedua>php artisan make:model Mahasiswa Model created successfully.</pre>
3	<p>Ubah model Mahasiswa.php pada folder App menjadi seperti berikut.</p>  <pre>Mahasiswa.php x 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</pre> <p>Keterangan:</p> <ul style="list-style-type: none">- Model Mahasiswa akan menangani tabel mahasiswa
4	<p>Buat controller baru yaitu MahasiswaController menggunakan php artisan</p> <p>php artisan make:controller MahasiswaController</p>  <pre>C:\xampp\htdocs\laravel-crud-kedua>php artisan make:controller MahasiswaController Controller created successfully.</pre>
5	<p>Buat method index pada MahasiswaController.php pada folder app/Http/Controllers</p>

```

MahasiswaController.php X
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         $mahasiswa = Mahasiswa::all();
12         return view('index', ['mahasiswa' => $mahasiswa]);
13     }
14 }
15

```

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

6

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)

```

index.blade.php  master.blade.php X
resources > views > master.blade.php
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" rel="stylesheet">
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                         <!-- bagian judul halaman -->
16                         <h2> @yield('judul_halaman')</h2>
17                     </div>
18                     <div class="card-body">
19                         <!-- bagian konten blog -->
20                         @yield('konten')
21                     </div>
22                 </div>
23             </div>
24         </div>
25     </div>
26 </body>
27 </html>

```

```

1  index.blade.php x
resources > views > index.blade.php
2  @extends('master')
3
4  <!-- Isi Title-->
5  @section('title','Home')
6
7  <!-- Isi bagian judul halaman-->
8  @section('judul_halaman','Data Mahasiswa')
9
10 <!-- Isi bagian konten-->
11 @section('konten')
12     <a href="/mahasiswa/tambah" class="btn btn-primary">Tambah Data Mahasiswa</a>
13     <br/>
14     <table class="table table-bordered table-hover table-striped">
15         <thead>
16             <tr>
17                 <th>Nama</th>
18                 <th>NIM</th>
19             </tr>
20         </thead>
21         <tbody>@foreach($mahasiswa as $mhs)
22             <tr>
23                 <td>{{ $mhs->nama }}</td>
24                 <td>{{ $mhs->nim }}</td>
25                 <td>
26                     <a href="/mahasiswa/detail/{{ $mhs->id }}" class="badge badge-info">Detail</a>
27                     <a href="/mahasiswa/edit/{{ $mhs->id }}" class="badge badge-warning">Edit</a>
28                     <a href="/mahasiswa/hapus/{{ $mhs->id }}" class="badge badge-danger">Hapus</a>
29                 </td>
30             </tr>
31         @endforeach
32     </tbody>
33 </table>
34 @endsection

```

7

Jalankan command prompt, tuliskan perintah untuk menjalankan project laravel-crud

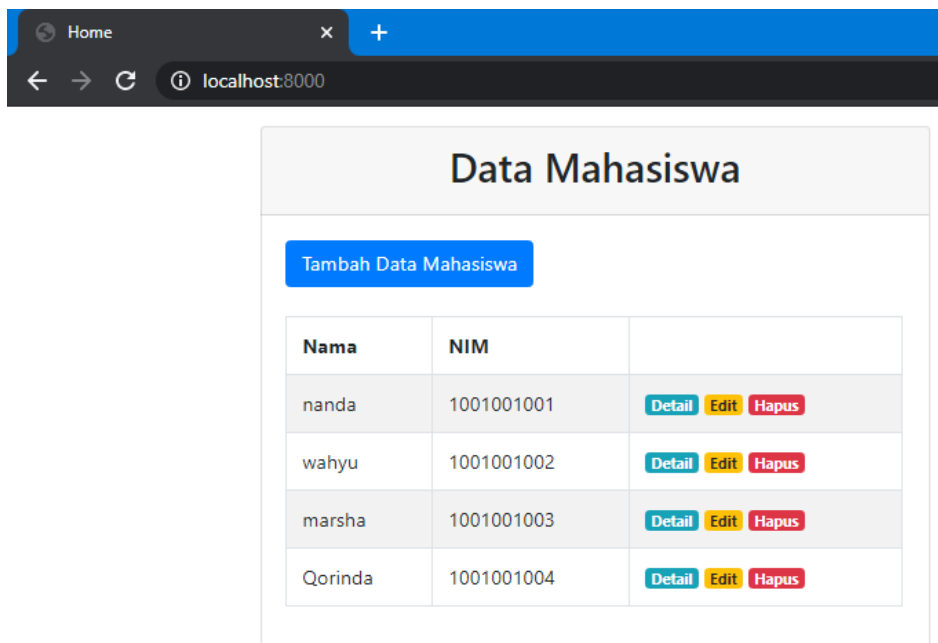
```
php artisan serve
```

```

C:\x9mbb\µfcqoc2\J99999-J-CLNq-κeqn9>bµb 99fJ299 26996

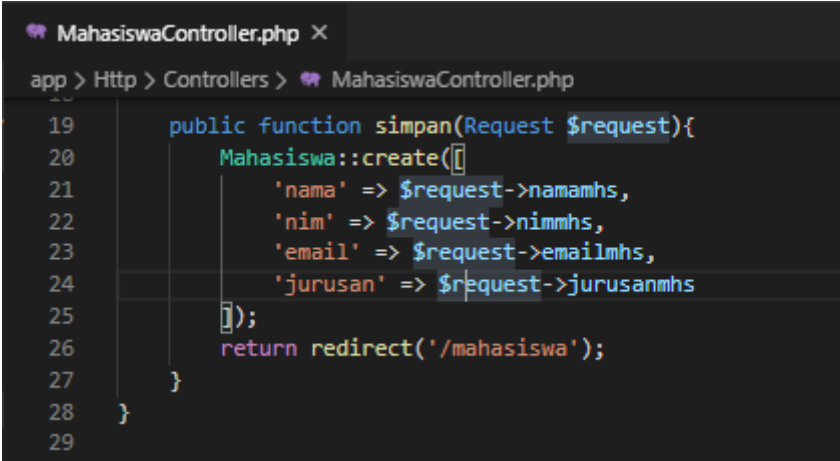
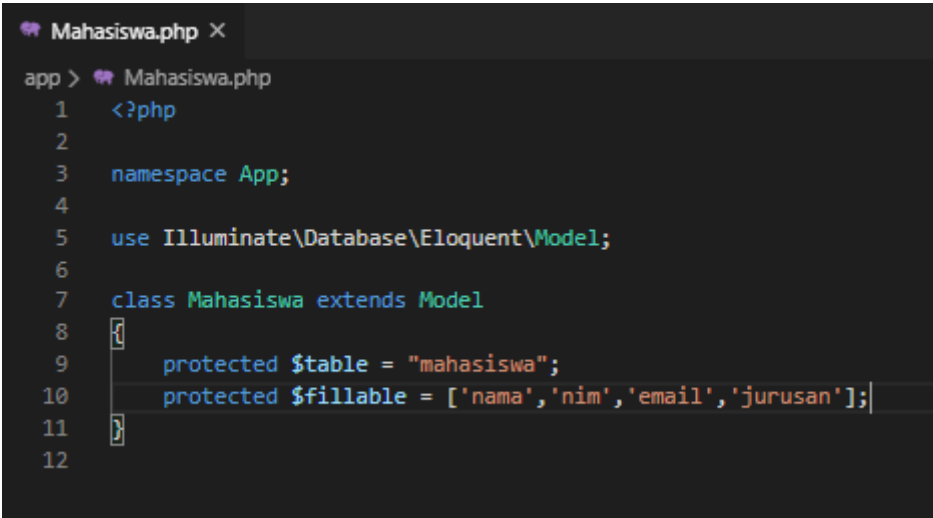
```

Buka browser dan ketikkan localhost:8000, maka akan tampil sebagai berikut



c. Memasukkan Data (Create) ke Database

Langkah	Keterangan
1	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/tambah yang akan menjalankan fungsi tambah pada MahasiswaController ketika tombol Tambah Data Mahasiswa ditekan.</p>  <pre> web.php routes > web.php 23 24 Route::get('/mahasiswa/tambah', 'MahasiswaController@tambah'); 25 </pre>
2	<p>Buat method tambah pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view tambah.</p>  <pre> MahasiswaController.php app > Http > Controllers > MahasiswaController.php 14 15 public function tambah(){ 16 return view('tambah'); 17 } 18 19 </pre>
3	<p>Kemudian buatlah view tambah.blade.php yang berisi form untuk memasukkan data baru pada folder resources/views.</p>  <pre> tambah.blade.php resources > views > tambah.blade.php 1 @extends('master') 2 3 <!-- Isi Title--> 4 @section('title','Tambah Data') 5 6 <!-- Isi bagian judul halaman--> 7 @section('judul_halaman','Tambah Data Mahasiswa') 8 9 <!-- Isi bagian konten--> 10 @section('konten') 11 Kembali 12
 13
 14 <form action="/mahasiswa/simpan" method="post"> 15 {{ csrf_field() }} 16 <div class="form-group"> 17 <label for="namamhs">Nama</label> 18 <input type="text" class="form-control" required="required" name="namamhs">
 19 </div> 20 <div class="form-group"> 21 <label for="nimhs">NIM</label> 22 <input type="text" class="form-control" required="required" name="nimhs">
 23 </div> 24 <div class="form-group"> 25 <label for="emailhs">E-mail</label> 26 <input type="text" class="form-control" required="required" name="emailhs">
 27 </div> 28 <div class="form-group"> 29 <label for="jurusanhs">Jurusan</label> 30 <input type="text" class="form-control" required="required" name="jurusanhs">
 31 </div> 32 <button type="submit" name="tambah" class="btn btn-primary float-right">Tambah Data</button> 33 </form> 34 35 36 @endsection </pre> <p>Kita lakukan copy paste dari tambah.blade.php di Bagian 1</p>
4	<p>Ketika tombol simpan ditekan, akan dipanggil routes /mahasiswa/simpan. Oleh karena itu, kita buat terlebih dahulu route tersebut.</p>

	 <pre> web.php x routes > web.php 26 Route::post('/mahasiswa/simpan', 'MahasiswaController@simpan'); 27 </pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - Pada route ini menggunakan metode post karena data mahasiswa dari form akan dikirim ke method simpan di MahasiswaController.php
5	<p>Buat method simpan pada MahasiswaController untuk menyimpan data ke database.</p>  <pre> MahasiswaController.php x app > Http > Controllers > MahasiswaController.php 19 public function simpan(Request \$request){ 20 Mahasiswa::create([21 'nama' => \$request->namamhs, 22 'nim' => \$request->nimmhs, 23 'email' => \$request->emailmhs, 24 'jurusan' => \$request->jurusanmhs 25]); 26 return redirect('/mahasiswa'); 27 } 28 } 29 </pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - 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
6	<p>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.</p>  <pre> Mahasiswa.php x 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 protected \$fillable = ['nama', 'nim', 'email', 'jurusan']; 11 } 12 </pre>
7	<p>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.</p>

Tambah Data Mahasiswa

[Kembali](#)

Nama
rahmat

NIM
1001001006

E-mail
rahmat@gmail.com

Jurusan
teknik mesin

[Tambah Data](#)

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

Data Mahasiswa

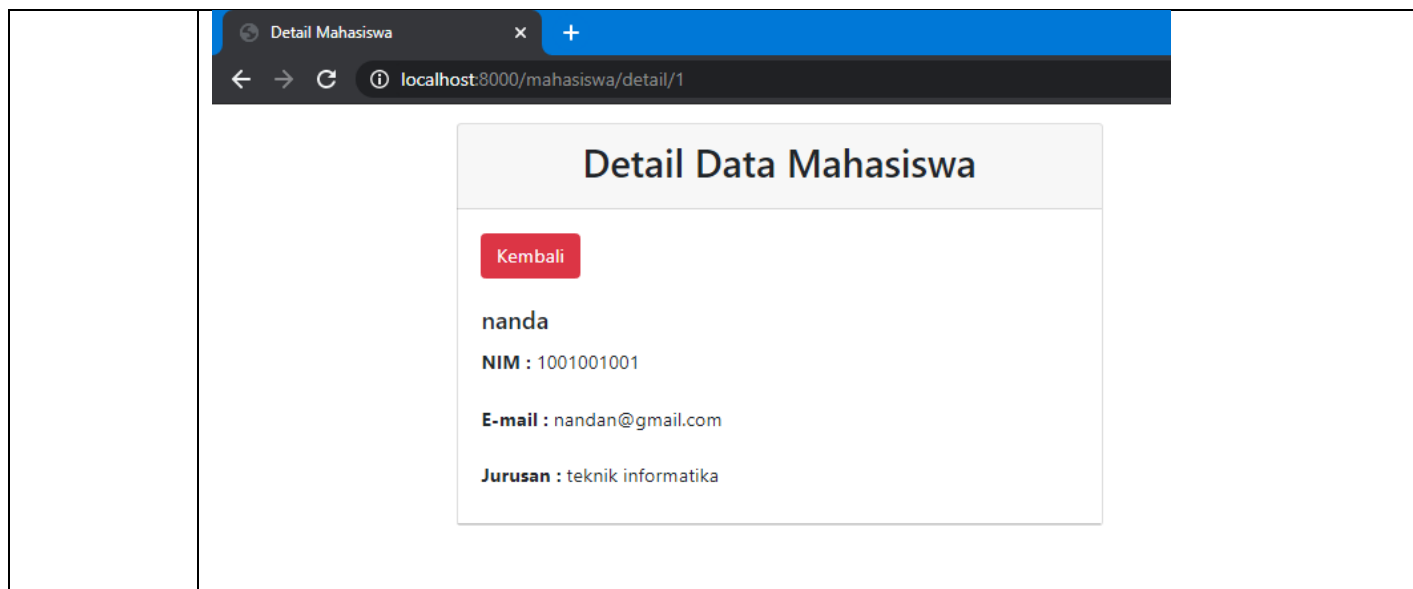
[Tambah Data Mahasiswa](#)

Nama	NIM	
nanda	1001001001	Detail Edit Hapus
wahyu	1001001002	Detail Edit Hapus
marsha	1001001003	Detail Edit Hapus
Qorinda	1001001004	Detail Edit Hapus
rahmat	1001001006	Detail Edit Hapus

d. Melihat Detail Data dari Database

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

	 <pre> web.php x routes > web.php 27 28 Route::get('/mahasiswa/detail/{id}', 'MahasiswaController@detail'); 29 </pre>
3	<p>Buat method detail pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan data mahasiswa pada view detail.blade.php.</p>  <pre> MahasiswaController.php app > Http > Controllers > MahasiswaController.php 28 29 public function detail(\$id){ 30 \$mahasiswa = Mahasiswa::find(\$id); 31 return view('detail', ['mahasiswa' => \$mahasiswa]); 32 } 33 } 34 </pre>
4	<p>Kemudian buatlah view detail.blade.php yang menampilkan detail data mahasiswa pada folder resources/views. View detail juga mengaplikasikan master.blade.php.</p>  <pre> detail.blade.php x resources > views > detail.blade.php 1 @extends('master') 2 3 <!-- Isi Title--> 4 @section('title','Detail Mahasiswa') 5 6 <!-- Isi bagian judul halaman--> 7 @section('judul_halaman','Detail Data Mahasiswa') 8 9 <!-- Isi bagian konten--> 10 @section('konten') 11 Kembali 12
 13
 14 <h5 class="card-title">{{ \$mahasiswa->nama }}</h5> 15 <p class="card-text"> 16 <label for="">NIM :</label> 17 {{ \$mahasiswa->nim }}</p> 18 <p class="card-text"> 19 <label for="">E-mail :</label> 20 {{ \$mahasiswa->email }}</p> 21 <p class="card-text"> 22 <label for="">Jurusan :</label> 23 {{ \$mahasiswa->jurusan }}</p> 24 25 26 @endsection </pre>
5	<p>Jalankan localhost:8000 dan pilih tombol 'Detail' di suatu data yang ingin kita lihat detailnya.</p>



e. Mengubah Data (Update) dari Database

Langkah	Keterangan
1	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/edit yang akan menjalankan fungsi edit pada MahasiswaController</p> <pre> web.php x routes > web.php 30 Route::get('/mahasiswa/edit/{id}', 'MahasiswaController@edit'); 31 </pre>
2	<p>Buat method edit pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view edit.</p> <pre> MahasiswaController.php x app > Http > Controllers > MahasiswaController.php 34 public function edit(\$id){ 35 \$mahasiswa = Mahasiswa::find(\$id); 36 return view('edit', ['mahasiswa' => \$mahasiswa]); 37 } 38 } 39 </pre> <p>Keterangan :</p> <ul style="list-style-type: none"> - Line 41 : fungsi eloquent untuk mengambil data mahasiswa berdasarkan id yang dipilih
3	<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>

```

editblade.php X
resources > views > editblade.php
1 @extends('master')
2
3 <!-- Isi Title-->
4 @section('title','Edit Data')
5
6 <!-- Isi bagian judul halaman-->
7 @section('judul_halaman','Edit Data Mahasiswa')
8
9 <!-- Isi bagian konten-->
10 @section('konten')
11 <a href="/mahasiswa" class="btn btn-danger">Kembali</a>
12 <br/>
13 <br/>
14
15 <form action="/mahasiswa/update/{{ $mahasiswa->id }}" method="post">
16     {{ csrf_field() }}
17     <input type="hidden" name="id" value="{{ $mahasiswa->id }}"><br/>
18     <div class="form-group">
19         <label for="namamhs">Nama</label>
20         <input type="text" name="namamhs" class="form-control" required="required" value="{{ $mahasiswa->nama }}"> <br/>
21     </div>
22     <div class="form-group">
23         <label for="nimhs">NIM</label>
24         <input type="number" name="nimhs" class="form-control" required="required" value="{{ $mahasiswa->nim }}"> <br/>
25     </div>
26     <div class="form-group">
27         <label for="emailhs">E-mail</label>
28         <input type="email" name="emailhs" class="form-control" required="required" value="{{ $mahasiswa->email }}"> <br/>
29     </div>
30     <div class="form-group">
31         <label for="jurusanhs">Jurusan</label>
32         <input type="text" name="jurusanhs" class="form-control" required="required" value="{{ $mahasiswa->jurusan }}"> <br/>
33     </div>
34     <button type="submit" name="tambah" class="btn btn-primary float-right">Simpan Data</button>
35 </form>
36
37 @endsection
38

```

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

4

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

```

web.php X
routes > web.php
31
32 Route::post('/mahasiswa/update/{id}', 'MahasiswaController@update');
33

```

Keterangan:

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

5

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

```

MahasiswaController.php X
app > Http > Controllers > MahasiswaController.php
39 public function update($id, Request $request)
40 {
41     $mahasiswa = Mahasiswa::find($id);
42     $mahasiswa->nama = $request->namamhs;
43     $mahasiswa->nim = $request->nimmhs;
44     $mahasiswa->email = $request->emailmhs;
45     $mahasiswa->jurusan = $request->jurusanmhs;
46     $mahasiswa->save();
47     return redirect('/mahasiswa');
48 }
49

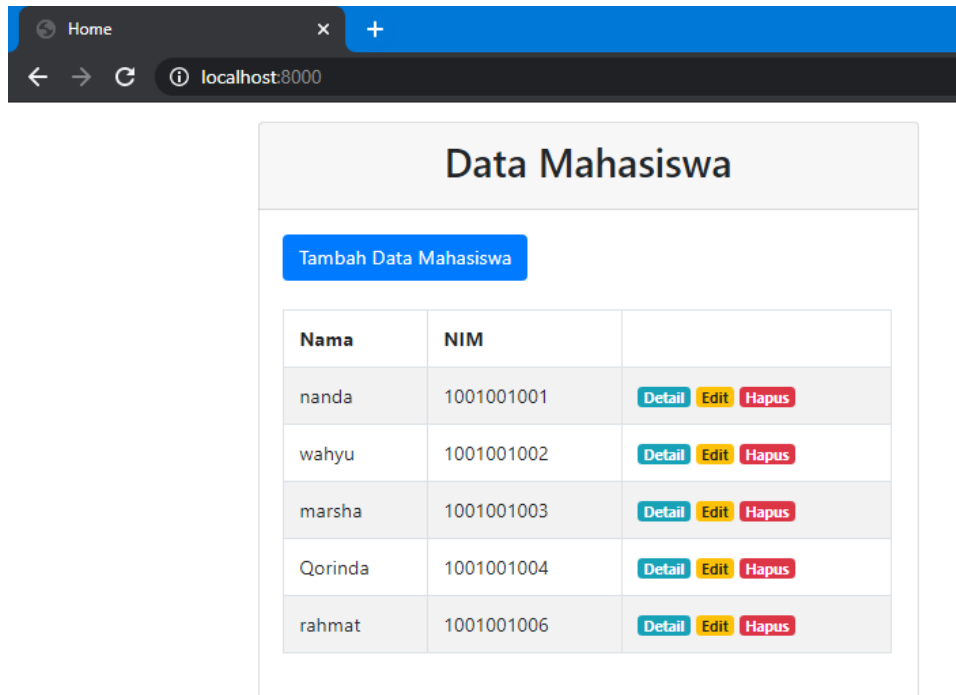
```

Keterangan:

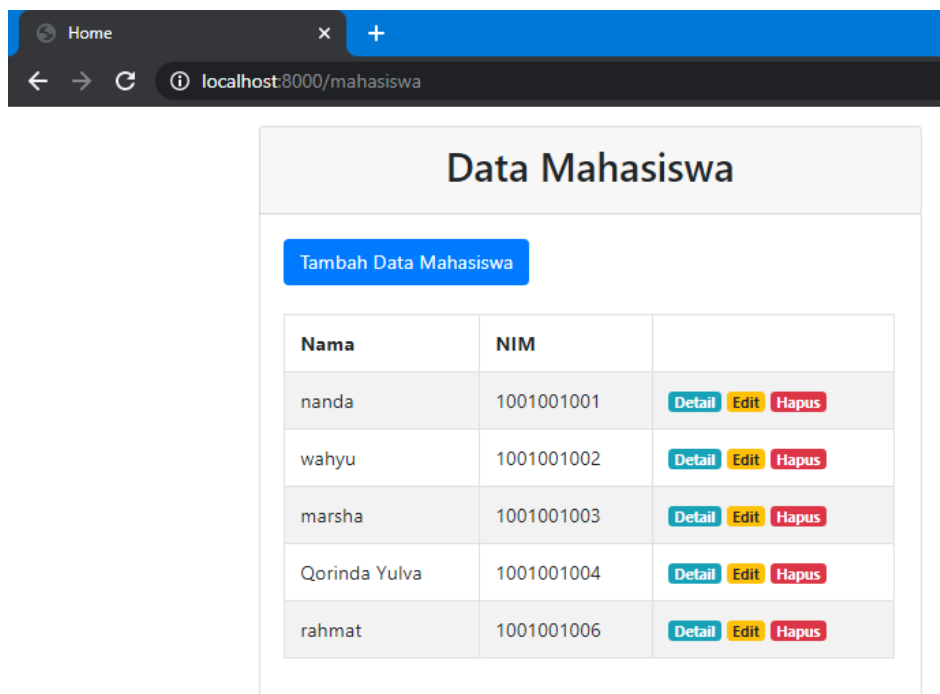
- Variabel \$request untuk menerima data yang akan ditambahkan ke database
- Line 48-52 merupakan fungsi eloquent untuk update data ke tabel mahasiswa

6

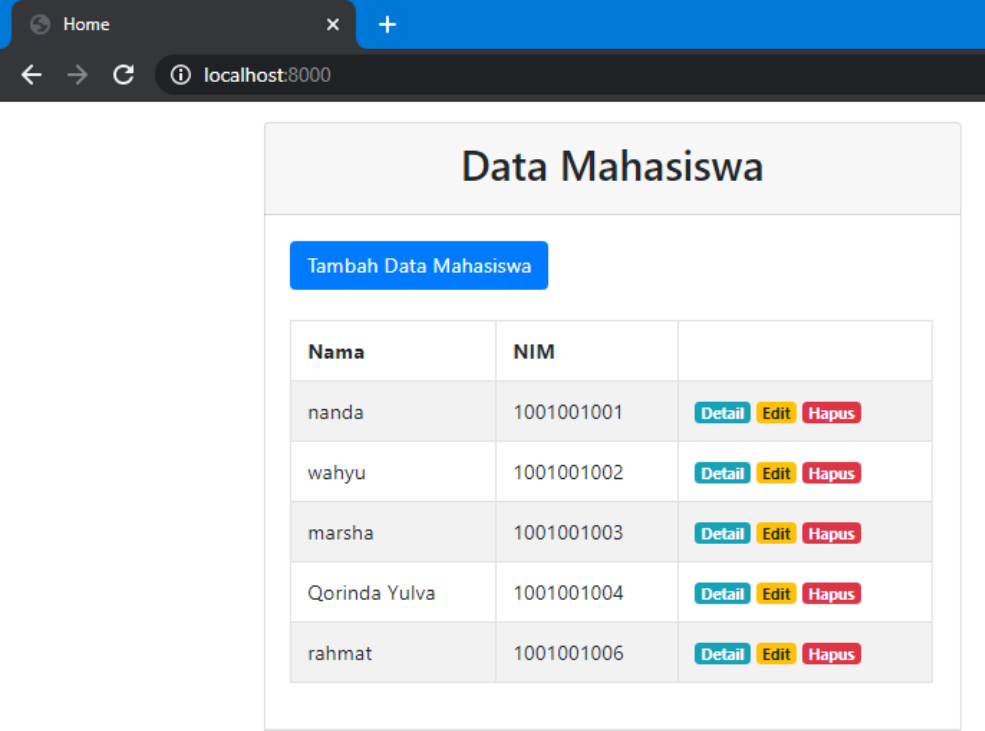
Jalankan localhost:8000 dan pilih tombol 'Edit', maka akan ditampilkan halaman edit yang berisi form untuk mengubah data baru.



Cobalah untuk mengedit data.



f. Menghapus Data (Delete) dari Database

Langkah	Keterangan
1	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/hapus yang akan menjalankan fungsi hapus pada MahasiswaController</p> <pre> web.php x routes > web.php 33 34 Route::get('/mahasiswa/hapus/{id}', 'MahasiswaController@hapus'); 35 </pre>
2	<p>Buat method hapus pada MahasiswaController.php di folder app/Http/Controllers yang akan menjalankan fungsi hapus.</p> <pre> MahasiswaController.php x app > Http > Controllers > MahasiswaController.php 48 49 public function hapus(\$id){ 50 \$mahasiswa = Mahasiswa::find(\$id); 51 \$mahasiswa->delete(); 52 return redirect('/mahasiswa'); 53 } 54 </pre> <p>Keterangan :</p> <ul style="list-style-type: none"> - Line 50 :fungsi eloquent untuk menghapus data mahasiswa berdasarkan id yang dipilih
3	<p>Jalankan localhost:8000 dan pilih tombol 'Hapus', maka data yang terpilih akan dihapus.</p> 

Setelah di Hapus

The screenshot shows a web browser window with the address bar displaying 'localhost:8000/mahasiswa'. The page title is 'Data Mahasiswa'. Below the title is a blue button labeled 'Tambah Data Mahasiswa'. A table displays the following data:

Nama	NIM	
nanda	1001001001	Detail Edit Hapus
wahyu	1001001002	Detail Edit Hapus
marsha	1001001003	Detail Edit Hapus
Qorinda Yulva	1001001004	Detail Edit Hapus