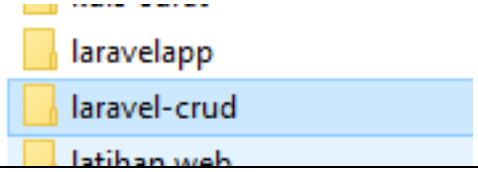
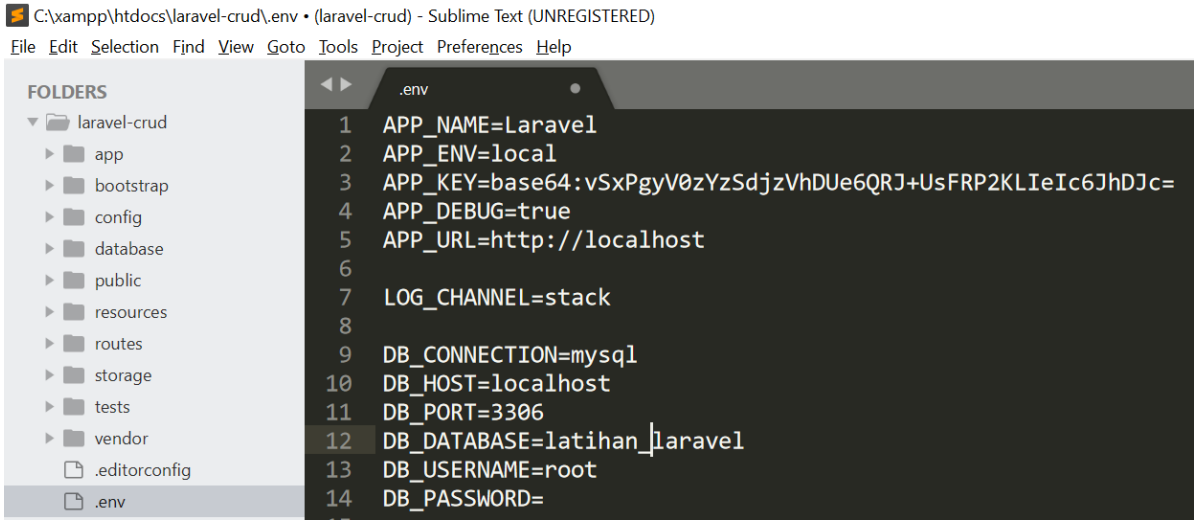
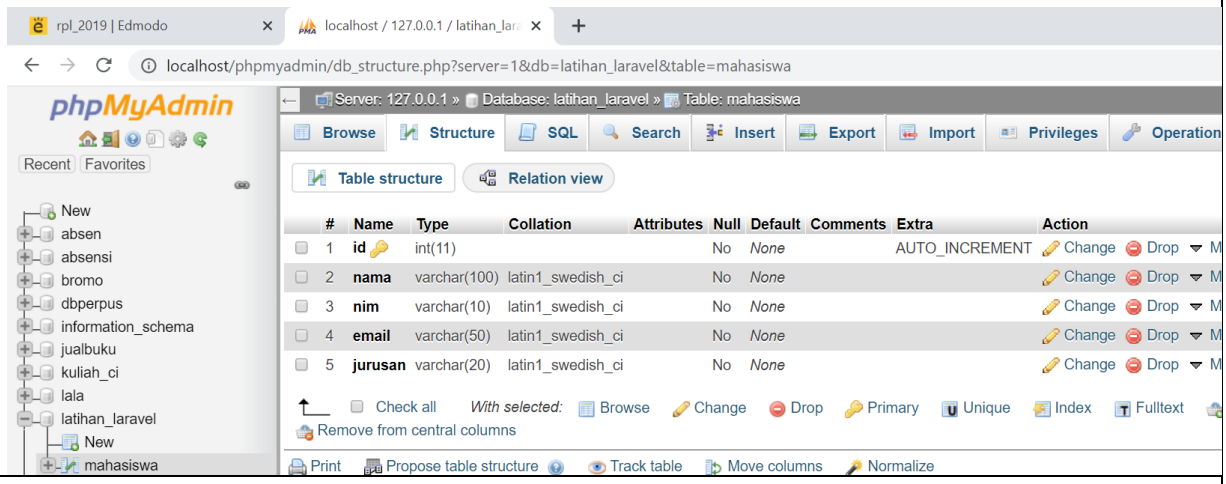
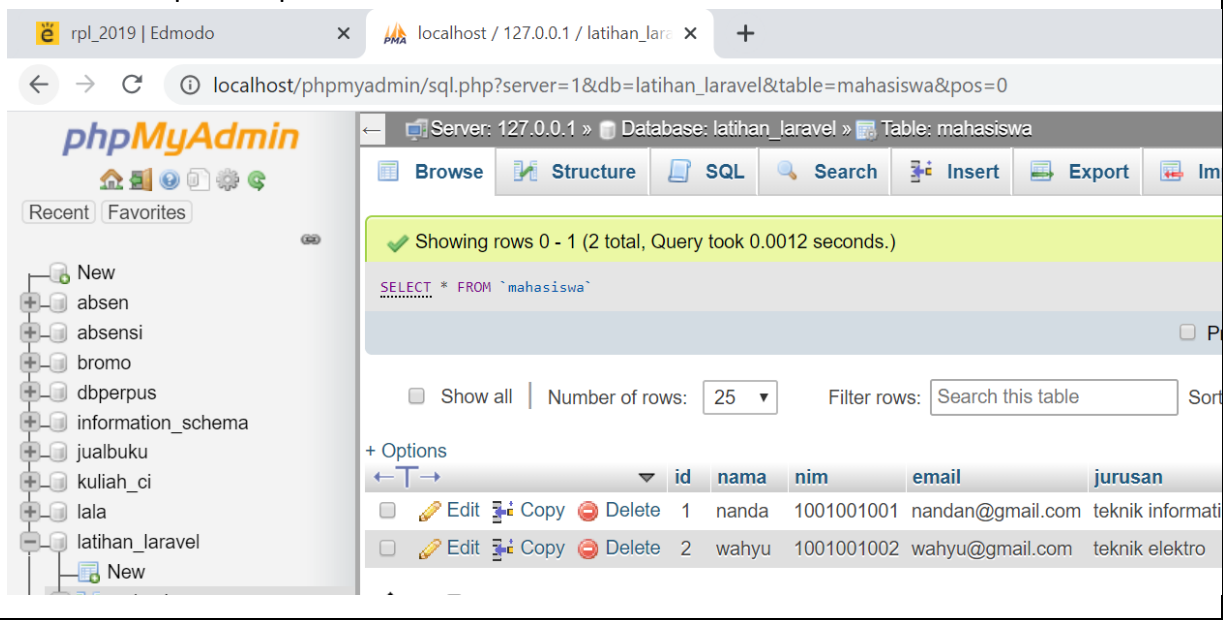
LAPORAN PRAKTIKUM
PEMROGRAMAN WEB LANJUT
Jobsheet-11
CRUD Laravel

NAMA : Dimas Tranggono Adi
NIM : 1841720199

a. Konfigurasi Database dan Pembuatan Tabel di MySQL

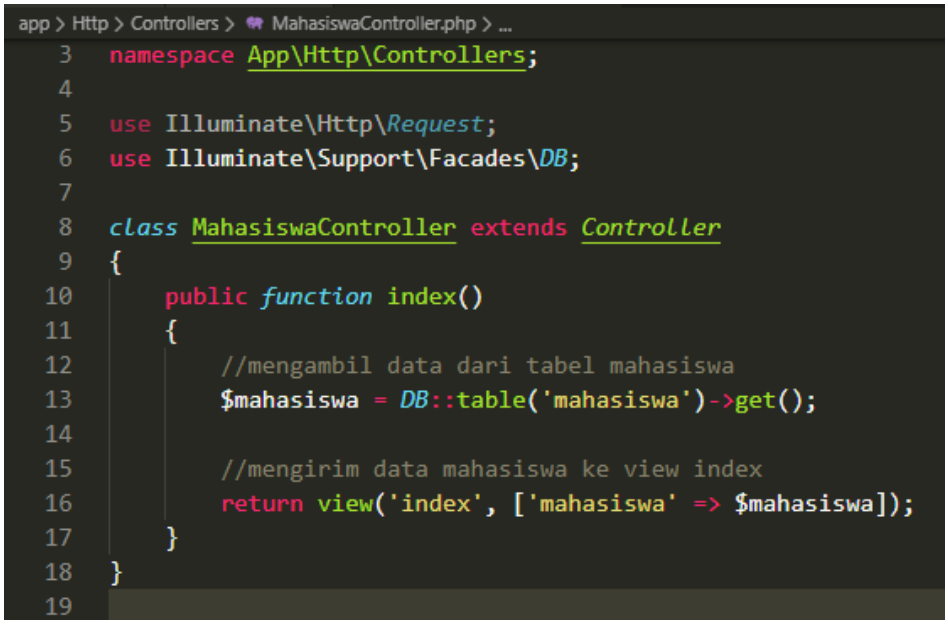
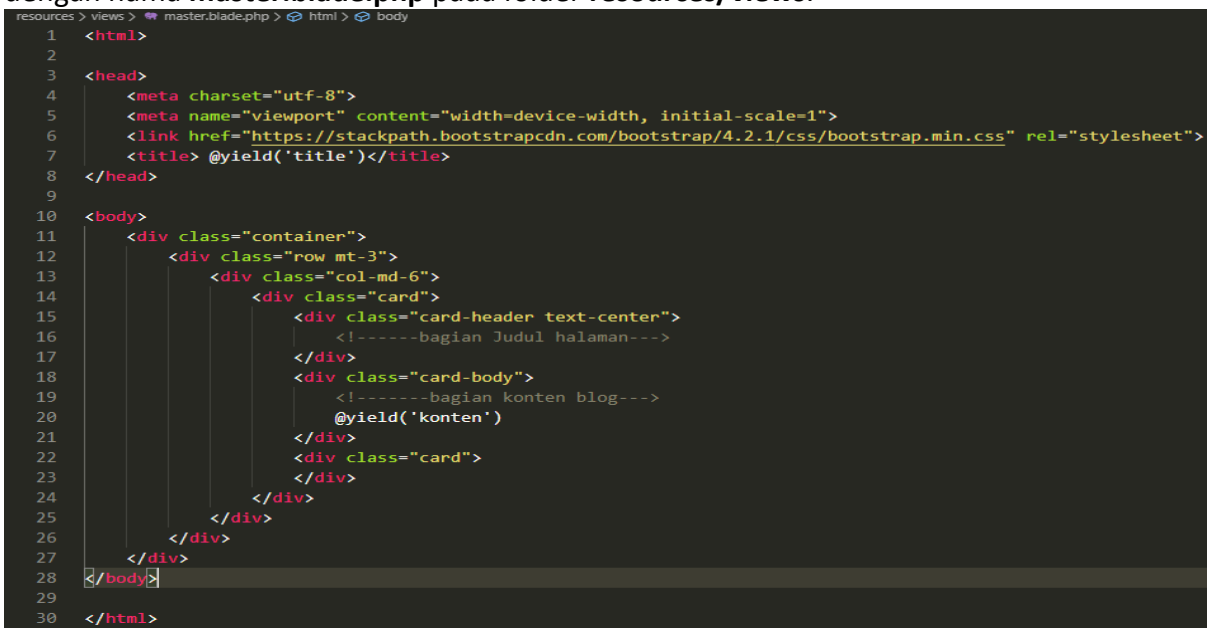
Langkah	Keterangan
1	Buatlah project Laravel baru dengan nama **laravel-crud**. Buka command prompt, tuliskan perintah berikut. **cd C:\xampp\htdocs laravel** **new laravel-crud** 
2	Selanjutnya kita lakukan konfigurasi database di Laravel. Untuk melakukan konfigurasi database, bukalah file **.env** pada project laravel-crud. Ubah seperti di bawah ini.  Keterangan:

	- 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. 
4	Isilah beberapa data pada tabel **mahasiswa** tersebut. 

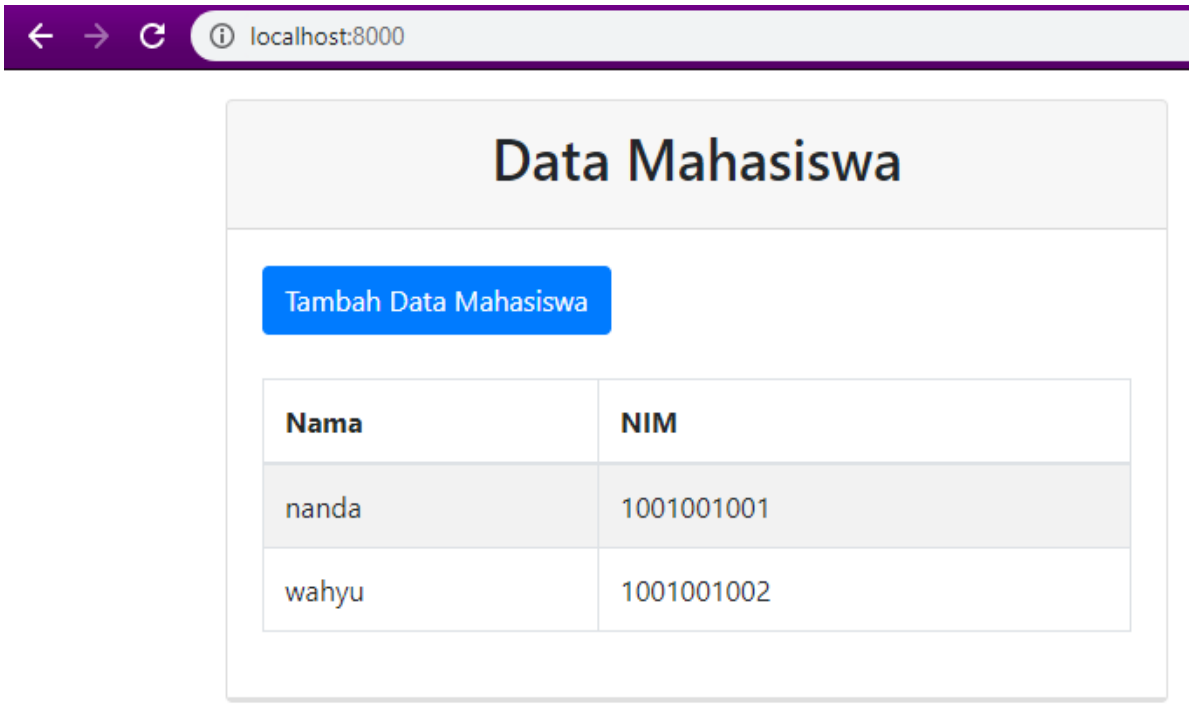
b. Menampilkan Data dari Database

Langkah	Keterangan
1	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. `20 Route::get('/', 'MahasiswaController@index');` Keterangan: - Ketika route (‘/’) diakses, akan dijalankan method **index** pada controller bernama **MahasiswaController**.

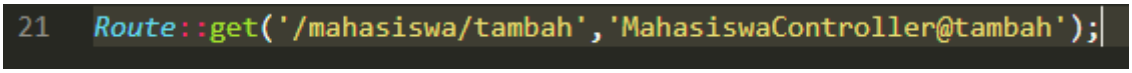
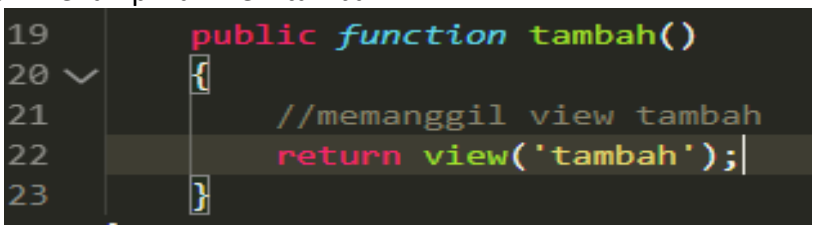
2	Buat controller baru menggunakan command prompt yaitu MahasiswaController menggunakan php artisan **cd laravel-crud** **php artisan make:controller MahasiswaController** `C:\xampp\htdocs\laravel-crud>php artisan make:controller MahasiswaController` `Controller created successfully.`

3	Buat method **index** pada MahasiswaController.php pada folder **app/Http/Controllers**  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
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**.  Keterangan: - Fungsi @yield pada line 6(title), 15(judul_halaman), dan 19(konten) berfungsi sebagai penanda bagian-bagian pada master blade. Nantinyabagian @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. 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--->@section('konten') 10 Tambah Data Mahasiswa 11 12 13 <table class="table table-bordered table-hover table-striped"> 14 <thead> 15 <tr> 16 <th>Nama</th> 17 <th>NIM</th> 18 </tr> 19 </thead> 20 <tbody> 21 @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 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
6	Jalankan command prompt, tuliskan perintah untuk menjalankan project laravel-crud **php artisan serve** C:\xampp\htdocs\laravel-crud>php artisan serve Laravel development server started: http://127.0.0.1:8000 Buka browser dan ketikkan localhost:8000, maka akan tampil sebagai berikut

c. Memasukkan Data (Create) ke Database

Langkah	Keterangan
1	Buatlah *route* baru pada **routes/web.php** dengan nama **/mahasiswa/tambah** yang akan menjalankan fungsi tambah pada MahasiswaController 
2	Buat method **tambah** pada **MahasiswaController.php** di folder **app/Http/Controllers** yang akan menampilkan view tambah. 
3	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 > ...

```blade
@extends('master')
<!-------isi title--->
@section('title', 'Tambah Data')

<!------isi bagian judul halaman---->
@section('judul_halaman', 'Tambah Data Mahasiswa')

<!------isi bagian konten------->
@section('konten')
<a href="/mahasiswa" class="btn btn-danger">Kembali</a>
<br />
<br />
<form action="/mahasiswa/simpan" method="post">
    {{csrf_field()}}
    <div class="form-group">
        <label for="namamhs">Nama</label>
        <input type="text" class="form-control" required="required" name="namamhs"> <br />
    </div>
    <div class="form-group">
        <label for="nimmhs">NIM</label>
        <input type="number" class="form-control" required="required" name="nimmhs"> <br />
    </div>
    <div class="form-group">
        <label for="emailmhs">E-mail</label>
        <input type="email" class="form-control" required="required" name="emailmhs"> <br />
    </div>
    <div class="form-group">
        <label for="jurusanmhs">Jurusan</label>
        <input type="text" class="form-control" required="required" name="jurusanmhs"> <br />
    </div>
    <button type="submit" name="tambah" class="btn btn-primary float-right">Tambah Data</button>
</form>
```

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*.

4

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

```php
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

5

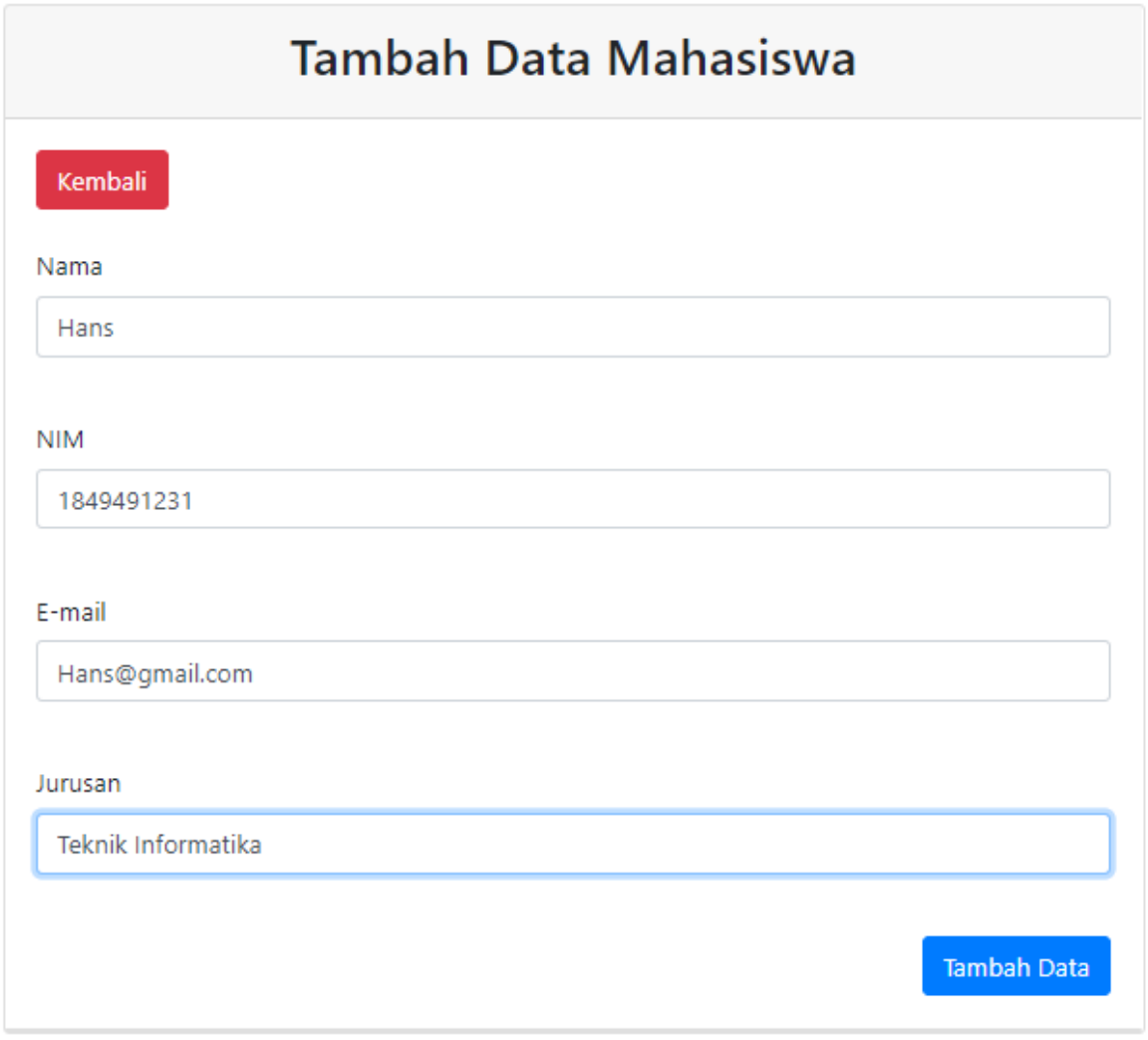
Buat method **simpan** pada **MahasiswaController** untuk menyimpan data ke database.

Keterangan:

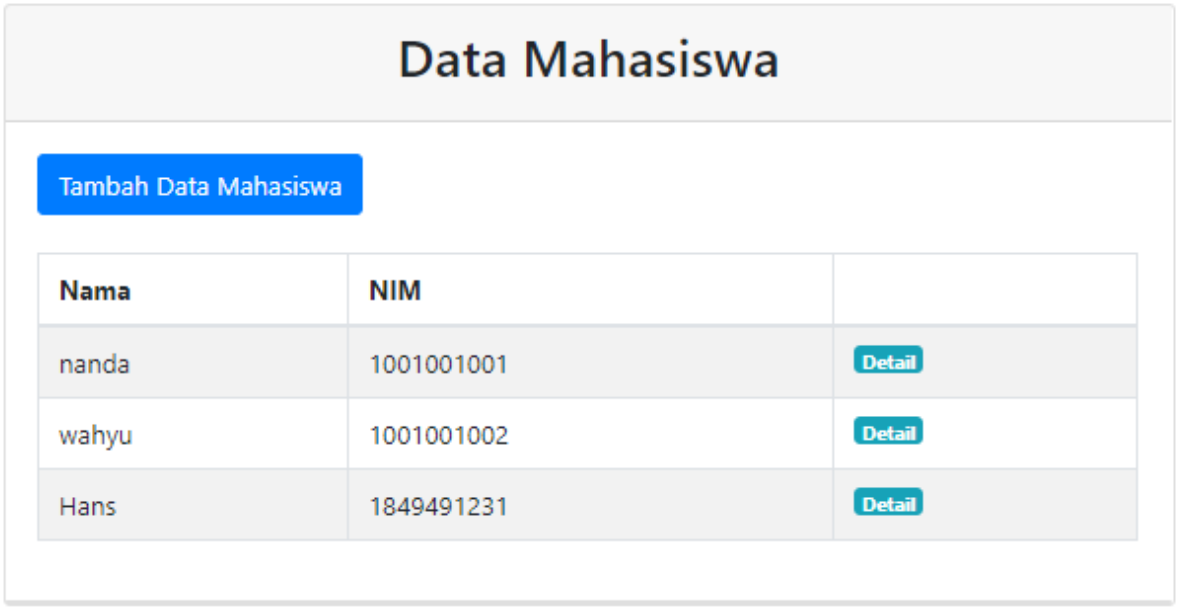
- Variabel $request untuk menerima data yang akan ditambahkan ke database
- Line 28-33 merupakan query builder untuk insert data ke tabel mahasiswa

6

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.



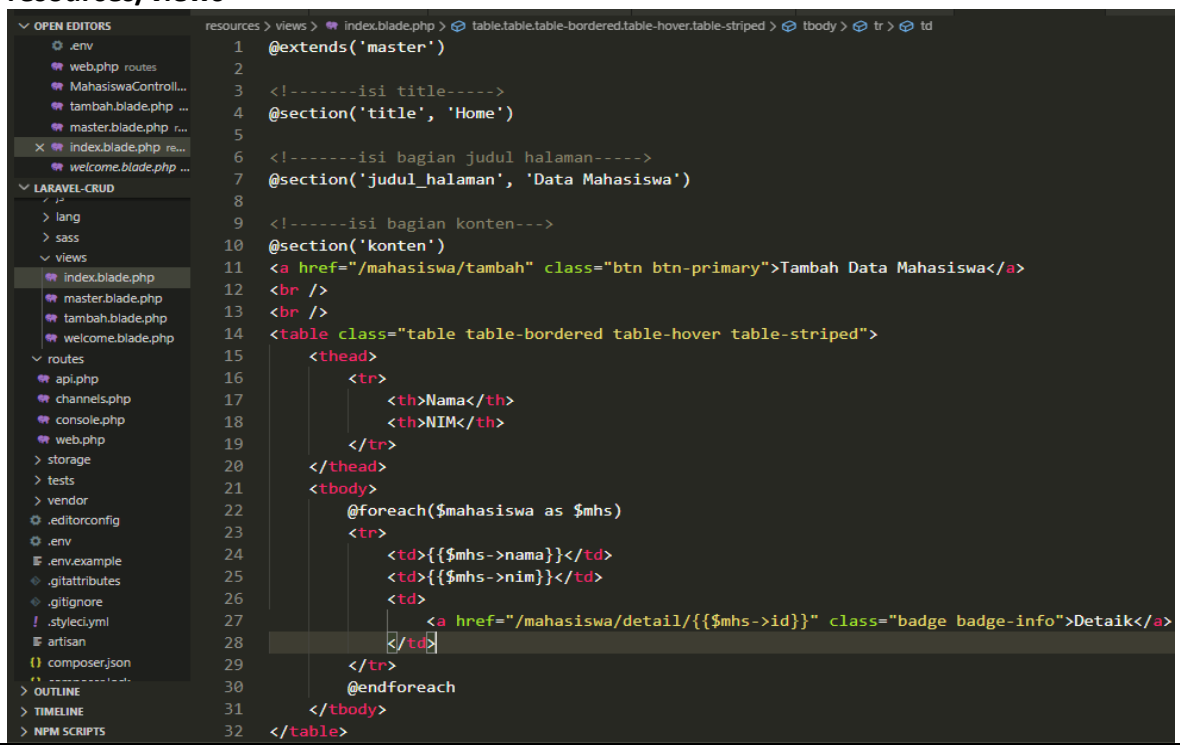
Setelah kita klik tombol tambah data, maka data baru akan ditampilkan pada halaman index.

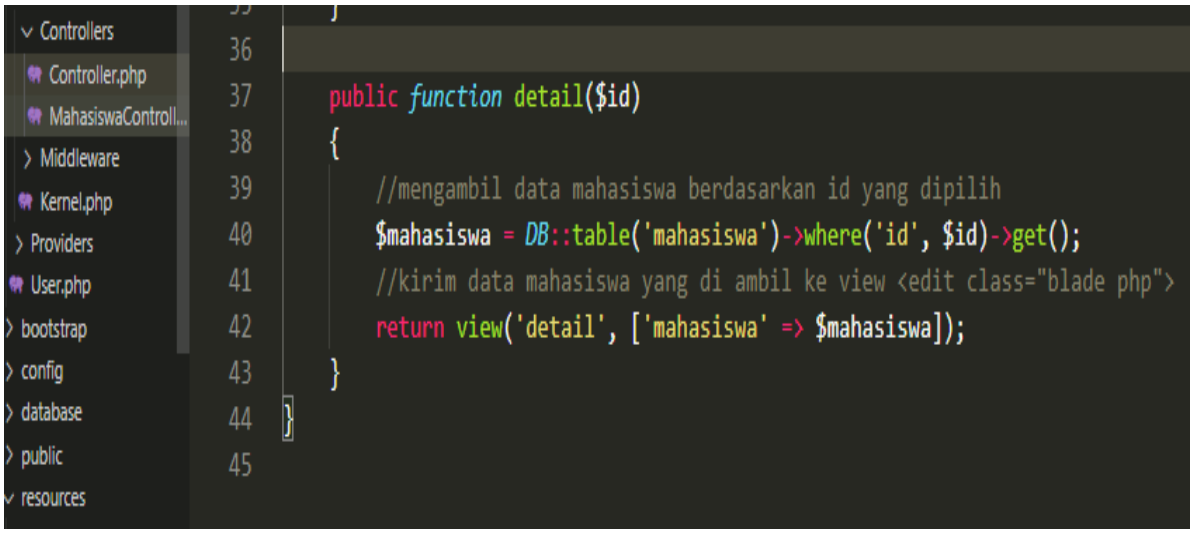


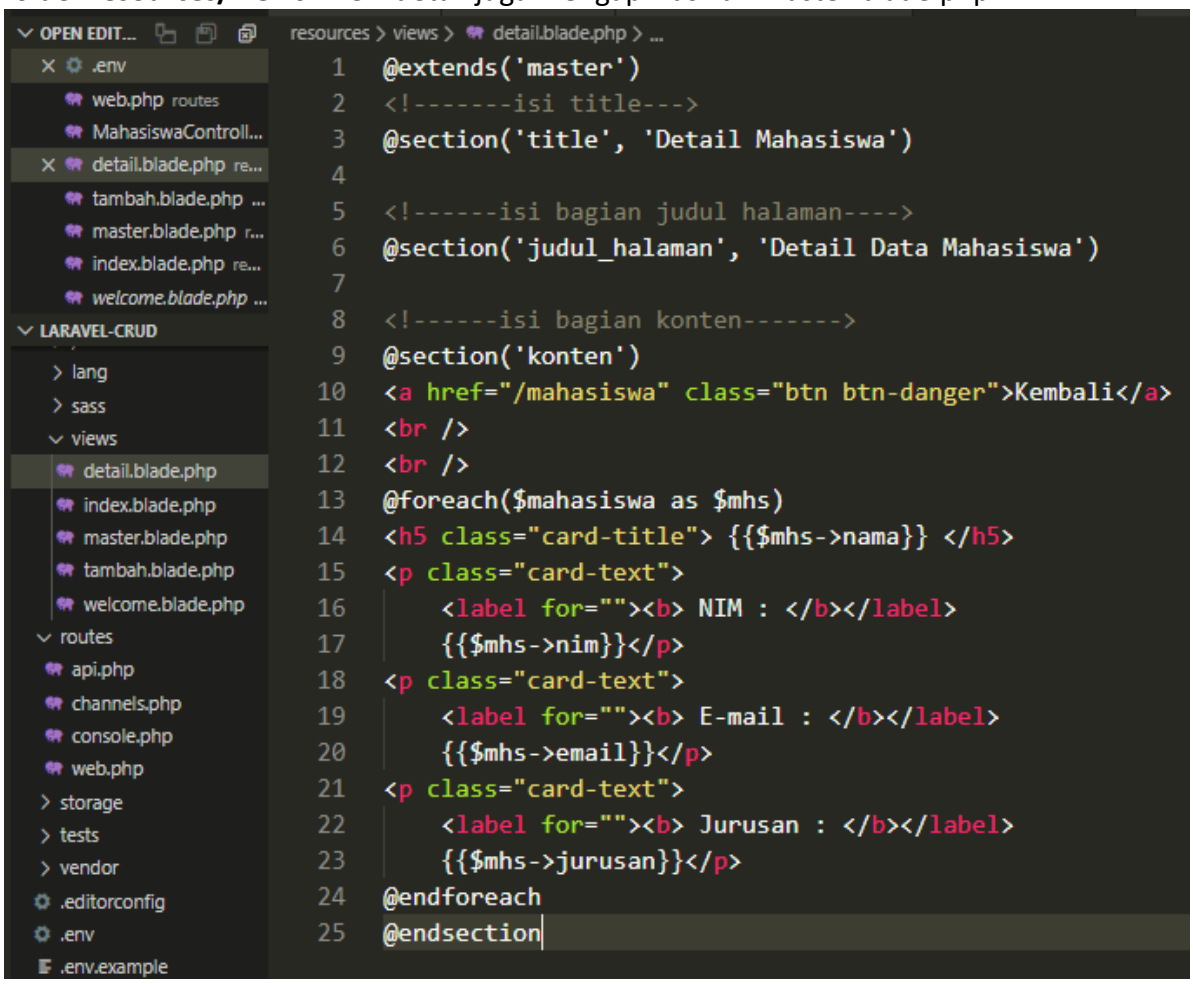
Kita dapat mencoba menambahkan beberapa data lagi agar jumlah data lebih banyak.

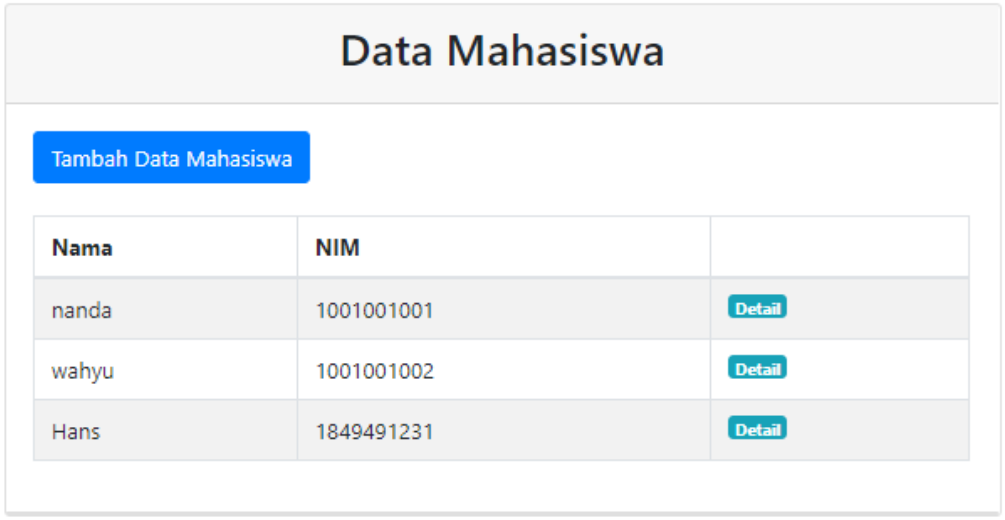
d. Melihat Detail Data dari Database

Langkah	Keterangan
1	Buatlah tombol untuk melihat detail data (Line 28) pada file **index.blade.php** di folder **resources/views**
2	Buatlah *route* baru pada **routes/web.php** dengan nama **/mahasiswa/detail** yang akan menjalankan fungsi detail pada MahasiswaController
3	Buat method **detail** pada **MahasiswaController.php** di folder **app/Http/Controllers** yang akan menampilkan data mahasiswa pada view detail.blade.php.




Controllers
Controller.php
MahasiswaControll...
Middleware
Kernel.php
Providers
User.php
bootstrap
config
database
public
resources
36
37 public function detail($id)
38 {
39 //mengambil data mahasiswa berdasarkan id yang dipilih
40 $mahasiswa = DB::table('mahasiswa')->where('id', $id)->get();
41 //kirim data mahasiswa yang di ambil ke view <edit class="blade php">
42 return view('detail', ['mahasiswa' => $mahasiswa]);
43 }
44 }
45

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

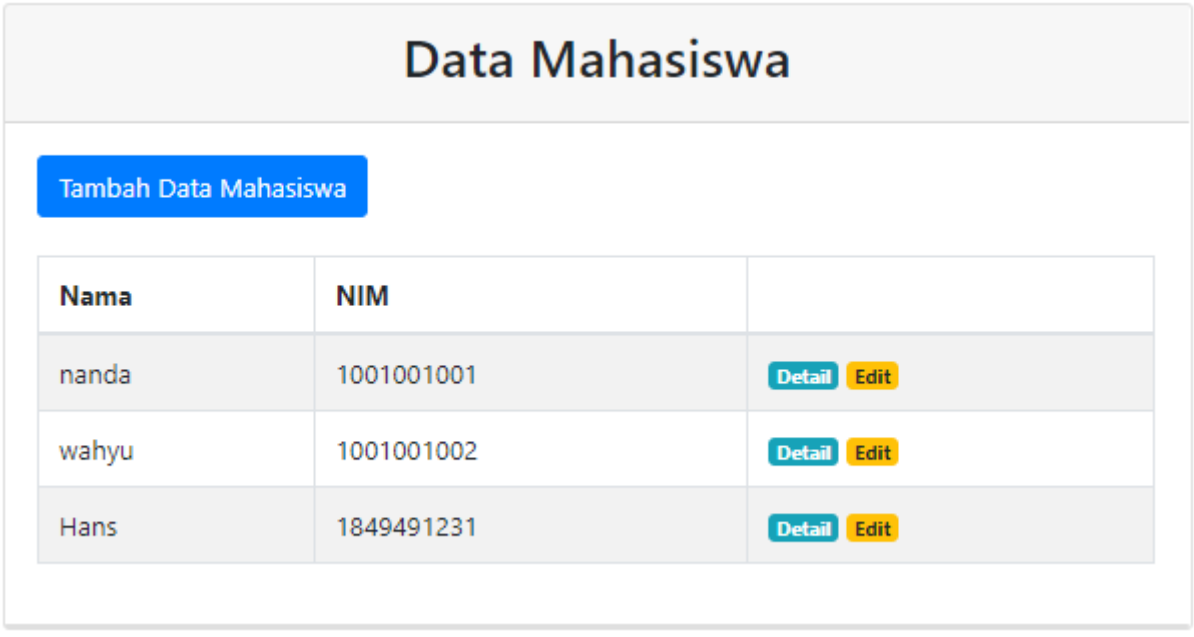


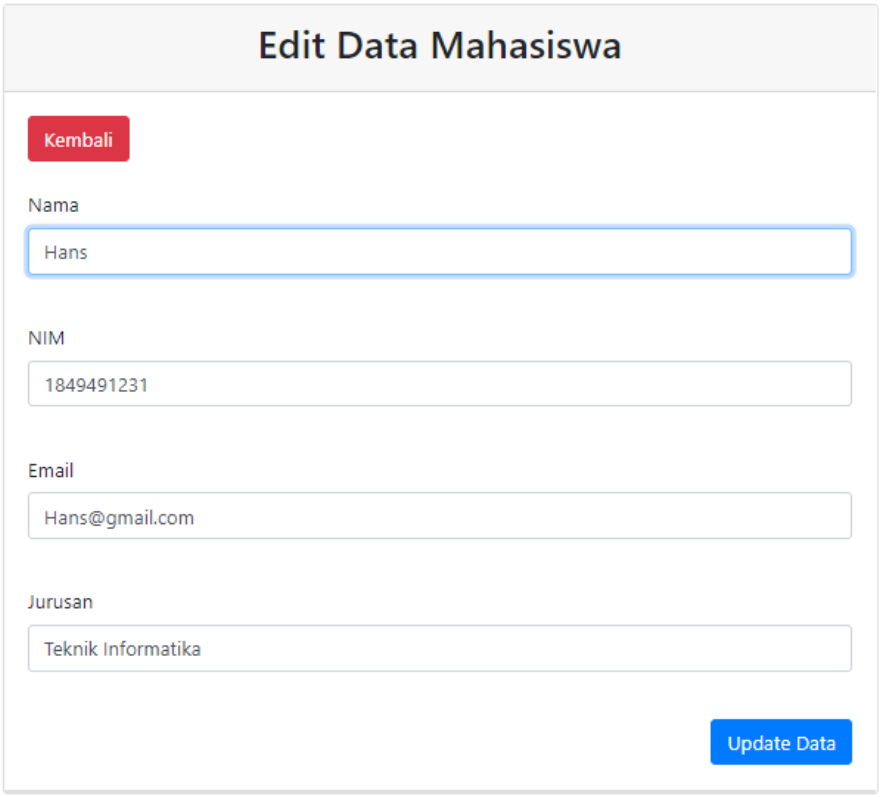
e. Mengubah Data (Update) dari Database

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

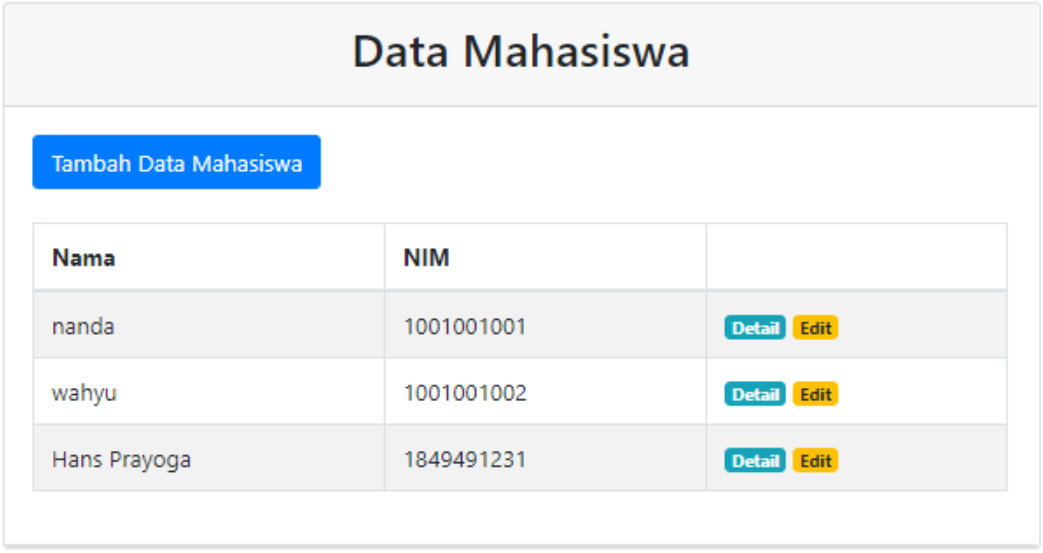
```
<!-- isi title -->
@section('title', 'Home')

<!-- isi bagian judul halaman -->
@section('judul_halaman', 'Data Mahasiswa')

<!-- isi bagian konten -->
@section('konten')
<a href="/mahasiswa/tambah" class="btn btn-primary">Tambah Data Mahasiswa</a>
<br />
<br />
<table class="table table-bordered table-hovel table-striped">
    <thead>
        <tr>
            <th>Nama</th>
            <th>NIM</th>
            <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach($mahasiswa as $mhs)
        <tr>
            <td>{{$mhs->nama}}</td>
            <td>{{$mhs->nim}}</td>
            <td>
                <a href="/mahasiswa/detail/{{ $mhs->id }}" class="badge badge-info">Detail</a>
                <a href="/mahasiswa/edit/{{ $mhs->id }}" class="badge badge-warning">Edit</a>
```

2	Buatlah *route* baru pada **routes/web.php** dengan nama **/mahasiswa/edit** yang akan menjalankan fungsi edit pada MahasiswaController
	``php Route::get('/mahasiswa/edit/{id}', 'MahasiswaController@edit'); ```
3	Buat method **edit** pada **MahasiswaController.php** di folder **app/Http/Controllers** yang akan menampilkan view edit. ```php public function edit($id) { // mengambil data mahasiswa berdasarkan id yang dipilih $mahasiswa = DB::table('mahasiswa')->where('id', $id)->get(); // kirim data mahasiswa yang diambil ke view edit.blade.php return view('edit', ['mahasiswa' => $mahasiswa]); } ``` Keterangan : - Line 49 : query builder untuk mengambil data mahasiswa berdasarkan id yang dipilih
4	Kemudian buatlah view **edit.blade.php** yang berisi form untuk mengubah data pada folder **resources/views.** View edit juga mengaplikasikan master.blade.php. esources > views > edit.blade.php ``` 1 @extends('master') 2 3 <!-- isi title --> 4 @section('title','Edit Data') 5 6 <!-- isi bagian judul halaman --> 7 @section('judul_halaman','Edit Data Mahasiswa') 8 9 <!-- isi bagian konten --> 10 @section('konten') 11 Kembali 12 13 14 <form action="/mahasiswa/update" method="POST"> 15 {{ csrf_field() }} 16 @foreach($mahasiswa as $mhs) 17 <input type="hidden" name="id" value="{{ $mhs->id }}"> 18 <div class="form-group"> 19 <label>Nama</label> 20 <input type="text" class="form-control" value="{{ $mhs->nama }}" required="required" name="namamhs"> 21 </div> 22 <div class="form-group"> 23 <label>NIM</label> 24 <input type="number" class="form-control" value="{{ $mhs->nim }}" required="required" name="nimmhs"> 25 </div> 26 <div class="form-group"> 27 <label>Email</label> 28 <input type="email" class="form-control" value="{{ $mhs->email }}" required="required" name="emailmhs"> 29 </div> 30 <div class="form-group"> 31 <label>Jurusan</label> 32 <input type="text" class="form-control" value="{{ $mhs->jurusan }}" required="required" name="jurusanmhs"> 33 </div> 34 <input type="submit" class="btn btn-primary float-right" value="Update Data"> 35 @endforeach 36 </form> 37 @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 MahaswaController

5	Ketika tombol simpan ditekan, akan dipanggil routes **/mahasiswa/update**. Oleh karena itu, kita buat terlebih dahulu route tersebut. `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. `public function update(Request $request)` `{` `  // update data mahasiswa` `  DB::table('mahasiswa')->where('id', $request->id)->update([` `  '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 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.  Klik edit di data pertama, kita akan mengubah namanya.



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



f. Menghapus Data (Delete) dari Database

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

```
@section('konten')
<a href="/mahasiswa/tambah" class="btn btn-primary">Tambah Data Mahasiswa</a>
<br />
<br />
<table class="table table-bordered table-hover table-striped">
    <thead>
        <tr>
            <th>Nama</th>
            <th>NIM</th>
            <th> </th>
        </tr>
    </thead>
    <tbody>
        @foreach($mahasiswa as $mhs)
        <tr>
            <td>{{$mhs->nama}}</td>
            <td>{{$mhs->nim}}</td>
            <td>
                <a href="/mahasiswa/detail/{{$mhs->id}}" class="badge badge-info">Detail</a>
                <a href="/mahasiswa/edit/{{$mhs->id}}" class="badge badge-warning">Edit</a>
                <a href="/mahasiswa/hapus/{{$mhs->id}}" class="badge badge-danger">Delete</a>
            </td>
        </tr>
        @endforeach
    </tbody>
</table>
```

2

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

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

3

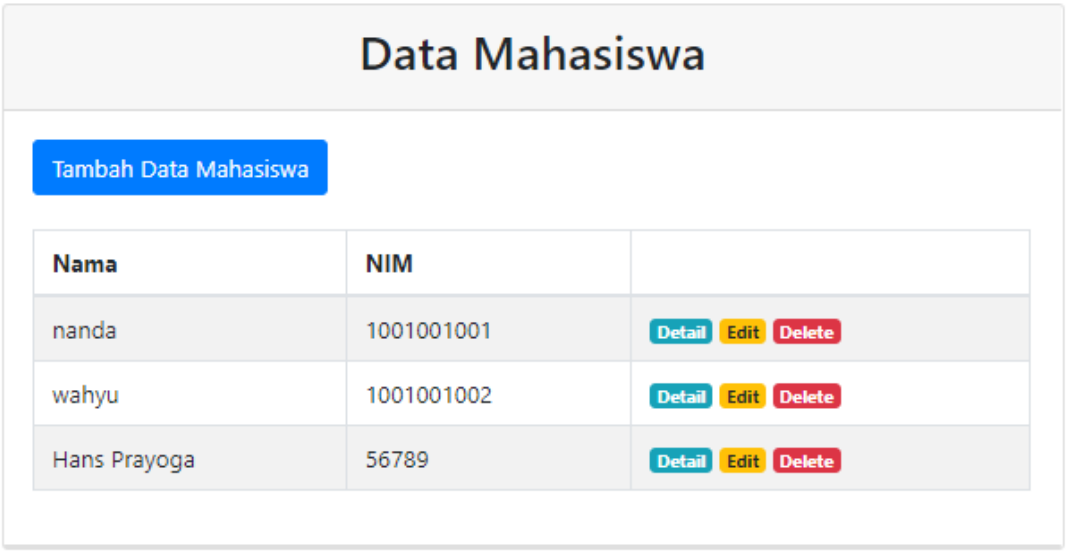
Buat method **hapus** pada **MahasiswaController.php** di folder **app/Http/Controllers** yang akan menjalankan fungsi hapus.

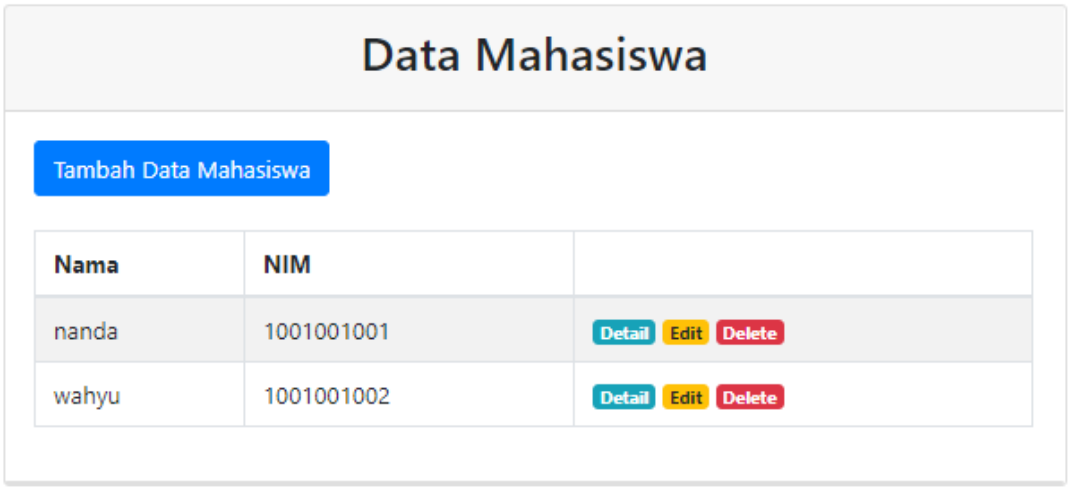
```
public function hapus($id)
{
    //menghapus  data mahasiswa berdasarkan id yang dipilih
    $mahasiswa = DB::table('mahasiswa')->where('id', $id)->delete();

    return redirect('/');
}
```

Keterangan :

- 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.





Praktikum – Bagian 2: Membuat CRUD di Laravel menggunakan Eloquent

Eloquent adalah sebuah fitur untuk mengelola data yang ada pada database. Eloquent ORM menyediakan fungsi-fungsi active record, atau fungsi-fungsi query sql untuk mengelola data pada database. Dengan menggunakan Eloquent pada Laravel, kita bisa mengelola data pada database dari hanya satu buah model.

a. Konfigurasi Database

Langkah	Keterangan
1	Buatlah project Laravel baru dengan nama **laravel-crud-kedua**. Buka command prompt, tuliskan perintah berikut. **cd C:\xampp\htdocs** **laravel new laravel-crud-kedua**
2	Selanjutnya kita lakukan konfigurasi database di Laravel. Untuk melakukan konfigurasi database, bukalah file **.env** pada project laravel-crud. Ubah seperti di bawah ini. `APP_NAME=Laravel` `APP_ENV=local` `APP_KEY=base64:jRLjpmPf3PVvLNpfqYo41+LW9ubPzpoAgji2HMD0v70=` `APP_DEBUG=true` `APP_URL=http://localhost` `LOG_CHANNEL=stack` `DB_CONNECTION=mysql` `DB_HOST=127.0.0.1` `DB_PORT=3306` `DB_DATABASE=latihan_laravel` `DB_USERNAME=root` `DB_PASSWORD=` 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

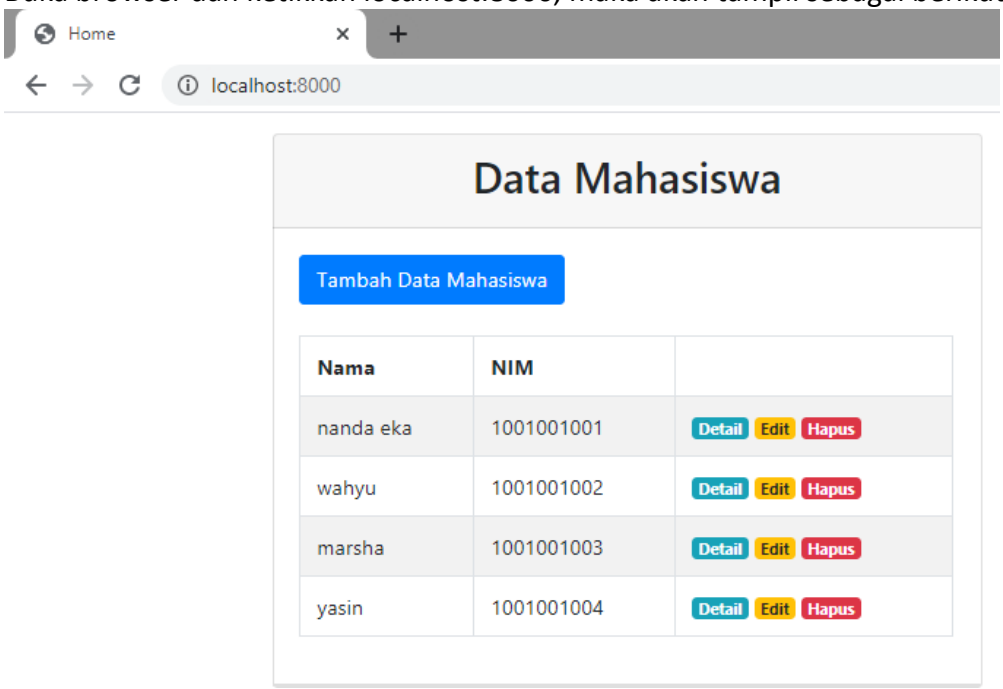
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		AUTO_INCREMENT
2	nama	varchar(100)	utf8mb4_general_ci		No	None		
3	nim	varchar(10)	utf8mb4_general_ci		No	None		
4	email	varchar(50)	utf8mb4_general_ci		No	None		
5	jurusan	varchar(20)	utf8mb4_general_ci		No	None		
6	created_at	timestamp			Yes	NULL		
7	updated_at	timestamp			Yes	NULL		

b. Menampilkan Data dari Database

Langkah	Keterangan
1	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. `Route::get('/', 'MahasiswaController@index');` `Route::get('/mahasiswa', 'MahasiswaController@index');`
2	Buat model menggunakan command prompt dengan nama Mahasiswa menggunakan php artisan **cd laravel-crud-kedua** **php artisan make:model Mahasiswa**
3	Ubah model **Mahasiswa.php** pada folder **App** menjadi seperti berikut. `<?php` `namespace App;` `use Illuminate\Database\Eloquent\Model;` `class Mahasiswa extends Model` `{` `protected $table = "mahasiswa";` Keterangan: - Model Mahasiswa akan menangani tabel mahasiswa
4	Buat controller baru yaitu MahasiswaController menggunakan php artisan **php artisan make:controller MahasiswaController**

5	Buat method **index** pada MahasiswaController.php pada folder **app/Http/Controllers**  Keterangan:

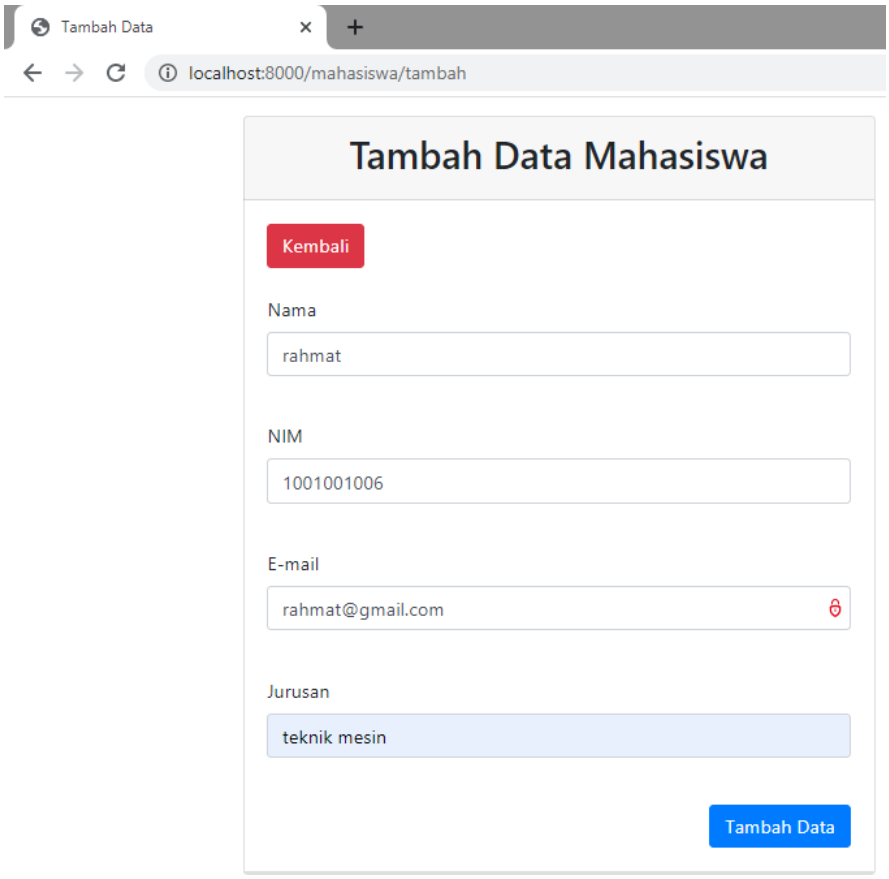
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Mahasiswa;
use Illuminate\Support\Facades\Redis;

class MahasiswaController extends Controller
{
    public function index()
    {
        $mahasiswa = Mahasiswa::all();
        return view('index', ['mahasiswa' => $mahasiswa]);
    }
```

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

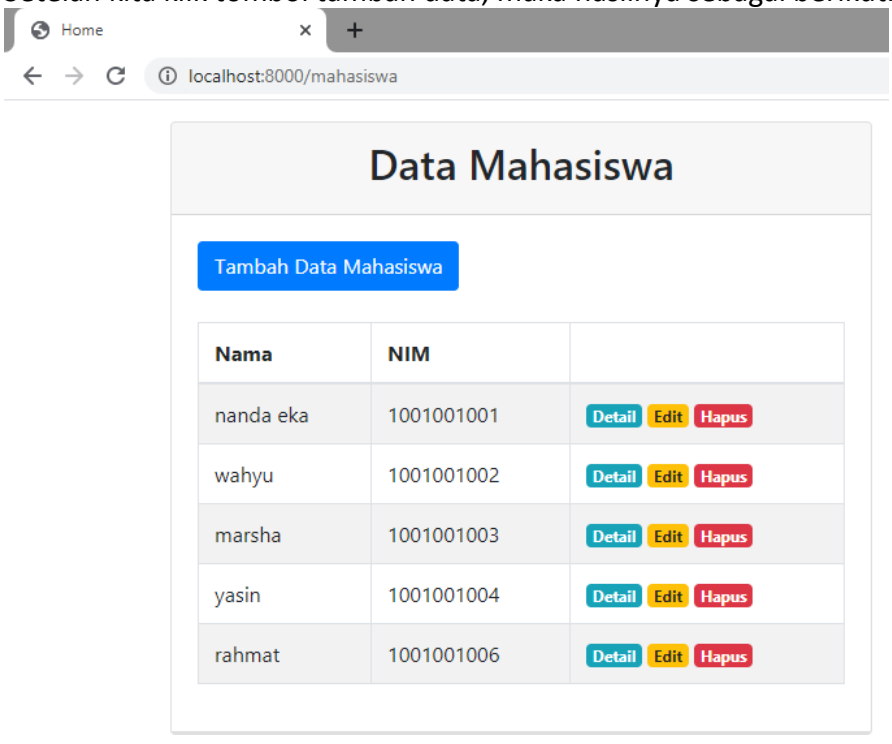
c. Memasukkan Data (Create) ke Database

Langkah	Keterangan
1	Buatlah *route* baru pada **routes/web.php** dengan nama **/mahasiswa/tambah** yang akan menjalankan fungsi tambah pada MahasiswaController ketika tombol Tambah Data Mahasiswa ditekan. `Route::get('/mahasiswa/tambah', 'MahasiswaController@tambah');`
2	Buat method **tambah** pada **MahasiswaController.php** di folder **app/Http/Controllers** yang akan menampilkan view tambah. `public function tambah()` `{` `  //memanggil view tambah` `  return view('tambah');` `}`

3	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
4	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
5	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
6	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. `<?php` `namespace App;` `use Illuminate\Database\Eloquent\Model;` `class Mahasiswa extends Model` `{` `  protected $table = "mahasiswa";` `  protected $fillable = ['nama', 'nim', 'email', 'jurusan'];`
7	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.



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



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

	`Route::get('/mahasiswa/detail/{id}', 'MahasiswaController@detail');`
3	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)` `{` `  //mengambil data mahasiswa berdasarkan id yang dipilih` `  $mahasiswa = Mahasiswa::find($id);` `  //kirim data mahasiswa yang di ambil ke view <edit class="b` `  return view('detail', ['mahasiswa' => $mahasiswa]);` `}`
4	Kemudian buatlah view **detail.blade.php** yang menampilkan detail data mahasiswa pada folder **resources/views**. View detail juga mengaplikasikan master.blade.php. `@extends('master')` `<!-------isi title--->` `@section('title', 'Detail Mahasiswa')` `<!------isi bagian judul halaman---->` `@section('judul_halaman', 'Detail Data Mahasiswa')` `<!------isi bagian konten------->` `@section('konten')` `<a href="/" class="btn btn-danger">Kembali</a>` ` ` ` ` `<h5 class="card-title"> {{$mahasiswa->nama}} </h5>` `<p class="card-text">` `  <label for=""><b> NIM : </b></label>` `  {{$mahasiswa->nim}}</p>` `<p class="card-text">` `  <label for=""><b> E-mail : </b></label>` `  {{$mahasiswa->email}}</p>` `<p class="card-text">` `  <label for=""><b> Jurusan : </b></label>` `  {{$mahasiswa->jurusan}}</p>` `@endsection`
5	Jalankan localhost:8000 dan pilih tombol ‘Detail’ di suatu data yang ingin kita lihat detailnya.

e. Mengubah Data (Update) dari Database

<table>
<tr><th>Langkah</th><th>Keterangan</th></tr>
<tr><td>1</td><td>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/edit yang akan menjalankan fungsi edit pada MahasiswaController

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

</td></tr>
<tr><td>2</td><td>Buat method edit pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view edit.</td></tr>
<tr><td></td><td>

```
public function edit($id)
{
    //mengambil data mahasiswa berdasarkan id yang dipilih
    $mahasiswa = Mahasiswa::find($id);
    //kirim data mahasiswa yang diambil ke view edit.blade.php
    return view('edit', ['mahasiswa' => $mahasiswa]);
}
```

Keterangan :
- Line 41 : fungsi eloquent untuk mengambil data mahasiswa berdasarkan id yang dipilih</td></tr>
<tr><td>3</td><td>Kemudian buatlah view edit.blade.php yang berisi form untuk mengubah data pada folder resources/views. View edit juga mengaplikasikan master.blade.php.

```
@extends('master')
<!--------isi title--->
@section('title', 'Edit Mahasiswa')

<!------isi bagian judul halaman---->
@section('judul_halaman', 'Edit Data Mahasiswa')

<!------isi bagian konten------->
@section('konten')
<a href="/" class="btn btn-danger">Kembali</a>
<br />
<br />
<form action="/mahasiswa/update/{{$mahasiswa->id}}" method="post">
    {{csrf_field()}}
    <input type="hidden" name="id" value="{{$mahasiswa->id}}"><br />
    <div class="form-group">
        <label for="namamhs">Nama</label>
        <input type="text" class="form-control" required="required" name="namamhs" value="{{$mahasiswa->nama}}"><br />
    </div>
    <div class="form-group">
        <label for="nimmhs">NIM</label>
        <input type="number" class="form-control" required="required" name="nimmhs" value="{{$mahasiswa->nim}}"><br />
    </div>
    <div class="form-group">
        <label for="emailmhs">E-mail</label>
        <input type="email" class="form-control" required="required" name="emailmhs" value="{{$mahasiswa->email}}"><br />
    </div>
    <div class="form-group">
        <label for="jurusanmhs">Jurusan</label>
        <input type="text" class="form-control" required="required" name="jurusanmhs" value="{{$mahasiswa->jurusan}}"><br />
    </div>
    <button type="submit" name="edit" class="btn btn-primary float-right">Simpan Data</button>
</form>
@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</td></tr>
</table>

4	Ketika tombol simpan ditekan, akan dipanggil routes **/mahasiswa/update/{id}**. Oleh karena itu, kita buat terlebih dahulu route tersebut. `Route::post('/mahasiswa/update/{id}', 'MahasiswaController@update');` 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.

	`` public function update($id, Request $request) { //update data mahasiswa $mahasiswa = Mahasiswa::find($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
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. <u>Menghapus Data (Delete) dari Database</u>

Langkah	Keterangan
1	Buatlah *route* baru pada **routes/web.php** dengan nama **/mahasiswa/hapus** yang akan menjalankan fungsi hapus pada MahasiswaController `Route::get('/mahasiswa/hapus/{id}', 'MahasiswaController@hapus');`
2	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 $mahasiswa = Mahasiswa::find($id); $mahasiswa->delete(); return redirect('/'); } ``` Keterangan :

	- Line 59 :fungsi eloquent untuk menghapus data mahasiswa berdasarkan id yang dipilih
3	Jalankan localhost:8000 dan pilih tombol ‘Hapus’, maka data yang terpilih akan dihapus.

-- Selamat Mengerjakan-