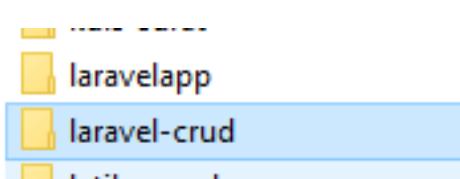
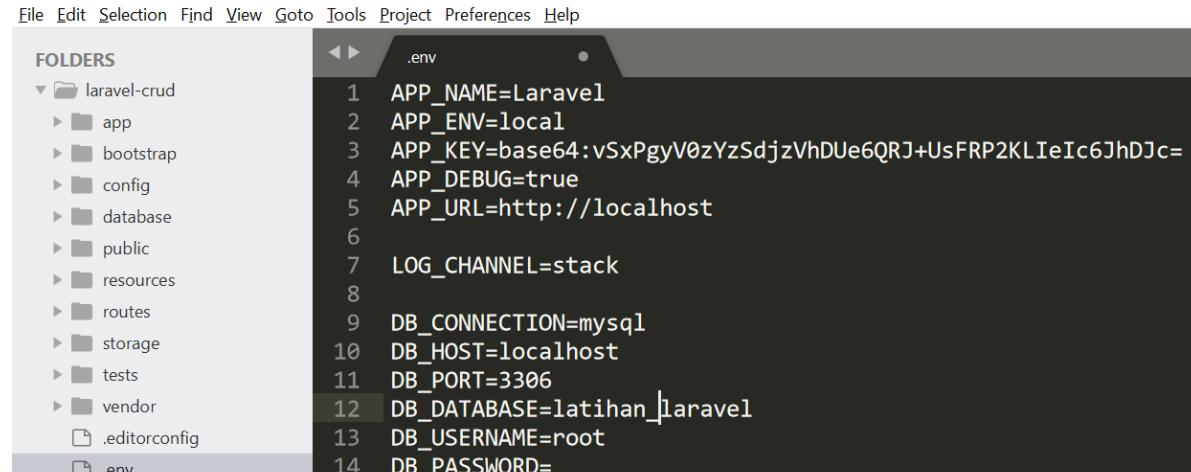


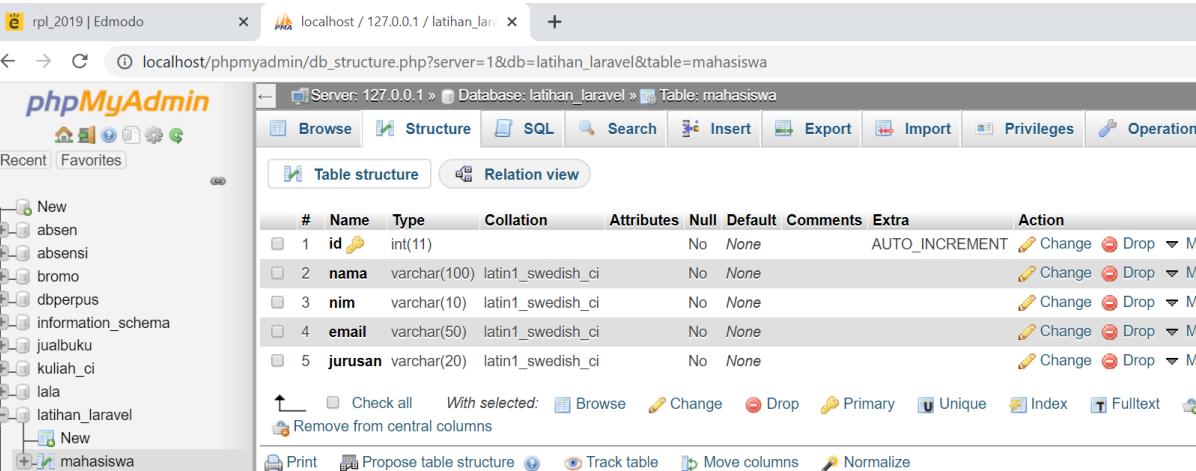
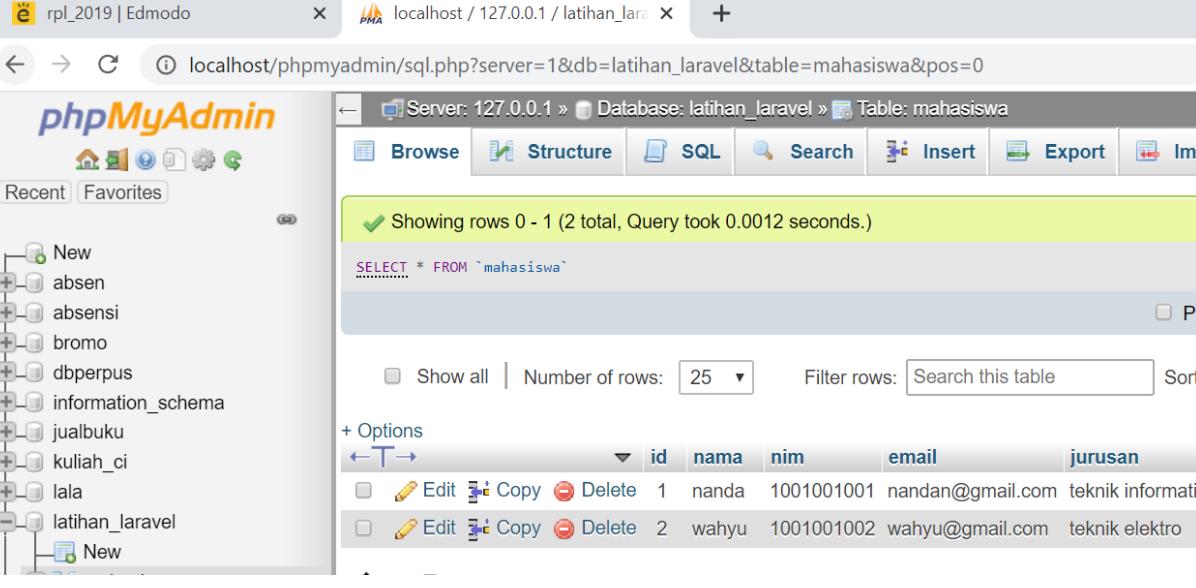
LAPORAN PRAKTIKUM
PEMROGRAMAN WEB LANJUT
Jobsheet-11
CRUD Laravel



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.  <p>Keterangan:</p>

	<ul style="list-style-type: none"> - Nama database yang akan digunakan adalah latihan_laravel dengan username root. 																																																												
3	Jalankanxampp, selanjutnya buat tabel dengan nama mahasiswa di mysql pada database latihan_laravel.  A screenshot of the phpMyAdmin interface. On the left, there's a tree view of databases and tables. The 'mahasiswa' table under the 'latihan_laravel' database is selected. On the right, the 'Structure' tab is active, showing the columns: id (int(11)), nama (varchar(100)), nim (varchar(10)), email (varchar(50)), and jurusan (varchar(20)). Each column has its properties like type, length, and collation listed. Below the table structure, there are buttons for various actions like 'Change', 'Drop', 'Primary', 'Unique', 'Index', 'Fulltext', and 'Normalize'. <p>The table structure for 'mahasiswa' is as follows:</p> <table border="1"> <thead> <tr> <th>#</th> <th>Name</th> <th>Type</th> <th>Collation</th> <th>Attributes</th> <th>Null</th> <th>Default</th> <th>Comments</th> <th>Extra</th> <th>Action</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>id</td> <td>int(11)</td> <td>latin1_swedish_ci</td> <td></td> <td>No</td> <td>None</td> <td></td> <td>AUTO_INCREMENT</td> <td>Change Drop M</td> </tr> <tr> <td>2</td> <td>nama</td> <td>varchar(100)</td> <td>latin1_swedish_ci</td> <td></td> <td>No</td> <td>None</td> <td></td> <td></td> <td>Change Drop M</td> </tr> <tr> <td>3</td> <td>nim</td> <td>varchar(10)</td> <td>latin1_swedish_ci</td> <td></td> <td>No</td> <td>None</td> <td></td> <td></td> <td>Change Drop M</td> </tr> <tr> <td>4</td> <td>email</td> <td>varchar(50)</td> <td>latin1_swedish_ci</td> <td></td> <td>No</td> <td>None</td> <td></td> <td></td> <td>Change Drop M</td> </tr> <tr> <td>5</td> <td>jurusan</td> <td>varchar(20)</td> <td>latin1_swedish_ci</td> <td></td> <td>No</td> <td>None</td> <td></td> <td></td> <td>Change Drop M</td> </tr> </tbody> </table>	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action	1	id	int(11)	latin1_swedish_ci		No	None		AUTO_INCREMENT	Change Drop M	2	nama	varchar(100)	latin1_swedish_ci		No	None			Change Drop M	3	nim	varchar(10)	latin1_swedish_ci		No	None			Change Drop M	4	email	varchar(50)	latin1_swedish_ci		No	None			Change Drop M	5	jurusan	varchar(20)	latin1_swedish_ci		No	None			Change Drop M
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action																																																				
1	id	int(11)	latin1_swedish_ci		No	None		AUTO_INCREMENT	Change Drop M																																																				
2	nama	varchar(100)	latin1_swedish_ci		No	None			Change Drop M																																																				
3	nim	varchar(10)	latin1_swedish_ci		No	None			Change Drop M																																																				
4	email	varchar(50)	latin1_swedish_ci		No	None			Change Drop M																																																				
5	jurusan	varchar(20)	latin1_swedish_ci		No	None			Change Drop M																																																				
4	Isilah beberapa data pada tabel mahasiswa tersebut.  A screenshot of the phpMyAdmin interface. The 'Browse' tab is active for the 'mahasiswa' table. It shows two rows of data: id 1, nama nanda, nim 1001001001, email nandan@gmail.com, and jurusan teknik informatika; and id 2, nama wahyu, nim 1001001002, email wahyu@gmail.com, and jurusan teknik elektro. There are buttons for 'Edit', 'Copy', 'Delete', and other table operations. The SQL query at the top is 'SELECT * FROM `mahasiswa`'. <p>The data in the 'mahasiswa' table is:</p> <table border="1"> <thead> <tr> <th>id</th> <th>nama</th> <th>nim</th> <th>email</th> <th>jurusan</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>nanda</td> <td>1001001001</td> <td>nandan@gmail.com</td> <td>teknik informatika</td> </tr> <tr> <td>2</td> <td>wahyu</td> <td>1001001002</td> <td>wahyu@gmail.com</td> <td>teknik elektro</td> </tr> </tbody> </table>	id	nama	nim	email	jurusan	1	nanda	1001001001	nandan@gmail.com	teknik informatika	2	wahyu	1001001002	wahyu@gmail.com	teknik elektro																																													
id	nama	nim	email	jurusan																																																									
1	nanda	1001001001	nandan@gmail.com	teknik informatika																																																									
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. Pertama, buatlah route pada routes/web.php sehingga ketika pertama kali project dijalankan akan terbuka halaman yang menampilkan data.</p> <pre>20 Route::get('/', 'MahasiswaController@index');</pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - 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	<p>Buat method index pada MahasiswaController.php pada folder app/Http/Controllers</p> <pre>app > Http > Controllers > MahasiswaController.php > ... 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 { 12 //mengambil data dari tabel mahasiswa 13 \$mahasiswa = DB::table('mahasiswa')->get(); 14 15 //mengirim data mahasiswa ke view index 16 return view('index', ['mahasiswa' => \$mahasiswa]); 17 } 18 } 19</pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - Tambahkan 'use Illuminate\Support\Facades\DB;' (line 6) agar <i>query builder</i> dapat digunakan - Line 13 untuk mengambil data dari tabel mahasiswa menggunakan <i>query builder</i> laravel dan akan disimpan di variabel \$mahasiswa - Line 16 : data akan dikirim ke blade view bernama index
4	<p>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.</p> <pre>resources > views > master.blade.php > html > body 1 <html> 2 3 <head> 4 <meta charset="utf-8"> 5 <meta name="viewport" content="width=device-width, initial-scale=1"> 6 <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css" rel="stylesheet"> 7 <title> @yield('title')</title> 8 </head> 9 10 <body> 11 <div class="container"> 12 <div class="row mt-3"> 13 <div class="col-md-6"> 14 <div class="card"> 15 <div class="card-header text-center"> 16 !----bagian Judul halaman--- 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 </div> 27 </div> 28 </body> 29 30 </html></pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - 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	<p>Sekarang buat file index.blade.php yang menerapkan <i>template</i> master.blade.php dan akan digunakan untuk menampilkan data.</p> <pre>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 </tr> 20 </thead> 21 <tbody> 22 @foreach(\$mahasiswa as \$mhs) 23 <tr> 24 <td>{{\$mhs->nama}}</td> 25 <td>{{\$mhs->nim}}</td> 26 </tr> 27 @endforeach 28 </tbody> 29 </table> 30 @endsection </pre>
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>

Nama	NIM
nanda	1001001001
wahyu	1001001002

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
	<pre>21 Route::get('/mahasiswa/tambah', 'MahasiswaController@tambah');</pre>
2	Buat method tambah pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view tambah.
	<pre>19 20 public function tambah() 21 { 22 //memanggil view tambah 23 return view('tambah');</pre>
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 > ...
1  @extends('master')
2  <!-----isi title---->
3  @section('title', 'Tambah Data')
4
5  <!-----isi bagian judul halaman---->
6  @section('judul_halaman', 'Tambah Data Mahasiswa')
7
8  <!-----isi bagian konten----->
9  @section('konten')
10 <a href="/mahasiswa" class="btn btn-danger">Kembali</a>
11 <br />
12 <br />
13 <form action="/mahasiswa/simpan" method="post">
14   {{csrf_field()}}
15   <div class="form-group">
16     <label for="namamhs">Nama</label>
17     <input type="text" class="form-control" required="required" name="namamhs"> <br />
18   </div>
19   <div class="form-group">
20     <label for="nimmhs">NIM</label>
21     <input type="number" class="form-control" required="required" name="nimmhs"> <br />
22   </div>
23   <div class="form-group">
24     <label for="emailmhs">E-mail</label>
25     <input type="email" class="form-control" required="required" name="emailmhs"> <br />
26   </div>
27   <div class="form-group">
28     <label for="jurusanmhs">Jurusan</label>
29     <input type="text" class="form-control" required="required" name="jurusanmhs"> <br />
30   </div>
31   <button type="submit" name="tambah" class="btn btn-primary float-right">Tambah Data</button>
32 </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.

```

22  Route::post('/mahasiswa/simpan', 'MahasiswaController@simpan');
23

```

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.

> Exceptions

 v Http

 v Controllers

 Controller.php

 MahasiswaControll...

 > Middleware

 Kernel.php

 > Providers

 User.php

 > bootstrap

 > config

 > database

25

26

27

28

29

30

31

32

33

34

35

```
public function simpan(Request $request)
```

```
{
```

```
    //insert data ke table mahasiswa
```

```
    DB::table('mahasiswa')->insert([
```

```
        'nama' => $request->namamhs,
```

```
        'nim' => $request->nimmhs,
```

```
        'email' => $request->emailmhs,
```

```
        'jurusan' => $request->jurusanmhs
```

```
    ]);
```

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

```
}
```

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.

Tambah Data Mahasiswa

[Kembali](#)

Nama

Hans

NIM

1849491231

E-mail

Hans@gmail.com

Jurusan

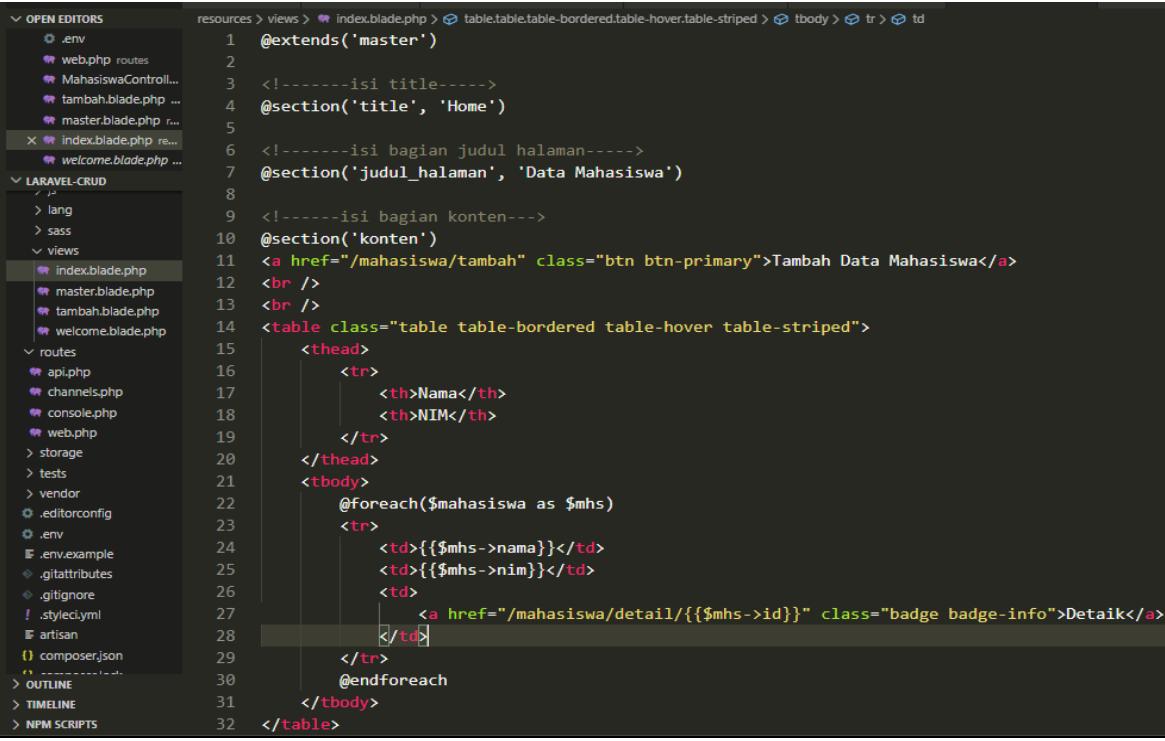
Teknik Informatika

[Tambah Data](#)

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

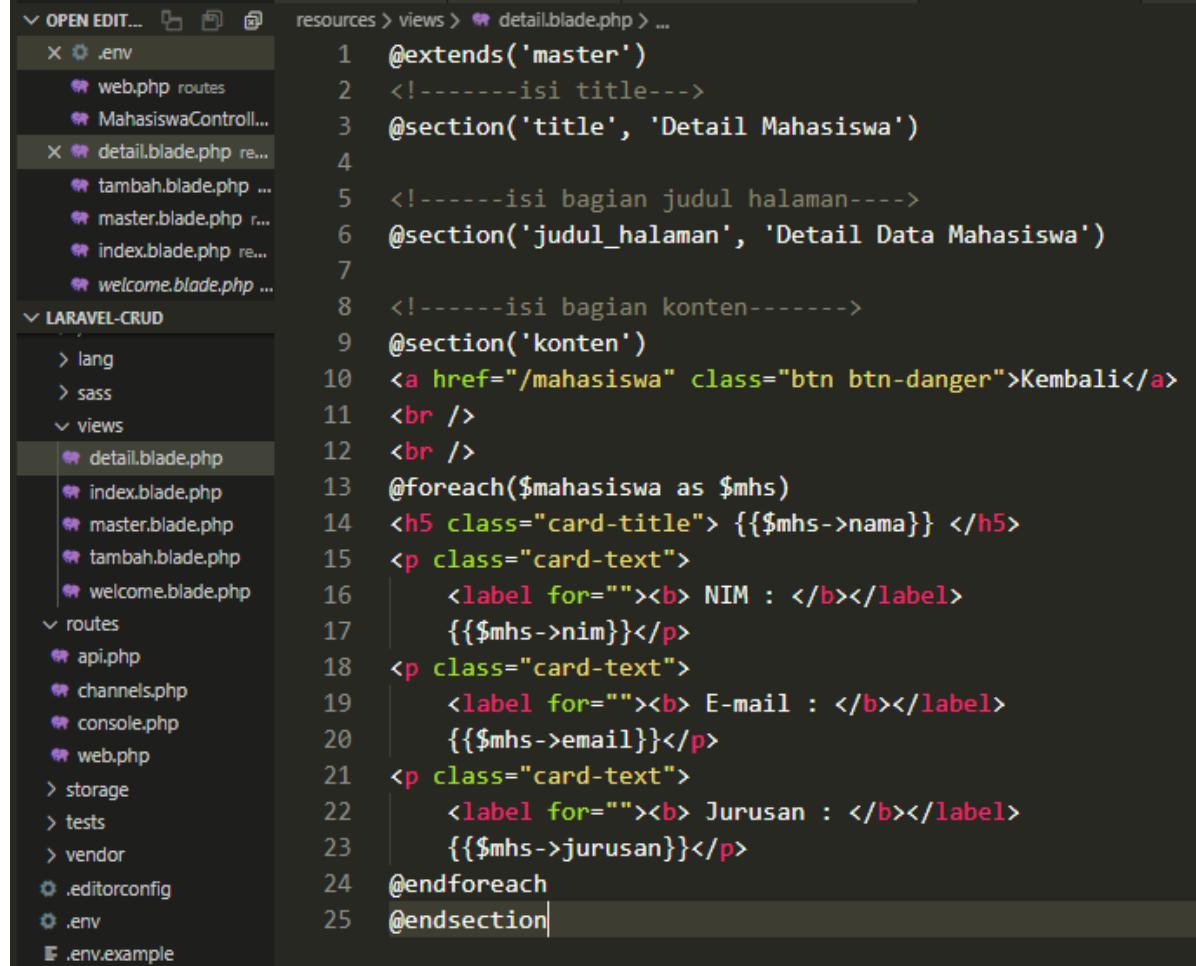
Data Mahasiswa														
Tambah Data Mahasiswa														
<table border="1"> <thead> <tr> <th>Nama</th><th>NIM</th><th></th></tr> </thead> <tbody> <tr> <td>nanda</td><td>1001001001</td><td>Detail</td></tr> <tr> <td>wahyu</td><td>1001001002</td><td>Detail</td></tr> <tr> <td>Hans</td><td>1849491231</td><td>Detail</td></tr> </tbody> </table>			Nama	NIM		nanda	1001001001	Detail	wahyu	1001001002	Detail	Hans	1849491231	Detail
Nama	NIM													
nanda	1001001001	Detail												
wahyu	1001001002	Detail												
Hans	1849491231	Detail												
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> resources > views > index.blade.php > table.table.table-bordered.table-hover.table-striped > tbody > tr > td 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 </tr> 20 </thead> 21 <tbody> 22 @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> </pre>
2	Buatlah <i>route</i> baru pada routes/web.php dengan nama /mahasiswa/detail yang akan menjalankan fungsi detail pada MahasiswaController
	<pre> 23 Route::get('/mahasiswa/detail/{id}', 'MahasiswaController@detail'); 24 </pre>
3	Buat method detail pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan data mahasiswa pada view detail.blade.php .

The screenshot shows a code editor interface with a sidebar on the left displaying a project structure. The structure includes 'Controllers' (with 'Controller.php' and 'MahasiswaController.php'), 'Middleware' (with 'Kernel.php'), 'Providers' (with 'User.php'), and several other directories like 'bootstrap', 'config', 'database', 'public', and 'resources'. The main panel displays a portion of the 'Controller.php' file:

```
35
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	<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> resources > views > detail.blade.php > ... 1 @extends('master') 2 <!--isi title--> 3 @section('title', 'Detail Mahasiswa') 4 5 <!--isi bagian judul halaman--> 6 @section('judul_halaman', 'Detail Data Mahasiswa') 7 8 <!--isi bagian konten--> 9 @section('konten') 10 Kembali 11
 12
 13 @foreach(\$mahasiswa as \$mhs) 14 <h5 class="card-title"> {{\$mhs->nama}} </h5> 15 <p class="card-text"> 16 <label for=""> NIM : </label> 17 {{\$mhs->nim}}</p> 18 <p class="card-text"> 19 <label for=""> E-mail : </label> 20 {{\$mhs->email}}</p> 21 <p class="card-text"> 22 <label for=""> Jurusan : </label> 23 {{\$mhs->jurusan}}</p> 24 @endforeach 25 @endsection </pre>
5	<p>Jalankan localhost:8000 dan pilih tombol ‘Detail’ di suatu data yang ingin kita lihat detailnya.</p>

Data Mahasiswa

[Tambah Data Mahasiswa](#)

Nama	NIM	
nanda	1001001001	Detail
wahyu	1001001002	Detail
Hans	1849491231	Detail

Detail Data Mahasiswa

[Kembali](#)

Hans

NIM : 1849491231

E-mail : Hans@gmail.com

Jurusan : Teknik Informatika

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

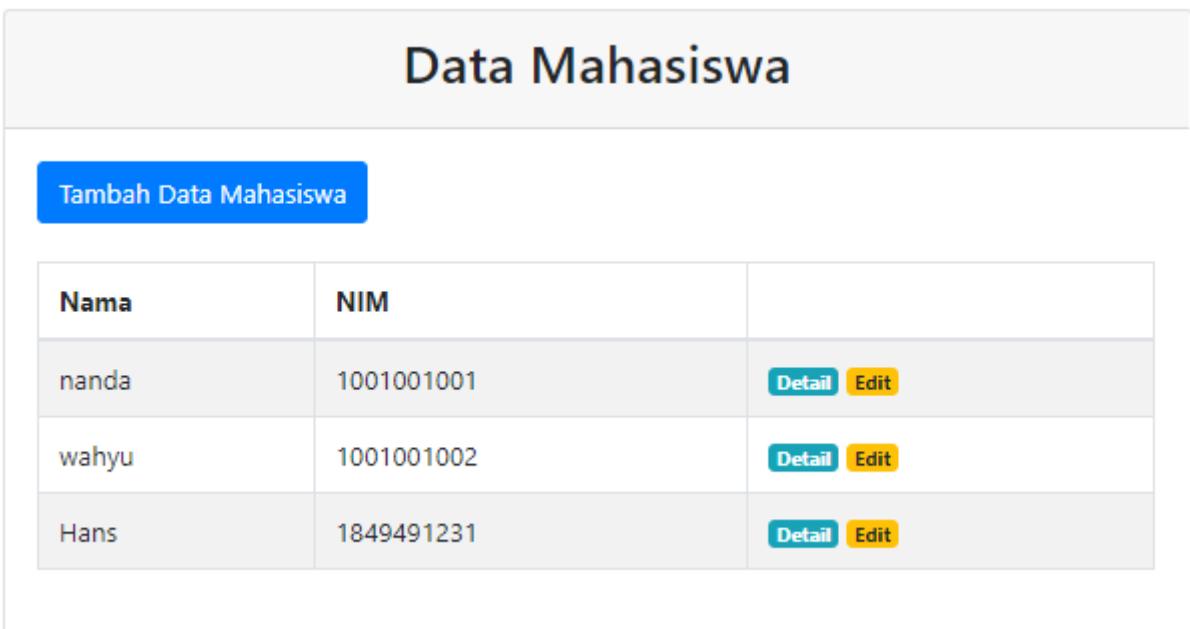
```
<!-- isi title -->
@section('title', 'Home')

<!-- 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>
      </td>
    </tr>
    @endforeach
  </tbody>
</table>

```

2	Buatlah route baru pada routes/web.php dengan nama /mahasiswa/edit yang akan menjalankan fungsi edit pada MahasiswaController
	<pre>Route::get('/mahasiswa/edit/{id}', 'MahasiswaController@edit');</pre>
3	<p>Buat method edit pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view edit.</p> <pre>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]); }</pre>
Keterangan :	<ul style="list-style-type: none"> - Line 49 : query builder untuk mengambil data mahasiswa berdasarkan id yang dipilih <p>4 Kemudian buatlah view edit.blade.php yang berisi form untuk mengubah data pada folder resources/views. View edit juga mengaplikasikan master.blade.php.</p> <pre>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 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</pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - 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	<p>Ketika tombol simpan ditekan, akan dipanggil routes <code>/mahasiswa/update</code>. Oleh karena itu, kita buat terlebih dahulu route tersebut.</p> <pre data-bbox="303 197 1318 265">30 Route::post('/mahasiswa/update', 'MahasiswaController@update'); 31</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 update di MahasiswaController.php 												
6	<p>Buat method update pada MahasiswaController untuk menyimpan data yang diubah ke database.</p> <pre data-bbox="334 518 1282 938">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('/'); }</pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - Variabel <code>\$request</code> untuk menerima data yang akan ditambahkan ke database - Line 58-63 merupakan query builder untuk update data ke tabel mahasiswa 												
7	<p>Jalankan <code>localhost:8000</code> dan pilih tombol ‘Edit’, maka akan ditampilkan halaman edit yang berisi form untuk mengubah data baru.</p>  <p>The screenshot shows a table with three rows of student data. Each row has columns for 'Nama' (Name) and 'NIM' (Student ID). To the right of each row are two buttons: 'Detail' (blue) and 'Edit' (yellow).</p> <table border="1" data-bbox="334 1432 1445 1731"> <thead> <tr> <th>Nama</th> <th>NIM</th> <th></th> </tr> </thead> <tbody> <tr> <td>nanda</td> <td>1001001001</td> <td>Detail Edit</td> </tr> <tr> <td>wahyu</td> <td>1001001002</td> <td>Detail Edit</td> </tr> <tr> <td>Hans</td> <td>1849491231</td> <td>Detail Edit</td> </tr> </tbody> </table> <p>Klik edit di data pertama, kita akan mengubah namanya.</p>	Nama	NIM		nanda	1001001001	Detail Edit	wahyu	1001001002	Detail Edit	Hans	1849491231	Detail Edit
Nama	NIM												
nanda	1001001001	Detail Edit											
wahyu	1001001002	Detail Edit											
Hans	1849491231	Detail Edit											

Edit Data Mahasiswa

[Kembali](#)

Nama

NIM

Email

Jurusan

[Update Data](#)

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

Data Mahasiswa

[Tambah Data Mahasiswa](#)

Nama	NIM	
nanda	1001001001	Detail Edit
wahyu	1001001002	Detail Edit
Hans Prayoga	1849491231	Detail Edit

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.

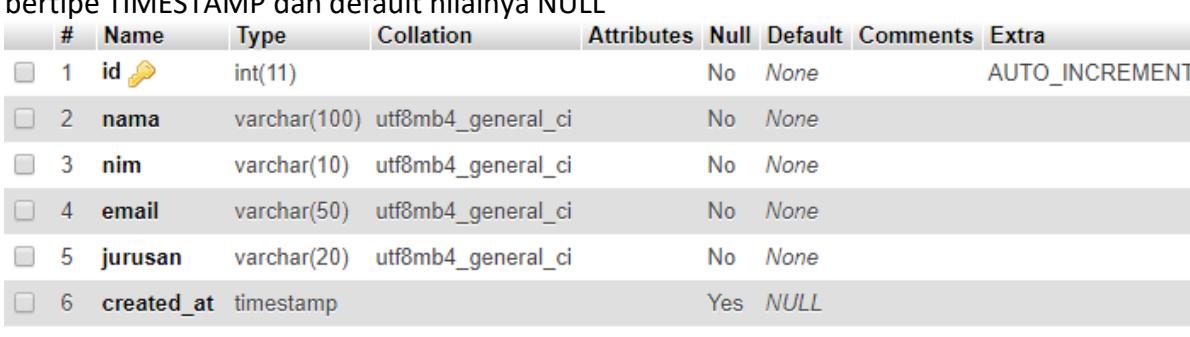
Data Mahasiswa			
Tambah Data Mahasiswa			
Nama	NIM		
nanda	1001001001	Detail	Edit
wahyu	1001001002	Detail	Edit
Hans Prayoga	56789	Detail	Edit

Data Mahasiswa			
Tambah Data Mahasiswa			
Nama	NIM		
nanda	1001001001	Detail	Edit
wahyu	1001001002	Detail	Edit

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. <pre>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=</pre> <p>Keterangan:</p> <ul style="list-style-type: none">- 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  <table border="1"><thead><tr><th>#</th><th>Name</th><th>Type</th><th>Collation</th><th>Attributes</th><th>Null</th><th>Default</th><th>Comments</th><th>Extra</th></tr></thead><tbody><tr><td>1</td><td>id 🏷</td><td>int(11)</td><td></td><td></td><td>No</td><td>None</td><td></td><td>AUTO_INCREMENT</td></tr><tr><td>2</td><td>nama</td><td>varchar(100)</td><td>utf8mb4_general_ci</td><td></td><td>No</td><td>None</td><td></td><td></td></tr><tr><td>3</td><td>nim</td><td>varchar(10)</td><td>utf8mb4_general_ci</td><td></td><td>No</td><td>None</td><td></td><td></td></tr><tr><td>4</td><td>email</td><td>varchar(50)</td><td>utf8mb4_general_ci</td><td></td><td>No</td><td>None</td><td></td><td></td></tr><tr><td>5</td><td>jurusan</td><td>varchar(20)</td><td>utf8mb4_general_ci</td><td></td><td>No</td><td>None</td><td></td><td></td></tr><tr><td>6</td><td>created_at</td><td>timestamp</td><td></td><td></td><td>Yes</td><td>NULL</td><td></td><td></td></tr><tr><td>7</td><td>updated_at</td><td>timestamp</td><td></td><td></td><td>Yes</td><td>NULL</td><td></td><td></td></tr></tbody></table>	#	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		
#	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	<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> <pre>Route::get('/', 'MahasiswaController@index'); Route::get('/mahasiswa', 'MahasiswaController@index');</pre>
2	<p>Buat model menggunakan command prompt dengan nama Mahasiswa menggunakan php artisan</p> <pre>cd laravel-crud-kedua php artisan make:model Mahasiswa</pre>
3	<p>Ubah model Mahasiswa.php pada folder App menjadi seperti berikut.</p> <pre>1 <?php 2 3 namespace App; 4 5 use Illuminate\Database\Eloquent\Model; 6 7 class Mahasiswa extends Model 8 { 9 protected \$table = "mahasiswa";</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> <pre>php artisan make:controller MahasiswaController</pre>

5

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

```
<?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]);
    }
}
```

Keterangan:

	<ul style="list-style-type: none"> - 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	<p>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).</p> <p>Pada bagian view kita buat seperti bagian 1 (copy-paste dari bagian 1)</p>															
7	<p>Jalankan command prompt, tuliskan perintah untuk menjalankan project laravel-crud php artisan serve</p> <p>Buka browser dan ketikkan localhost:8000, maka akan tampil sebagai berikut</p> <table border="1"> <thead> <tr> <th>Nama</th> <th>NIM</th> <th></th> </tr> </thead> <tbody> <tr> <td>nanda eka</td> <td>1001001001</td> <td>Detail Edit Hapus</td> </tr> <tr> <td>wahyu</td> <td>1001001002</td> <td>Detail Edit Hapus</td> </tr> <tr> <td>marsha</td> <td>1001001003</td> <td>Detail Edit Hapus</td> </tr> <tr> <td>yasin</td> <td>1001001004</td> <td>Detail Edit Hapus</td> </tr> </tbody> </table>	Nama	NIM		nanda eka	1001001001	Detail Edit Hapus	wahyu	1001001002	Detail Edit Hapus	marsha	1001001003	Detail Edit Hapus	yasin	1001001004	Detail Edit Hapus
Nama	NIM															
nanda eka	1001001001	Detail Edit Hapus														
wahyu	1001001002	Detail Edit Hapus														
marsha	1001001003	Detail Edit Hapus														
yasin	1001001004	Detail Edit Hapus														

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>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>public function tambah() { //memanggil view tambah return view('tambah'); }</pre>

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. <pre>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	Buat method simpan pada MahasiswaController untuk menyimpan data ke database. <pre>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('/'); }</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	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. <pre><?php namespace App; use Illuminate\Database\Eloquent\Model; class Mahasiswa extends Model { protected \$table = "mahasiswa"; protected \$fillable = ['nama', 'nim', 'email', 'jurusan'];</pre>
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.

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 eka	1001001001	Detail Edit Hapus
wahyu	1001001002	Detail Edit Hapus
marsha	1001001003	Detail Edit Hapus
yasin	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><code>Route::get('/mahasiswa/detail/{id}', 'MahasiswaController@detail');</code></pre>
3	Buat method detail pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan data mahasiswa pada view detail.blade.php . <pre><code>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]); }</code></pre>
4	Kemudian buatlah view detail.blade.php yang menampilkan detail data mahasiswa pada folder resources/views . View detail juga mengaplikasikan master.blade.php . <pre><code>@extends('master') <!--isi title--> @section('title', 'Detail Mahasiswa') <!--isi bagian judul halaman--> @section('judul_halaman', 'Detail Data Mahasiswa') <!--isi bagian konten--> @section('konten') Kembali

 <h5 class="card-title"> {{\$mahasiswa->nama}} </h5> <p class="card-text"> <label for=""> NIM : </label> {{\$mahasiswa->nim}}</p> <p class="card-text"> <label for=""> E-mail : </label> {{\$mahasiswa->email}}</p> <p class="card-text"> <label for=""> Jurusan : </label> {{\$mahasiswa->jurusan}}</p> @endsection</code></pre>
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 route baru pada routes/web.php dengan nama /mahasiswa/edit yang akan menjalankan fungsi edit pada MahasiswaController <pre>Route::get('/mahasiswa/edit/{id}', 'MahasiswaController@edit');</pre>
2	Buat method edit pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view edit.
	<pre>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]); }</pre> <p>Keterangan :</p> <ul style="list-style-type: none"> - Line 41 : fungsi eloquent untuk mengambil data mahasiswa berdasarkan id yang dipilih
3	Kemudian buatlah view edit.blade.php yang berisi form untuk mengubah data pada folder resources/views . View edit juga mengaplikasikan master.blade.php. <pre>@extends('master') <!--isi title--> @section('title', 'Edit Mahasiswa') <!--isi bagian judul halaman--> @section('judul_halaman', 'Edit Data Mahasiswa') <!--isi bagian konten--> @section('konten') Kembali <form action="/mahasiswa/update/{{ \$mahasiswa->id }}" method="post"> {{ csrf_field() }} <input type="hidden" name="id" value="{{ \$mahasiswa->id }}>
 <div class="form-group"> <label for="namamhs">Nama</label> <input type="text" class="form-control" required="required" name="namamhs" value="{{ \$mahasiswa->nama }}>
 </div> <div class="form-group"> <label for="nimmhs">NIM</label> <input type="number" class="form-control" required="required" name="nimmhs" value="{{ \$mahasiswa->nim }}>
 </div> <div class="form-group"> <label for="emailmhs">E-mail</label> <input type="email" class="form-control" required="required" name="emailmhs" value="{{ \$mahasiswa->email }}>
 </div> <div class="form-group"> <label for="jurusanmhs">Jurusan</label> <input type="text" class="form-control" required="required" name="jurusanmhs" value="{{ \$mahasiswa->jurusan }}>
 </div> <button type="submit" name="edit" class="btn btn-primary float-right">Simpan Data</button> </form> @endsection</pre> <p>Keterangan:</p> <ul style="list-style-type: none"> - 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	<p>Ketika tombol simpan ditekan, akan dipanggil routes <code>/mahasiswa/update/{id}</code>. Oleh karena itu, kita buat terlebih dahulu route tersebut.</p> <pre><code>Route::post('/mahasiswa/update/{id}', 'MahasiswaController@update');</code></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 update di MahasiswaController.php
5	Buat method update pada MahasiswaController untuk menyimpan data yang diubah ke database.

	<pre><code>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('/'); }</code></pre>
	<p>Keterangan:</p> <ul style="list-style-type: none"> - Variabel <code>\$request</code> untuk menerima data yang akan ditambahkan ke database - Line 48-52 merupakan fungsi eloquent untuk update data ke tabel mahasiswa
6	Jalankan <code>localhost:8000</code> 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 <i>route</i> baru pada routes/web.php dengan nama <code>/mahasiswa/hapus</code> yang akan menjalankan fungsi hapus pada MahasiswaController</p> <pre><code>Route::get('/mahasiswa/hapus/{id}', 'MahasiswaController@hapus');</code></pre>
2	<p>Buat method hapus pada MahasiswaController.php di folder app/Http/Controllers yang akan menjalankan fungsi hapus.</p> <pre><code>public function hapus(\$id) { //menghapus data mahasiswa berdasarkan id yang dipilih \$mahasiswa = Mahasiswa::find(\$id); \$mahasiswa->delete(); return redirect('/'); }</code></pre> <p>Keterangan :</p>

	<ul style="list-style-type: none">- 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-*

