

# LAPORAN TUGAS PEMROGRAMAN WEB LANJUT

Jobsheet 10  
Membuat CRUD menggunakan Laravel



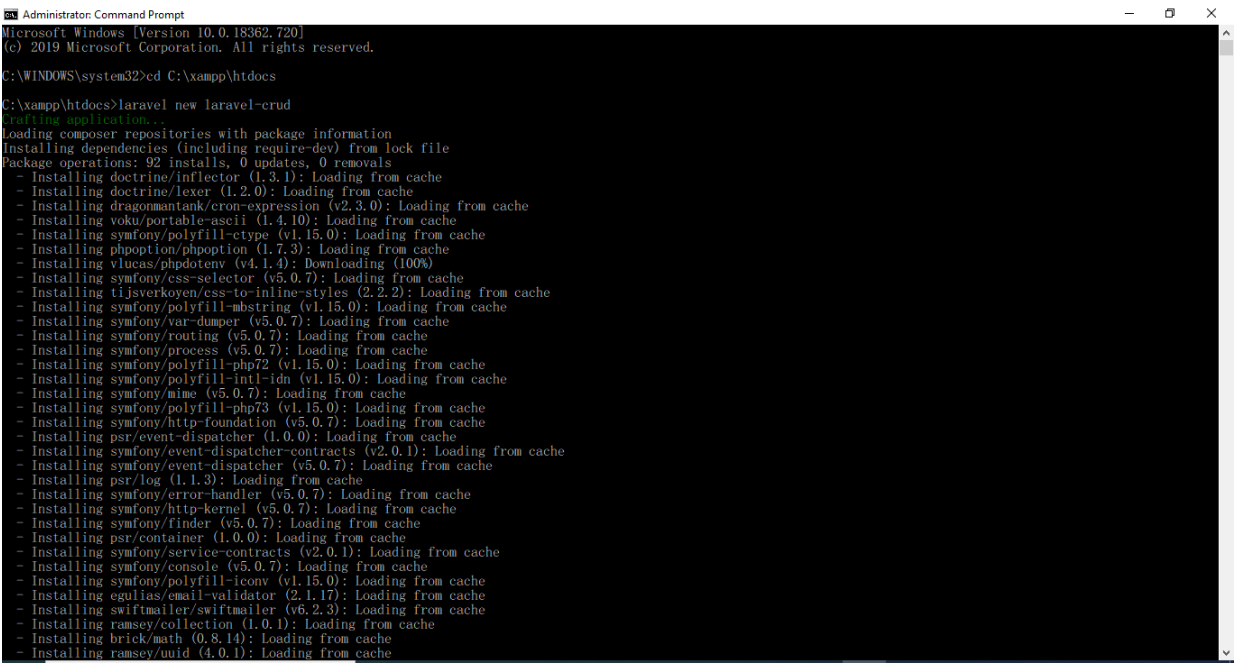
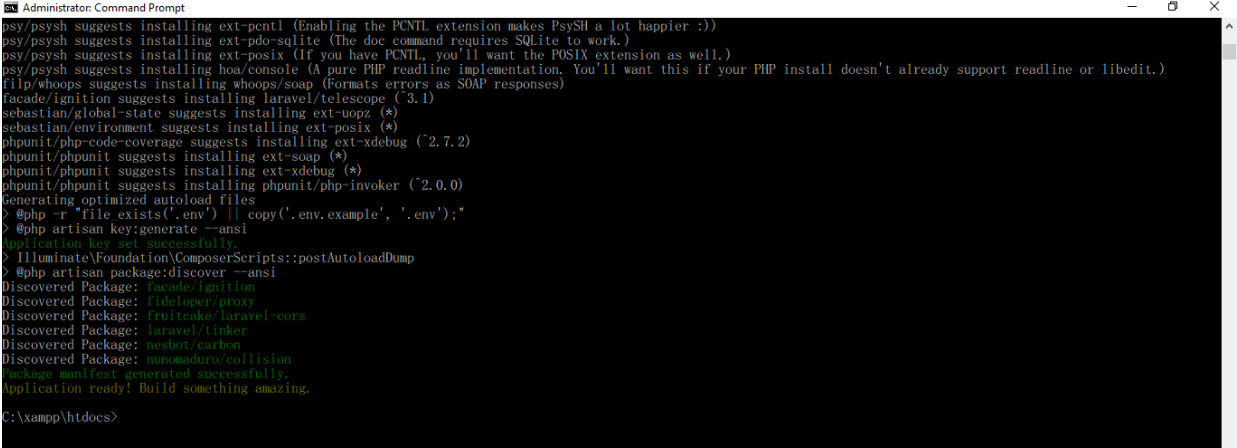
Disusun Oleh :

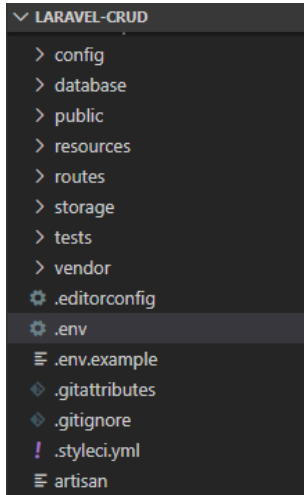
Nama : Hunayn Risatayn  
NIM : 1841720148  
Kelas : TI-2B

Program Studi D-IV Teknik Informatika  
Jurusan Teknologi Informasi  
Politeknik Negeri Malang

## 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> <p><b>cd C:\xampp\htdocs</b></p> <p><b>laravel new laravel-crud</b></p>   <p>Selanjutnya kita lakukan konfigurasi database di Laravel. Untuk melakukan konfigurasi database, bukalah file <b>.env</b> pada project laravel-crud. Ubah seperti di bawah ini.</p> <p>Keterangan:</p> <ul style="list-style-type: none"><li>- Nama database yang akan digunakan adalah latihan_laravel dengan username root</li></ul>



```

.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:KtFYiucTCVb7stNox2c678vG4fDx/aspLL5Qtd+CBM=
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
16 BROADCAST_DRIVER=log

```

Jalankan xampp, selanjutnya buat tabel dengan nama mahasiswa di mysql pada database latihan\_laravel.

Server: 127.0.0.1 » Database: latihan\_laravel » Table: mahasiswa

Table structure


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

Isilah beberapa data pada tabel mahasiswa tersebut.

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

Check all With selected: Edit Copy Delete Export

## 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 <b>routes/web.php</b> sehingga ketika pertama kali project dijalankan akan terbuka halaman yang menampilkan data.</p> <p>Keterangan:</p> <ul style="list-style-type: none"><li>- Ketika route ('/mahasiswa') diakses, akan dijalankan method index pada controller bernama <b>MahasiswaController</b>.</li></ul> <pre>routes &gt; web.php &gt; ... 1  &lt;?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('/mahasiswa', 'MahasiswaController@index'); 21</pre> <p>Buat controller baru menggunakan command prompt yaitu MahasiswaController menggunakan php artisan</p> <p> Command Prompt</p> <pre>Microsoft Windows [Version 10.0.18362.720] (c) 2019 Microsoft Corporation. All rights reserved.  C:\Users\User&gt;cd C:\xampp\htdocs\laravel-crud  C:\xampp\htdocs\laravel-crud&gt;php artisan make:controller MahasiswaController Controller created successfully.  C:\xampp\htdocs\laravel-crud&gt;</pre> <p>Buat method index pada <b>MahasiswaController.php</b> pada folder <b>app/Http/Controllers</b></p> <p>Keterangan:</p> <ul style="list-style-type: none"><li>- Tambahkan 'use Illuminate\Support\Facades\DB;' (line 6) agar query builder dapat digunakan</li><li>- Line 13 untuk mengambil data dari tabel mahasiswa menggunakan query builder laravel dan akan disimpan di variabel \$mahasiswa</li><li>- Line 16 : data akan dikirim ke blade view bernama index</li></ul>

```

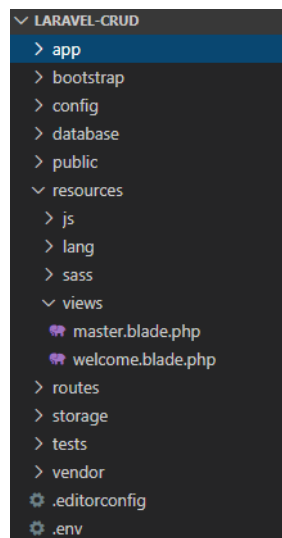
app > Http > Controllers > 🐞 MahasiswaController.php > PHP Intelephense > {} App\Http\Controllers
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use Illuminate\Support\Facades\DB;
7
8  class MahasiswaController extends Controller
9  {
10     //
11     public function index()
12     {
13         $mahasiswa = DB::table('mahasiswa')->get();
14         return view('index', ['mahasiswa' => $mahasiswa]);
15     }
16 }
17

```

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. Nantinya bagian @yield ini akan diisi sesuai dengan halaman view yang menerapkan master.blade.php



```

resources > views > master.blade.php > html
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                         <h2> @yield('judul_halaman') </h2>
17                     </div>
18                     <div class="card-body">
19                         @yield('konten')
20                     </div>
21                 </div>
22             </div>
23         </div>
24     </div>
25 </body>
26
27 </html>

```

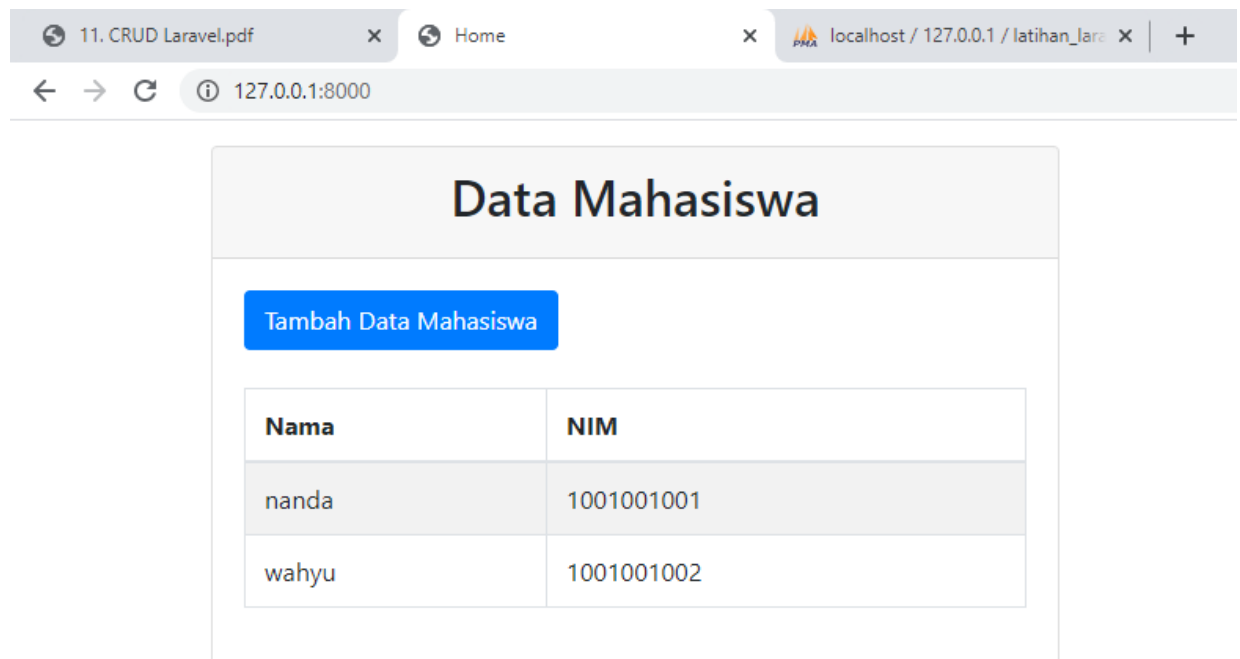
Sekarang buat file **index.blade.php** yang menerapkan template master.blade.php dan akan digunakan untuk menampilkan data.

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

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

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	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/tambah yang akan menjalankan fungsi tambah pada MahasiswaController</p> <pre> 21 22 Route::get('/mahasiswa/tambah', 'MahasiswaController@tambah'); 23 </pre>

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

```
app > Http > Controllers > MahasiswaController.php > PHP Intelephense > MahasiswaController > index
17     public function tambah()
18     {
19         return view('tambah');
20     }
21 }
```

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

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.

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

Activate Windows  
Go to Settings to activate Windows

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

Keterangan:

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



```
routes > web.php > ...
```

```
23
24 Route::post('/mahasiswa/simpan', 'MahasiswaController@simpan');
25
```

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

```
app > Http > Controllers > MahasiswaController.php > ...
```

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

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

NIM

E-mail

Jurusan

Tambah Data

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

## Data Mahasiswa

Tambah Data Mahasiswa

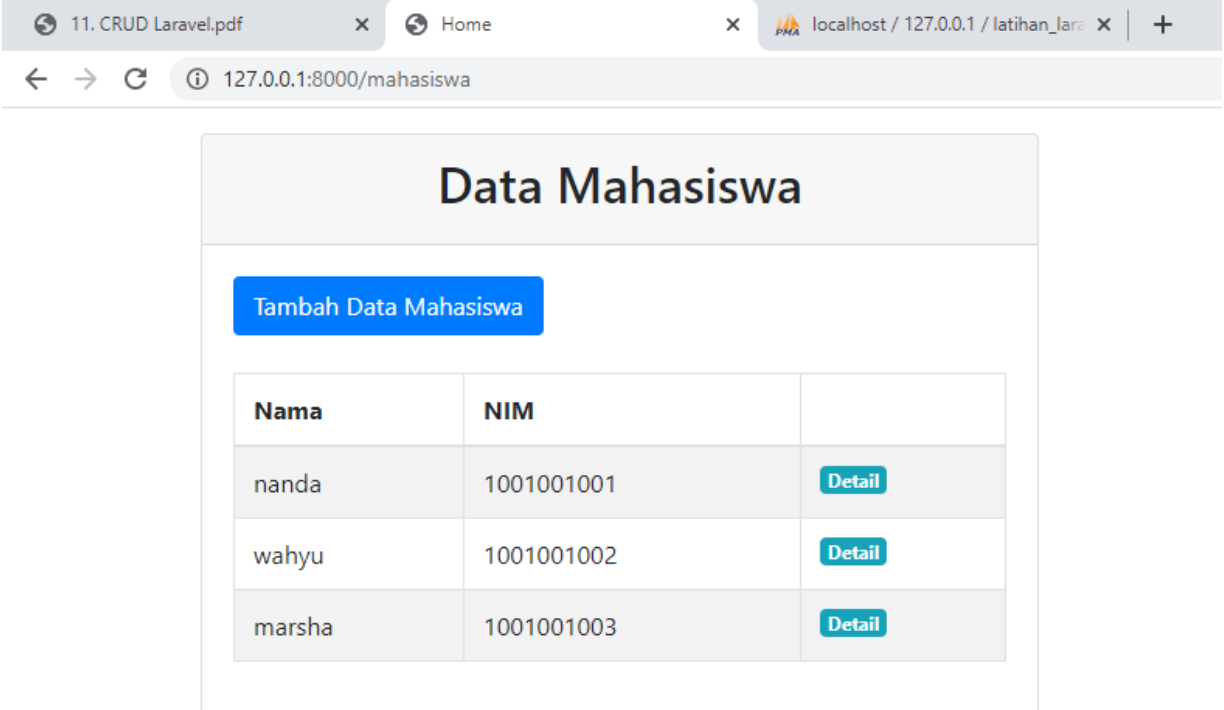
Nama	NIM	
nanda	1001001001	<span>Detail</span>
wahyu	1001001002	<span>Detail</span>
marsha	1001001003	<span>Detail</span>

#### d. Melihat Detail Data dari Database

Langkah	Keterangan
1	<p>Buatlah tombol untuk melihat detail data (Line 28) pada file index.blade.php di folder resources/views</p> <pre>resources &gt; views &gt; index.blade.php &gt; table.table-bordered.table-hover.table-striped 1  @extends('master') 2 3  @section('title', 'Home') 4 5  @section('judul_halaman', 'Data Mahasiswa') 6 7  @section('konten') 8      &lt;a href="/mahasiswa/tambah" class="btn btn-primary"&gt;Tambah Data Mahasiswa&lt;/a&gt; 9      &lt;br&gt; 10     &lt;br&gt; 11     &lt;table class="table table-bordered table-hover table-striped"&gt; 12         &lt;thead&gt; 13             &lt;tr&gt; 14                 &lt;th&gt;Nama&lt;/th&gt; 15                 &lt;th&gt;NIM&lt;/th&gt; 16             &lt;/tr&gt; 17         &lt;/thead&gt; 18         &lt;tbody&gt; 19             @foreach(\$mahasiswa as \$mhs) 20                 &lt;tr&gt; 21                     &lt;td&gt;{{ \$mhs-&gt;nama }}&lt;/td&gt; 22                     &lt;td&gt;{{ \$mhs-&gt;nim }}&lt;/td&gt; 23                     &lt;td&gt; 24                         &lt;a href="/mahasiswa/detail/{{ \$mhs-&gt;id }}" class="badge badge-info"&gt;Detail&lt;/a&gt; 25                     &lt;/td&gt; 26                 &lt;/tr&gt; 27             @endforeach 28         &lt;/tbody&gt; 29     &lt;/table&gt; 30 @endsection</pre> <p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/detail yang akan menjalankan fungsi detail pada MahasiswaController</p> <pre>routes &gt; web.php &gt; ... 25 26 Route::get('/mahasiswa/detail/{id}', 'MahasiswaController@detail'); 27</pre> <p>Buat method detail pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan data mahasiswa pada view detail.blade.php.</p> <pre>app &gt; Http &gt; Controllers &gt; MahasiswaController.php &gt; PHP Intelephense &gt; MahasiswaController &gt; simpan 33 public function detail(\$id) 34 { 35     \$mahasiswa = DB::table('mahasiswa')-&gt;where('id', \$id)-&gt;get(); 36     return view('detail', ['mahasiswa' =&gt; \$mahasiswa]); 37 } 38 39</pre> <p>Kemudian buatlah view detail.blade.php yang menampilkan detail data mahasiswa pada folder resources/views. View detail juga mengaplikasikan master.blade.php.</p>

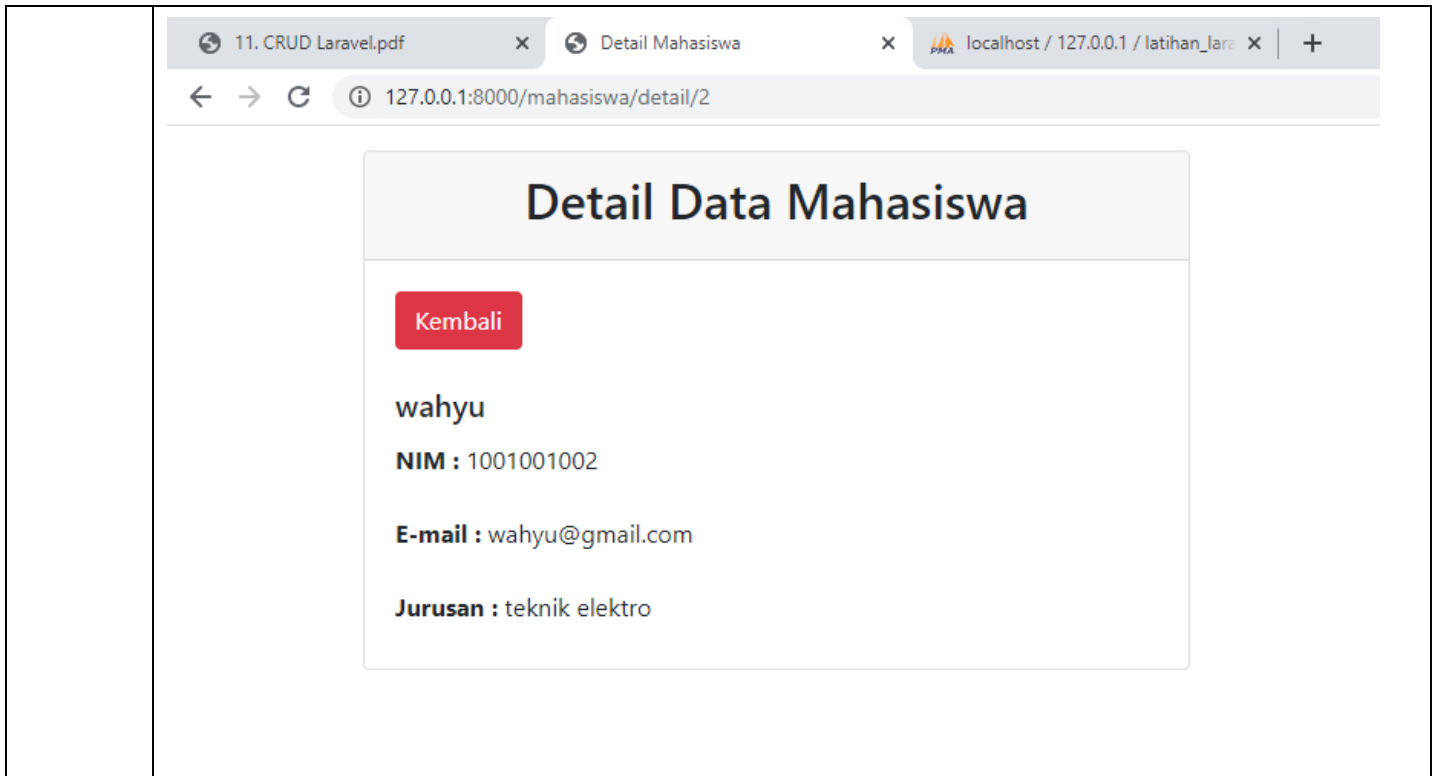
```
resources > views > detail.blade.php > ...
1  @extends('master')
2
3  @section('title', 'Detail Mahasiswa')
4
5  @section('judul_halaman', 'Detail Data Mahasiswa')
6
7  @section('konten')
8  <a href="/mahasiswa" class="btn btn-danger">Kembali</a>
9  <br>
10 <br>
11 @foreach($mahasiswa as $mhs)
12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

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



The screenshot shows a web browser window with the URL `127.0.0.1:8000/mahasiswa`. The page displays a section titled "Data Mahasiswa". Inside this section, there is a blue button labeled "Tambah Data Mahasiswa". Below the button is a table with three rows of student data. Each row contains the student's name, NIM, and a "Detail" button.

Nama	NIM	
nanda	1001001001	<a href="#">Detail</a>
wahyu	1001001002	<a href="#">Detail</a>
marsha	1001001003	<a href="#">Detail</a>



#### 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> resources &gt; views &gt; index.blade.php &gt; table.table.table-bordered.table-hover.table-striped 1  @extends('master') 2 3  @section('title', 'Home') 4 5  @section('judul_halaman', 'Data Mahasiswa') 6 7  @section('konten') 8  &lt;a href="/mahasiswa/tambah" class="btn btn-primary"&gt;Tambah Data Mahasiswa&lt;/a&gt; 9  &lt;br&gt; 10 &lt;br&gt; 11 &lt;table class="table table-bordered table-hover table-striped"&gt; 12     &lt;thead&gt; 13         &lt;tr&gt; 14             &lt;th&gt;Nama&lt;/th&gt; 15             &lt;th&gt;NIM&lt;/th&gt; 16             &lt;th&gt;&lt;/th&gt; 17         &lt;/tr&gt; 18     &lt;/thead&gt; 19     &lt;tbody&gt; 20         @foreach(\$mahasiswa as \$mhs) 21             &lt;tr&gt; 22                 &lt;td&gt;{{ \$mhs-&gt;nama }}&lt;/td&gt; 23                 &lt;td&gt;{{ \$mhs-&gt;nim }}&lt;/td&gt; 24                 &lt;td&gt; 25                     &lt;a href="/mahasiswa/detail/{{ \$mhs-&gt;id }}" class="badge badge-info"&gt;Detail&lt;/a&gt; 26                     &lt;a href="/mahasiswa/edit/{{ \$mhs-&gt;id }}" class="badge badge-warning"&gt;Edit&lt;/a&gt; 27                 &lt;/td&gt; 28             &lt;/tr&gt; 29             @endforeach 30         &lt;/tbody&gt; 31     &lt;/table&gt; 32 @endsection </pre> <p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/edit yang akan menjalankan fungsi edit pada MahasiswaController</p>

```

routes > web.php > ...
27
28 Route::get('/mahasiswa/edit/{id}', 'MahasiswaController@edit');
29

```

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

Keterangan :

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

```

app > Http > Controllers > MahasiswaController.php > PHP Intelephense > MahasiswaController > detail
39 public function edit($id)
40 {
41     $mahasiswa = DB::table('mahasiswa')->where('id', $id)->get();
42     return view('edit', ['mahasiswa' => $mahasiswa]);
43 }
44 }
45

```

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

```

resources > views > edit.blade.php > form > div.form-group > input.form-control
1 @extends('master')
2
3 @section('title', 'Edit Data')
4
5 @section('judul_halaman', 'Edit Data Mahasiswa')
6
7 @section('konten')
8 <a href="/mahasiswa" class="btn btn-danger">Kembali</a>
9 <br>
10 <br>
11 @foreach($mahasiswa as $mhs)
12 <form action="/mahasiswa/update" method="post">
13     {{ csrf_field() }}
14     <input type="hidden" name="id" value="{{ $mhs->id }}"><br>
15     <div class="form-group">
16         <label for="namamhs">Nama</label>
17         <input type="text" class="form-control" required="required" name="namamhs" value="{{ $mhs->nama }}"><br>
18     </div>
19     <div class="form-group">
20         <label for="nimhs">NIM</label>
21         <input type="number" class="form-control" required="required" name="nimhs" value="{{ $mhs->nim }}"><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" value="{{ $mhs->email }}"><br>
26     </div>

```

```

resources > views > edit.blade.php > form > div.form-group > input.form-control
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" value="{{ $mhs->email }}"><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"
30         value="{{ $mhs->jurusan }}"><br>
31 </div>
32 <button type="submit" name="edit" class="btn btn-primary float-right">Simpan Data</button>
33 </form>
34 @endforeach
35 @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

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

Keterangan:

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

```
routes > web.php > ...  
29  
30 Route::post('/mahasiswa/update', 'MahasiswaController@update');  
31
```

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

Keterangan:

- Variabel \$request untuk menerima data yang akan ditambahkan ke database

- Line 58-63 merupakan query builder untuk update data ke tabel mahasiswa

```
app > Http > Controllers > MahasiswaController.php > PHP Intelephense > MahasiswaController > edit  
45 public function update(Request $request)  
46 {  
47     DB::table('mahasiswa')->where('id', $request->id)->update([  
48         'nama' => $request->namamhs,  
49         'nim' => $request->nimmhs,  
50         'email' => $request->emailmhs,  
51         'jurusan' => $request->jurusanmhs  
52     ]);  
53     return redirect('/mahasiswa');  
54 }  
55 }  
56
```

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

11. CRUD Laravel.pdf

Home

localhost / 127.0.0.1 / latihan\_lara

127.0.0.1:8000/mahasiswa

## Data Mahasiswa

Tambah Data Mahasiswa

Nama	NIM	
nanda	1001001001	<a href="#">Detail</a> <a href="#">Edit</a>
wahyu	1001001002	<a href="#">Detail</a> <a href="#">Edit</a>
marsha	1001001003	<a href="#">Detail</a> <a href="#">Edit</a>

11. CRUD Laravel.pdf

Edit Data

localhost / 127.0.0.1 / latihan\_lara

127.0.0.1:8000/mahasiswa/edit/2

## Edit Data Mahasiswa

Kembali

Nama

wahyu

NIM

1001001002

E-mail

wahyu@gmail.com

Jurusan

teknik elektro

Simpan Data



The screenshot displays a web application interface for editing student data. The browser window shows multiple tabs, with the active one being 'Edit Data'. The URL in the address bar is '127.0.0.1:8000/mahasiswa/edit/2'. The form itself is titled 'Edit Data Mahasiswa' and contains several elements: a red button labeled 'Kembali' (Back), four text input fields for 'Nama' (filled with 'wahyu'), 'NIM' (filled with '1001001002'), 'E-mail' (filled with 'wahyu@gmail.com'), and 'Jurusan' (filled with 'teknik kimia'), and a blue button labeled 'Simpan Data' (Save Data) at the bottom right.

**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

```
resources > views > index.blade.php > table.table-bordered.table-hover.table-striped > thead > tr
19 <tbody>
20     @foreach($mahasiswa as $mhs)
21         <tr>
22             <td>{{ $mhs->nama }}</td>
23             <td>{{ $mhs->nim }}</td>
24             <td>
25                 <a href="/mahasiswa/detail/{{ $mhs->id }}" class="badge badge-info">Detail</a>
26                 <a href="/mahasiswa/edit/{{ $mhs->id }}" class="badge badge-warning">Edit</a>
27                 <a href="/mahasiswa/hapus/{{ $mhs->id }}" class="badge badge-danger">Hapus</a>
28             </td>
29         </tr>
30     @endforeach
31 </tbody>
32 </table>
33 @endsection
```

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

```
routes > web.php > ...
31
32 Route::get('/mahasiswa/hapus/{id}', 'MahasiswaController@hapus');
33
```

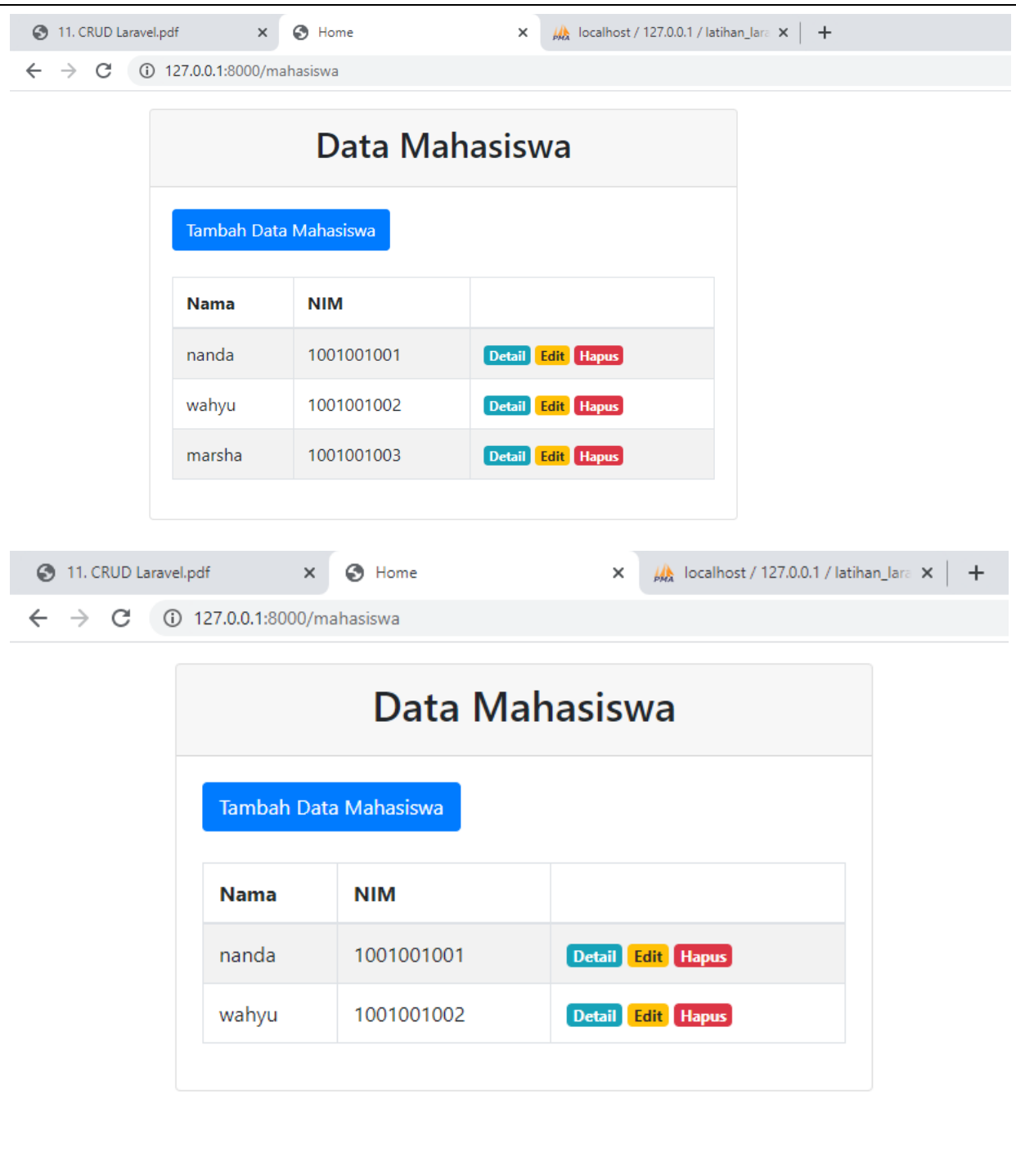
Buat method hapus pada MahasiswaController.php di folder app/Http/Controllers yang akan menjalankan fungsi hapus.

Keterangan :

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

```
app > Http > Controllers > MahasiswaController.php > PHP Intelephense > MahasiswaController > update
56 public function hapus($id)
57 {
58     DB::table('mahasiswa')->where('id', $id)->delete();
59     return redirect('/mahasiswa');
60 }
61
62
```

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

### 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 laravel new laravel-crud-kedua</pre>

```
Command Prompt
C:\xampp\htdocs>laravel-crud>cd C:\xampp\htdocs
C:\xampp\htdocs>laravel new laravel-crud-kedua
Creating 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/inferno (1.3.1): Loading from cache
- Installing doctrine/lexer (1.2.0): Loading from cache
- Installing dragonmantank/cron-expression (v2.3.0): Loading from cache
- Installing voku/portable-ascii (1.4.10): Loading from cache
- Installing symfony/polyfill-ctype (v1.15.0): Loading from cache
- Installing phpoption/phpoption (1.7.3): Loading from cache
- Installing vlucas/phpdotenv (v4.1.4): Loading from cache
- Installing symfony/ess-selector (v5.0.7): Loading from cache
- Installing tijsverkoyen/css-to-inline-styles (2.2.2): Loading from cache
- Installing symfony/polyfill-mbstring (v1.15.0): Loading from cache
- Installing symfony/var-dumper (v5.0.7): Loading from cache
- Installing symfony/routing (v5.0.7): Loading from cache
- Installing symfony/process (v5.0.7): Loading from cache
- Installing symfony/polyfill-php72 (v1.15.0): Loading from cache
- Installing symfony/polyfill-intl-idn (v1.15.0): Loading from cache
- Installing symfony/mime (v5.0.7): Loading from cache
- Installing symfony/polyfill-php73 (v1.15.0): Loading from cache
- Installing symfony/http-foundation (v5.0.7): Loading from cache
- Installing psr/event-dispatcher (1.0.0): Loading from cache
- Installing symfony/event-dispatcher-contracts (v2.0.1): Loading from cache
- Installing symfony/event-dispatcher (v5.0.7): Loading from cache
- Installing psr/log (1.1.3): Loading from cache
- Installing symfony/error-handler (v5.0.7): Loading from cache
- Installing symfony/http-kernel (v5.0.7): Loading from cache
- Installing symfony/finder (v5.0.7): Loading from cache
- Installing psr/container (1.0.0): Loading from cache
- Installing symfony/service-contracts (v2.0.1): Loading from cache
- Installing symfony/console (v5.0.7): Loading from cache
- Installing symfony/polyfill-iconv (v1.15.0): Loading from cache
- Installing egulias/email-validator (2.1.17): Loading from cache
- Installing swiftmailer/swiftmailer (v6.2.3): Loading from cache
- Installing ramsey/collection (1.0.1): Loading from cache
- Installing brick/math (0.8.14): Loading from cache
- Installing ramsey/uuid (4.0.1): Loading from cache
- Installing psr/simple-cache (1.0.1): Loading from cache
- Installing opis/closure (3.5.1): Loading from cache
- Installing symfony/translation-contracts (v2.0.1): Loading from cache
```

```
Command Prompt
laravel/framework suggests installing ext-memcached (Required to use the memcache cache driver.)
laravel/framework suggests installing ext-pcntl (Required to use all features of the queue worker.)
laravel/framework suggests installing ext-posix (Required to use all features of the queue worker.)
laravel/framework suggests installing ext-redis (Required to use the Redis cache and queue drivers (^4.0|^5.0).)
laravel/framework suggests installing league/flysystem-aws-s3-v3 (Required to use the Flysystem S3 driver (^1.0).)
laravel/framework suggests installing league/flysystem-azure-blob (Required to use the Flysystem cache (^1.0).)
laravel/framework suggests installing league/flysystem-sftp (Required to use the Flysystem SFTP driver (^1.0).)
laravel/framework suggests installing moonshot/math (Required to use ordered UUIDs (^1.1).)
laravel/framework suggests installing nyholm/psr7 (Required to use PSR-7 bridging features (^1.2).)
laravel/framework suggests installing pda/pheanstalk (Required to use the beanstalk queue driver (^4.0).)
laravel/framework suggests installing pusher/pusher-php-server (Required to use the Pusher broadcast driver (^4.0).)
laravel/framework suggests installing symfony/cache (Required to use PSR-6 cache bridge (^5.0).)
laravel/framework suggests installing symfony/psr-http-message-bridge (Required to use PSR-7 bridging features (^2.0).)
laravel/framework suggests installing wildbit/swiftmailer-postmark (Required to use Postmark mail driver (^3.0).)
guzzlehttp/guzzle suggests installing zendframework/zend-httphandler (Emit PSR-7 responses)
psr/psr7 suggests installing ext-intl (Required for Internationalized Domain Name (IDN) support)
psr/psr7 suggests installing ext-pcntl (Enabling the PCNTL extension makes PsySH a lot happier :))
psr/psr7 suggests installing ext-pdo-sqlite (The doc command requires SQLite to work.)
psr/psr7 suggests installing ext-posix (If you have PCNTL, you'll want the POSIX extension as well.)
psr/psr7 suggests installing htda/console (A pure PHP readline implementation. You'll want this if your PHP install doesn't already support readline or libedit.)
filp/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
> @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.
```

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.

```
.env
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
```

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`

The screenshot shows the 'Table structure' view for the 'mahasiswa' table in the 'latihan\_laravel' database. The table has 7 columns: id, nama, nim, email, jurusan, created\_at, and updated\_at. The 'created\_at' and 'updated\_at' columns are of type 'timestamp' and have a default value of 'NULL'.

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

## 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

```
routes > web.php > #Function#69cd17c
20 Route::get('/', 'MahasiswaController@index');
21 Route::get('/mahasiswa', 'MahasiswaController@index');
22
```

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

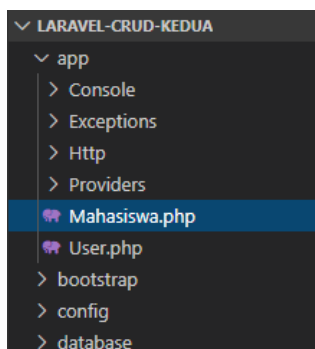
`php artisan make:model Mahasiswa`

```
C:\xampp\htdocs>cd C:\xampp\htdocs\laravel-crud-kedua
C:\xampp\htdocs\laravel-crud-kedua>php artisan make:model Mahasiswa
Model created successfully.
C:\xampp\htdocs\laravel-crud-kedua>
```

Ubah model `Mahasiswa.php` pada folder `App` menjadi seperti berikut.

Keterangan:

- Model Mahasiswa akan menangani tabel mahasiswa



```

app > Mahasiswa.php > ...
1  <?php
2
3  namespace App;
4
5  use Illuminate\Database\Eloquent\Model;
6
7  class Mahasiswa extends Model
8  {
9      //
10     protected $table = "mahasiswa";
11 }
12

```

Buat controller baru yaitu MahasiswaController menggunakan php artisan

**php artisan make:controller MahasiswaController**

```

C:\xampp\htdocs\laravel-crud-kedua>php artisan make:controller MahasiswaController
Controller created successfully.

```

```

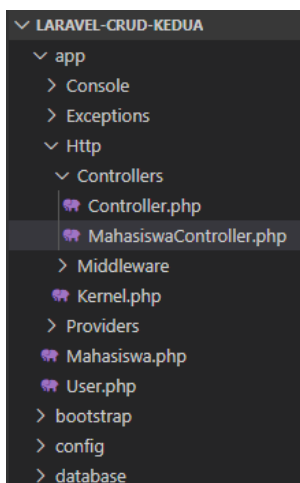
C:\xampp\htdocs\laravel-crud-kedua>

```

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

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



```

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     //
11     public function index()
12     {
13         $mahasiswa = Mahasiswa::all();
14         return view('index', ['mahasiswa' => $mahasiswa]);
15     }
16 }
17

```

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

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

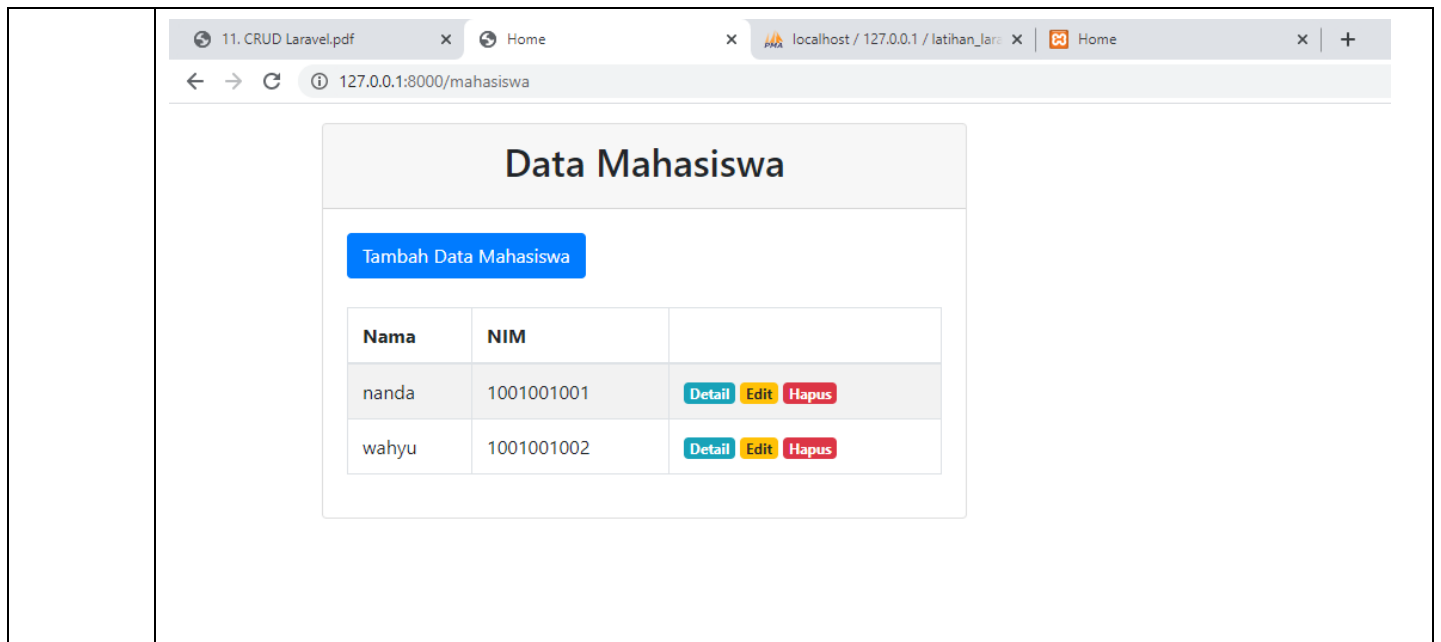
```

Administrator: Command Prompt - php artisan serve
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd C:\xampp\htdocs\laravel-crud-kedua

C:\xampp\htdocs\laravel-crud-kedua>php artisan serve
Laravel development server started: http://127.0.0.1:8000
[Mon Apr 13 11:12:50 2020] 127.0.0.1:52412 [200]: /favicon.ico
[Mon Apr 13 11:12:54 2020] 127.0.0.1:52418 [200]: /favicon.ico
[Mon Apr 13 11:13:02 2020] 127.0.0.1:52424 [200]: /favicon.ico
[Mon Apr 13 11:13:08 2020] 127.0.0.1:52429 [200]: /favicon.ico

```



### 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> routes &gt; web.php 22 23 Route::get('/mahasiswa/tambah', 'MahasiswaController@tambah'); 24 </pre> <p>Buat method tambah pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view tambah.</p> <pre> app &gt; Http &gt; Controllers &gt; MahasiswaController.php &gt; PHP Intelephense &gt; MahasiswaController &gt; index 17 public function tambah() 18 { 19     return view('tambah'); 20 } 21 22 </pre> <p>Kemudian buatlah view tambah.blade.php yang berisi form untuk memasukkan data baru pada folder resources/views.</p> <p>Kita lakukan copy paste dari tambah.blade.php di Bagian 1</p> <p>Ketika tombol simpan ditekan, akan dipanggil routes /mahasiswa/simpan. Oleh karena itu, kita buat terlebih dahulu route tersebut.</p> <p>Keterangan:</p> <ul style="list-style-type: none"> <li>- Pada route ini menggunakan metode post karena data mahasiswa dari form akan dikirim ke method simpan di MahasiswaController.php</li> </ul>



```
routes > web.php
```

```
24
25 Route::post('/mahasiswa/simpan', 'MahasiswaController@simpan');
26
```

Buat method simpan pada MahasiswaController untuk menyimpan data ke database

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

```
app > Http > Controllers > MahasiswaController.php > PHP Intelephense > MahasiswaController > tambah
```

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

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.

```
app > Mahasiswa.php > PHP Intelephense > Mahasiswa
```

```
7 class Mahasiswa extends Model
8 {
9     //
10    protected $table = "mahasiswa";
11    protected $fillable = ['nama', 'nim', 'email', 'jurusan'];
12 }
13
```

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.

11. CRUD Laravel.pdf
Tambah Data
localhost / 127.0.0.1 / latihan\_lar...
Home
127.0.0.1:8000/mahasiswa/tambah

## Tambah Data Mahasiswa

Kembali

Nama

NIM

E-mail

Jurusan

Tambah Data

11. CRUD Laravel.pdf
Home
localhost / 127.0.0.1 / latihan\_lar...
Home
127.0.0.1:8000/mahasiswa

## Data Mahasiswa

Tambah Data Mahasiswa

Nama	NIM	
nanda	1001001001	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
wahyu	1001001002	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
rahmat	1001001006	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>

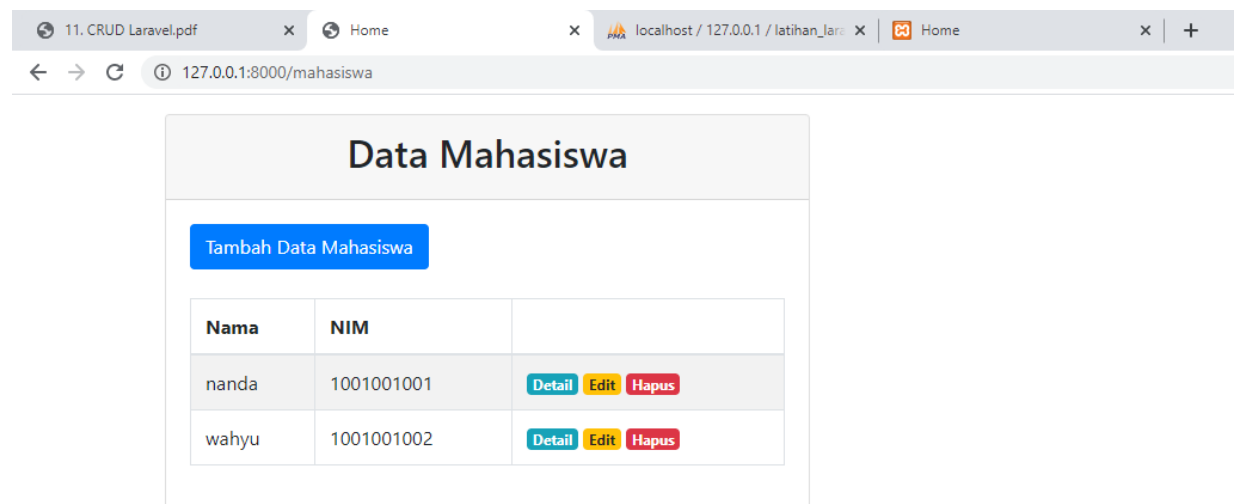
Jalankan command prompt, tuliskan perintah untuk menjalankan project laravel-crud php artisan serve  
Buka browser dan ketikkan localhost:8000, maka akan tampil sebagai berikut

Administrator: Command Prompt - php artisan serve

```
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd C:\xampp\htdocs\laravel-crud-kedua

C:\xampp\htdocs\laravel-crud-kedua>php artisan serve
Laravel development server started: http://127.0.0.1:8000
[Mon Apr 13 11:12:50 2020] 127.0.0.1:52412 [200]: /favicon.ico
[Mon Apr 13 11:12:54 2020] 127.0.0.1:52418 [200]: /favicon.ico
[Mon Apr 13 11:13:02 2020] 127.0.0.1:52424 [200]: /favicon.ico
[Mon Apr 13 11:13:08 2020] 127.0.0.1:52429 [200]: /favicon.ico
```



#### d. Melihat Detail Data dari Database

Langkah	Keterangan
1	<p>Buatlah route baru pada routes/web.php dengan nama /mahasiswa/detail yang akan menjalankan fungsi detail pada MahasiswaController</p> <pre>routes &gt; web.php 26 27 Route::get('/mahasiswa/detail/{id}', 'MahasiswaController@detail'); 28</pre> <p>Buat method detail pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan data mahasiswa pada view detail.blade.php.</p>

```

app > Http > Controllers > MahasiswaController.php > PHP Intelephense > MahasiswaController > simpan
33     public function detail($id)
34     {
35         $mahasiswa = Mahasiswa::find($id);
36         return view('detail', ['mahasiswa' => $mahasiswa]);
37     }
38 }
39

```

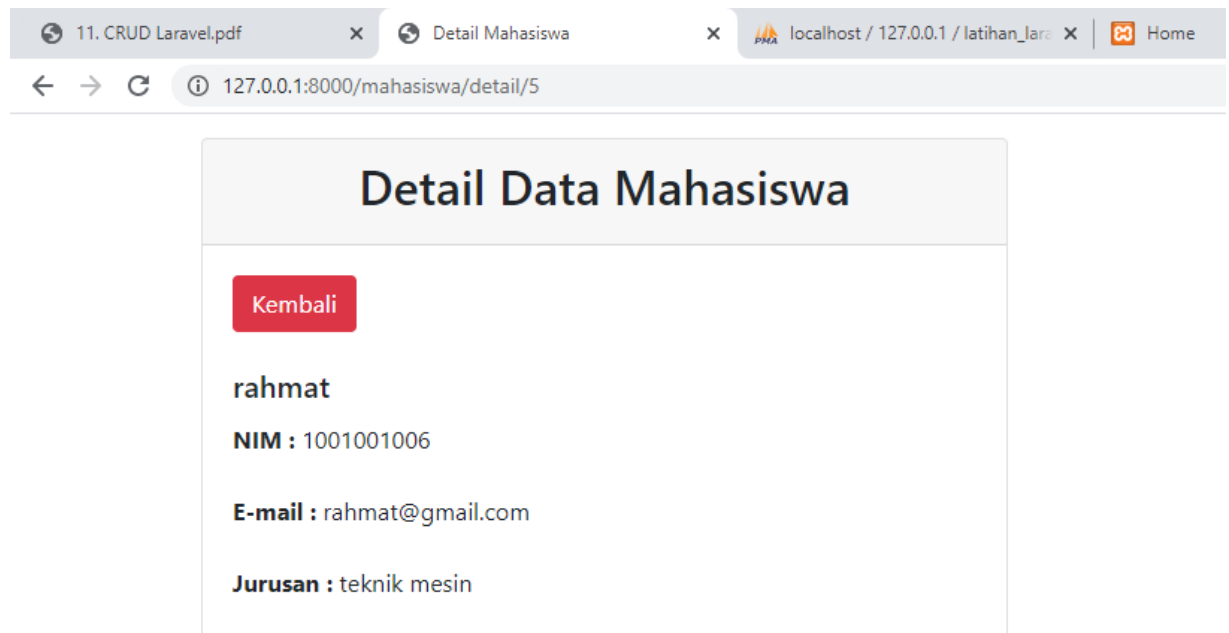
Kemudian buatlah view detail.blade.php yang menampilkan detail data mahasiswa pada folder resources/views. View detail juga mengaplikasikan master.blade.php.

```

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

```

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



### 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> routes &gt; web.php 28 29 Route::get('/mahasiswa/edit/{id}', 'MahasiswaController@edit'); 30 </pre> <p>Buat method edit pada MahasiswaController.php di folder app/Http/Controllers yang akan menampilkan view edit.</p> <pre> app &gt; Http &gt; Controllers &gt; MahasiswaController.php &gt; PHP Intelephense &gt; MahasiswaController &gt; detail 39 public function edit(\$id) 40 { 41     \$mahasiswa = Mahasiswa::find(\$id); 42     return view('edit', ['mahasiswa' =&gt; \$mahasiswa]); 43 } 44 } 45 </pre> <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> <p>Keterangan:</p> <ul style="list-style-type: none"> <li>- Line 11-31 merupakan form untuk memasukkan data mahasiswa berupa nama, nim, email, dan jurusan</li> <li>- Line 11 terdapat action="/mahasiswa/update/{{ \$mahasiswa-&gt;id }}" yang menunjukkan routes /mahasiswa/update/{id} dimana data pada form tersebut akan dikirimkan ke fungsi update pada controller MahasiswaController</li> </ul> <pre> resources &gt; views &gt; edit.blade.php &gt; form &gt; div.form-group &gt; input.form-control 1 @extends('master') 2 3 @section('title', 'Edit Data') 4 5 @section('judul_halaman', 'Edit Data Mahasiswa') 6 7 @section('konten') 8 &lt;a href="/mahasiswa" class="btn btn-danger"&gt;Kembali&lt;/a&gt; 9 &lt;br&gt; 10 &lt;br&gt; 11 &lt;form action="/mahasiswa/update/{{ \$mahasiswa-&gt;id }}" method="post"&gt; 12     {{ csrf_field() }} 13     &lt;input type="hidden" name="id" value="{{ \$mahasiswa-&gt;id }}"&gt;&lt;br&gt; 14     &lt;div class="form-group"&gt; 15         &lt;label for="namamhs"&gt;Nama&lt;/label&gt; 16         &lt;input type="text" class="form-control" required="required" name="namamhs" 17             value="{{ \$mahasiswa-&gt;nama }}"&gt;&lt;br&gt; 18     &lt;/div&gt; 19     &lt;div class="form-group"&gt; 20         &lt;label for="nimmhs"&gt;NIM&lt;/label&gt; 21         &lt;input type="number" class="form-control" required="required" name="nimmhs" 22             value="{{ \$mahasiswa-&gt;nim }}"&gt;&lt;br&gt; 23     &lt;/div&gt; 24     &lt;div class="form-group"&gt; 25         &lt;label for="emailmhs"&gt;E-mail&lt;/label&gt; 26         &lt;input type="email" class="form-control" required="required" name="emailmhs" 27             value="{{ \$mahasiswa-&gt;email }}"&gt;&lt;br&gt; 28     &lt;/div&gt; </pre>

```
resources > views > edit.blade.php > form > div.form-group > input.form-control
19 <div class="form-group">
20 <label for="nimmhs">NIM</label>
21 <input type="number" class="form-control" required="required" name="nimmhs"
22 value="{{ $mahasiswa->nim }}"><br>
23 </div>
24 <div class="form-group">
25 <label for="emailmhs">E-mail</label>
26 <input type="email" class="form-control" required="required" name="emailmhs"
27 value="{{ $mahasiswa->email }}"><br>
28 </div>
29 <div class="form-group">
30 <label for="jurusanmhs">Jurusan</label>
31 <input type="text" class="form-control" required="required" name="jurusanmhs"
32 value="{{ $mahasiswa->jurusan }}"><br>
33 </div>
34 <button type="submit" name="edit" class="btn btn-primary float-right">Simpan Data</button>
35 </form>
36 @endsection
```

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

Keterangan:

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

```
routes > web.php
30
31 Route::post('/mahasiswa/update/{id}', 'MahasiswaController@update');
32
```

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

Keterangan:

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

```
app > Http > Controllers > MahasiswaController.php > PHP Intelephense > MahasiswaController > edit
45 public function update($id, Request $request)
46 {
47     $mahasiswa = Mahasiswa::find($id);
48     $mahasiswa->nama = $request->namamhs;
49     $mahasiswa->nim = $request->nimmhs;
50     $mahasiswa->email = $request->emailmhs;
51     $mahasiswa->jurusan = $request->jurusanmhs;
52     $mahasiswa->save();
53     return redirect('/mahasiswa');
54 }
55 }
56
```

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

11. CRUD Laravel.pdf x Edit Data x localhost / 127.0.0.1 / latihan\_lars x Home

127.0.0.1:8000/mahasiswa/edit/5

## Edit Data Mahasiswa

[Kembali](#)

Nama

NIM

E-mail

Jurusan

[Simpan Data](#)

11. CRUD Laravel.pdf

Edit Data

localhost / 127.0.0.1 / latihan\_lar... Home

127.0.0.1:8000/mahasiswa/edit/5

## Edit Data Mahasiswa

Kembali

Nama

rahmat

NIM

1001001006

E-mail

rahmat@gmail.com

Jurusan

teknik sipil

Simpan Data

11. CRUD Laravel.pdf

Home

localhost / 127.0.0.1 / latihan\_lar... Home

127.0.0.1:8000/mahasiswa

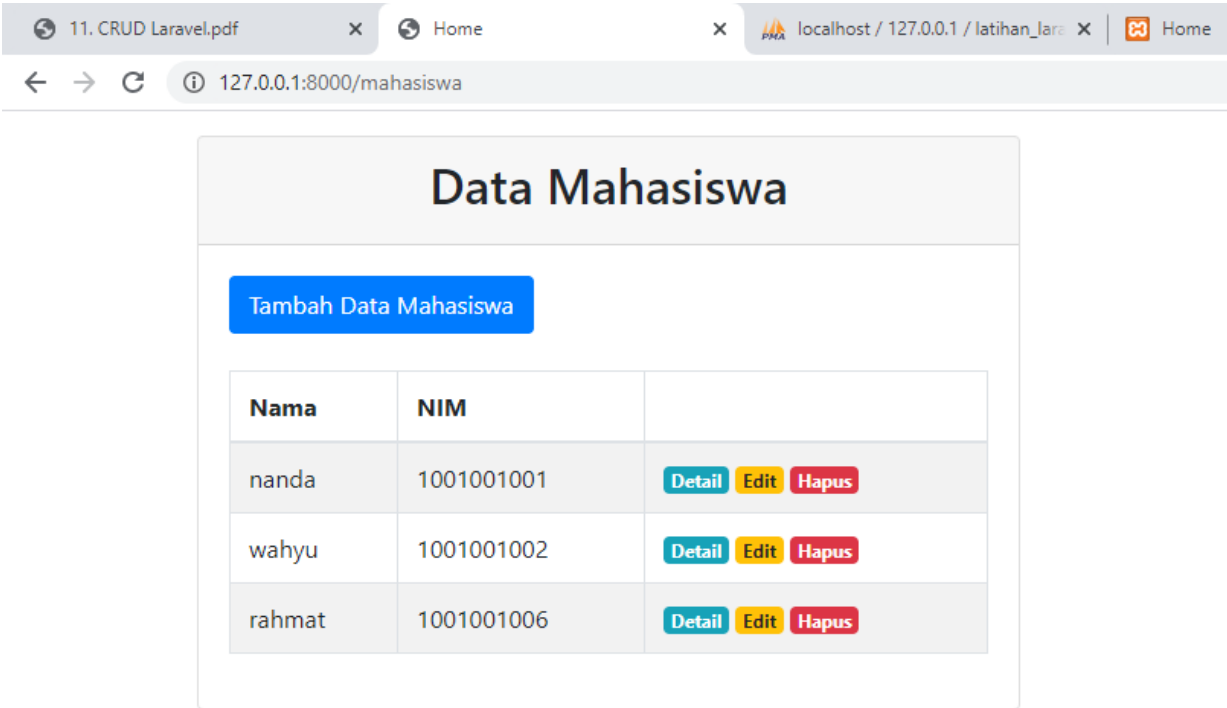
## Data Mahasiswa

Tambah Data Mahasiswa

Nama	NIM	
nanda	1001001001	<div>Detail Edit Hapus</div>
wahyu	1001001002	<div>Detail Edit Hapus</div>
rahmat	1001001006	<div>Detail Edit Hapus</div>



## 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> routes &gt; web.php 32 33 Route::get('/mahasiswa/hapus/{id}', 'MahasiswaController@hapus'); 34 </pre> <p>Buat method hapus pada MahasiswaController.php di folder app/Http/Controllers yang akan menjalankan fungsi hapus.</p> <p>Keterangan :</p> <ul style="list-style-type: none"> <li>- Line 59 :fungsi eloquent untuk menghapus data mahasiswa berdasarkan id yang dipilih</li> </ul> <pre> app &gt; Http &gt; Controllers &gt; MahasiswaController.php &gt; PHP Intelephense &gt; MahasiswaController &gt; update 56 public function hapus(\$id) 57 { 58     \$mahasiswa = Mahasiswa::find(\$id); 59     \$mahasiswa-&gt;delete(); 60     return redirect('/mahasiswa'); 61 } 62 63 </pre> <p>Jalankan localhost:8000 dan pilih tombol 'Hapus', maka data yang terpilih akan dihapus.</p> 

11. CRUD Laravel.pdf

Home

localhost / 127.0.0.1 / latihan\_lar... Home

127.0.0.1:8000/mahasiswa

# Data Mahasiswa

Tambah Data Mahasiswa

Nama	NIM	
nanda	1001001001	<div>Detail Edit Hapus</div>
wahyu	1001001002	<div>Detail Edit Hapus</div>

