

# LARAVEL 9

CARA MUDAH MEMBUAT WEB BAGI  
KAUM REBAHAN

Muharir, M.Kom

# APLIKASI PENGGAJIAN

UNIVERSITAS ISLAM KALIMANTAN M.A.B  
FAKULTAS TEKNOLOGI INFORMASI  
**2023**

## DAFTAR ISI

DAFTAR ISI .....	1
KEBUTUHAN MINIMAL DEVELOPMENT .....	2
MENGINSTALL LARAVEL FRAMEWORK .....	1
MELAKUKAN KONFIGURASI DATABASE .....	2
MENGINSTALL ADMIN LTE TEMPLATE. ....	5
MENGKONFIGURASI TEMPLATE ADMIN LTE.....	10
MENAMPILKAN DATA PENGGUNA (CRUD USER).....	11
MENGINSTALL SWEETALERT, DATATABLE DAN SELECT2 .....	16
MEMBUAT TAMBAH DATA PENGGUNA .....	18
MEMBUAT EDIT PENGGUNA .....	24
MEMBUAT HAPUS PENGGUNA .....	28
MEMBUAT CRUD JABATAN.....	30
MEMBUAT CETAK PDF DATA JABATAN.....	42
MEMBUAT GRAFIK JABATAN .....	46
EXPORT JABATAN KE EXCEL (BONUS) .....	49
TENTANG PENULIS .....	53

## **Kebutuhan Minimal Development**

Agar development berjalan lancar maka ada beberapa requirement yang perlu diperhatikan di antaranya,

- Browser yang dapat digunakan bisa beragam seperti Google Chrome, Mozilla Firefox, dan Safari.
- Text editor dapat menggunakan Vscode maupun text editor lainnya. Namun pada case kali ini penulis menggunakan Vscode dengan tambahan extensions di antaranya
  - o Auto close tag
  - o Auto rename tag
  - o PHP Intelephense
  - o Laravel Extension Pack
- Composer 2.3.5 atau yang lebih tinggi sebagai dependency manager untuk menginstall library-library yang di butuhkan pada project yang akan dibangun.
- PHP Versi 8.1 atau yang lebih tinggi.
- Nodejs untuk compile asset pada laravel mix.
- Anda dapat menggunakan Laragon pada Windows, agar proses penginstalannya lebih mudah dan Minimal Developmentnya terpenuhi.
- Pada Linux dapat menggunakan XAMPP ataupun install tools diatas secara terpisah.
- Pada Mac anda dapat menggunakan MAMP ataupun XAMPP . Atau kalian dapat menggunakan tools alternative yang lain yaitu Docker.

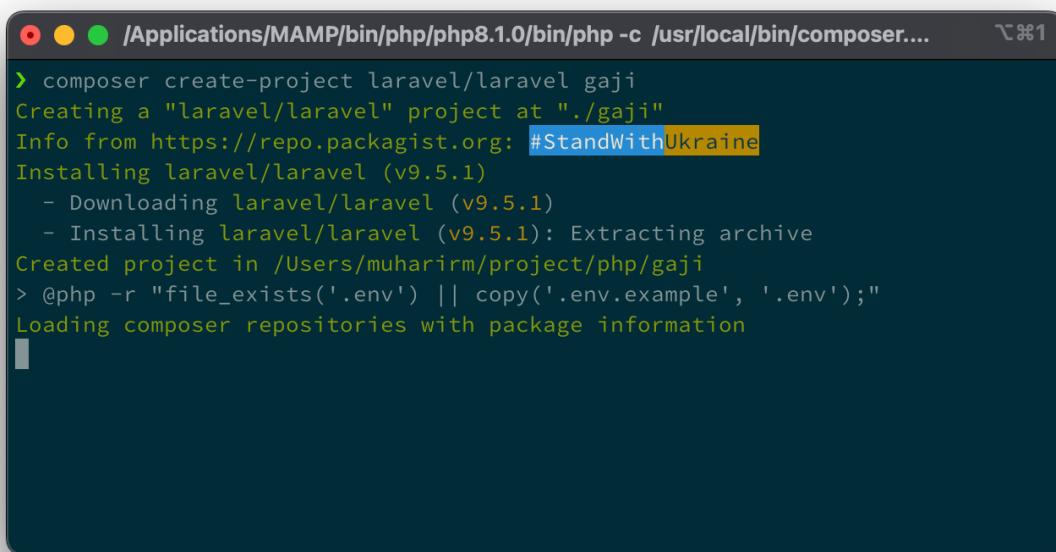
-

## Menginstall Laravel Framework

Untuk menginstall laravel framework dapat dilakukan dengan beberapa cara, hal ini dapat kalian lihat pada dokumentasi laravel di website laravel <https://laravel.com/docs/9.x>. Pada praktik kali ini kita akan menggunakan composer untuk menginstall laravel.

Ketikkan perintah berikut pada CMD / Terminal CMDER laragon.

```
composer create-project laravel/laravel gaji
```



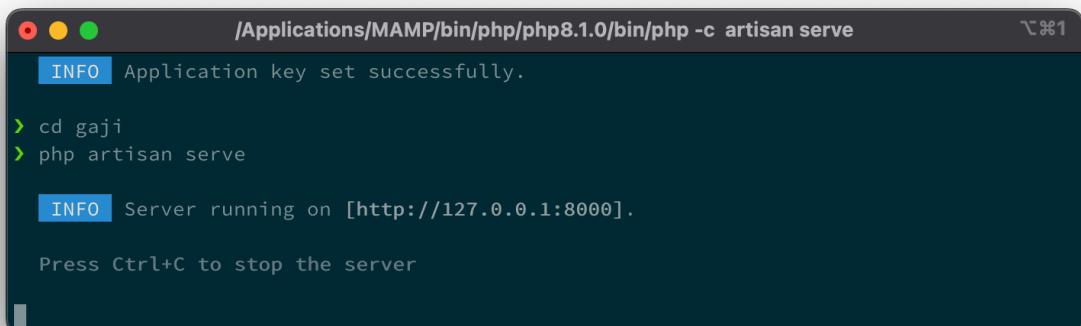
```
composer create-project laravel/laravel gaji
Creating a "laravel/laravel" project at "./gaji"
Info from https://repo.packagist.org: #StandWithUkraine
Installing laravel/laravel (v9.5.1)
- Downloading laravel/laravel (v9.5.1)
- Installing laravel/laravel (v9.5.1): Extracting archive
Created project in /Users/muharirm/project/php/gaji
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
```

Jika sudah selesai kita bisa masuk ke project laravel yang sudah di buat dengan cara melakukan perintah

```
cd gaji
```

Untuk menjalankan project bisa dengan melakukan perintah

```
php artisan serve
```

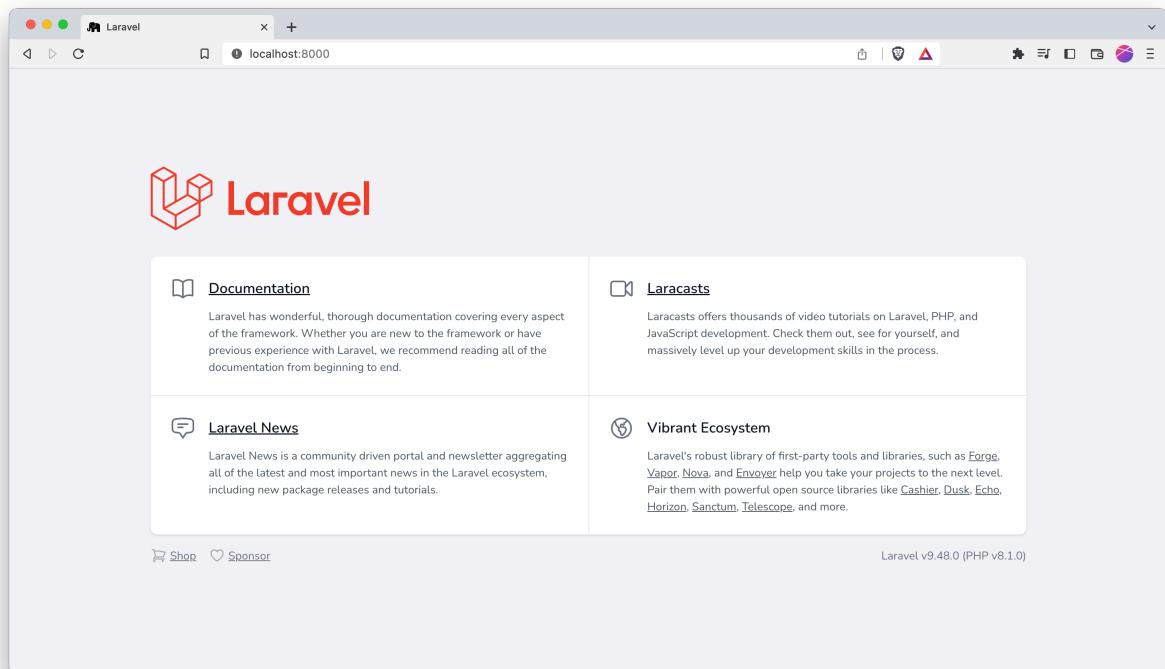


```
INFO Application key set successfully.

> cd gaji
> php artisan serve

INFO Server running on [http://127.0.0.1:8000].
Press Ctrl+C to stop the server
```

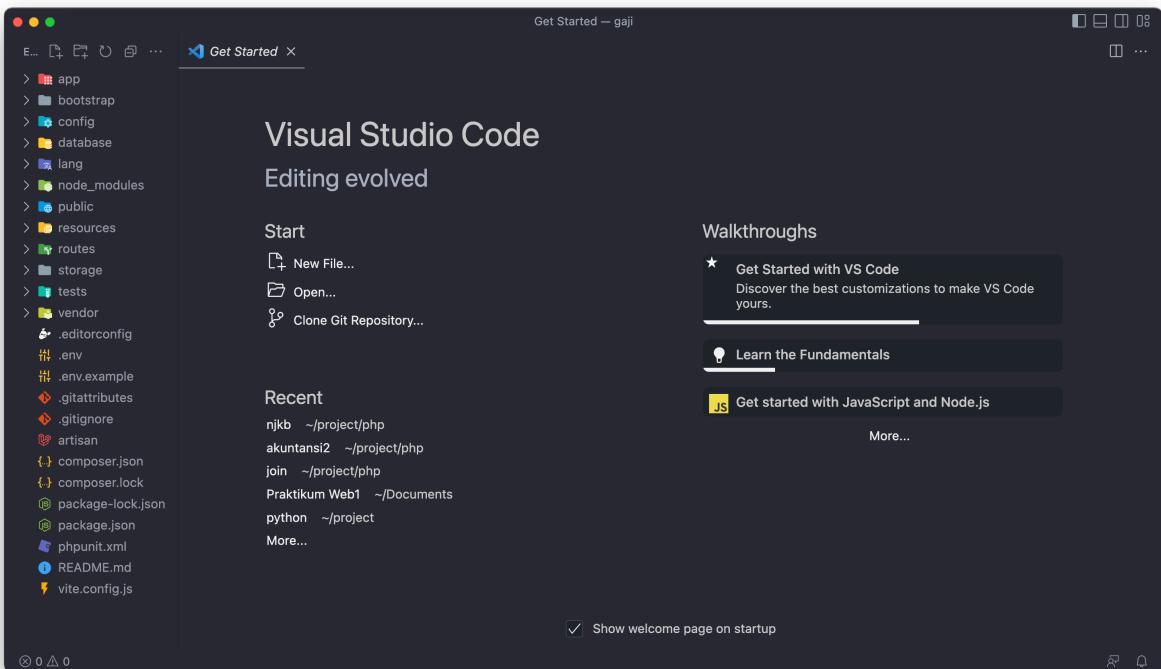
Kalian dapat melihat project yang sudah di buat dengan membuka alamat <http://localhost:8000> pada browser kalian.



Selamat kalian sudah berhasil menginstall project laravel. Saat ini project sudah siap di gunakan untuk membuat project yang kalian inginkan.

## Melakukan Konfigurasi Database

Untuk melakukan konfigurasi database. Buka project dengan text editor Vscode. Pilih **File - Open Folder** dan arahkan ke direktori project yang di buat yaitu folder **gaji**.

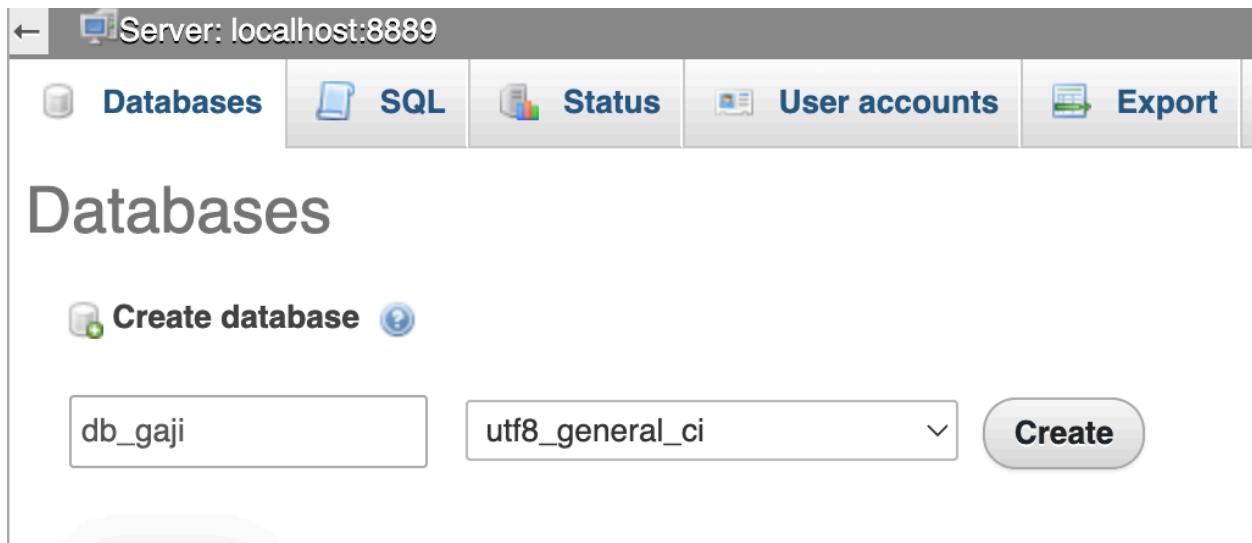


Buka file **.env** dan lakukan setting seperti berikut :

```
10
11 DB_CONNECTION=mysql
12 DB_HOST=127.0.0.1
13 DB_PORT=3306
14 DB_DATABASE=db_gaji
15 DB_USERNAME=root
16 DB_PASSWORD=
17
```

Pada bagian DB\_DATABASE ubah menjadi **db\_gaji**.

Berikutnya mari kita buat database baru. Kalian bisa menggunakan PhpMyAdmin, Navicat, HeidiSql ataupun tools manajemen database yang lain untuk membuatnya. Pada praktikum kali ini saya menggunakan PhpMyAdmin sebagai tools untuk manajemen databasenya.



Karena pada praktikum kali ini kita tidak menggunakan **migrations** melainkan menggunakan database yang sudah exist pada praktikum database sebelumnya yaitu membuat aplikasi gaji. Maka bagi kalian yang belum mempunyai table-table yang di perlukan kalian perlu melakukan import datanya terlebih dahulu.

[https://drive.google.com/file/d/1WmQf2ZjENVFmpdpU9OXIzA\\_5VCj6RnDo/view?usp=sharing](https://drive.google.com/file/d/1WmQf2ZjENVFmpdpU9OXIzA_5VCj6RnDo/view?usp=sharing)

Diatas adalah link untuk download database gaji yang sudah exist. Lakukan import database tersebut ke **db\_gaji** yang sudah kita buat.

## Importing into the database "db\_gaji"

### File to import:

File may be compressed (gzip, bzip2, zip) or uncompressed.

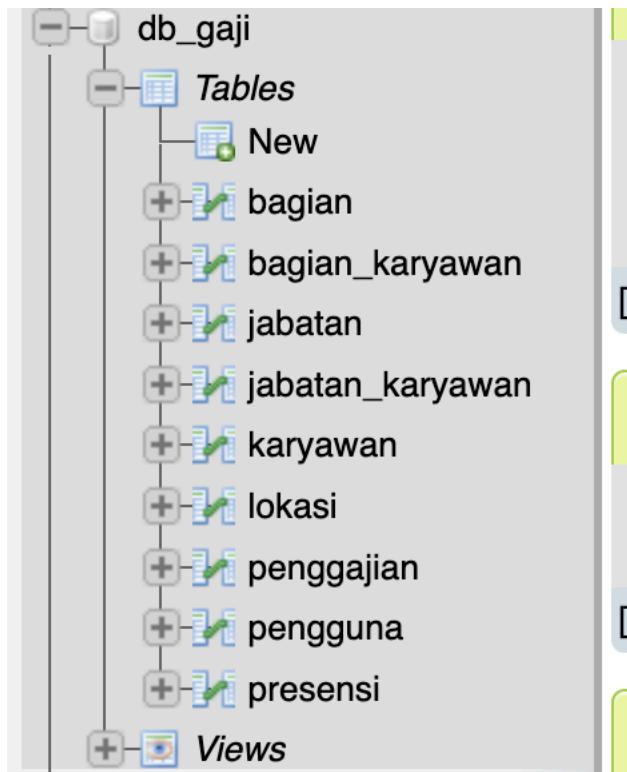
A compressed file's name must end in **.[format].[compression]**. Example: **.sql.zip**

Browse your computer:  backup\_gaji.sql (Max: 8,192KiB)

You may also drag and drop a file on any page.

Character set of the file:

Jika telah berhasil maka table akan muncul pada database yang kita buat seperti pada gambar dibawah ini



## Menginstall Admin LTE Template.

Aplikasi yang akan kita buat menggunakan Admin Lte Template agar kita lebih fokus pada bussines logic aplikasi. Tidak berkutat pada templating dan memecah template menjadi beberapa bagian untuk aplikasi yang akan kita buat. Agar hal ini dapat diwujudkan kita dapat menggunakan package <https://github.com/jeroennoten/Laravel-AdminLTE/wiki>. Dengan menggunakan package ini kita akan lebih mudah dalam manajemen template tanpa harus mendownload dan membagi bagian-bagian template Admin LTE kedalam aplikasi kita.

Untuk menginstallnya jalankan perintah berikut :

```
composer require jeroennoten/laravel-adminlte
```

```
muharirm@Muharirs-MacBook-Pro:~/project/php/gaji
> composer require jeroennoten/laravel-adminlte
Info from https://repo.packagist.org: #StandWithUkraine
Using version ^3.8 for jeroennoten/laravel-adminlte
./composer.json has been updated
Running composer update jeroennoten/laravel-adminlte
Loading composer repositories with package information
Updating dependencies
Lock file operations: 2 installs, 0 updates, 0 removals
- Locking almasaeed2010/adminlte (v3.2.0)
- Locking jeroennoten/laravel-adminlte (v3.8.5)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 2 installs, 0 updates, 0 removals
- Installing almasaeed2010/adminlte (v3.2.0): Extracting archive
- Installing jeroennoten/laravel-adminlte (v3.8.5): Extracting archive
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi

[INFO] Discovering packages.

jeroennoten/laravel-adminlte
laravel/sail
laravel/sanctum
laravel/tinker
nesbot/carbon
nunomaduro/collision
nunomaduro/termwind
spatie/laravel-ignition

80 packages you are using are looking for funding.
Use the 'composer fund' command to find out more!
> @php artisan vendor:publish --tag=laravel-assets --ansi --force

[INFO] No publishable resources for tag [laravel-assets].
```

Jika proses penginstalan telah selesai kita dapat melanjutkan ke tahapan yang selanjutnya yaitu menjalankan perintah

```
php artisan adminlte:install
```

```
> php artisan adminlte:install
Basic assets installed successfully.
Configuration file installed successfully.
Translation files installed successfully.
The installation is complete.
```

Dan berikutkan jalankan 2 perintah berikut

```
composer require laravel/ui
php artisan ui bootstrap -auth
```

Jika perintah berhasil di eksekusi akan muncul tampilan seperti dibawah ini pada command prompt ataupun terminalnya.

```
[INFO] Authentication scaffolding generated successfully.

[INFO] Bootstrap scaffolding installed successfully.

[WARN] Please run [npm install && npm run dev] to compile your fresh scaffolding.
```

Berikutnya jalankan perintah **npm install** dan **npm run dev**.

```
found 0 vulnerabilities
> npm run dev

> dev
> vite

VITE v4.0.4  ready in 486 ms

→ Local:  http://localhost:5173/
→ Network: use --host to expose
→ press h to show help

LARAVEL v9.48.0  plugin v0.7.3

→ APP_URL: http://localhost
```

Untuk keluar dari perintah **npm run dev** kita dapat menekan **CTRL + C** pada keyboard. Lalu jalankan kembali perintah dibawah ini agar asset baik css dan js di compile menggunakan vite engine.

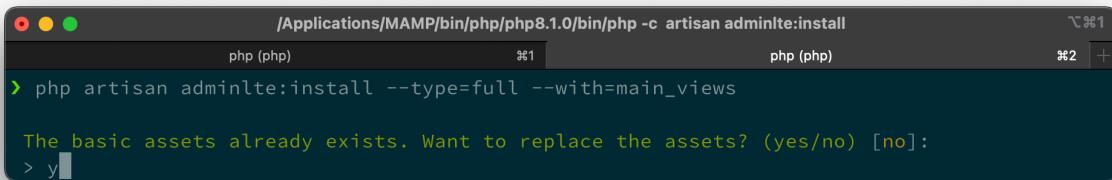
### **npm run build**

Jika kita tidak menjalankan perintah tersebut, konsekuensinya kita harus menjalankan perintah **npm run dev** secara terus menerus dan tidak boleh di stop. Karena vite beranggapan kita masih dalam mode development. Sedangkan jika kita menggunakan perintah **npm run build**. Kita dianggap berada pada mode production sehingga asset yang kita miliki baik js dan css di compile (mix) oleh vite engine.

Masuk ketahapan terakhir jalankan perintah berikut :

### **php artisan adminlte:install --type=full --with=main\_views**

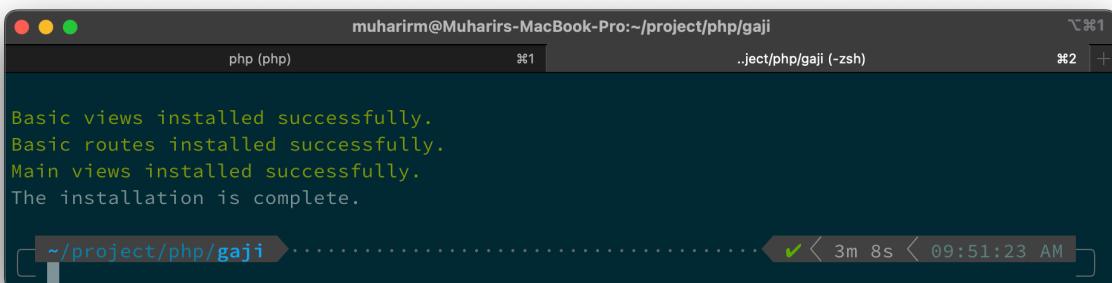
Jika muncul notifikasi pada terminal seperti ini, ketik y dan enter. Begitu seterusnya jika muncul kembali secara berulang.



```
/Applications/MAMP/bin/php/php8.1.0/bin/php -c artisan adminlte:install
php (php)          #1           php (php)          #2 +
```

```
> php artisan adminlte:install --type=full --with=main_views
The basic assets already exists. Want to replace the assets? (yes/no) [no]: > y
```

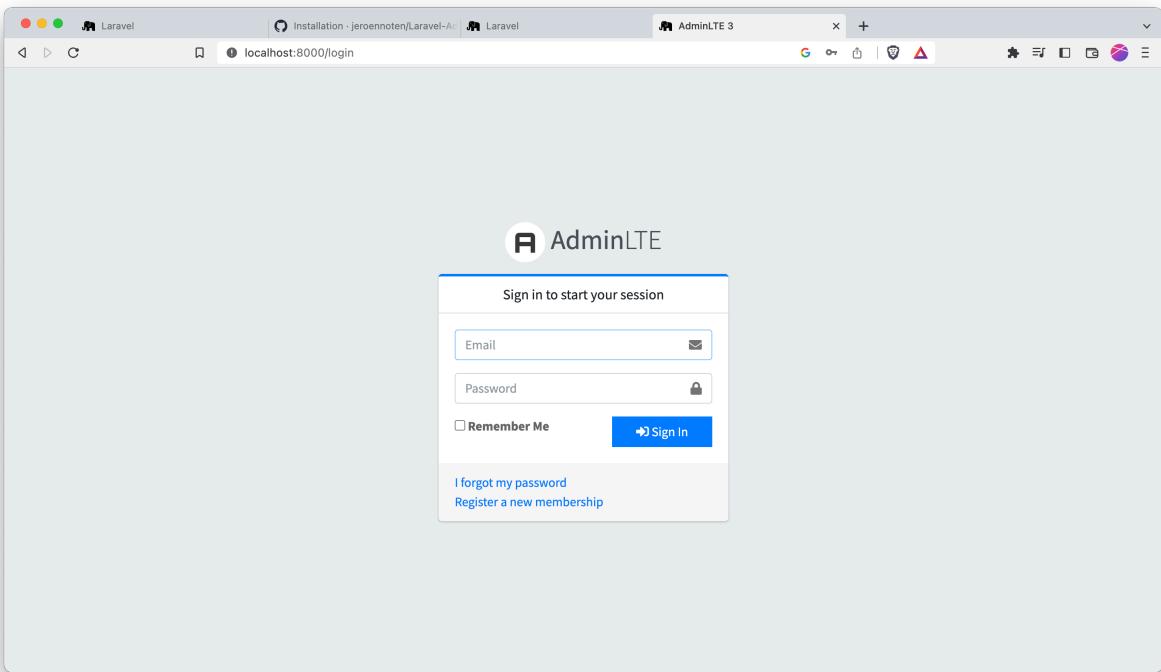
Hingga muncul notifikasi bahwa penginstalannya sukses



```
muharirm@Muharirs-MacBook-Pro:~/project/php/gaji
php (php)          #1           ..ect/php/gaji (-zsh)          #2 +
```

```
Basic views installed successfully.
Basic routes installed successfully.
Main views installed successfully.
The installation is complete.
```

Sekarang kalian bisa menjalankan kembali servernya dengan menggunakan perintah **php artisan serve** pada terminal. Buka browser <http://localhost:8000> dan kalian akan melihat pada sisi kanan atas dari aplikasi terdapat link login dan register. Kalian dapat mencoba klik dan lihatlah bahwa template Admin LTE telah terpasang dengan cantik.



Wah mudah bukan, daripada kita harus menginstal manual dan melakukan cut section pada bagian-bagian templatanya. Cara yang kita gunakan ini jauh lebih efektif dan jauh lebih cepat.

Untuk dapat mencoba login dan registernya kita harus menjalankan migration bawaan laravel untuk membuat table users. Lakukan perintah berikut untuk menjalankan migration bawaan laravel pada terminal atau cmd.

```
php artisan migrate
```

```
> php artisan migrate

INFO Preparing database.

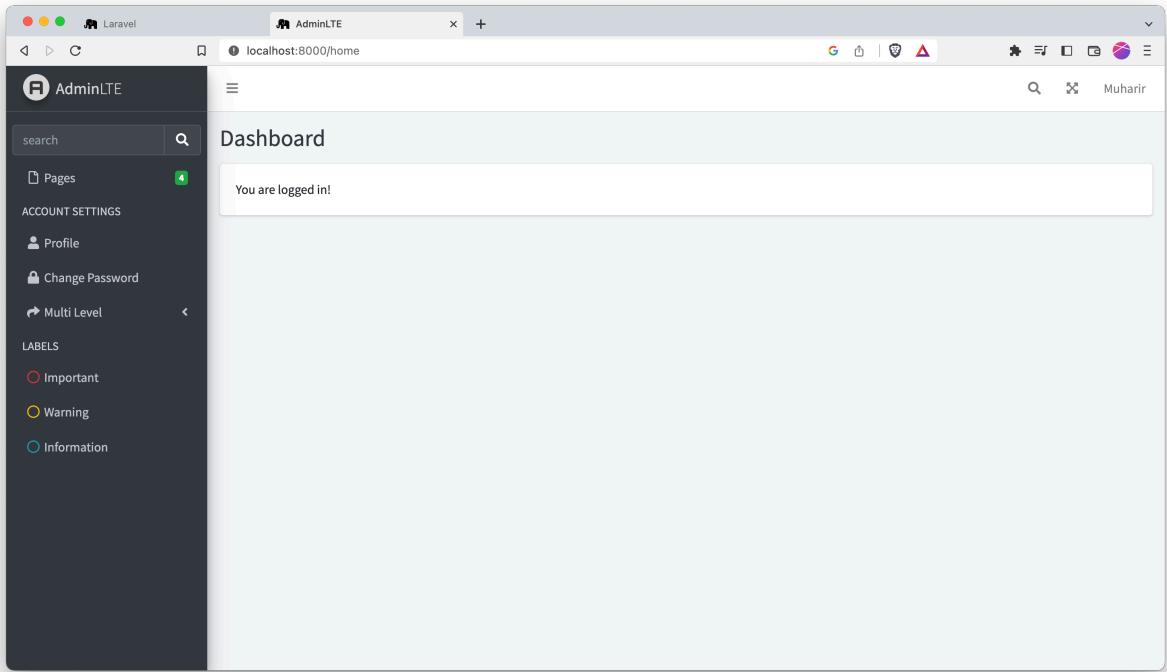
Creating migration table                                         DONE

INFO Running migrations.

2014_10_12_000000_create_users_table                           DONE
2014_10_12_100000_create_password_resets_table               DONE
2019_08_19_000000_create_failed_jobs_table                   DONE
2019_12_14_000001_create_personal_access_tokens_table       DONE
```

Nah sekarang kita bisa melakukan register dan login pada aplikasi. Selamat Mencoba.

Jika berhasil melakukan register dan login, maka akan muncul halaman seperti dibawah ini.

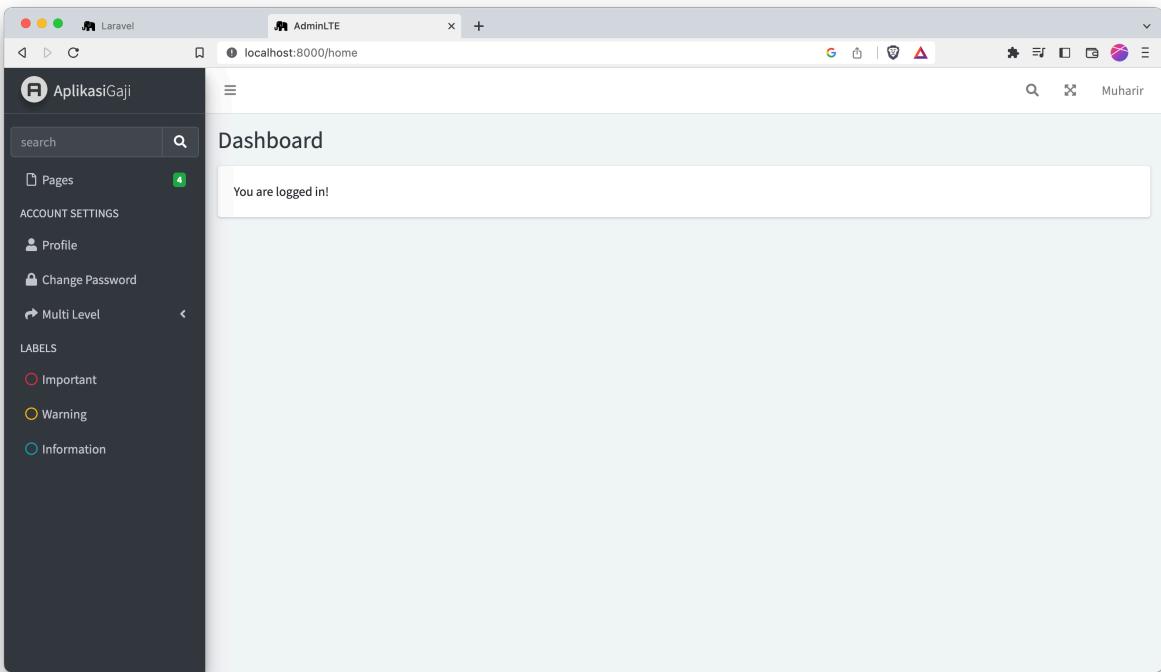


Wah keren kan ? 😎

## Mengkonfigurasi Template Admin LTE

Jika kalian melihat tampilan sebelumnya masih menggunakan brand AdminLTE kita akan mengubahnya dan mengkonfigurasinya. Pada project pilih folder **config – adminlte.php**

Konfigurasi Properties	Value
title	Aplikasi Gaji
logo	<b>Aplikasi</b>Gaji,
preloader enabled	False



Sekarang Brand Aplikasi telah berubah. Anda dapat menyesuaikan konfigurasi-konfigurasi yang lain pada file adminlte.php ini. Silahkan kalian coba-coba untuk menghilangkan search bar dan menu yang lainnya jika tidak diperlukan.

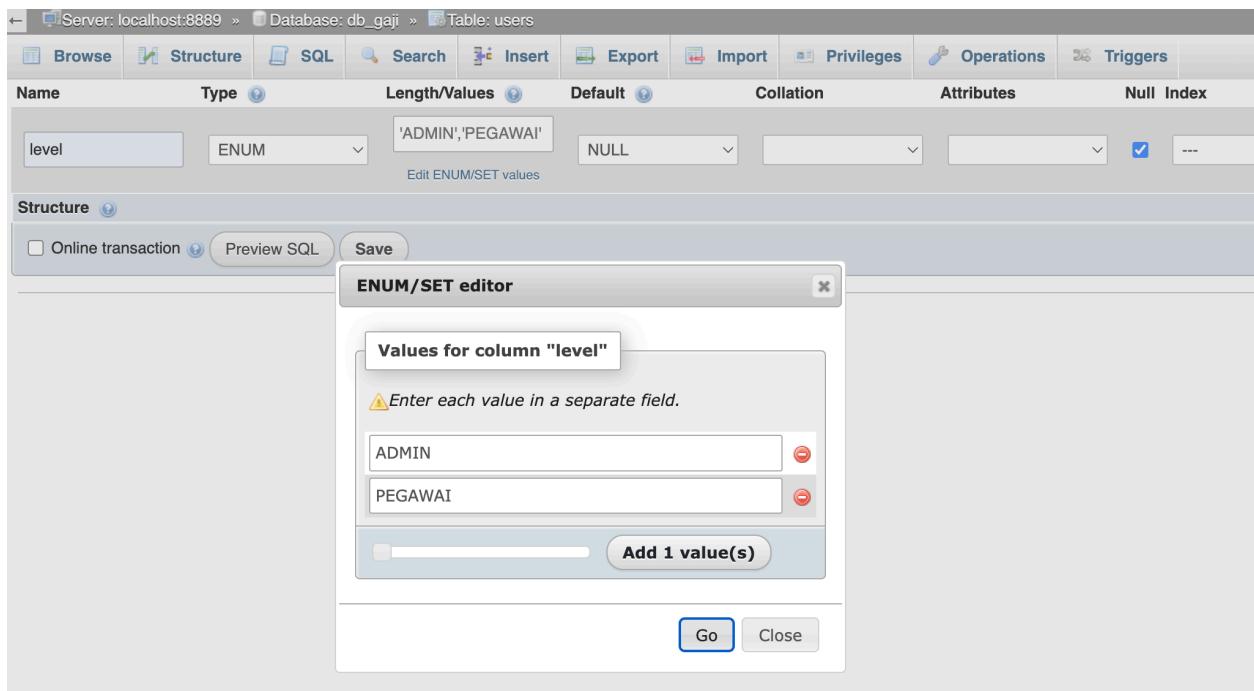
### Menampilkan Data Pengguna (CRUD User)

Untuk membuat CRUD pada table users kita perlu membuat model User terlebih dahulu. Namun karena laravel telah membuatkannya untuk kita (default) maka kita tidak perlu membuatnya kembali. Silahkan buka folder **app - Models - User.php**. Kita tambahkan level properti pada bagian fillable arraynya.

```
20  <?php  
21      protected $fillable = [  
22          'name',  
23          'email',  
24          'password',  
25          'level'  
26      ];
```

Hal ini dimaksudkan untuk kita menambahkan field baru bernama level dengan tipe data Enum pada table user. Karena kita tidak menggunakan migrations maka

kita perlu menambahkannya secara manual pada table users. Silahkan buka PhpMyAdmin dan tambahkan kolom level pada table users. Dan ijin NULL pada kolom tersebut.



The screenshot shows the 'Structure' tab for the 'users' table in PhpMyAdmin. A modal window titled 'ENUM/SET editor' is open, showing the values 'ADMIN' and 'PEGAWAI' for the 'level' column. The 'Add 1 value(s)' button is visible at the bottom right of the modal.

Kembali pada CMD / Terminal kita akan membuat Controller dengan nama UserController.php dengan cara memasukkan perintah :

```
php artisan make:controller UserController --resource
```

```
> php artisan make:controller UserController --resource
INFO Controller [app/Http/Controllers/UserController.php] created successfully.
```

Setelah berhasil di buat kita dapat melihatnya pada folder **app/Http/Controllers/UserController.php**

Silahkan buka file tersebut dan edit seperti dibawah ini untuk method index dan tambahkan pemanggilan model User pada bagian atas.

```
<?php

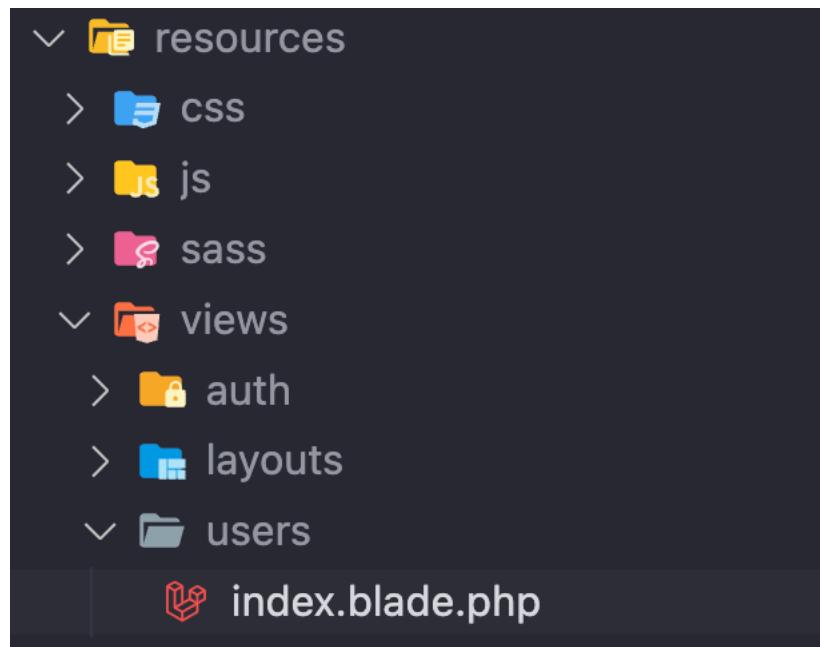
namespace App\Http\Controllers;

use App\Models\User;
use Illuminate\Http\Request;

class UserController extends Controller
```

```
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $users = User::all();
        return view('users.index', compact('users'));
    }
}
```

Jika sudah selesai berikutnya kita akan membuat view untuk di tampilkan pada user. Dengan cara pada folder **resources - views** buat folder baru dengan nama **users**. Lalu buat file baru dengan nama **index.blade.php**



Pada file **index.blade.php** buat menjadi seperti berikut. Untuk memudahkan kalian bisa copy paste isi pada [home.blade.php](#) dan pastekan ke dalam **index.blade.php** lalu ubah menjadi seperti berikut :

```
@extends('adminlte::page')

@section('title', 'Data Pengguna')

@section('content_header')
    <h1 class="m-0 text-dark">Data Pengguna</h1>
@stop

@section('content')
```

```

<div class="row">
    <div class="col-12">
        <div class="card">

            <div class="card-header">
                <h2 class="card-title"><strong>Table Data Pengguna</strong></h2>
                <a href="{{ route('pengguna.create') }}" class="btn btn-primary btn-md float-right"> Tambah Pengguna</a>
            </div>
            <div class="card-body">

                <div class="table-responsive">
                    <table class="table table-bordered">
                        <tr>
                            <th>NO.</th>
                            <th>NAMA LENGKAP</th>
                            <th>EMAIL</th>
                            <th>LEVEL</th>
                            <th class="text-center">AKSI</th>
                        </tr>
                        @php
                            $no = 1;
                        @endphp
                        @foreach ($users as $user)
                        <tr>
                            <td>{{ $no++ }}</td>
                            <td>{{ $user->name }}</td>
                            <td>{{ $user->email }}</td>
                            <td>{{ $user->level }}</td>
                            <td class="text-center">
                                <a href="#" class="btn btn-md btn-warning">EDIT</a>
                                <a href="#" class="btn btn-md btn-danger">HAPUS</a>
                            </td>
                        </tr>
                        @endforeach
                    </table>
                </div>
            </div>
        </div>
    </div>
</div>
@stop

```

Selanjutnya tambahkan routing pengguna pada folder **routes/web.php** . Seperti berikut :

```

Route::group(['prefix' => 'admin', 'middleware' => ['auth']], function ()
{
    Route::resource('pengguna', UserController::class);
}

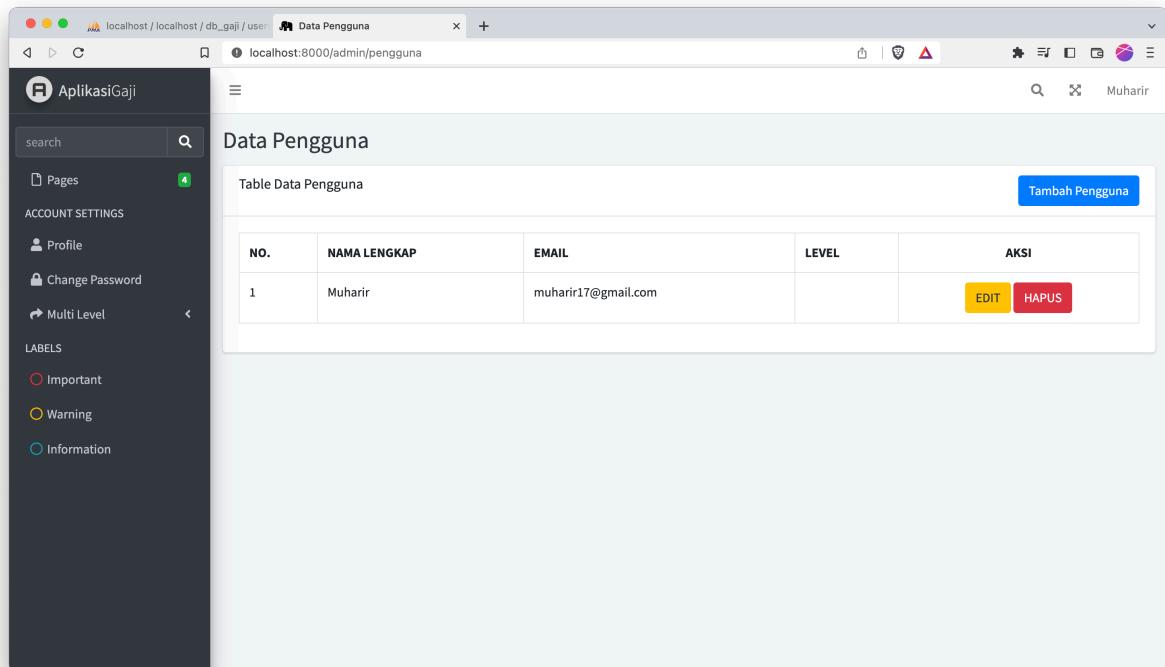
```

```
});
```

Jangan lupa untuk menambahkan use UserController pada bagian atas file **web.php**. Seperti pada gambar dibawah ini.

```
routes > php web.php > ...
1   <?php
2
3   use App\Http\Controllers\UserController;
4   use Illuminate\Support\Facades\Route;
5
6   /*
7   |-----|
8   | Web Routes
9   |-----|
```

Pada bagian routing kita menambahkan routing group dengan prefix admin sehingga untuk mengakses halaman pengguna kita dapat menggunakan alamat : <http://localhost:8000/admin/pengguna> Kalian dapat mencobanya pada browser masing-masing untuk melihat hasilnya.



Kita sudah berhasil menampilkan data pengguna dengan menggunakan laravel. Nah berikutnya karena kita akan membuat tambah data, edit, dan hapus. Kita perlu

menambahkan beberapa package untuk mempercantik notifikasi jika terjadi sukses atau error nya setiap operasi yang dilakukan.

## Menginstall Sweetalert, Datatable dan Select2

Untuk menginstall Sweetalert anda dapat menggunakan perintah perintah berikut :

```
composer require realrashid/sweetalert
```

Selanjutnya tambahkan `@include('sweetalert::alert')` pada layout **master.blade.php** letak file ini ada pada

```
resources > views > vendor > adminlte > master.blade.php > ...
```

Tambahkan di bagian bawah seperti berikut :

```
106
107      @include('sweetalert::alert')
108      {{-- Custom Scripts --}}
109      @yield('adminlte_js')
110
111    </body>
112
113  </html>
114
```

Selanjutnya jalankan perintah berikut pada command prompt atau terminal :

```
php artisan sweetalert:publish
```

Kita sudah berhasil menginstall Sweetalert, agar kita tidak terulang-ulang menginstall package-package yang lain, maka kita akan menginstall semua plugin yang akan kita butuhkan selama development aplikasi penggajian. Berikutnya kita lanjutkan untuk menginstall **Datatable** dan **Select2**. Jalankan perintah berikut pada command prompt :

```
composer require yajra/laravel-datatables-oracle:^10.0
```

Berikutnya jalankan perintah berikut :

```
php artisan vendor:publish --tag=datatables
```

Untuk asset kita dapat menggunakan asset bawaan dari template dengan menggunakan perintah :

```
php artisan adminlte:plugins install --plugin=datatables --
plugin=select2

> php artisan adminlte:plugins install --plugin=datatables --plugin=select2
2/2 [██████████] 100%
The plugins installation is complete. Summary:

+-----+-----+
| Plugin Key | Status   |
+-----+-----+
| datatables | Installed |
| select2     | Installed |
+-----+-----+
```

Oke sudah lengkap plugin-plugin yang kita butuhkan dalam pembuatan aplikasi penggajian. Baik kita lanjutkan ke proses tambah data, edit, dan hapus pada aplikasi penggajian.

## Membuat Tambah Data Pengguna

Buat file baru pada folder **resources - views - users**. Dengan nama **tambah.blade.php** dan isi kode seperti berikut :

### tambah.blade.php

```
@extends('adminlte::page')

@section('title', 'Data Pengguna')

@section('content_header')
    <h1 class="m-0 text-dark">Data Pengguna</h1>
@stop

@section('content')
    <div class="row">
        <div class="col-12">
            <div class="card">

                <div class="card-header">
                    <h3 class="card-title"><strong>Tambah Data Pengguna</strong></h3>
                </div>
                <div class="card-body">
                    <form action="{{ route('pengguna.store') }}" method="post">
                        @csrf
                        <div class="form-group row">
                            <label class="form-label col-sm-2">Nama Lengkap</label>
                            <div class="col-sm-4">
                                <div class="input-group">
                                    <input type="text" name="name" placeholder="Masukkan Nama Lengkap" class="form-control" value="{{ old('name') }}" required>
                                </div>
                            </div>

                            <label class="form-label col-sm-2">Email</label>
                            <div class="col-sm-4">
                                <input type="email" name="email" placeholder="Masukkan Email" class="form-control" value="{{ old('email') }}" required>
                            </div>
                        </div>

                        <div class="form-group row">
                            <label class="form-label col-sm-2">Password</label>
                            <div class="col-sm-4">
                                <div class="input-group">
```

```

        <input type="password" name="password" placeholder="Masukkan
Password" class="form-control" value="{{ old('password') }}" required>
            </div>
        </div>

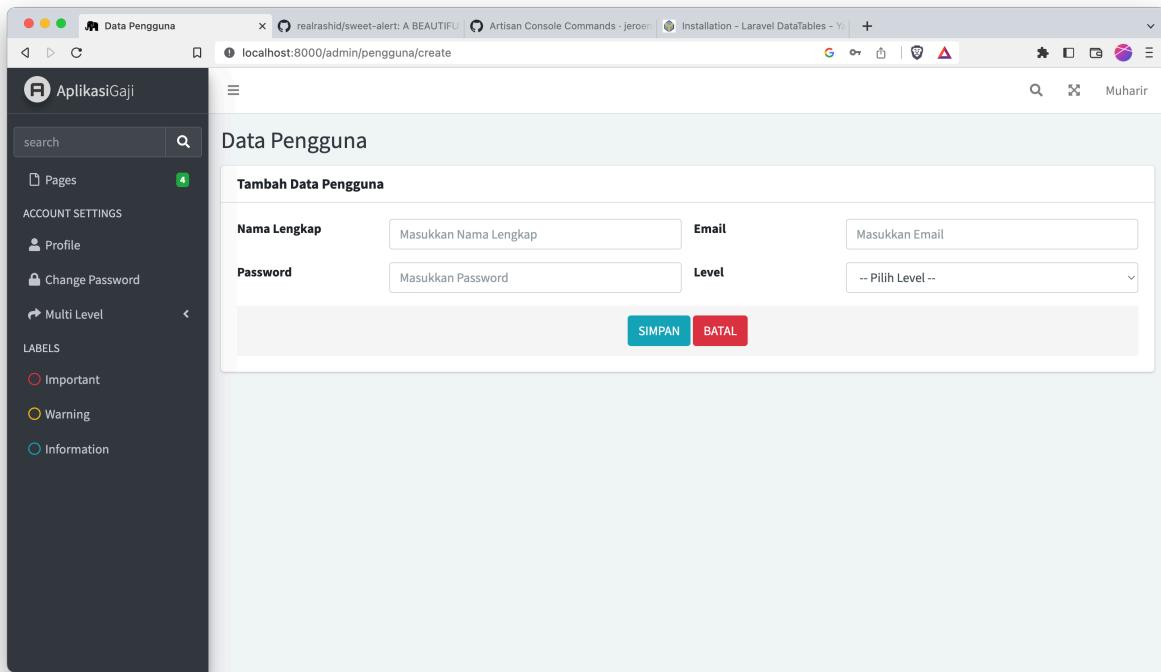
        <label class="form-label col-sm-2">Level</label>
        <div class="col-sm-4">
            <select name="level" class="form-control" required>
                <option value="" selected disabled>-- Pilih Level --
</option>
                <option value="ADMIN">ADMIN</option>
                <option value="PEGAWAI">PEGAWAI</option>
            </select>
        </div>
    </div>

    <div class="card-footer text-center">
        <button type="submit" class="btn btn-info"
id="simpan">SIMPAN</button>
        <a href="{{ route('pengguna.index') }}" class="btn btn-
danger">BATAL</a>
    </div>

    </form>
</div>
</div>
</div>
</div>
@stop

```

Jika dijalankan akan menghasilkan tampilan seperti berikut ini :



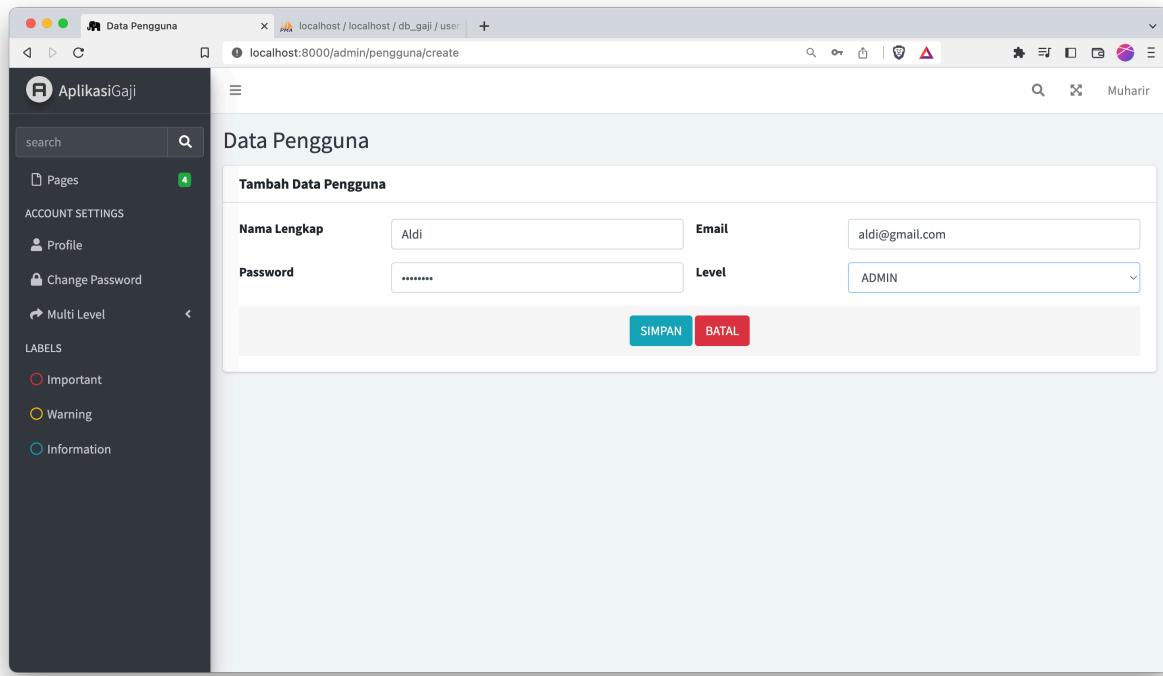
Untuk selanjutnya mari kita handle semua inputnya pada controller. Buka UserController.php pada method store tambahkan kode seperti berikut :

```
public function store(Request $request)
{
    // memvalidasi inputan
    $this->validate($request, [
        'name'      => 'required',
        'email'     => 'required|email',
        'password'  => 'required',
        'level'     => 'required'
    ]);

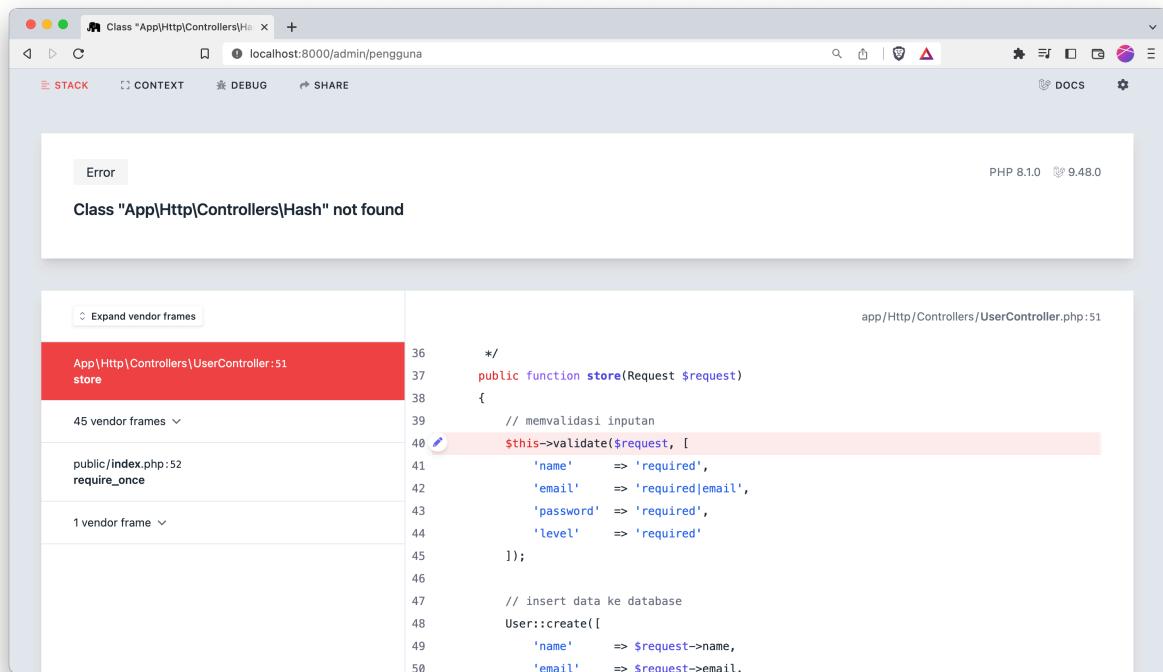
    // insert data ke database
    User::create([
        'name'      => $request->name,
        'email'     => $request->email,
        'password'  => Hash::make($request->password),
        'level'     => $request->level,
    ]);

    Alert::success('Sukses', 'Berhasil Menambahkan User Baru');
    return redirect()->route('pengguna.index');
}
```

Bisa kalian coba untuk menambahkan data pengguna baru.



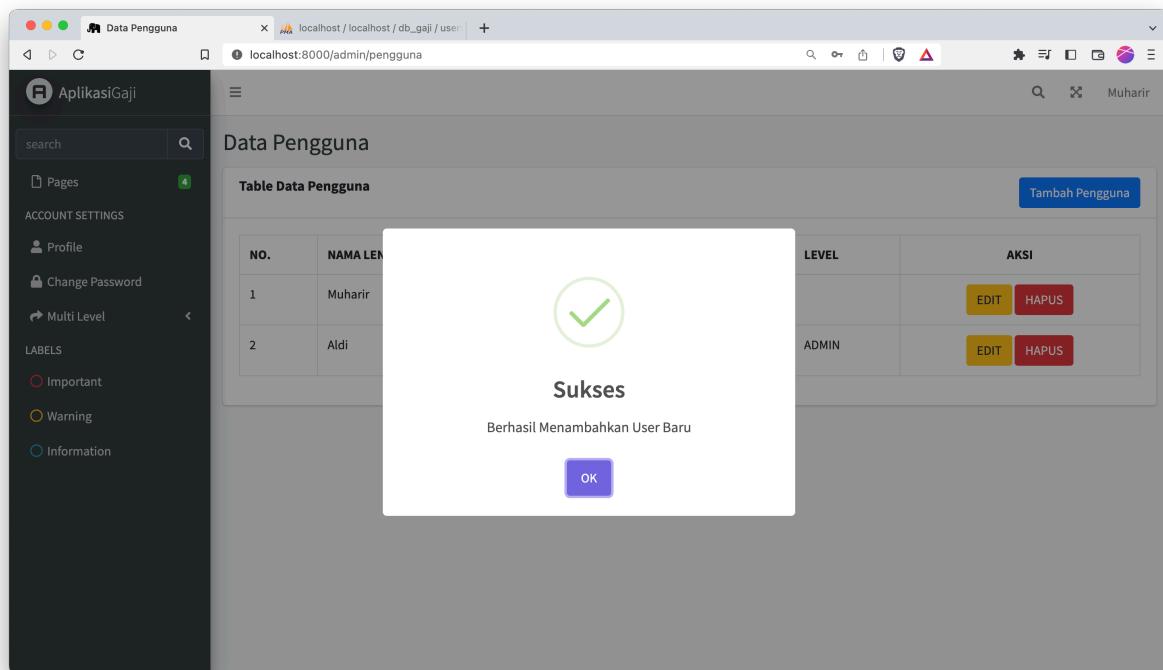
Jika muncul error seperti berikut ini



Ini diakibatkan kita belum mengimport **Hash** class untuk enkripsi password yang kita masukkan. Tambahkan **use Hash;** dan **use Alert;** Untuk membuat notifikasi SweetAlert Pada bagian atas file **UserController.php** seperti berikut ini :

```
app > Http > Controllers > UserController.php > ...
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Models\User;
6  use Illuminate\Http\Request;
7  use Hash;
8  use Alert;
9
10 class UserController extends Controller
11 {
```

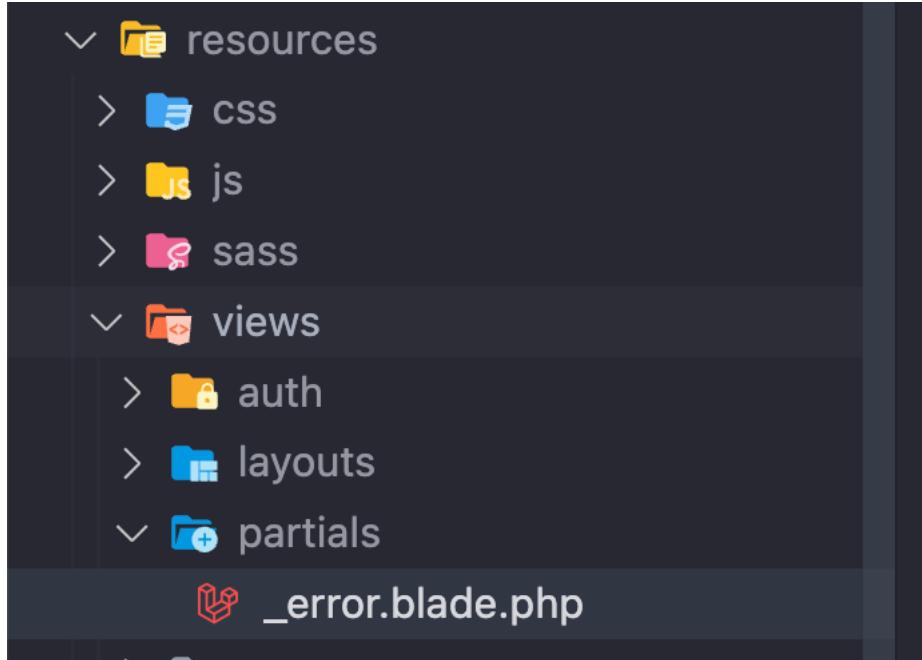
Lalu coba untuk mengulangi simpan data pengguna. Jika berhasil akan muncul notifikasi SweetAlert seperti berikut :



Kita sudah berhasil menambahkan data baru pada aplikasi.

Namun saat kita mengisikan data yang salah misalnya email, aplikasi tidak memunculkan kesalahan, oleh karena itu kita akan membuat alert terpisah pada aplikasi.

Silahkan buat folder baru dengan nama **partials** pada folder **resources - views**. Dan tambahkan file baru dengan nama **\_error.blade.php**.



Dan tambahkan kode berikut di dalamnya

```
@if ($errors->any())
<div class="alert bg-danger alert-dismissible">
    <button type="button" class="close" data-dismiss="alert" aria-
hidden="true">x</button>
    <h5><i class="icon fas fa-times"></i>Upps Error !!!</h5>
    @foreach ($errors->all() as $error)
        <li>
            {{ $error }}
        </li>
    @endforeach
</div>
@endif
```

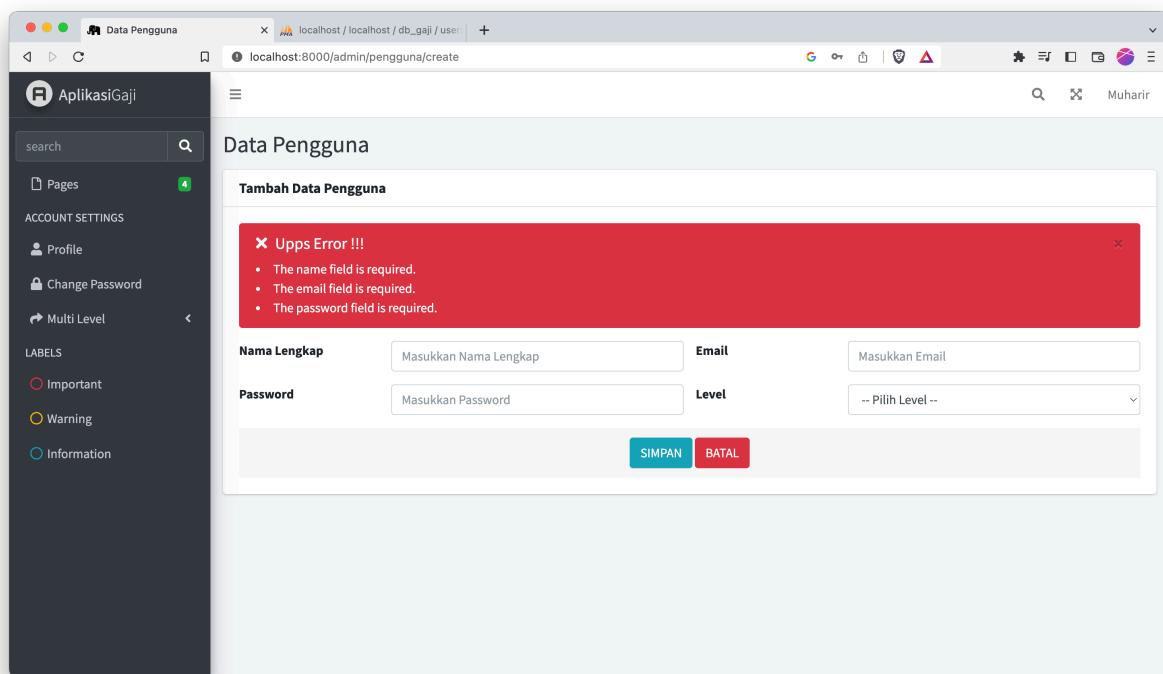
Lalu modifikasi **tambah.blade.php**, tambahkan **@include('partials.error')** di dalamnya seperti berikut :

```

14 <div class="card-header">
15   <h3 class="card-title"><strong>Tambah Data Pengguna</strong></h3>
16 </div>
17 <div class="card-body">
18
19   @include('partials._error')
20
21   <form action="{{ route('pengguna.store') }}" method="post">
22     @csrf
23     <div class="form-group row">

```

Jika sudah, bisa di pastikan jika ada kesalahan input maka akan muncul seperti berikut ini :



## Membuat Edit Pengguna

Berikutnya kita akan membuat edit data pada pengguna. Langkah pertama modifikasi **index.blade.php** pada folder **users** . Ubah pada bagian tombol action (edit dan hapus) menjadi seperti berikut :

```

@php
    $no = 1;
@endphp
@foreach ($users as $user)
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| {{ $no++ }} | {{ $user->name }} | {{ $user->email }} | {{ $user->level }} | <form action="{{ route('pengguna.destroy', $user->id) }}" method="post">             @method('DELETE')             @csrf             <a href="{{ route('pengguna.edit', $user->id) }}" class="btn btn-md btn-warning">EDIT</a>             <button type="submit" class="btn btn-md btn-danger">HAPUS</button>         </form> |

@endforeach

```

Berikutnya pada **views-users** copy paste file **tambah.blade.php** dan rename menjadi **edit.blade.php** lalu ubah isinya menjadi seperti berikut :

```

@extends('adminlte::page')

@section('title', 'Data Pengguna')

@section('content_header')
    <h1 class="m-0 text-dark">Data Pengguna</h1>
@stop

@section('content')
    <div class="row">
        <div class="col-12">
            <div class="card">

                <div class="card-header">
                    <h3 class="card-title"><strong>Edit Data Pengguna</strong></h3>
                </div>
                <div class="card-body">
                    @include('partials._error')
                    <form action="{{ route('pengguna.update', $pengguna->id) }}" method="post">
                        @method('PUT')
                        @csrf
                        <div class="form-group row">
                            <label class="form-label col-sm-2">Nama Lengkap</label>
                            <div class="col-sm-4">
                                <div class="input-group">
                                    <input type="text" name="name" placeholder="Masukkan Nama Lengkap" class="form-control" value="{{ isset($pengguna) ? $pengguna->name : old('name') }}" required>
                                </div>
                            </div>
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>

```

```

        </div>

        <label class="form-label col-sm-2">Email</label>
        <div class="col-sm-4">
            <input type="email" name="email" placeholder="Masukkan Email"
class="form-control" value="{{ isset($pengguna) ? $pengguna->email : old('email') }}" required>
            </div>
        </div>

        <div class="form-group row">
            <label class="form-label col-sm-2">Password</label>
            <div class="col-sm-4">
                <div class="input-group">
                    <input type="password" name="password" placeholder="Masukkan
Password" class="form-control" value="{{ old('password') }}>
                    </div>
                </div>
            </div>

            <label class="form-label col-sm-2">Level</label>
            <div class="col-sm-4">
                <select name="level" class="form-control" required>
                    <option value="" selected disabled>-- Pilih Level --
</option>
                    <option value="ADMIN" {{ $pengguna->level == 'ADMIN' ?
'selected' : '' }}>ADMIN</option>
                    <option value="PEGAWAI" {{ $pengguna->level == 'PEGAWAI' ?
'selected' : '' }}>PEGAWAI</option>
                </select>
            </div>
        </div>

        <div class="card-footer text-center">
            <button type="submit" class="btn btn-info"
id="simpan">SIMPAN</button>
            <a href="{{ route('pengguna.index') }}" class="btn btn-
danger">BATAL</a>
        </div>
    </form>
</div>
</div>
</div>
</div>
@stop

```

Selanjutnya isi method **edit** dan **update** pada **UserController.php** seperti berikut :

```

public function edit(User $pengguna)
{
    return view('users.edit', compact('pengguna'));
}

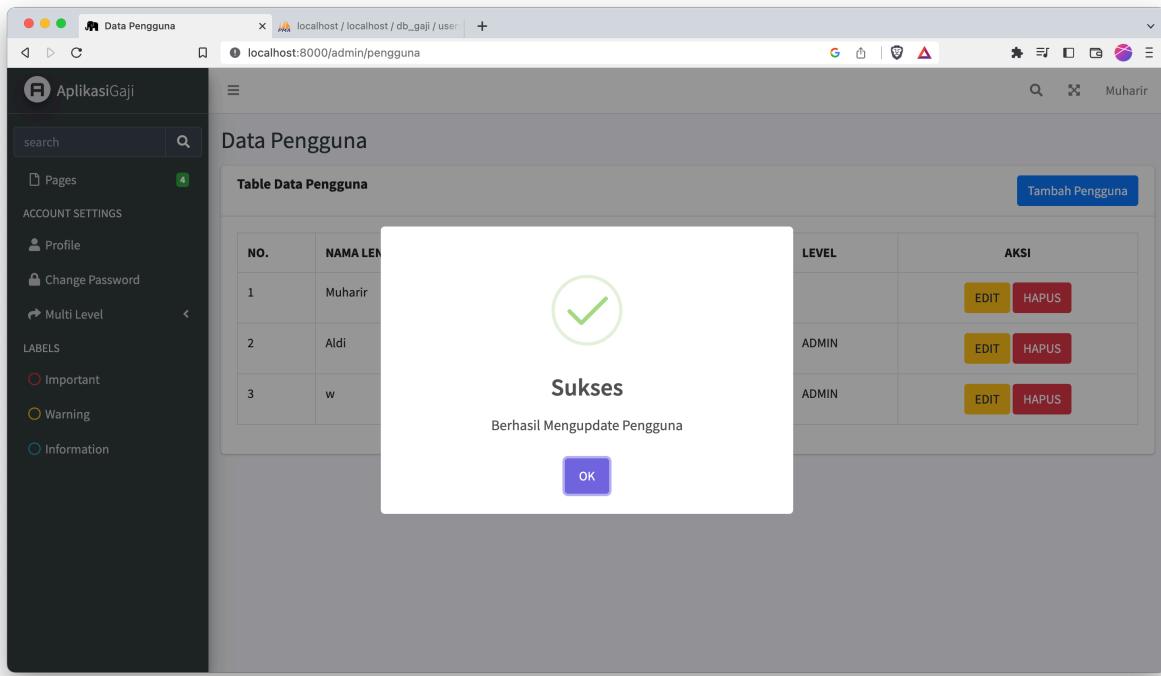
public function update(Request $request, User $pengguna)
{
    // memvalidasi inputan
    $this->validate($request, [
        'name'      => 'required',
        'email'     => 'required|email',
        'password'  => 'nullable',
        'level'     => 'required',
    ]);

    // update data ke database
    $pengguna->update([
        'name'      => $request->name,
        'email'     => $request->email,
        'password'  => is_null($request->password) ? $pengguna->password :
Hash::make($request->password),
        'level'     => $request->level,
    ]);

    Alert::success('Sukses', 'Berhasil Mengupdate Pengguna');
    return redirect()->route('pengguna.index');
}

```

Berikutnya mari kita coba untuk mengedit data pengguna.

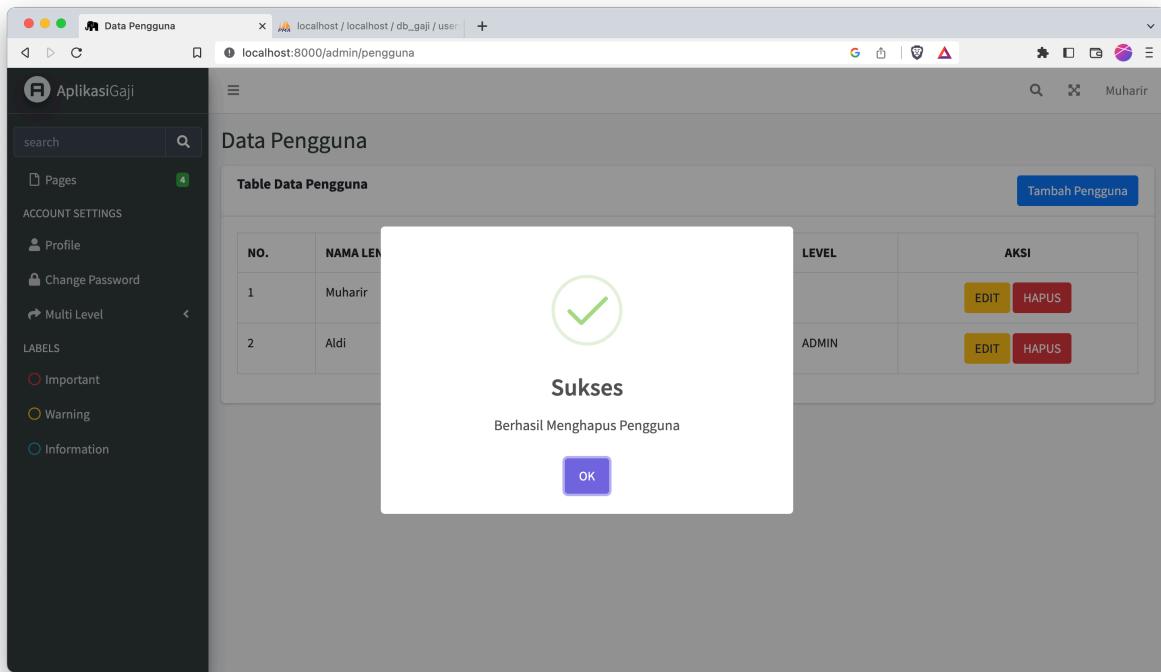


## Membuat Hapus Pengguna

Untuk menambahkan hapus data. Isi metode destroy pada **UserController.php** seperti berikut

```
public function destroy(User $pengguna)
{
    $pengguna->destroy($pengguna->id);
    Alert::success('Sukses', 'Berhasil Mengupdate Pengguna');
    return redirect()->route('pengguna.index');
}
```

Dan coba hapus salah satu data yang kalian miliki.



Berhasil menghapus data pengguna. Berikutnya kita akan menambahkan menu user pada Sidebar. Buka **file adminlte.php** pada folder **config**

Hapus array **blog** dan **pages**

```
// Sidebar items:  
[  
    'type' => 'sidebar-menu-search',  
    'text' => 'search',  
  
,  
[  
    'text' => 'blog',  
    'url' => 'admin/blog',  
    'can' => 'manage-blog',  
,  
[  
    'text' => 'pages',  
    'url' => 'admin/pages',  
    'icon' => 'far fa-fw fa-file',  
    'label' => 4,  
    'label_color' => 'success',  
,
```

Lalu tambahkan menu baru seperti berikut :

```
// Sidebar items:
[
  {
    'type' => 'sidebar-menu-search',
    'text' => 'search',
  },
  [
    ['header' => 'MASTER DATA'],
    [
      {
        'text' => 'Pengguna',
        'route' => 'pengguna.index',
        'icon' => 'fas fa-fw fa-user',
      },
    ],
  ],
]
```

Maka menu Pengguna kini telah muncul pada sidebar kiri template.

NO.	NAMA LENGKAP	EMAIL	LEVEL	AKSI
1	Muharir	muharir17@gmail.com		<button>EDIT</button> <button>HAPUS</button>
2	Aldi	aldi@gmail.com	ADMIN	<button>EDIT</button> <button>HAPUS</button>

## Membuat CRUD Jabatan

Sebelum kita menambahkan operasi CRUD pada table jabatan sebaiknya kita membuat Model dan controllernya terlebih dahulu.

Jalankan 2 perintah berikut :

```
php artisan make:model Jabatan
php artisan make:controller JabatanController -resource
```

```
> php artisan make:model Jabatan
[INFO] Model [app/Models/Jabatan.php] created successfully.

> php artisan make:controller JabatanController --resource
[INFO] Controller [app/Http/Controllers/JabatanController.php] created successfully.
```

Pada model **Jabatan.php** tambahkan kode berikut

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Jabatan extends Model
{
    use HasFactory;

    protected $table    = 'jabatan';
    protected $fillable = ['nama_jabatan', 'gapok_jabatan', 'tunjangan_jabatan',
'uang_makan'];

    public $timestamps = false;
}
```

Berikutnya buat folder baru pada **resources - views** dengan nama **masterdata** lalu pada folder **masterdata** buat folder baru dengan nama **jabatan**. Pada folder jabatan buat file **index.blade.php** , **tambah.blade.php** dan **edit.blade.php**. Untuk mempercepat kalian bisa copy paste dari **users**.

## Membuat Tampil Data Dengan Datatable

Sebelum menambahkan method pada **JabatanController.php** kita akan menyiapkan partial actionnya terlebih dahulu untuk nantinya kita embed kedalam Datatable. Buat file baru dengan nama **\_action.blade.php** pada folder **partials** dan isi kodennya seperti berikut :

```

<div class="btn-group" role="group">
    <form action="{{ route('jabatan.destroy', $form_url) }}" method="post" class="form-inline">
        @method('DELETE')
        @csrf
        <a class="btn btn-sm btn-warning" href="{{ $edit_url }}><i class="fa fa-edit"></i>
Edit</a> &nbsp;
        <button type="submit" onclick="return confirm('Anda Yakin?')" class="btn btn-sm btn-
danger"><i class="fa fa-trash"></i> Hapus</button>
    </form>
</div>

```

Buka **JabatanController.php** dan tambahkan berikut di dalamnya

```

1  <?php
2
3  namespace App\Http\Controllers;
4  use App\Models\Jabatan;
5  use DataTables;
6  use Session;
7

```

Lalu pada method index tambahkan kode seperti berikut dan buat method baru dengan nama **getJabatan()** seperti berikut :

```

public function index()
{
    return view('masterdata.jabatan.index');
}

public function getJabatan(Request $request)
{
    if ($request->ajax()) {
        $jabatan = Jabatan::all();
        return DataTables::of($jabatan)
            ->editColumn('aksi', function ($jabatan) {
                return view('partials._action', [
                    'model' => $jabatan,
                    'form_url' => route('jabatan.destroy', $jabatan->id),
                    'edit_url' => route('jabatan.edit', $jabatan->id),
                ]);
            })
            ->addIndexColumn()
            ->rawColumns(['aksi'])
            ->make(true);
    }
}

```

Setelah itu pada file **jabatan/index.php** di ubah menjadi seperti berikut :

```
@extends('adminlte::page')

@section('title', 'Data Jabatan')
@section('plugins.Datatables', true)

@section('content_header')
    <h1 class="m-0 text-dark">Data Jabatan</h1>
@stop

@section('content')
    <div class="row">
        <div class="col-12">
            <div class="card">

                <div class="card-header">
                    <h2 class="card-title"><strong>Table Data Jabatan</strong></h2>
                    <a href="{{ route('jabatan.create') }}" class="btn btn-primary btn-md float-right"> Tambah Jabatan</a>
                </div>
                <div class="card-body">

                    <div class="table-responsive">
                        <table class="table table-bordered table-striped" id="jabatan">
                            <thead>
                                <tr>
                                    <th>NO.</th>
                                    <th>NAMA JABATAN</th>
                                    <th>GAJI POKOK</th>
                                    <th>TUNJANGAN</th>
                                    <th>UANG MAKAN</th>
                                    <th class="text-center">AKSI</th>
                                </tr>
                            </thead>
                            </table>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
@stop

@push('js')
<script type="text/javascript">
```

```

$(document).ready(function() {

    var dataTable = $('#jabatan').DataTable({
        processing: true,
        serverSide: true,
        autoWidth: false,
        stateSave: true,
        // scrollX: true,
        "order": [
            [0, "desc"]
        ],
        ajax: '{{ route('get.jabatan') }}',
        columns: [
            {
                data: 'DT_RowIndex',
                name: 'DT_RowIndex',
                orderable: false,
                searchable: false
            },
            {
                data: 'nama_jabatan',
                name: 'nama_jabatan'
            },
            {
                data: 'gapok_jabatan',
                name: 'gapok_jabatan'
            },
            {
                data: 'tunjangan_jabatan',
                name: 'tunjangan_jabatan'
            },
            {
                data: 'uang_makan',
                name: 'uang_makan'
            },
            {
                data: 'aksi',
                name: 'aksi',
                orderable: false,
                searchable: false,
                'sClass': 'text-center'
            }
        ]
    });
});

```

</script>

@endpush

Berikutnya kita akan menambahkan routing jabatan pada **routes - web.php**

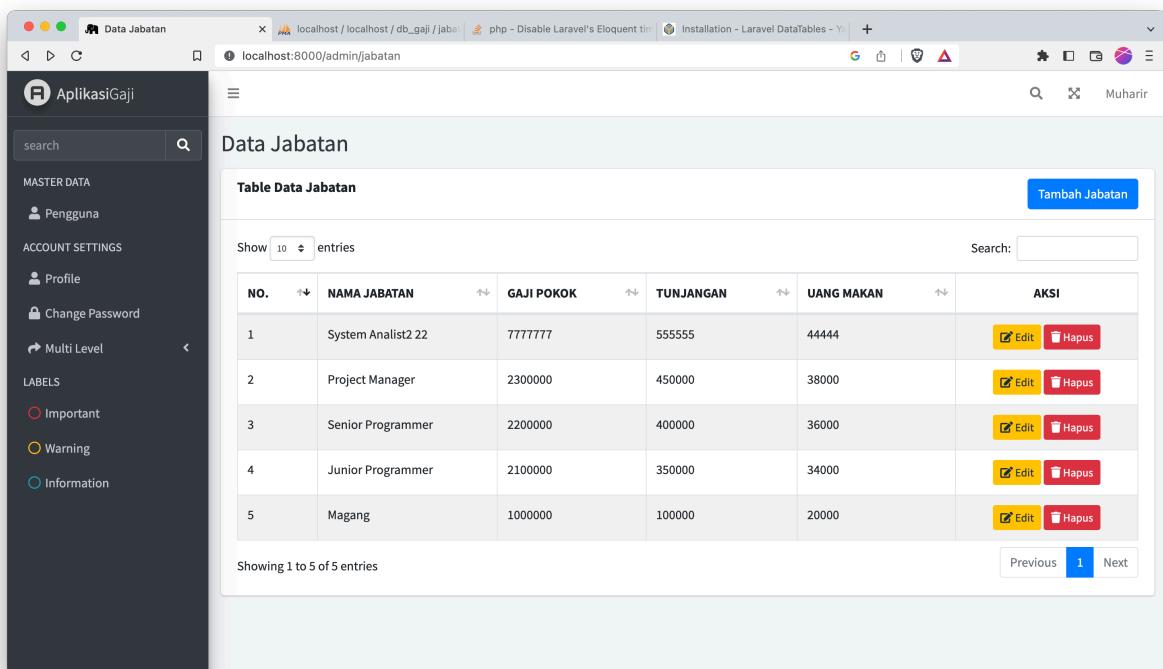
```
36 Route::group( attributes: ['prefix' => 'admin', 'middleware' => ['auth']], routes: function () {  
37     Route::resource( name: 'pengguna', controller: UserController::class);  
38     Route::resource( name: 'jabatan', controller: JabatanController::class);  
39     Route::get( uri: 'get-jabatan', action: [JabatanController::class, 'getJabatan'])->name( name: 'get.jabatan');  
40 } );  
41  
42 } ;|  
43
```

Disana kita menambahkan routing **resource jabatan** dan **get.jabatan** yang digunakan sebagai request **ajax**.

Jangan lupa untuk menambahkan **use JabatanController.php** pada bagian atas

```
routes >  web.php > ...  
1  <?php  
2  
3  use App\Http\Controllers\JabatanController;  
4  use App\Http\Controllers\UserController;  
5  use Illuminate\Support\Facades\Route;  
6
```

Berikutnya buka halaman jabatan pada browser, dan akan tampil seperti berikut



Yeay, berikutnya kita akan menambahkan menunya pada **adminlte.php** di folder **config**.

```
[ 'header' => 'MASTER DATA'],
[
    [
        'text' => 'Pengguna',
        'route'=> 'pengguna.index',
        'icon' => 'fas fa-fw fa-user',
    ],
    [
        'text' => 'Jabatan',
        'route'=> 'jabatan.index',
        'icon' => 'fas fa-fw fa-star',
    ],
]
```

NO.	NAMA JABATAN	GAJI POKOK	TUNJANGAN	UANG MAKAN	AKSI
1	System Analyst	7777777	555555	44444	<button>Edit</button> <button>Hapus</button>
2	Project Manager	2300000	450000	38000	<button>Edit</button> <button>Hapus</button>
3	Senior Programmer	2200000	400000	36000	<button>Edit</button> <button>Hapus</button>
4	Junior Programmer	2100000	350000	34000	<button>Edit</button> <button>Hapus</button>
5	Magang	1000000	100000	20000	<button>Edit</button> <button>Hapus</button>

Sekarang menu jabatan sudah muncul pada sidebar sebelah kiri.

### Menambahkan Tambah Data Jabatan

Tambahkan kode berikut pada method **create** dan **store** di **JabatanController.php**

```
public function create()
{
    return view('masterdata.jabatan.tambah');
}
```

```

public function store(Request $request)
{
    // memvalidasi inputan
    $this->validate($request, [
        'nama_jabatan'      => 'required',
        'gapok_jabatan'     => 'required|numeric',
        'tunjangan_jabatan' => 'required|numeric',
        'uang_makan'        => 'required|numeric',
    ]);

    // insert data ke database
    Jabatan::create($request->all());

    Alert::success('Sukses', 'Berhasil Menambahkan Jabatan Baru');
    return redirect()->route('jabatan.index');
}

```

Karena kita menambahkan alert pastikan **use Alert;** Sudah kalian tambahkan pada bagian atas controller.

```

app > Http > Controllers >  JabatanController.php > ...
1  <?php
2
3  namespace App\Http\Controllers;
4  use App\Models\Jabatan;
5  use DataTables;
6  use Session;
7  use Alert;
8

```

Kemudian sesuaikan view pada **tambah.blade.php** menjadi seperti berikut

```

@extends('adminlte::page')

@section('title', 'Data Jabatan')

@section('content_header')
    <h1 class="m-0 text-dark">Data Jabatan</h1>
@stop

@section('content')
    <div class="row">
        <div class="col-12">
            <div class="card">

                <div class="card-header">

```

```

        <h3 class="card-title"><strong>Tambah Data Jabatan</strong></h3>
    </div>
    <div class="card-body">
        <form action="{{ route('jabatan.store') }}" method="post">
            @csrf
            <div class="form-group row">
                <label class="form-label col-sm-2">Nama Jabatan</label>
                <div class="col-sm-4">
                    <div class="input-group">
                        <input type="text" name="nama_jabatan" placeholder="Manager Produksi" class="form-control" value="{{ old('nama_jabatan') }}" required>
                    </div>
                </div>

                <label class="form-label col-sm-2">Gaji Pokok</label>
                <div class="col-sm-4">
                    <input type="number" name="gapok_jabatan" placeholder="Rp.0" class="form-control" value="{{ old('gapok_jabatan') }}" required>
                </div>
            </div>

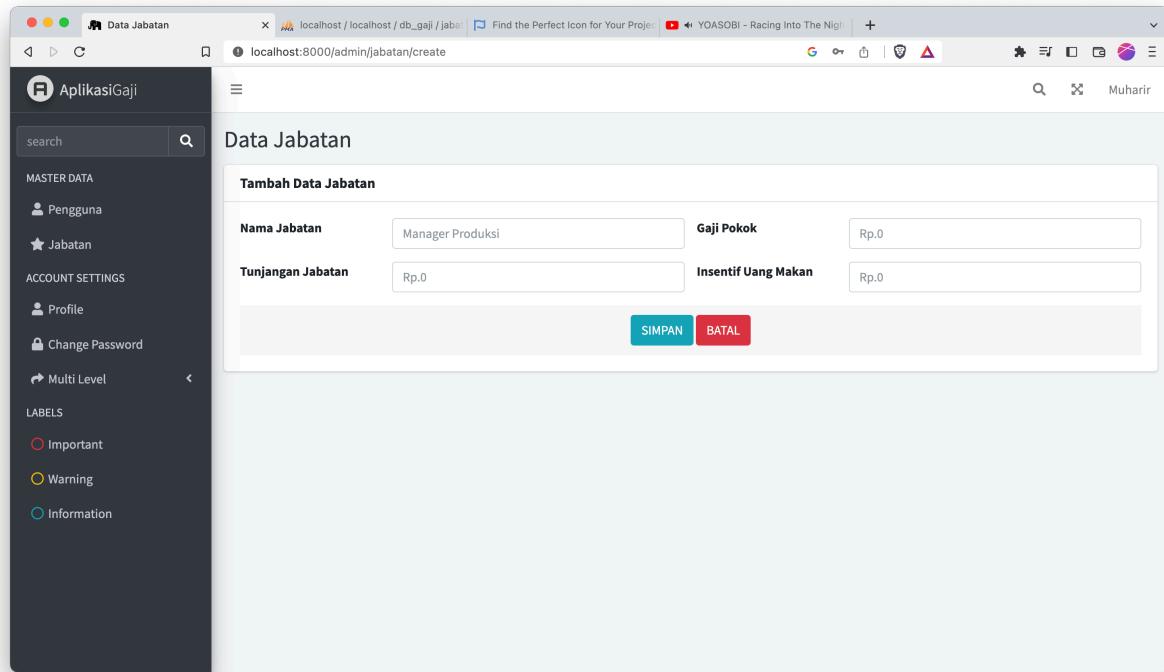
            <div class="form-group row">
                <label class="form-label col-sm-2">Tunjangan Jabatan</label>
                <div class="col-sm-4">
                    <div class="input-group">
                        <input type="number" name="tunjangan_jabatan" placeholder="Rp.0" class="form-control" value="{{ old('tunjangan_jabatan') }}" required>
                    </div>
                </div>

                <label class="form-label col-sm-2">Insentif Uang Makan</label>
                <div class="col-sm-4">
                    <div class="input-group">
                        <input type="number" name="uang_makan" placeholder="Rp.0" class="form-control" value="{{ old('uang_makan') }}" required>
                    </div>
                </div>
            </div>

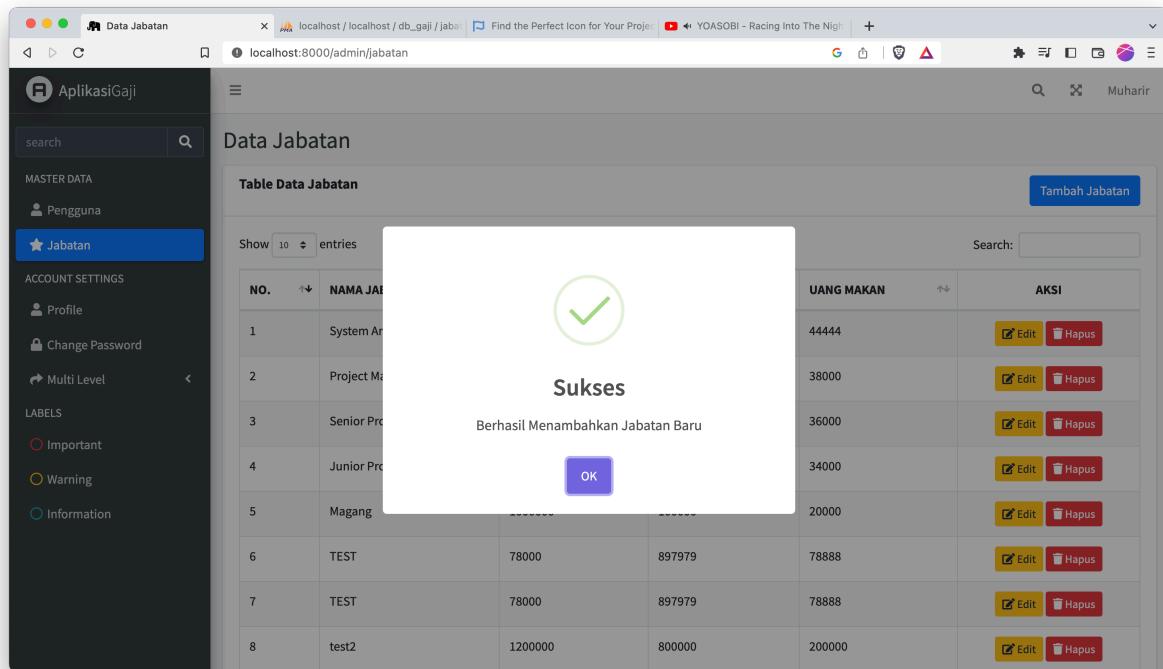
            <div class="card-footer text-center">
                <button type="submit" class="btn btn-info" id="simpan">SIMPAN</button>
                <a href="{{ route('jabatan.index') }}" class="btn btn-danger">BATAL</a>
            </div>
        </form>
    </div>
</div>
```

```
        </div>
    </div>
@stop
```

Jika dijalankan akan menampilkan halaman seperti berikut, coba mulai menambahkan data baru.



Dan jika semua tahapan benar, maka akan muncul notifikasi sweetalert seperti berikut



## Menambahkan Edit Dan Hapus Jabatan

Ubah isi method **edit()**, **update()**, **destroy()** pada **JabatanController.php** seperti berikut,

```

public function edit(Jabatan $jabatan)
{
    return view('masterdata.jabatan.edit', compact('jabatan'));
}

public function update(Request $request, Jabatan $jabatan)
{
    // memvalidasi inputan
    $this->validate($request, [
        'nama_jabatan'      => 'required',
        'gapok_jabatan'     => 'required|numeric',
        'tunjangan_jabatan' => 'required|numeric',
        'uang_makan'        => 'required|numeric',
    ]);

    // insert data ke database
    $jabatan->update($request->all());

    Alert::success('Sukses', 'Berhasil Mengupdate Jabatan ');
    return redirect()->route('jabatan.index');
}

```

```

public function destroy(Jabatan $jabatan)
{
    $jabatan->destroy($jabatan->id);
    Alert::success('Sukses', 'Berhasil Menghapus Jabatan ');
    return redirect()->route('jabatan.index');
}

```

Lalu berikutnya perbaiki **edit.blade.php** pada folder jabatan seperti berikut

```

@extends('adminlte::page')

@section('title', 'Data Jabatan')

@section('content_header')
    <h1 class="m-0 text-dark">Data Jabatan</h1>
@stop

@section('content')
    <div class="row">
        <div class="col-12">
            <div class="card">

                <div class="card-header">
                    <h3 class="card-title"><strong>Edit Data Jabatan</strong></h3>
                </div>
                <div class="card-body">
                    <form action="{{ route('jabatan.update', $jabatan->id) }}" method="post">
                        @method('PUT')
                        @csrf
                        <div class="form-group row">
                            <label class="form-label col-sm-2">Nama Jabatan</label>
                            <div class="col-sm-4">
                                <div class="input-group">
                                    <input type="text" name="nama_jabatan" placeholder="Manager Produksi" class="form-control" value="{{ isset($jabatan) ? $jabatan->nama_jabatan : old('nama_jabatan') }}" required>
                                </div>
                            </div>

                            <label class="form-label col-sm-2">Gaji Pokok</label>
                            <div class="col-sm-4">
                                <input type="number" name="gapok_jabatan" placeholder="Rp. 0" class="form-control" value="{{ isset($jabatan) ? $jabatan->gapok_jabatan : old('gapok_jabatan') }}" required>
                            </div>
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>

```

```

<div class="form-group row">
    <label class="form-label col-sm-2">Tunjangan Jabatan</label>
    <div class="col-sm-4">
        <div class="input-group">
            <input type="number" name="tunjangan_jabatan"
placeholder="Rp.0" class="form-control" value="{{ isset($jabatan) ? $jabatan->tunjangan_jabatan : old('tunjangan_jabatan') }}" required>
        </div>
    </div>

    <label class="form-label col-sm-2">Insentif Uang Makan</label>
    <div class="col-sm-4">
        <div class="input-group">
            <input type="number" name="uang_makan" placeholder="Rp.0"
class="form-control" value="{{ isset($jabatan) ? $jabatan->uang_makan : old('uang_makan') }}" required>
        </div>
    </div>
</div>

<div class="card-footer text-center">
    <button type="submit" class="btn btn-info" id="simpan">SIMPAN</button>
    <a href="{{ route('jabatan.index') }}" class="btn btn-
danger">BATAL</a>
</div>

</form>
</div>
</div>
</div>
</div>
@stop

```

Sekarang bisa kalian coba tombol edit dan hapusnya, jika step-step diatas dijalankan dengan benar maka data dapat di edit dan dihapus.

### **Membuat Cetak PDF Data Jabatan**

Untuk membuat cetak pdf kalian dapat menginstall package DOMPDF, berikut tatacara penginstalannya. Jalankan perintah ini pada command prompt ataupun terminal.

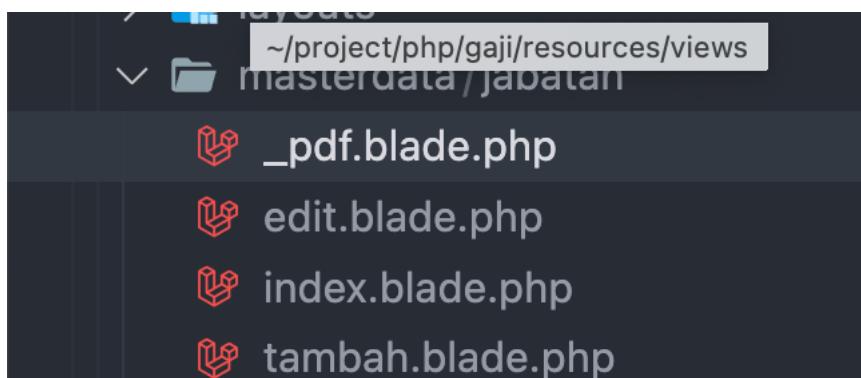
```
composer require barryvdh/laravel-dompdf
```

Berikutnya setelah penginstalan selesai dilakukan, kita akan mengimportnya pada **JabatanController.php** dengan menambahkan `use PDF;` pada bagian atas file **JabatanController.php** lalu buat method baru dengan nama **printPdf()**

```
public function printPdf()
{
    $jabatan = Jabatan::all();

    $pdf = PDF::loadView('masterdata.jabatan._pdf', compact('jabatan'));
    $pdf->setPaper('A4', 'landscape');
    return $pdf->stream('Data Gaji.pdf', array("Attachment" => false));
}
```

Berikutnya buat file baru dengan **nama \_pdf.blade.php** pada **views – masterdata – jabatan.**



Lalu kita buat tampilan cetak pdfnya pada file **\_pdf.blade.php** seperti berikut

```
<!DOCTYPE html>
<html lang="en">

<head>
    <title>Laporan Data Jabatan</title>
    <style>
        table {
            font-family: arial, sans-serif;
            border-collapse: collapse;
            width: 100%;
        }

        td,
        th {
            border: 1px solid #dddddd;
            text-align: left;
            padding: 8px;
        }
    </style>

```

```

    }

    tr:nth-child(even) {
        background-color: #dddddd;
    }

```

</head>

<body>

    <h2 style="text-align: center">Laporan Data Jabatan</h2>

    <hr>

    <table>

        <tr>

            <th>NO.</th>

            <th>NAMA JABATAN</th>

            <th>GAJI POKOK</th>

            <th>TUNJANGAN</th>

            <th>UANG MAKAN</th>

        </tr>

        @php

            \$no = 1;

        @endphp

        @foreach (\$jabatan as \$jb)

            <tr>

                <td>{{ \$no++ }}</td>

                <td>{{ \$jb->nama\_jabatan }}</td>

                <td>{{ \$jb->gapok\_jabatan }}</td>

                <td>{{ \$jb->tunjangan\_jabatan }}</td>

                <td>{{ \$jb->uang\_makan }}</td>

            </tr>

        @endforeach

    </table>

</body>

</html>

Berikutnya buat route untuk menampilkan print pdfnya seperti berikut

```

Route::resource('jabatan', Controller: JabatanController::class);
Route::get('uri: 'get-jabatan', action: [JabatanController::class, 'getJabatan'])->name( name: 'get.jabatan');
Route::get('uri: 'print-pdf', action: [JabatanController::class, 'printPdf'])->name( name: 'print.jabatan');

Route::get('uri: 'grafik-jabatan', action: [JabatanController::class, 'grafikJabatan'])->name( name: 'grafik.jabatan');
Route::get('uri: 'get-grafik', action: [JabatanController::class, 'getGrafik'])->name( name: 'get.grafik.jabatan');

```

Route diatas juga ditambahkan 1 route untuk lihat grafik dan 1 ajax untuk mengolah grafik yang akan di tampilkan, agar nanti tidak perlu membuat kembali.

Berikutnya kita akan memodifikasi tampilan pada **index.blade.php** di folder **views - masterdata - jabatan** untuk membuat tombol cetak pdfnya dan lihat grafik.

```
<div class="card-header">
    <h2 class="card-title"><strong>Table Data Jabatan</strong></h2>
    <div class="form-group float-right">
        <a href="{{ route('jabatan.create') }}" class="btn btn-primary btn-md"> Tambah Jabatan</a>
        <a href="{{ route('print.jabatan') }}" class="btn btn-success btn-md"> Print Jabatan</a>
        <a href="{{ route('grafik.jabatan') }}" class="btn btn-danger btn-md"> Grafik Jabatan</a>
    </div>
</div>
```

Jika dijalankan akan menampilkan tampilan seperti berikut

NO.	NAMA JABATAN	GAJI POKOK	TUNJANGAN	UANG MAKAN	AKSI
1	System Analist2 22	7777777	555555	44444	<button>Edit</button> <button>Hapus</button>
2	Project Manager	2300000	450000	38000	<button>Edit</button> <button>Hapus</button>
3	Senior Programmer	2200000	400000	36000	<button>Edit</button> <button>Hapus</button>
4	Junior Programmer	2100000	350000	34000	<button>Edit</button> <button>Hapus</button>
5	Magang	1000000	100000	20000	<button>Edit</button> <button>Hapus</button>
6	TEST	78000	897979	78888	<button>Edit</button> <button>Hapus</button>
7	TEST3	78000	897979	78888	<button>Edit</button> <button>Hapus</button>
8	test2	1200000	800000	200000	<button>Edit</button> <button>Hapus</button>

Buka aplikasi dan coba untuk print jabatan (Klik Tombol Print Jabatan), jika berhasil akan muncul halaman seperti berikut

NO.	NAMA JABATAN	GAJI POKOK	TUNJANGAN	UANG MAKAN
1	System Analyst22	7777777	555555	44444
2	Project Manager	2300000	450000	38000
3	Senior Programmer	2200000	400000	36000
4	Junior Programmer	2100000	350000	34000
5	Magang	1000000	100000	20000
6	TEST	78000	897979	78888
7	TEST3	78000	897979	78888
8	test2	1200000	800000	200000

Yeay kita telah berhasil menampilkan print data jabatan. **What Next ?**

### Membuat Grafik Jabatan

Untuk membuat grafik jabatan disini kita akan menggunakan **chart.js** sebagai librarynya. Nah pada adminlte template yang kita gunakan kita dapat menambahkannya dengan menggunakan perintah :

```
php artisan adminlte:plugins install --plugin=chartJs
```

```
> php artisan adminlte:plugins install --plugin=chartJs
1/1 [██████████] 100%
The plugins installation is complete. Summary:

+-----+-----+
| Plugin Key | Status   |
+-----+-----+
| chartJs    | Installed |
+-----+-----+
```

Jika sudah kita akan membuat file baru dengan nama **chart.blade.php** pada folder **views - masterdata - jabatan**. Isikan kode berikut di dalamnya.

```

@extends('adminlte::page')

@section('title', 'Data Jabatan')
@section('plugins.Chartjs', true)

@section('content_header')
    <h1 class="m-0 text-dark">Grafik Pendapatan Per Jabatan</h1>
@stop

@section('content')
    <div class="row">
        <div class="col-12">
            <div class="card">

                <div class="card-header">
                    <h2 class="card-title"><strong>Grafik Pendapatan Perjabatan</strong></h2>
                </div>
                <div class="card-body">
                    <div style="width: 800px; margin: 0px auto;">
                        <canvas id="myChart"></canvas>
                    </div>
                </div>
            </div>
        </div>
    </div>
@stop

@push('js')
    <script type="text/javascript">
        $(document).ready(function() {
            $.ajax({
                type: "GET",
                url: "{{ route('get.grafik.jabatan') }}",
                success: function(response) {
                    var labels = response.data.map(function(e) {
                        return e.nama_jabatan
                    })

                    var data = response.data.map(function(e) {
                        return e.gapok_jabatan
                    })

                    var ctx = $('#myChart');
                    var config = {
                        type: 'bar',
                        data: {
                            labels: labels,

```

```

        datasets: [{  

            label: 'Transaksi Status',  

            data: data,  

            backgroundColor: 'rgba(75, 192, 192, 1)',  

        }]  

    }  

};  

var chart = new Chart(ctx, config);  

},  

error: function(xhr) {  

    console.log(xhr.responseJSON);  

}  

});  

});  

</script>  

@endpush

```

Jika sudah tambahkan 2 method pada **JabatanController.php**

```

public function grafikJabatan()  

{  

    return view('masterdata.jabatan.chart');  

}  

public function getGrafik()  

{  

    $jabatan = Jabatan::select('nama_jabatan', 'gapok_jabatan')->get();  

    return response()->json([  

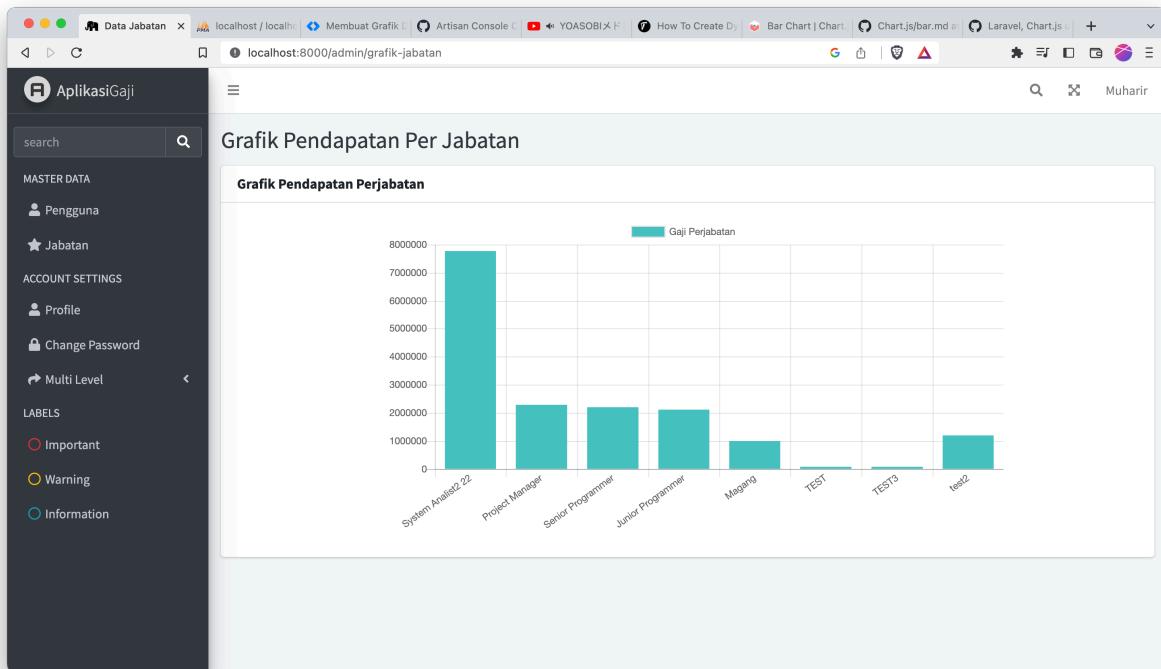
        'data' => $jabatan  

    ]);  

}

```

Berikut adalah tampilan grafik Gaji (Pendapatan) Per jabatan.



## Export Jabatan ke Excel (Bonus)

Untuk membuat export data ke excel ini sangat mudah. Kita dapat memanfaatkan view `_pdf.blade.php` dengan mengcoponya dan merename dengan nama `_excel.blade.php` untuk membuat excelnnya. Langkah berikutnya buat method baru pada `JabatanController.php` dengan nama `exportExcel()`

```
public function exportExcel()
{
    $jabatan = Jabatan::all();
    return view('masterdata.jabatan._excel', compact('jabatan'));
}
```

Berikutnya ubah view `_excel.blade.php` seperti berikut :

```
@php
    header("Content-type: application/vnd-ms-excel");
    header("Content-Disposition: attachment; filename=export_gaji_per_jabatan.xls");
@endphp
<!DOCTYPE html>
<html lang="en">

<head>
    <title>Laporan Data Jabatan</title>
```

```

<style>
    table {
        font-family: arial, sans-serif;
        border-collapse: collapse;
        width: 100%;
    }

    td,
    th {
        border: 1px solid #dddddd;
        text-align: left;
        padding: 8px;
    }

    tr:nth-child(even) {
        background-color: #dddddd;
    }
</style>
</head>

<body>
    <h2 style="text-align: center">Laporan Data Jabatan</h2>
    <hr>
    <table>
        <tr>
            <th>NO.</th>
            <th>NAMA JABATAN</th>
            <th>GAJI POKOK</th>
            <th>TUNJANGAN</th>
            <th>UANG MAKAN</th>
        </tr>
        @php
            $no = 1;
        @endphp
        @foreach ($jabatan as $jb)
            <tr>
                <td>{{ $no++ }}</td>
                <td>{{ $jb->nama_jabatan }}</td>
                <td>{{ $jb->gapok_jabatan }}</td>
                <td>{{ $jb->tunjangan_jabatan }}</td>
                <td>{{ $jb->uang_makan }}</td>
            </tr>
        @endforeach
    </table>
</body>

</html>

```

Berikutnya tambahkan route export.excel baru seperti berikut,

```
Route::get('uri: "grafik-jabatan", action: [JabatanController::class, 'grafikJabatan'])->name( name: 'grafik.jabatan');
Route::get('uri: "get-grafik", action: [JabatanController::class, 'getGrafik'])->name( name: 'get.grafik.jabatan');
Route::get('uri: "export-excel", action: [JabatanController::class, 'exportExcel'])->name( name: 'export.excel');
```

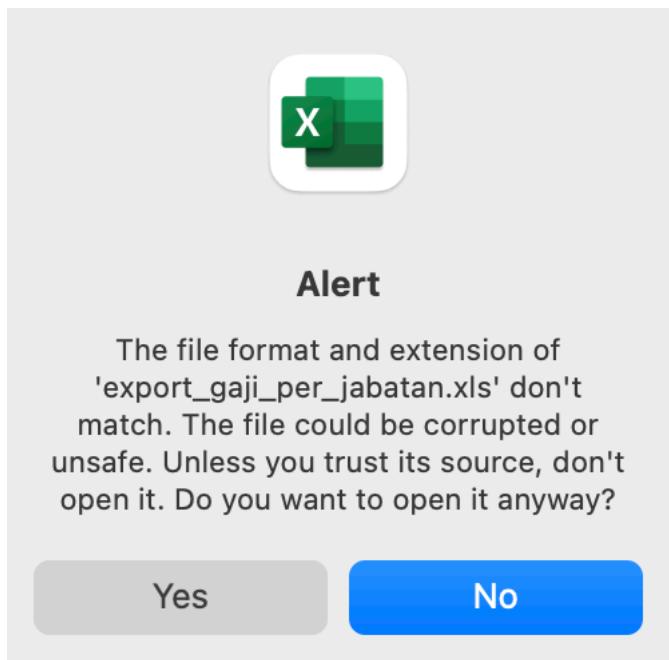
Terakhir buat tombol baru pada **index.blade.php** seperti berikut

```
<div class="form-group float-right">
    <a href="{{ route('jabatan.create') }}" class="btn btn-primary btn-md"> Tambah Jabatan</a>
    <a href="{{ route('print.jabatan') }}" class="btn btn-success btn-md"> Print Jabatan</a>
    <a href="{{ route('grafik.jabatan') }}" class="btn btn-danger btn-md"> Grafik Jabatan</a>
    <a href="{{ route('export.excel') }}" class="btn btn-info btn-md"> Export Excel</a>
</div>
```

Tampilan jika di jalankan, maka akan muncul 1 tombol baru yaitu export excel seperti pada gambar di bawah ini. Mari kita coba untuk klik tombol tersebut. Apakah file excelnnya benar-benar akan di export .

NO.	NAMA JABATAN	GAJI POKOK	TUNJANGAN	UANG MAKAN	AKSI
1	System Analyst22	7777777	555555	44444	<a href="#">Edit</a> <a href="#">Hapus</a>
2	Project Manager	2300000	450000	38000	<a href="#">Edit</a> <a href="#">Hapus</a>
3	Senior Programmer	2200000	400000	36000	<a href="#">Edit</a> <a href="#">Hapus</a>
4	Junior Programmer	2100000	350000	34000	<a href="#">Edit</a> <a href="#">Hapus</a>
5	Magang	1000000	100000	20000	<a href="#">Edit</a> <a href="#">Hapus</a>
6	TEST	78000	897979	78888	<a href="#">Edit</a> <a href="#">Hapus</a>
7	TEST3	78000	897979	78888	<a href="#">Edit</a> <a href="#">Hapus</a>
8	test2	1200000	800000	200000	<a href="#">Edit</a> <a href="#">Hapus</a>

Jika berhasil maka file akan terdownload dan jika membua file tersebut ada prompt seperti berikut klik yes saja



Maka semua datanya akan muncul pada file excel. Yeay Selamat ☺

## Tentang Penulis



### **Muharir**

Penulis lahir di Petanahan, 17 Januari 1994. Penulis tercatat sebagai pengajar pada salah satu Perguruan Tinggi di Kalimantan Selatan pada prodi Teknologi Infomasi (UNISKA).

Penulis juga bekerja sebagai software developer pada instansi pemerintahan. Disela-sela waktu luangnya, penulis selalu meng-update informasi tentang teknologi secara umum maupun web dan mobile teknologi secara khusus.

Diluar itu semua, penulis juga memiliki hobi ngoding dan telah membuat beberapa proyek yang dapat dilihat pada halaman [Github](#) penulis.

Penulis dapat dihubungi melalui kontak berikut :

Whatsapp : +62 822 9012 1212

Email : [muharir17@gmail.com](mailto:muharir17@gmail.com)

LinkedIn : [Muharir](#)