**LAPORAN PEMBUATAN APLIKASI**

**PENJUALAN BENGKEL KOMPUTER**

**DI SMK NEGERI 1 KARANG BARU**

**TAHUN 2024**



**DISUSUN OLEH:**

**Nama : Indah Rahayu**

**NISN : 0074586930**

**Program Keahlian : Teknik Komputer dan Informatika**

**Kompetensi Keahlian : Rekayasa Perangkat Lunak**

**PEMERINTAH ACEH**

**DINAS PENDIDIKAN**

**SEKOLAH MENENGAH KEJURUAN (SMK) NEGERI 1 KARANG BARU**

**KABUPATEN ACEH TAMIANG**

**TAHUN 2024**

# LEMBAR PENGESAHAN

**LAPORAN PEMBUATAN APLIKASI**

**PENJUALAN BENGKEL KOMPUTER**

**DI SMKN 1 KARANG BARU**

**TAHUN 2024**

**Nama : Indah rahayu**

**NISN : 0074586930**

**Program Keahlian : Teknik Komputer dan Informatika**

**Kompetensi Keahlian : Rekayasa Perangkat Lunak**

Telah disahkan tanggal, ……………………………….

Menyutujui:

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  | Pembimbing Sekolah  **Ahmadi Muslim, MP**  NIP. 19860505 201403 1 003 |

# KATA PENGANTAR

Puji syukur saya ucapkan Kehadirat Allah SWT. karena dengan rahmat dan karunia-Nya penulis dapat menyelesaikan laporan praktek kerja industri (Prakerin) yang telah dilaksanakan di SMKN 1 Karang Baru Kabupaten Aceh Tamiang.

Sehubungan dengan terlaksanakannya Prakerin ini tidak terlepas dari bantuan dan dorongan dari semua pihak secara moril maupun materil, oleh karena itu penulis mengucapkan terima kasih sebesar-besarnya kepada:

1. Bapak Fahmi Putra, S.Pd selaku Kepala SMK Negeri 1 Karang Baru
2. Bapak Ahmadi Muslim, MP selaku Kepala Badan Kepegawaian Dan Pengembangan Sumber daya Manusia Kabupaten Aceh Tamiang
3. Bapak Safrizal, S.ST selaku Kepala Bidang Rekayasa Perangkat Lunak SMK Negeri 1 Karang Baru
4. Bapak Ahmadi Muslim, MP selaku Pembimbing di SMK Negeri 1 Karang Baru
5. Bapak Ahmadi Muslim, MP selaku pembimbing di SMKN 1 Karang Baru Aceh Tamiang
6. Bapak/Ibu Guru di SMK Negeri 1 Karang Baru
7. Para pegawai di SMKN 1 Karang Baru

Penulis menyadari bahwa laporan ini masih jauh dari kesempurnaan. Oleh karena itu penulis menghrapkan kritik dan saran yang sifatnya membangun bagi pembaca, penulis berharap semoga laporan ini bermanfaat.

Karang Baru, 20 November 2023

Penyusun,

**Ahmadi Muslim**

# DAFTAR ISI

[**LEMBAR PENGESAHAN i**](#_Toc164929218)

[**KATA PENGANTAR ii**](#_Toc164929219)

[**DAFTAR ISI iii**](#_Toc164929220)

[**BAB I PENDAHULUAN 1**](#_Toc164929221)

[A. Latar Belakang 1](#_Toc164929222)

[B. Tujuan dan Manfaat Aplikasi 1](#_Toc164929223)

[C. Cara Kerja Aplikasi 1](#_Toc164929224)

[D. Dokumen Pendukung Perancangan Aplikasi 1](#_Toc164929225)

[BA**B II PERANCANGAN BASIS DATA** 2](#_Toc164929226)

[A. Entitas Relationship Diagram 2](#_Toc164929228)

[B. Desain Tabel Sebelum Normalisasi 2](#_Toc164929229)

[C. Normalisasi 2](#_Toc164929230)

[D. Design Tabel Setelah Normalisasi 2](#_Toc164929231)

[E. Migration 3](#_Toc164929232)

[**BAB III TAMPILAN APLIKASI 8**](#_Toc164929233)

[A. Fitur Utama / Dashboard 8](#_Toc164929235)

[B. Fitur Petugas 8](#_Toc164929236)

[C. Fitur Login 8](#_Toc164929237)

[D. Fitur Barang 8](#_Toc164929238)

[E. Fitur Pelanggan 8](#_Toc164929239)

[F. Fitur Transaksi 8](#_Toc164929240)

[G. Fitur Detiltransaksi 8](#_Toc164929241)

[H. Fitur Laporan 8](#_Toc164929242)

# BAB I PENDAHULUAN

## Latar Belakang

Selama ini sistem penjualan di Bengkel Komputer SMKN 1 Karang Baru masih sering dilakukan secara offline dan manual. Hal ini tentenya sangat berpotensi terjadinya resiko kesalahan dalam pendataan dan lambat dalam proses pelayanan. Menanggapi hal tersebut pembuatan aplikasi POS di Bengkel Komputer SMKN 1 Karang Baru dirasa sangat perlu untuk dilakukan.

## Tujuan dan Manfaat Aplikasi

1. Untuk mempercepat dalam proses pelayanan terhadap konsumen/pelanggan
2. Memudahkan dan lebih praktis dalam proses perhitungan jumlah pembayaran

## Cara Kerja Aplikasi

* Seorang petugas dapat menjual banyak layanan
* Seorang pembeli dapat membeli lebih dari satu layanan yang dijual oleh petugas
* Seorang petugas dapat merekap banyak transaksi
* Transaksi memiliki detiltransaksi
* Stok layanan akan berkurang sesuai jumlah layanan yang dibeli
* Nomor telepon petugas boleh lebih dari satu
* Petugas, pembeli, layanan dan transaksi dapat di identifikasi dengan id yang berbeda (unik)

## Dokumen Pendukung Perancangan Aplikasi

Barang yang diperjualbelikan

Daftar harga barang

# BAB II PERANCANGAN BASIS DATA



## Entitas Relationship Diagram (ERD)



**Penjelasan ERD:**

* *Seorang petugas dapat menjual banyak barang*
* *Seorang pelanggan dapat membeli lebih dari satu barang yang dijual oleh petugas*
* *Seorang petugas dapat merekap banyak transaksi*

## Desain Tabel Aplikasi



## Migration

**Versi 1: Skema Relasi by Table Migration files**

Skema relasi by tabel adalah pembuatan tabel berbarengan dengan relasi tabel. Penyusunan skema relasi dari tabel child ke tabel parent dibuat menjadi satu file dengan pembuatan atribut/fields pada tabel. Skema ini dilakukan dengan membuat tabel induk/parent terlebih dahulu, baru diikuti dengan membuat tabel child.

* + 1. **Tabel User**

Untuk tabel user sebenarnya tidak wajib dibuat, karena secara default sudah disediakan oleh laravel. Akan tetapi jika harus dibuat, maka dapat dituliskan seperti berikut ini

1. Memahami desain tabel dari model ERT



* *Tabel user merupakan tabel induk yang akan berisi tentang daftar user yang akan login untuk mengelola aplikasi.*
* *Tabel ini nantinya akan direferensi oleh tabel transaksi sehingga kedudukannya adalah* ***parent****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model User -m

1. Buka file create\_users\_table.php yang terdapat dalam folder migration di folder Database.
2. Kemudian ubah seperti berikut:

public function up(): void

    {

        Schema::create('users', function (Blueprint $table) {

            $table->id();

            $table->string('nama');

            $table->string('email');

            $table->string('password');

            $table->timestamps();

        });

    }

* + 1. **Tabel Pelanggan**

1. Memahami desain tabel dari model ERT



* *Tabel pelanggan merupakan tabel yang akan berisi tentang daftar pelanggan yang akan menjadi pembeli/stakeholder/customer pada proses penjualan.*
* *Tabel ini nantinya akan direferensi oleh tabel transaksi sehingga kedudukannya adalah* ***parent****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model pelanggan -m

1. Buka file create\_pelanggans\_table.php yang terdapat dalam folder migration di folder Database.
2. Kemudian ubah seperti berikut:

 public function up(): void

    {

        Schema::create('pelanggans', function (Blueprint $table) {

            $table->id();

          $table->string('nama');

          $table->string('hp')->nullable();

            $table->text('alamat')->nullable();

            $table->text('kec')->nullable();

            $table->timestamps();

        });

    }

* + 1. **Tabel Barang BELOM**

1. Memahami desain tabel dari model ERT



* *Tabel barang merupakan tabel yang akan berisi tentang daftar menu barang yang akan dijual oleh user dan dibeli oleh pelanggan.*
* *Tabel ini nantinya akan direferensi oleh tabel detiltransaksi sekaligus mereferensi tabel suplier dna tabel teknisi sehingga kedudukannya adalah* ***parent*** *sekaligus* ***child****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model barang –m

1. Buka file create\_barangs\_table.php yang terdapat dalam folder migration di folder Database.
2. Kemudian ubah seperti berikut:

public function up(): void

    {

        Schema::create('layanans', function (Blueprint $table) {

            $table->id();

            $table->string('nama');

            $table->string('jeniskategori');

            $table->bigInteger('suplier\_id')->unsigned();

            $table->string('description')->nullable();

            $table->integer('stock');

            $table->double('harga');

            $table->timestamps();

        });

Schema::table('layanans',function(Blueprint $table){

            $table->foreign('suplier\_id')->references('id')->on('supliers')

            ->onUpdate('cascade')->onDelete('cascade');

        });

        Schema::table('layanans',function(Blueprint $table){

            $table->foreign('teknisi\_id')->references('id')->on('teknisis')

            ->onUpdate('cascade')->onDelete('cascade');

        });

1. Untuk function down() menjadi seperti berikut:

public function down(): void

    {

       Schema::table('layanans', function(Blueprint $table) {

            $table->dropForeign('layanans\_suplier\_id\_foreign');

        });

        Schema::table('layanans', function(Blueprint $table) {

            $table->dropIndex('layanans\_suplier\_id\_foreign');

        });

        Schema::table('layanans', function(Blueprint $table) {

            $table->dropForeign('layanans\_teknisi\_id\_foreign');

        });

        Schema::table('layanans', function(Blueprint $table) {

            $table->dropIndex('layanans\_teknisi\_id\_foreign');

        });

        Schema::dropIfExists('layanans');

    }

* + 1. **Tabel Transaksi**

1. Memahami desain tabel dari model ERT



* *Tabel transaksi merupakan tabel yang akan berisi tentang transaksi barang yang dijual oleh user dan dibeli oleh pelanggan.*
* *Tabel ini nantinya akan direferensi oleh tabel detiltransaksi sekaligus mereferensi tabel suplier dan tabel teknisi sehingga kedudukannya adalah* ***parent*** *sekaligus* ***child****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model transaksi -m

1. Ubah file migration transaksi menjadi seperti berikut:

public function up(): void

    {

        Schema::create('transaksis', function (Blueprint $table) {

            $table->id();

            $table->string('invoice')->unique();

            $table->bigInteger('pelanggans\_id')->unsigned();

            $table->bigInteger('users\_id')->unsigned();

            $table->double('total');

            $table->timestamps();

        });

        Schema::table('transaksis',function(Blueprint $table){

            $table->foreign('pelanggans\_id')->references('id')->on('pelanggans')

            ->onUpdate('cascade')->onDelete('cascade');

            $table->foreign('users\_id')->references('id')->on('users')

            ->onUpdate('cascade')->onDelete('cascade');

        });

    }

1. Untuk function down() menjadi seperti berikut:

 public function down(): void

    {

        Schema::table('transaksis', function(Blueprint $table) {

            $table->dropForeign('transaksis\_pelanggans\_id\_foreign');

        });

        Schema::table('transaksis', function(Blueprint $table) {

            $table->dropIndex('transaksis\_pelanggans\_id\_foreign');

        });

        Schema::table('transaksis', function(Blueprint $table) {

            $table->dropForeign('transaksis\_users\_id\_foreign');

        });

        Schema::table('transaksis', function(Blueprint $table) {

            $table->dropIndex('transaksis\_users\_id\_foreign');

        });

        Schema::dropIfExists('transaksis');

    }

};

* + 1. **Tabel Detiltransaksi**

1. Memahami desain tabel dari model ERT



* *Tabel detiltransaksi merupakan tabel yang akan berisi tentang transaksi layanan yang dijual oleh user dan dibeli oleh pelanggan.*
* *Tabel ini nantinya akan mereferensi tabel transakssi dan tabel layanan sehingga kedudukannya adalah* ***child****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model detiltransaksi -m

1. Ubah file migration detiltransaksi menjadi seperti berikut:

public function up(): void

    {

        Schema::create('detiltransaksis', function (Blueprint $table) {

            $table->id();

            $table->bigInteger('transaksis\_id')->unsigned();

            $table->bigInteger('barangs\_id')->unsigned();

            $table->integer('qty');

            $table->double('harga');

            $table->timestamps();

        });

        Schema::table('detiltransaksis', function(Blueprint $table) {

            $table->foreign('transaksis\_id')->references('id')->on('transaksis')

                   ->onUpdate('cascade') ->onDelete('cascade');

        $table->foreign('barangs\_id')->references('id')->on('barangs')

                   ->onUpdate('cascade') ->onDelete('cascade');

           });

       }

1. Kemudian untuk function down() menjadi seperti berikut:

    {

        Schema::table('detiltransaksis', function(Blueprint $table) {

            $table->dropForeign('detiltransaksis\_transaksis\_id\_foreign');

        });

        Schema::table('detiltransaksis', function(Blueprint $table) {

            $table->dropIndex('detiltransaksis\_transaksis\_id\_foreign');

        });

        Schema::table('detiltransaksis', function(Blueprint $table) {

            $table->dropForeign('detiltransaksis\_barangs\_id\_foreign');

        });

        Schema::table('detiltransaksis', function(Blueprint $table) {

            $table->dropIndex('detiltransaksis\_barangs\_id\_foreign');

        });

        Schema::dropIfExists('detiltransaksis');

    }

};

* + 1. **Migrasi/migrate ke database server**

php artisan migrate

* + 1. **Cek hasilnya di localhost/phpmyadmin**



**Versi 2: Skema Relasi out Table Migration Files**

Skema relasi out tabel adalah pembuatan database tanpa memperhatikan atau mengesampingkan dahulu skema relasi tabel-tabel pada database. Penyusunan skema relasi dari tabel child ke tabel parent dibuat di dalam satu file tersendiri melalui skema yang sudah disusun dalam rencana permodelan ERT setelah semua tabel yang dibutuhkan pada aplikasi selesai dikerjakan. Penerapan model ini dirasa lebih mudah untuk menyusun skema tabel pada databse karena tidak akan terjadi error akibat tabel anak/child mencari referensi ke tabel induk/parent.

* + 1. **Tabel User**

Untuk tabel user sebenarnya tidak wajib dibuat, karena secara default sudah disediakan oleh laravel. Akan tetapi jika harus dibuat, maka dapat dituliskan seperti berikut ini

1. Memahami desain tabel dari model ERT



* *Tabel user merupakan tabel induk yang akan berisi tentang daftar user yang akan login untuk mengelola aplikasi.*
* *Tabel ini nantinya akan direferensi oleh tabel transaksi sehingga kedudukannya adalah* ***parent****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model User -m

1. Buka file create\_users\_table.php yang terdapat dalam folder migration di folder Database.
2. Kemudian ubah seperti berikut:

public function up(): void

    {

        Schema::create('users', function (Blueprint $table) {

            $table->id();

            $table->string('nama');

            $table->string('email');

            $table->string('password');

            $table->timestamps();

        });

    }

* + 1. **Tabel Pelanggan**

1. Memahami desain tabel dari model ERT



* *Tabel pelanggan merupakan tabel yang akan berisi tentang daftar pelanggan yang akan menjadi pembeli/stakeholder/customer pada proses penjualan.*
* *Tabel ini nantinya akan direferensi oleh tabel transaksi sehingga kedudukannya adalah* ***parent****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model pelanggan -m

1. Buka file create\_pelanggans\_table.php yang terdapat dalam folder migration di folder Database.
2. Kemudian ubah seperti berikut:

public function up(): void

    {

        Schema::create('pelanggans', function (Blueprint $table) {

            $table->id();

            $table->string('nama');

            $table->string('hp')->nullable();

            $table->text('alamat')->nullable();

            $table->text('kec')->nullable();

            $table->timestamps();

        });

    }

* + 1. **Tabel Barang**

1. Memahami desain tabel dari model ERT



* *Tabel layanan merupakan tabel yang akan berisi tentang daftar menu layanan yang akan dijual oleh user dan dibeli oleh pelanggan.*
* *Tabel ini nantinya akan direferensi oleh tabel detiltransaksi sekaligus mereferensi tabel suplier dna tabel teknisi sehingga kedudukannya adalah* ***parent*** *sekaligus* ***child****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model barang -m

1. Buka file create\_barangs\_table.php yang terdapat dalam folder migration di folder Database.
2. Kemudian ubah seperti berikut:

public function up(): void

    {

        Schema::create('barangs', function (Blueprint $table) {

            $table->id();

            $table->string('barang');

            $table->string('description')->nullable();

            $table->double('harga');

            $table->timestamps();

        });

    }

* + 1. **Tabel Transaksi**

1. Memahami desain tabel dari model ERT



* *Tabel transaksi merupakan tabel yang akan berisi tentang transaksi layanan yang dijual oleh user dan dibeli oleh pelanggan.*
* *Tabel ini nantinya akan direferensi oleh tabel detiltransaksi sekaligus mereferensi tabel suplier dan tabel teknisi sehingga kedudukannya adalah* ***parent*** *sekaligus* ***child****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model transaksi -m

1. Ubah file migration transaksi menjadi seperti berikut:

public function up(): void

    {

        Schema::create('transaksis', function (Blueprint $table) {

            $table->id();

            $table->string('invoice')->unique();

            $table->bigInteger('pelanggans\_id')->unsigned();

            $table->bigInteger('users\_id')->unsigned();

            $table->double('total');

            $table->timestamps();

        });

  }

* + 1. **Tabel Detiltransaksi**

1. Memahami desain tabel dari model ERT



* *Tabel detiltransaksi merupakan tabel yang akan berisi tentang transaksi layanan yang dijual oleh user dan dibeli oleh pelanggan.*
* *Tabel ini nantinya akan mereferensi tabel transakssi dan tabel layanan sehingga kedudukannya adalah* ***child****.*

1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:model detiltransaksi -m

1. Ubah file migration detiltransaksi menjadi seperti berikut:

public function up(): void

    {

        Schema::create('detiltransaksis', function (Blueprint $table) {

            $table->id();

            $table->bigInteger('transaksis\_id')->unsigned();

            $table->bigInteger('barangs\_id')->unsigned();

            $table->integer('qty');

            $table->double('harga');

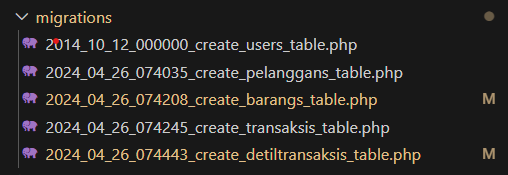
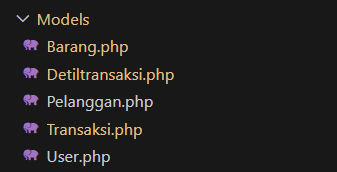
            $table->timestamps();

        });

}

* + 1. **Tabel Relasi pada Folder Migrasi**

Hasil tabel migrasi jika kita berhasil makan akan muncul pada folder model dan migrasi

* + 1. **Tabel Relasi Migrasi**

1. Memahami desain tabel dari model ERT



1. Menuliskan prompt berikut pada terminal VS Code:

php artisan make:migration relasi

1. Ubah file migration relasi menjadi seperti berikut:

    public function up()

    {

        Schema::table('layanans',function(Blueprint $table){

            $table->foreign('suplier\_id')->references('id')->on('supliers')

            ->onUpdate('cascade')->onDelete('cascade');

        });

        Schema::table('layanans',function(Blueprint $table){

            $table->foreign('teknisi\_id')->references('id')->on('teknisis')

            ->onUpdate('cascade')->onDelete('cascade');

        });

        Schema::table('transaksis', function (Blueprint $table) {

            $table->foreign('pelanggan\_id')->references('id')->on('pelanggans')

                ->onUpdate('cascade')->onDelete('cascade');

            $table->foreign('user\_id')->references('id')->on('Users')

                ->onUpdate('cascade')->onDelete('cascade');

        });

        Schema::table('detiltransaksis', function (Blueprint $table) {

            $table->foreign('transaksi\_id')->references('id')->on('transaksis')

                ->onUpdate('cascade')->onDelete('cascade');

            $table->foreign('layanan\_id')->references('id')->on('layanans')

                ->onUpdate('cascade')->onDelete('cascade');

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*/

    public function down(): void

    {

        Schema::table('layanans', function(Blueprint $table) {

            $table->dropForeign('layanans\_suplier\_id\_foreign');

        });

        Schema::table('layanans', function(Blueprint $table) {

            $table->dropIndex('layanans\_suplier\_id\_foreign');

        });

        Schema::table('layanans', function(Blueprint $table) {

            $table->dropForeign('layanans\_teknisi\_id\_foreign');

        });

        Schema::table('layanans', function(Blueprint $table) {

            $table->dropIndex('layanans\_teknisi\_id\_foreign');

        });

        Schema::dropIfExists('layanans');

        Schema::table('transaksis', function (Blueprint $table) {

            $table->dropForeign('transaksi\_pelanggan\_id\_foreign');

        });

        Schema::table('transaksis', function (Blueprint $table) {

            $table->dropIndex('transaksi\_pelanggan\_id\_foreign');

        });

        Schema::table('transaksis', function (Blueprint $table) {

            $table->dropForeign('transaksi\_user\_id\_foreign');

        });

        Schema::table('transaksis', function (Blueprint $table) {

            $table->dropIndex('transaksi\_user\_id\_foreign');

        });

        Schema::dropIfExists('transaksis');

        Schema::table('detiltransaksis', function (Blueprint $table) {

            $table->dropForeign('detiltransaksi\_transaksi\_id\_foreign');

        });

        Schema::table('detiltransaksis', function (Blueprint $table) {

            $table->dropIndex('detiltransaksi\_transaksi\_id\_foreign');

        });

        Schema::table('detiltransaksis', function (Blueprint $table) {

            $table->dropForeign('detiltransaksi\_layanan\_id\_foreign');

        });

        Schema::table('detiltransaksis', function (Blueprint $table) {

            $table->dropIndex('detiltransaksi\_layanan\_id\_foreign');

        });

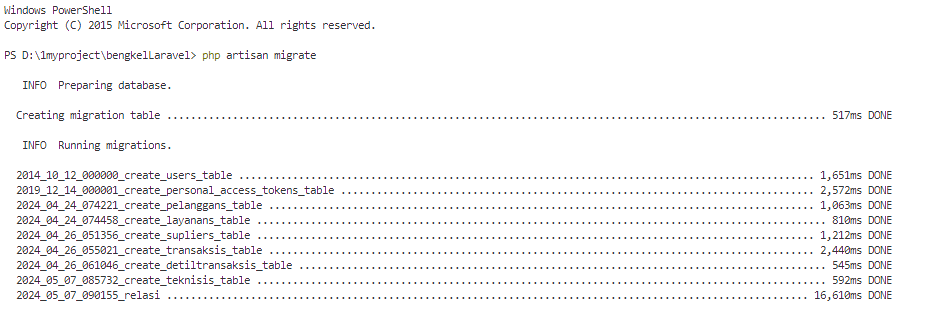
        Schema::dropIfExists('detiltransaksis');

    }

};

* + 1. **Migrasi/migrate ke database server**

php artisan migrate



* + 1. **Cek hasilnya di localhost/phpmyadmin**



# BAB IV BACKUP APLIKASI



## Cara Install Git di Windows

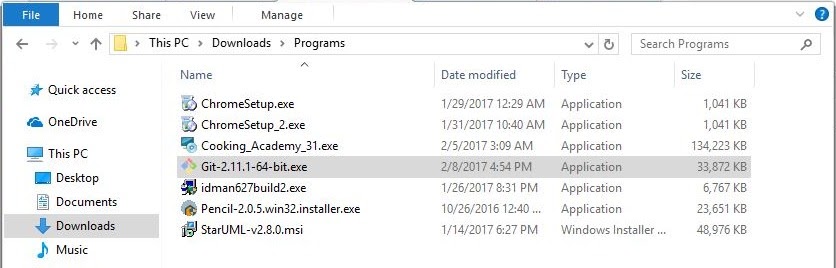
Instalasi Git di Windows memang tidak seperti di Linux yang ketik perintah langsung terinstal.

Kita harus men-download dulu, kemudian melakukan ritual next>next>finish.

Tapi dalam ritual tersebut, ada pilihan yang harus diperhatikan agar perintah git dapat dikenali di CMD.

Silahkan buka website resminya Git ( [git-scm.com](https://git-scm.com/)). Kemudian unduh Git sesuai dengan arsitektur komputer kita. Kalau menggunakan 64bit, unduh yang 64bit. Begitu juga kalau menggunakan 32bit.

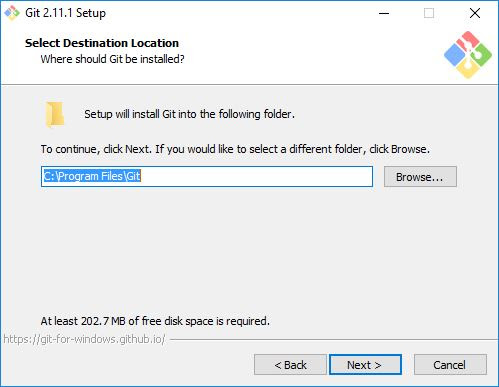
Baiklah, mari kita mulai ritual instalnya. Silahkan klik 2x file instaler Git yang sudah diunduh.



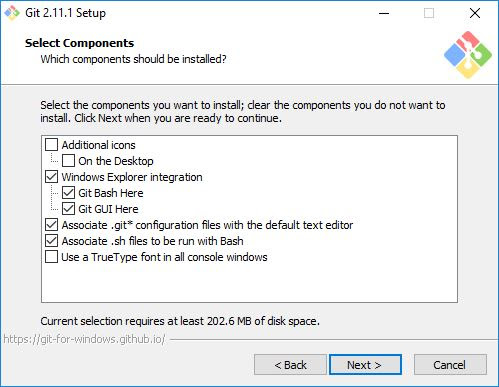
Maka akan muncul infomasi lisensi Git, klik Next > untuk melanjutkan.



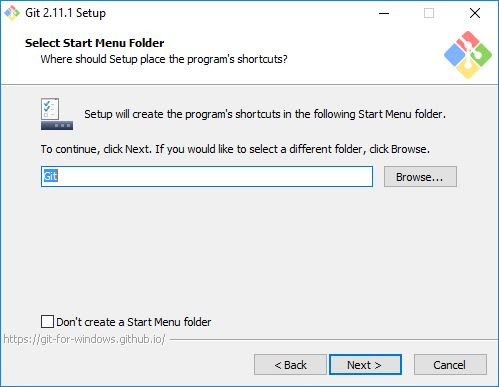
Selanjutnya menentukan lokasi instalasi. Biarkan saja apa adanya, kemudian klik Next >.



Selanjutnya pemilihan komoponen, biarkan saja seperti ini kemudian klik Next >.



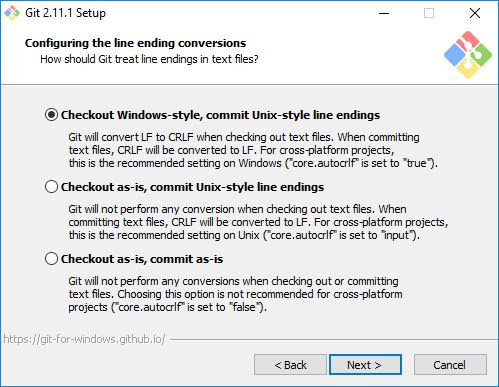
Selanjutnya pemlilihan direktori start menu, klik Next >.



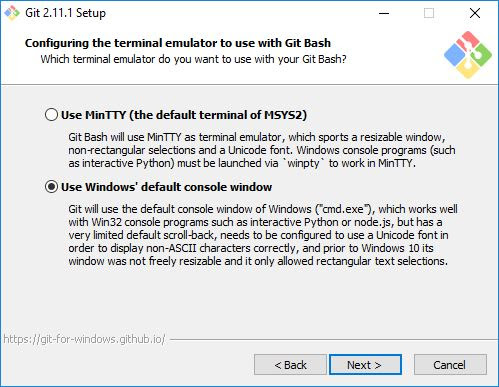
Selanjutnya pengaturan PATH Environment. Pilih yang tengah agar perintah git dapat di kenali di Command Prompt (CMD). Setelah itu klik Next >.



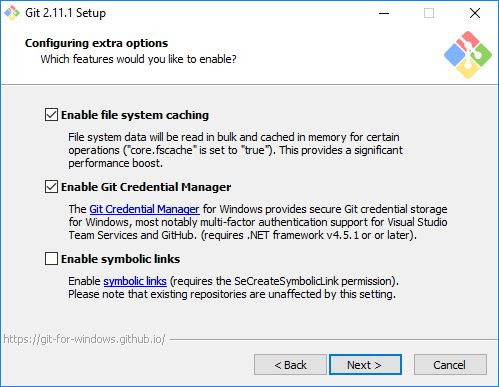
Selanjutnya konversi line ending. Biarkan saja seperti ini, kemudian klik Next >.



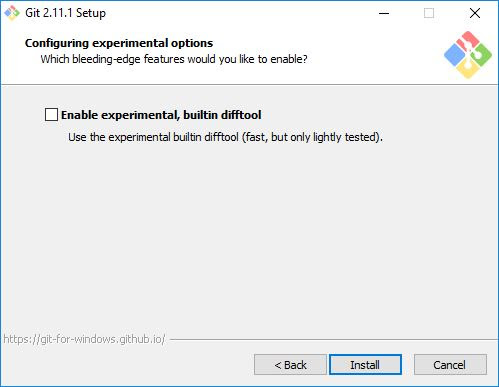
Selanjutnya pemilihan emulator terminal. Pilih saja yang bawah, kemudian klik Next >.



Selanjutnya pemilihan opsi ekstra. Klik saja Next >.



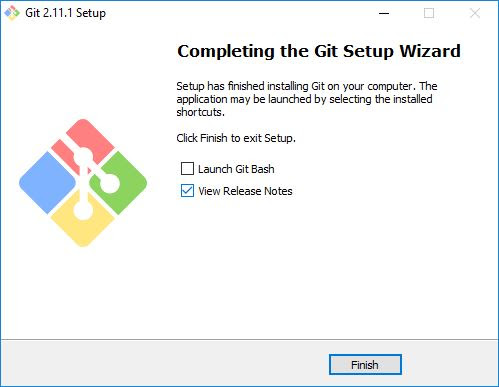
Selanjutnya pemilihan opsi ekspreimental, langsung saja klik Install untuk memaulai instalasi.



Tunggu beberapa saat, instalasi sedang dilakukan.



Setelah selesai, kita bisa langsung klik Finish.



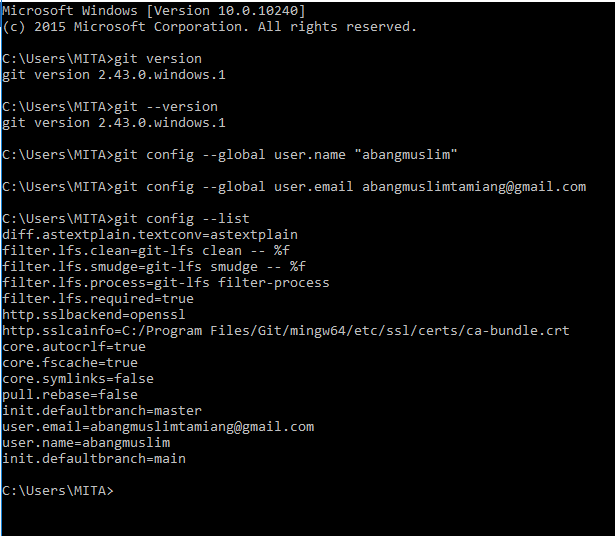
## Setting Dan Konfigurasi Github

Berikut adalah cara melakukan setingan konfigurasi git local desktop. Setingan ini akan menentukan bagaimana cara git terhubung ke github dan lain sebagainya.

1. **Melihat version dan default branch**

Git –v : untuk melihat versi github, jika belum terinstal maka akan muncul pesan error.

Git config –list : untuk melihat daftar konfigurasi setingan yang sudah dilakukan.



1. **Melakukan setting Configurasi Global**

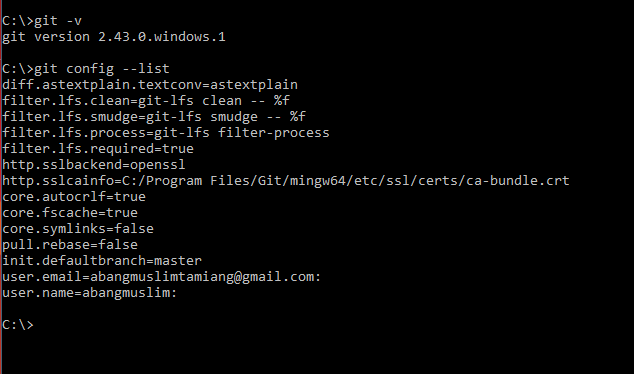
Lakukan setingan configurasi global jika belum ada config ke usesrname di github. Username dan email yang digunakan harus benar benar yang sudah terdaftar.

**Git config --global user.email “email\_user”**:

**Git config --global user.name “nama\_user”**:

**Git config --global user.email “abangmuslimtamiang@gmail.com”**:

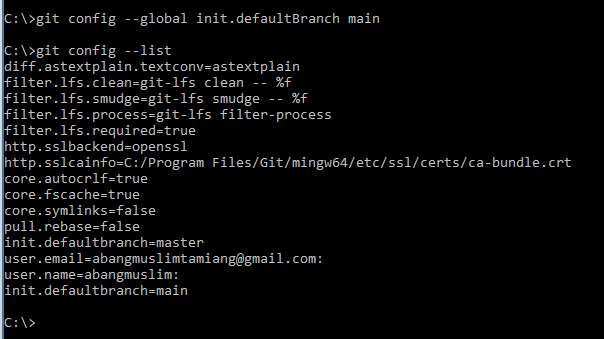
**Git config --global user.name “abangmuslim”**:



1. **Melakukan setting Default Branch Main di Configurasi Global**

Lakukan setingan default branch main agar branch yang di update adalah branch main.

git config --global init.defaultBranch main



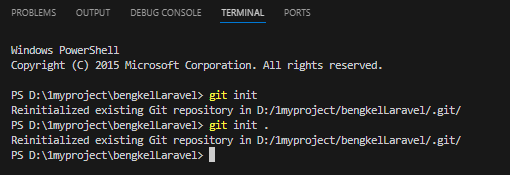
## Bekerja dengan Git dan Github

Berikut adalah cara bekerja dengan git local desktop dan sekaligus mengupdate/singkronisasi di github.

1. **Git Init**

Lakukan git init . untuk menginisialisasi project yang akan di update.

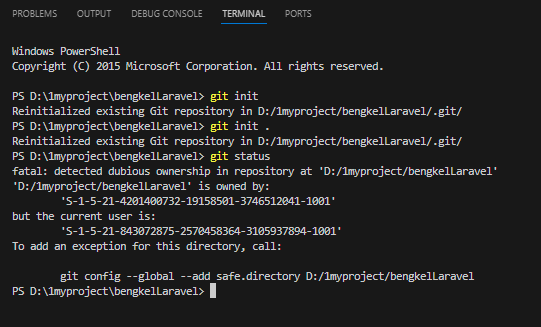
git init.



1. **Git status**

Lakukan git status . untuk melihat status project yang akan di update.

git status.

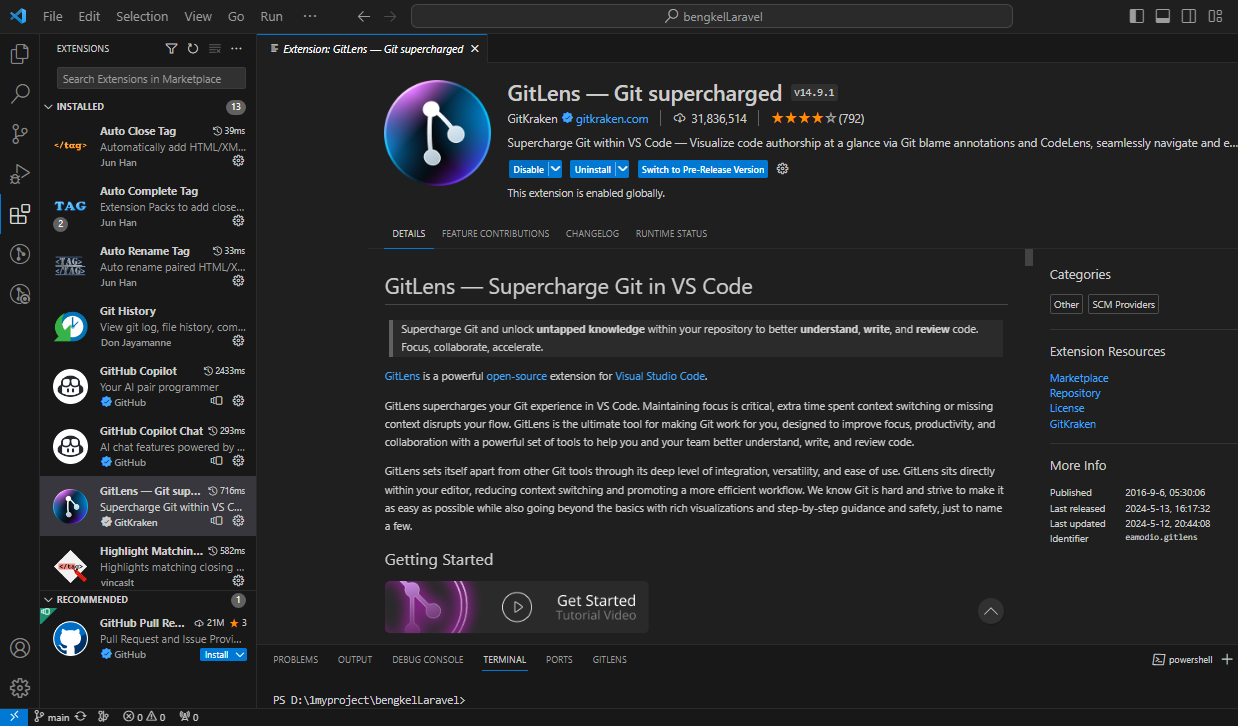


1. **Git Add**

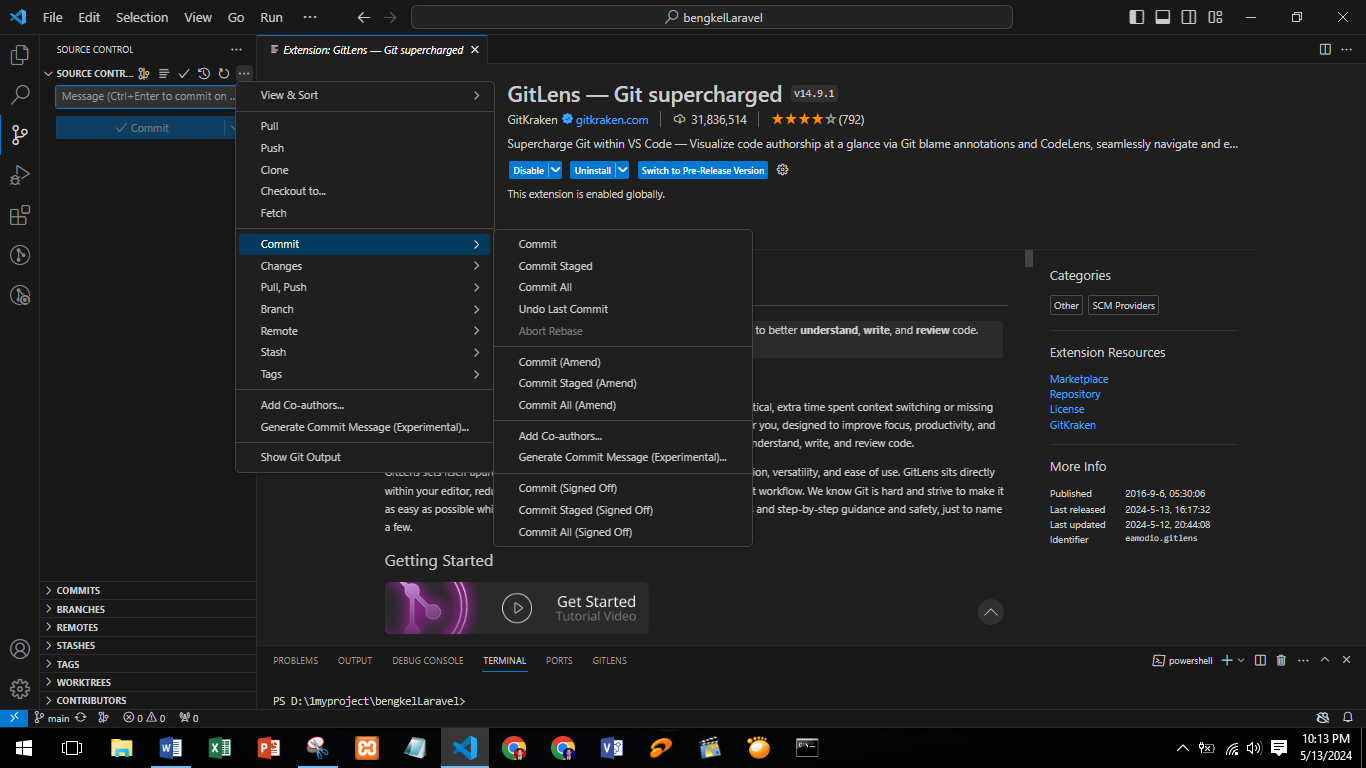
Melakukan git add . untuk melakukan penambahan project yang akan di update.

git add.

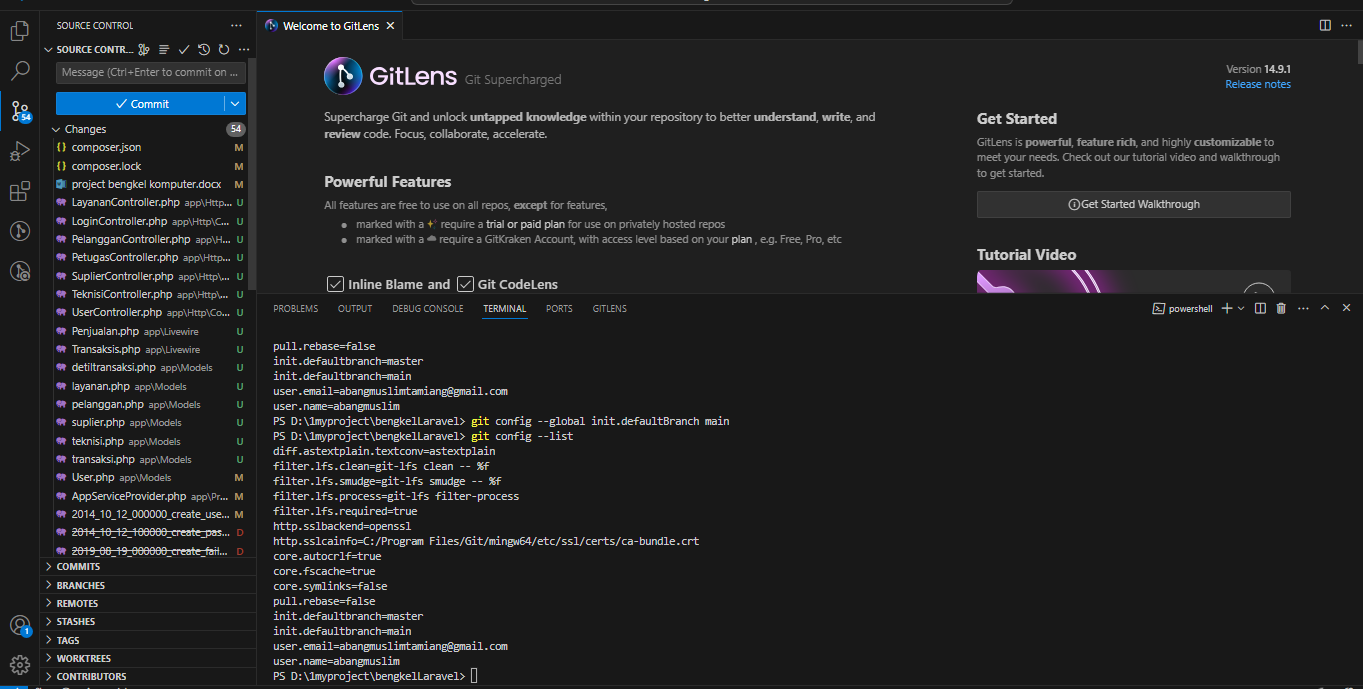
Namun, ada cara yang lebih efektif dan effisien adalah dengan menggunakan extensi dari vs code yaitu **Git Lens**.



Di Ekstensi Ini tidak perlu mengetikkan prompt git add untuk melihat jumlah perubahan yang masuk ke katalog commit. Melainkan hanya perlu mengklik di bagian git tab sidebar kiri vs code.



Selanjutnya hanya perlu mengklik perintah commit jika terlihat bagian yang akan di update ke github sudah sesuai.



Lalu tinggal tunggu proses git di vs code melakukan updat/singkronisasi ke main branch kita yang sudah terinisialisasi di github.

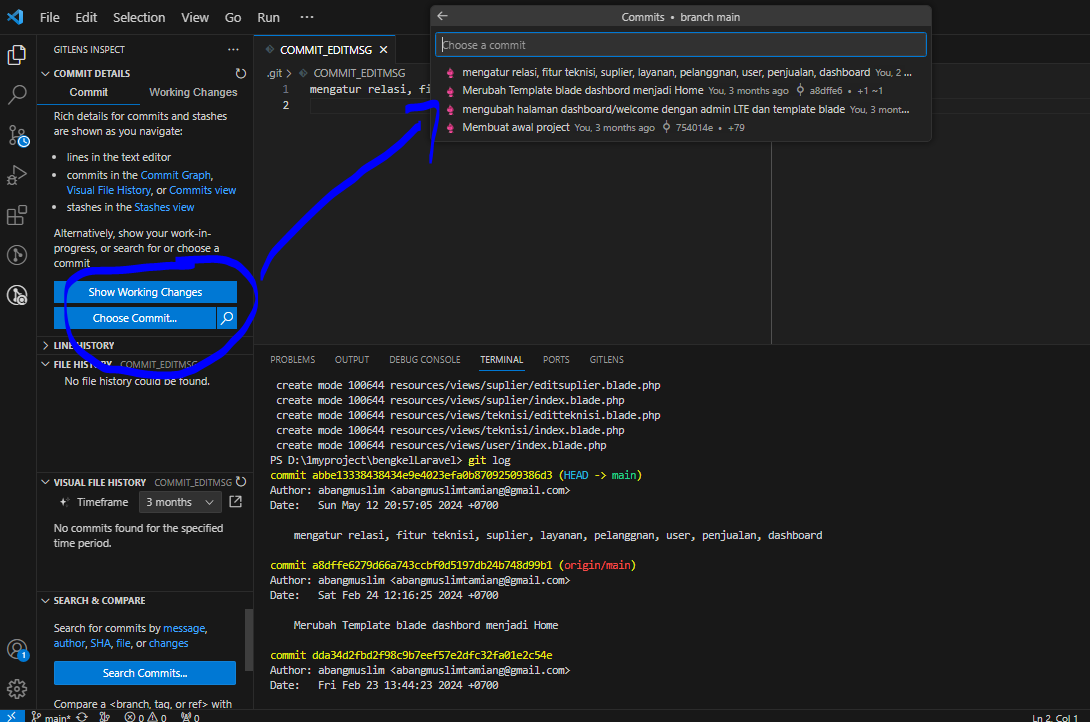


1. **Git Commit**

Melakukan git commit . untuk melakukan commit project yang akan di update.

git commit –m “ komentar atau keterangan mengenai update perubahan yang dilakukan di project ”

Agar lebih efektif dan effisien kita dapat menggunakan git lense. Dan Untuk melakukan commit Jangan lupa memberikan nama pada commit yang kita buat agar perubahan (check point) menjadi lebih dapat dimengerti semua orang, terutama kita sendiri.

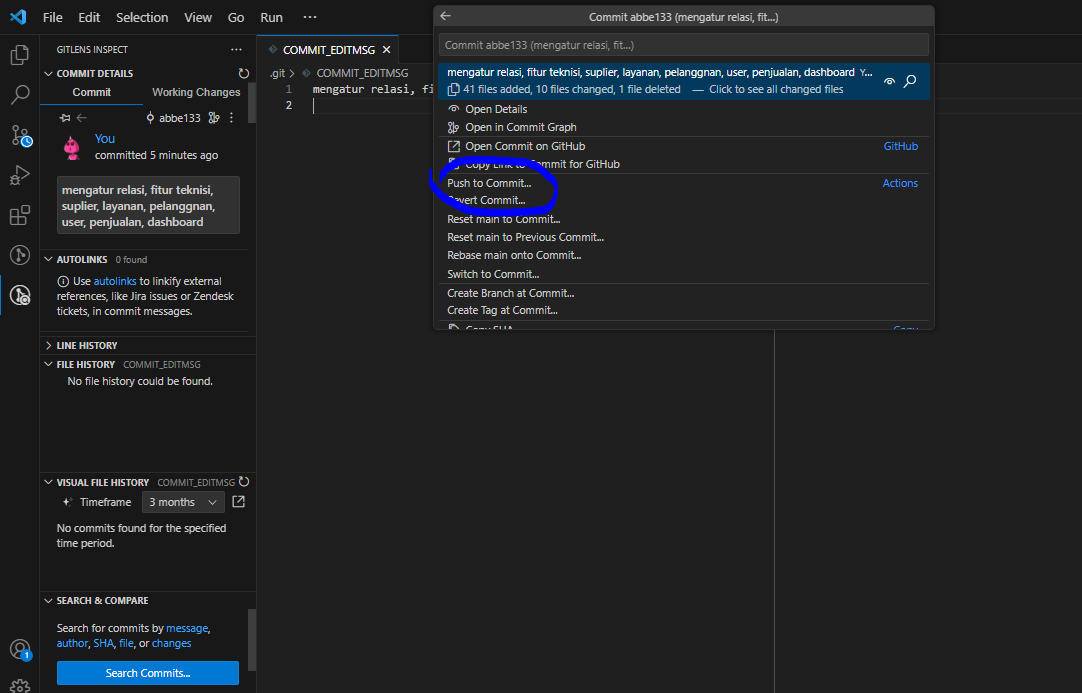


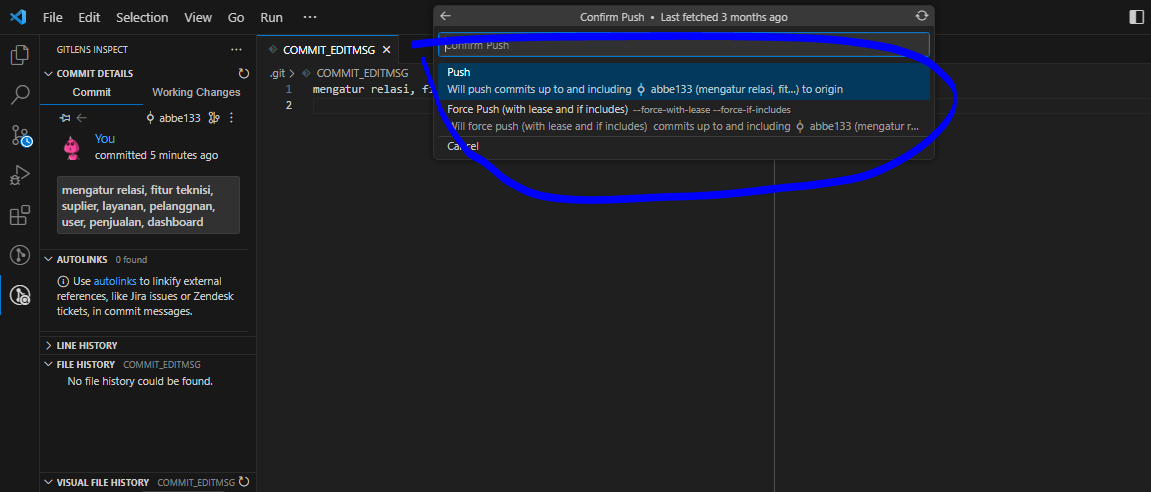
1. **Git push**

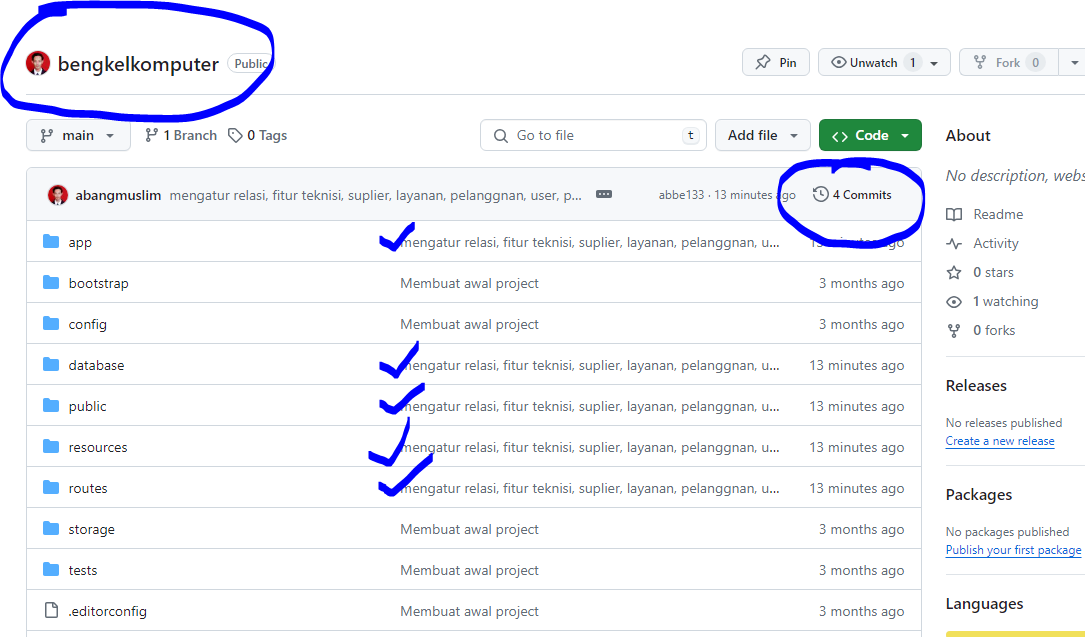
Melakukan git push . untuk melakukan update project kita ke github.

git push.

Agar lebih efektif dan effisien kita dapat menggunakan extensi git lens yang sudah terinstal di vs code.







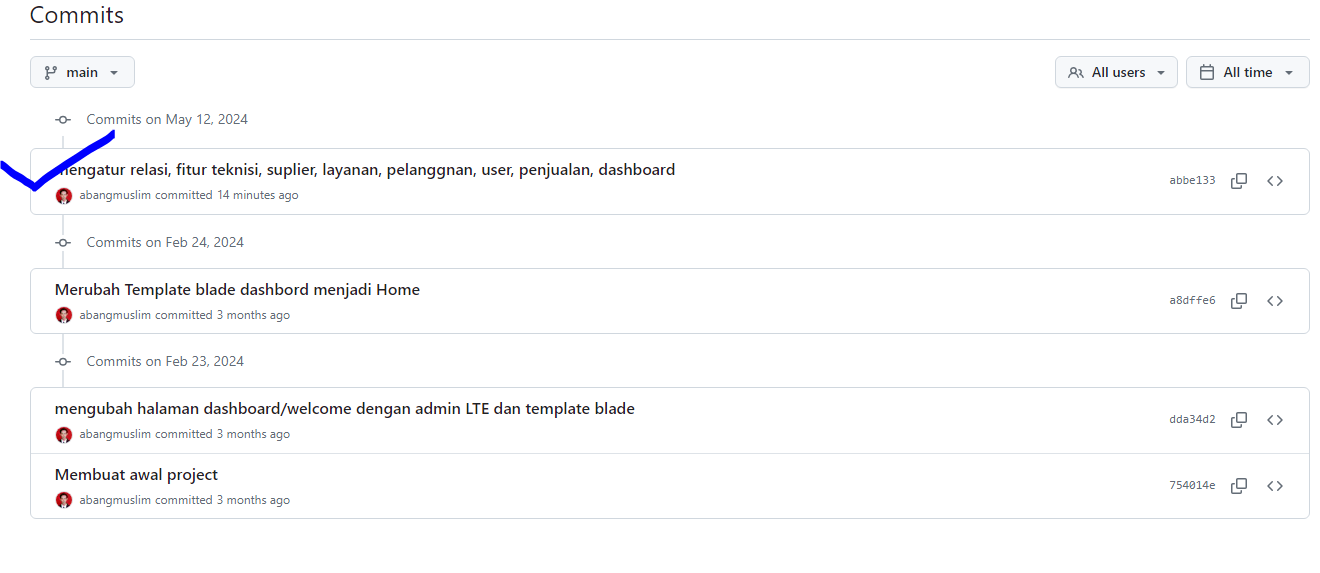
1. **Git log**

Melakukan git log . untuk melihat/check update file project kita ke github.

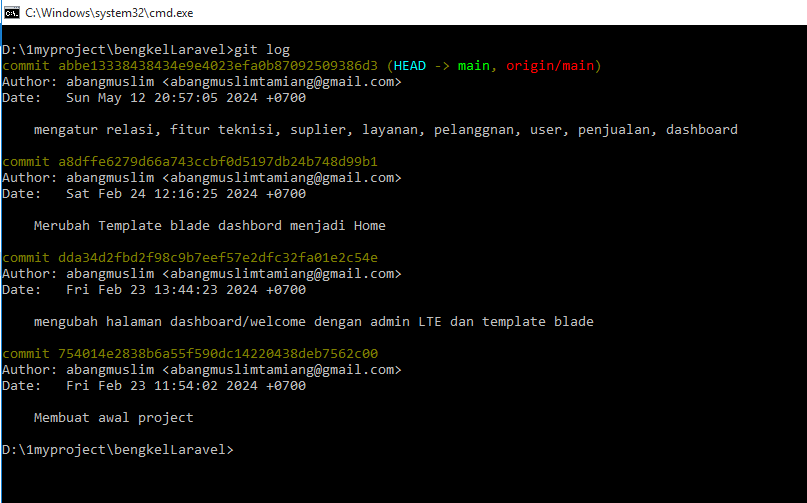
git log.

### Agar lebih efektif dan effisien kita dapat menggunakan extensi git lens yang sudah terinstal di vs code.

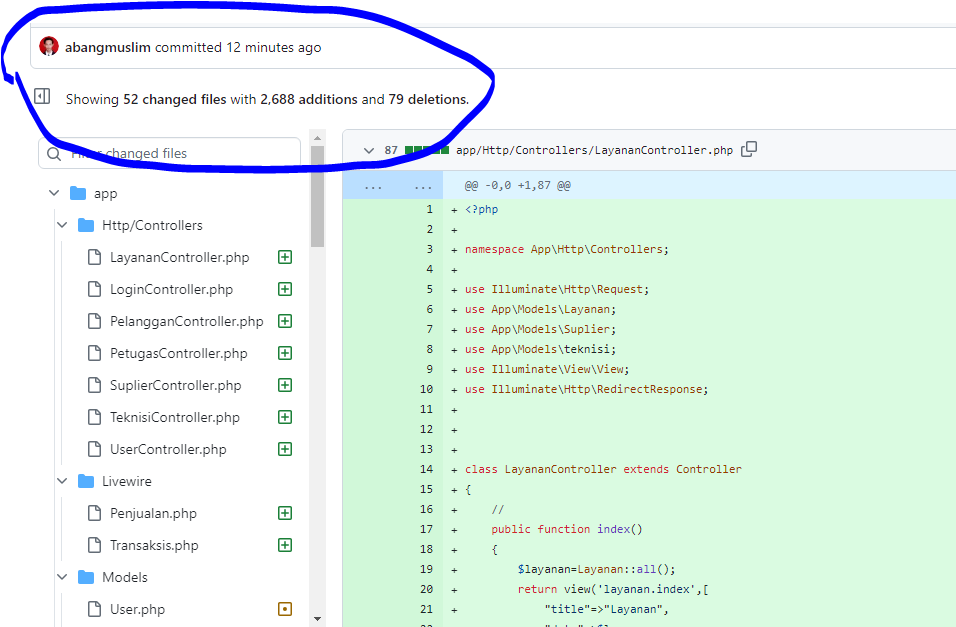
### Namun kita bisa juga langsung mengecek ke repository kita di github. Yaitu dengan mengklik bagian commit.



Dengan menggunakan git log di comand prompt dapat pula ita lihat riwayat commit yang telah kita lakukan



### Di menu commit pada github kita dapat juga melihat riwayat dan jumlah commit yang pernah kita buat, lengkap dengan tanggal dan keterangan dari penamaan commit tersebut.



1. **Menghapus Konfigurasi global remote**

### Menghapus configurasi username

git config --global --replace-all user.name "FName LName"

### Menghapus configurasi email

git config --global --replace-all user.email "email-id@domain.com"

### Menghapus configurasi remote origin

git remote remove origin

git remote set-url origin NEW\_URL

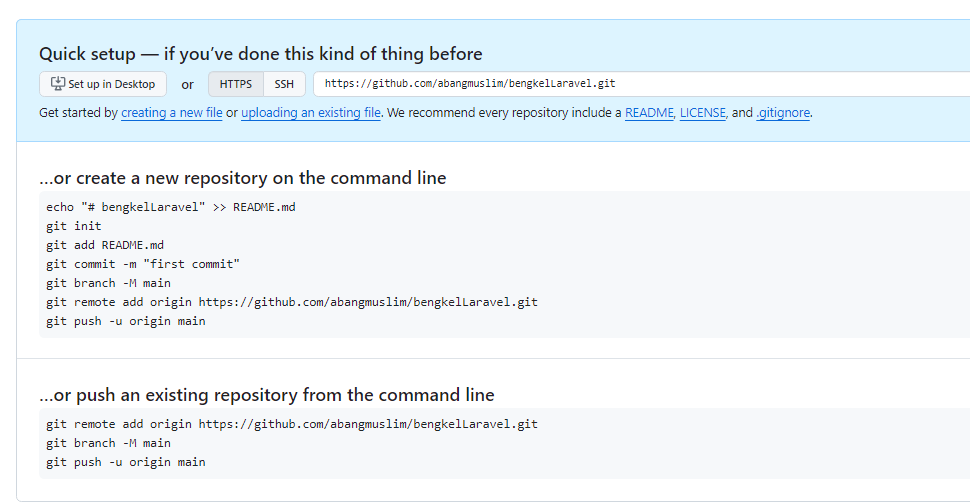
git remote set-url origin <https://github.com/abangmuslim/bengkelLaravel.git>

### paksa singkron ke master origin

git push origin master --force

1. **Merubah Repository dari Local Remote Global Config**

### Membuat repository baru, kemudian mengikuti comand/prompt yang tersedia



Hapus dulu git remote origin dengan:

git remote remove origin

### …or create a new repository on the command line

echo "# bengkelLaravel" >> README.md

git init

git add README.md

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/abangmuslim/bengkelLaravel.git

git push -u origin main

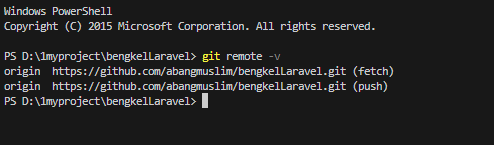
### …or push an existing repository from the command line

git remote add origin https://github.com/abangmuslim/bengkelLaravel.git

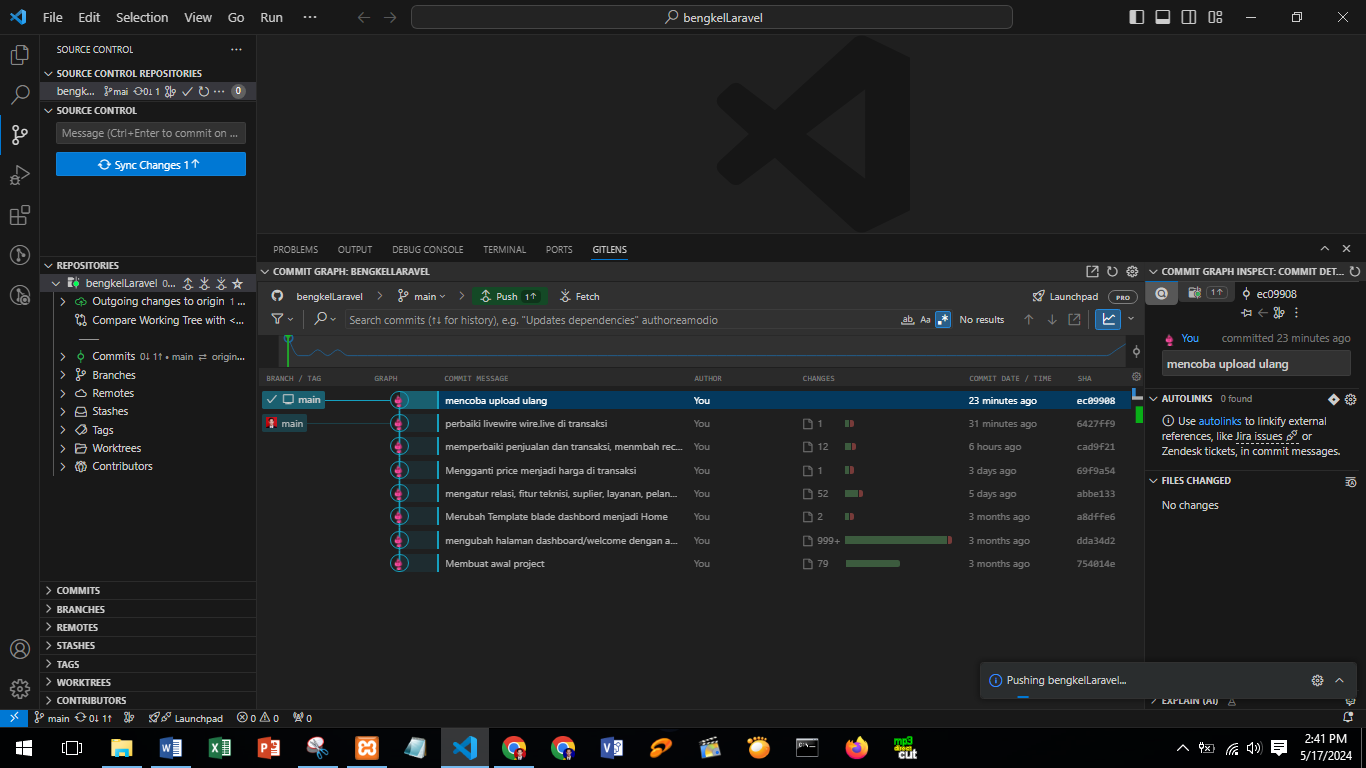
git branch -M main

git push -u origin main

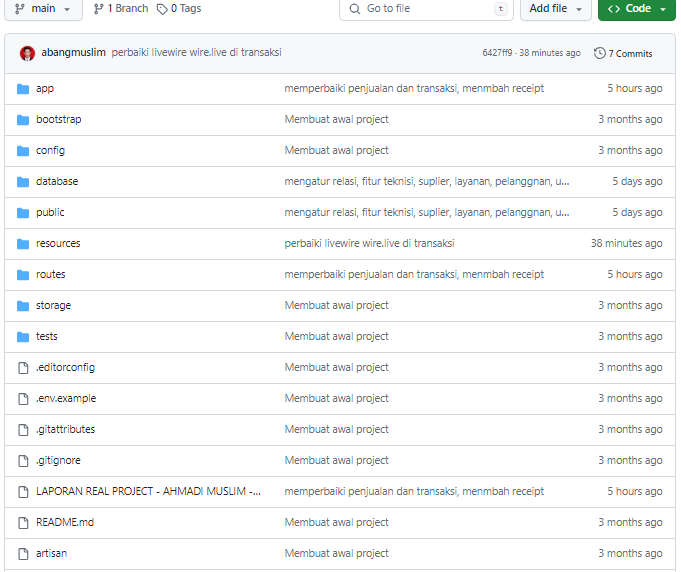
### Melihat remote origin main branch pada setingan project



### Melakukan singkronisasi pada menu **Repository** di bagian **Source Control** project



### Perubahan repository project dapat dilihat pada github

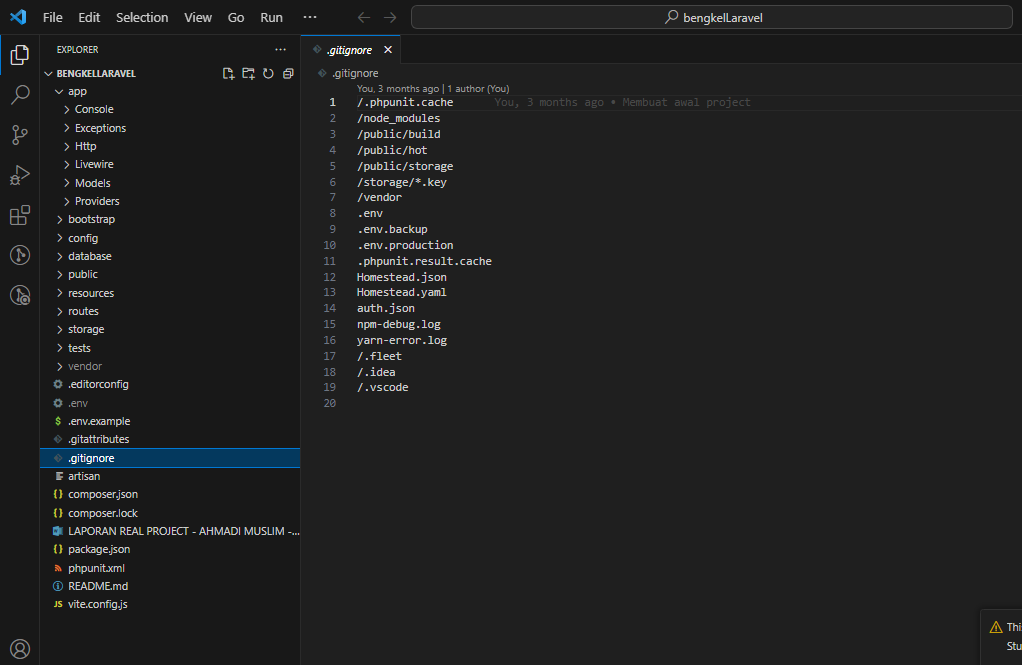


## Git Ignore

Git ignore adalah file yang ditinggalkan saat mengupdate/singkronisasi di github. Artinya file ini tidak akan ikut terupdate ke github.

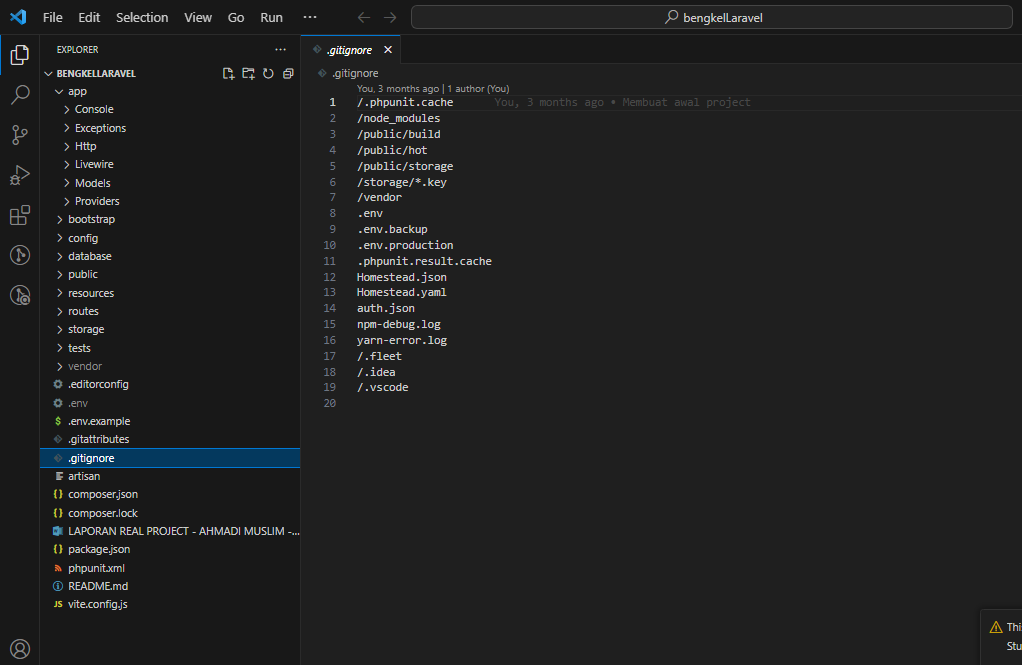
1. **Menambahkan File di Git Ignore**

### Membuat repository baru, kemudian mengikuti comand/prompt yang tersedia



1. **Menghapus File di Git Ignore**

### Membuat repository baru, kemudian mengikuti comand/prompt yang tersedia



1. **Clonning file dari github**

### Membuat repository baru, kemudian mengikuti comand/prompt yang tersedia

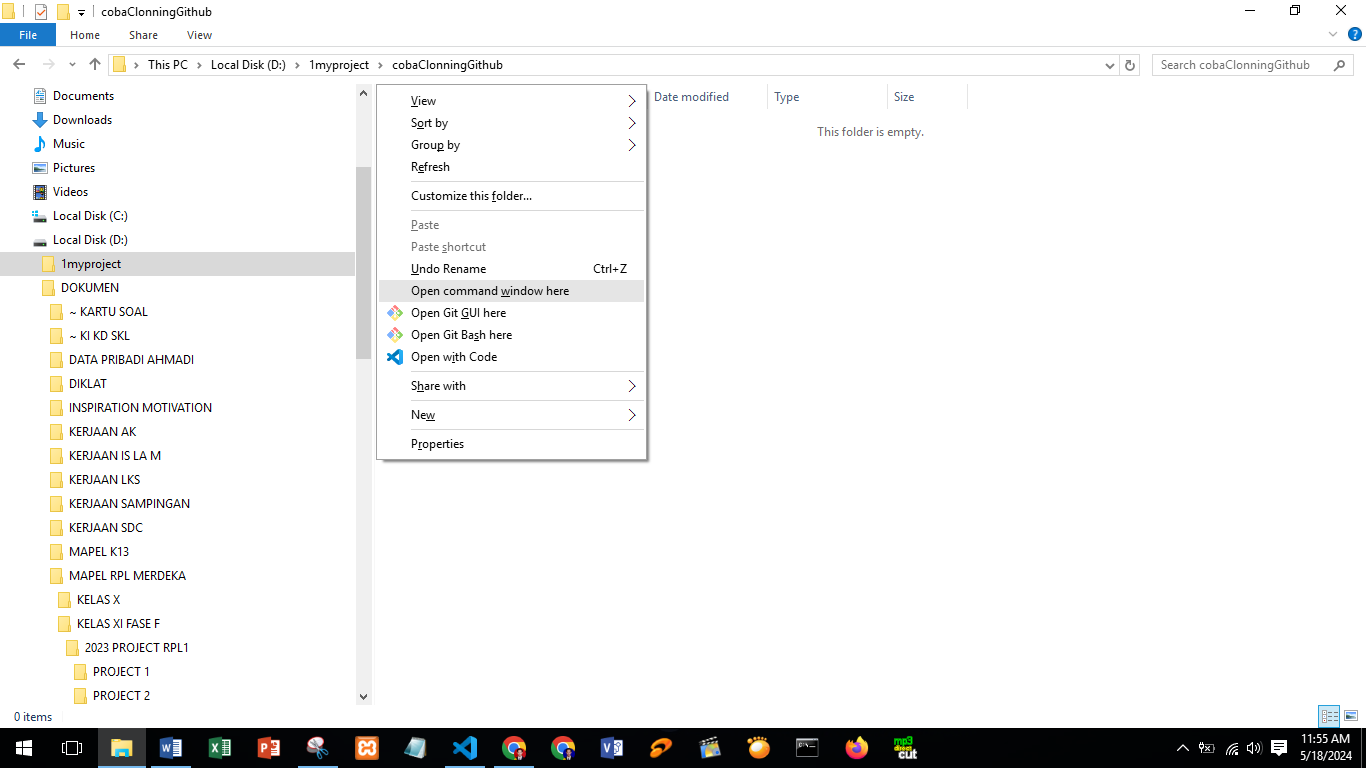
### Cara Mengkloning Repositori Github

### Dari Repositori Github, klik Klon

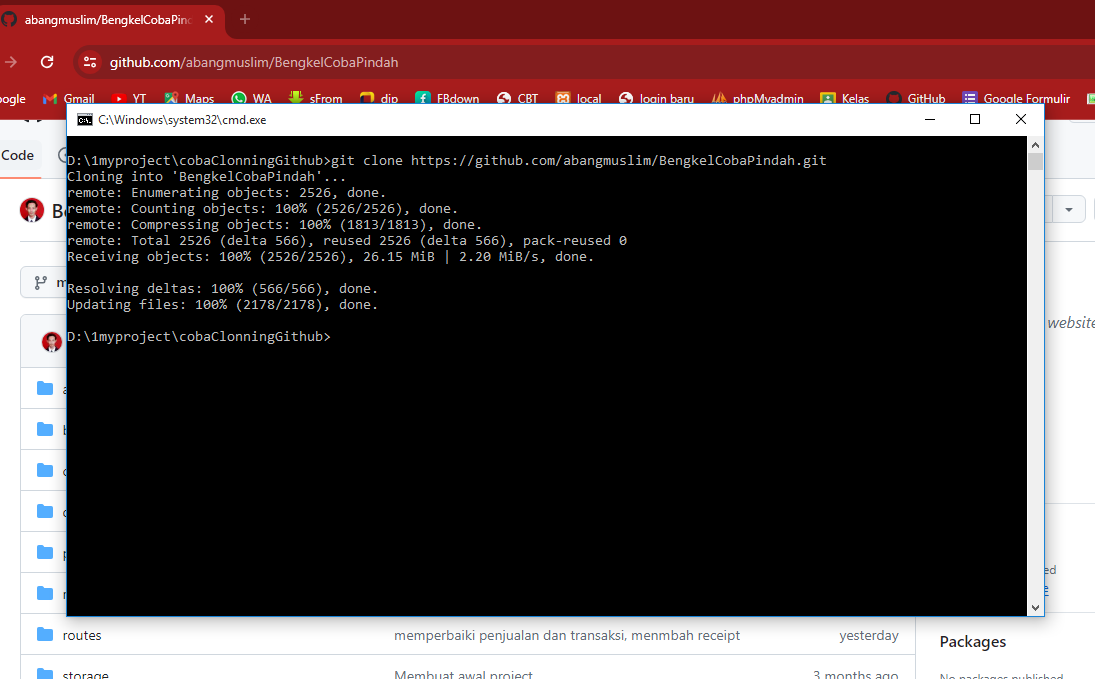
### Salin URL klon

### Di Terminal (Mac) atau baris perintah (Windows git bash), pindah ke direktori lokal

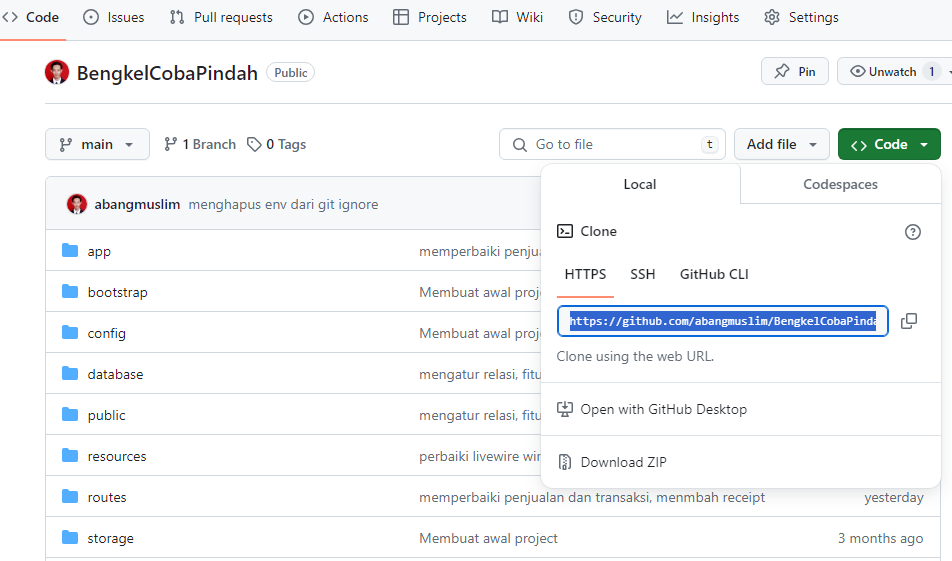
### Gunakan perintah git clone bersama dengan URL yang disalin



git clone https://github.com/abangmuslim/bengkelLaravel.git



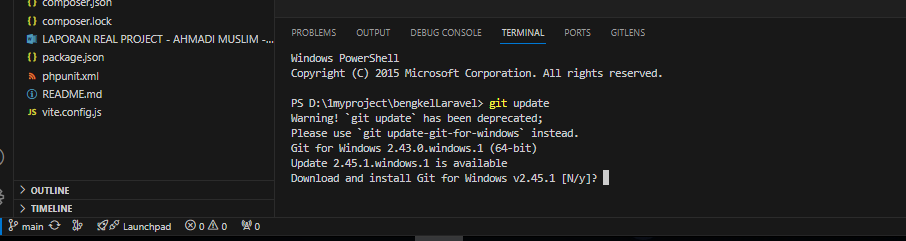




1. **Git Update / Git Instal**

### Membuat repository baru, kemudian mengikuti comand/prompt yang tersedia

git clone https://github.com/abangmuslim/bengkelLaravel.git



# BAB III TAMPILAN APLIKASI

## Alur Proses Pembuatan Fitur/Menu Pada Aplikasi

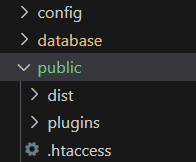
Berikut adalah urutan/tahapan yang dilakukan untuk pembuatan aplikasi. Urutan ini sangat mempengaruhi mana fitur yang lebih dahulu dibuat, sehingga proses pembuatan aplikasi dapat berjalan dengan baik.



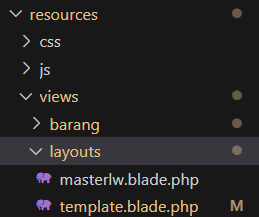
1. Fitur/menu pertama yang dibuat adalah **Dashboard /Home**, fitur ini perlu dibuat sebagai halaman/pangkalan utama (halaman sentral) yang akan diakses setelah login atau menjadi tempat kembali dari mengakses sebuah menu. Disamping itu dapat pula menjadi halaman rekap dan laporan ringkasan/resume dari aplikasi yang sedang dikerjakan.
2. Fitur/menu selanjutnya yang dibuat adalah **Barang** dan **Pelanggan**. Setelah menu suplier dan teknisi selesai data record dari menu barang dan pelanggan nantinya akan diakses oleh menu transaksi, sehingga menu transaksi nantinya tidak akan dapat berjalan normal jika menu barang dan pelanggan belum selesai.
3. Fitur/menu selanjutnya yang dibuat menu **User**. Menu user nantinya akan menjadi menu yang sangat krusial karena menjadi menu yang menangani data pengguna yang bertugas untuk menjalankan aplikasi, oleh karena itu data user pada menu ini akan dilengkapi dengan akun dan passwordnya. Disamping itu menu user nantinya akan dilengkapi dengan authentikasi melalui login untuk melindungi isi dari aplikasi sebagi kendali dan keamanan.
4. Fitur/menu selanjutnya yang dibuat adalah **Transaksi** dan **Detiltransaksi**. Kedua menu ini sebenarnya tergabung dalam 1 menu, yaitu penjualan. Namun dalam pengerjaan melibatkan dua buah tabel child yaitu tabel transaksi dan detiltransaksi.
5. Setelah semua fitur/menu selsai dibuat, maka perlu adanya sebuah **Receipt** atau nota cash bon untuk menjadi bukti/struck pembayaran untuk pelanggan dan penjual. Sebenarnya menu ini sudah include menjadi satu di dalam menu penjualan. Namun dalam pembuatannya menu ini dapat ditempatkan khusus agar setiap level user dapat melacak laporan dari transaksi penjualan yang sudah dijalankan melalui aplikasi.

## Fitur Utama / Dashboard

1. Copy 2 buah folder dari Admin LTE yaitu **plugins** dan **dist** ke dalam folder **public** yang terdapat di project yang telah kita kerjakan.
2. Sehingga di dalam folder public project kita terdapat 2 buah folder tambahan yaitu plugins dan dist.



1. Membuat folder baru dengan nama **layouts** di dalam folder **resources/views**.
2. Lalu buatlah file baru dengan nama **template.blade.php**.



1. Tuliskan script berikut pada file **template.blade.php**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <title>{{$title}}</title>

    <!-- Google Font: Source Sans Pro -->

    <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">

    <!-- Font Awesome Icons -->

    <link rel="stylesheet" href="{{ asset('plugins/fontawesome-free/css/all.min.css') }}">

    @yield('tambahanCSS')

    <!-- IonIcons -->

    <link rel="stylesheet" href="https://code.ionicframework.com/ionicons/2.0.1/css/ionicons.min.css">

    <!-- Theme style -->

    <link rel="stylesheet" href="{{ asset('dist/css/adminlte.min.css')}}">

</head>

<body class="hold-transition sidebar-mini" style="background-image: url('../backgroundlogin.jpg')">

    <div class="wrapper">

        <!-- Navbar -->

        <nav class="main-header navbar navbar-expand navbar-white navbar-light">

            <!-- Left navbar links -->

            <ul class="navbar-nav">

                <li class="nav-item">

                    <a class="nav-link" data-widget="pushmenu" href="#" role="button"><i class="fas fa-bars"></i></a>

                </li>

            </ul>

            <!-- Right navbar links -->

            <ul class="navbar-nav ml-auto">

                <li class="nav-item dropdown">

                    <a class="nav-link" data-toggle="dropdown" href="#">

                        <i class="far fa-user mr-2"></i>

                        <span class="badge badge-warning navbar-badge"></span>

                    </a>

                    <div class="dropdown-menu dropdown-menu-lg dropdown-menu-right">

                        <span class="dropdown-item dropdown-header">User Menu</span>

                        <div class="dropdown-divider"></div>

                        <a href="#" class="dropdown-item">

                            <i class="fas fa-user mr-2"></i>

                            <span class="float-right text-muted text-sm"></span>

                        </a>

                        <div class="dropdown-divider"></div>

                        <form action="logout" method="POST">

                            @csrf

                            <button type="submit" class="dropdown-item">

                                <i class="fas fa-sign-out-alt mr-2"></i>Logout

                            </button>

                        </form>

                    </div>

                </li>

                <li class="nav-item">

                    <a class="nav-link" data-widget="fullscreen" href="#" role="button">

                        <i class="fas fa-expand-arrows-alt"></i>

                    </a>

                </li>

            </ul>

        </nav>

        <!-- /.navbar -->

        <!-- Main Sidebar Container -->

        <aside class="main-sidebar sidebar-dark-primary elevation-4">

            <!-- Brand Logo -->

            <a href="index3.html" class="brand-link">

                <img src="{{ asset('dist/img/AdminLTELogo.png')}}" alt="AdminLTE Logo" class="brand-image img-circle elevation-3" style="opacity: .8">

                <span class="brand-text font-weight-light">POS</span>

            </a>

            <!-- Sidebar -->

            <div class="sidebar">

                <!-- Sidebar user panel (optional) -->

                <div class="user-panel mt-3 pb-3 mb-3 d-flex">

                    <div class="image">

                        <img src="{{ asset('dist/img/user2-160x160.jpg')}}" class="img-circle elevation-2" alt="User Image">

                    </div>

                    <div class="info">

                        <a href="#" class="d-block">Indah</a>

                    </div>

                </div>

                <!-- Sidebar Menu -->

                <nav class="mt-2">

                    <ul class="nav nav-pills nav-sidebar flex-column" data-widget="treeview" role="menu" data-accordion="false">

                        <!-- Add icons to the links using the .nav-icon class

               with font-awesome or any other icon font library -->

                        <li class="nav-item">

                            <a href="/" class="nav-link {{ ($title==='welcome')?'active':'' }}">

                                <i class="nav-icon fas fa-tachometer-alt"></i>

                                <p>

                                    Dashboard

                                </p>

                            </a>

                        </li>

                        <li class="nav-item">

                            <a href="#" class="nav-link {{($title==='barang')?'active':''}}">

                                <i class="nav-icon fas fa-boxes"></i>

                                <p>

                                    Barang

                                    <i class="fas fa-angle-left right"></i>

                                </p>

                            </a>

                            <ul class="nav nav-treeview">

                                <li class="nav-item">

                                    <a href="{{route('barang.index')}}" class="nav-link">

                                        <i class="far fa-circle nav-icon"></i>

                                        <p>Input Barang</p>

                                    </a>

                                </li>

                                <li class="nav-item">

                                    <a href="#" class="nav-link">

                                        <i class="far fa-circle nav-icon"></i>

                                        <p>Laporan Barang</p>

                                    </a>

                                </li>

                            </ul>

                        </li>

                        <li class="nav-item">

                            <a href="{{ route('pelanggan.index')}}" class="nav-link {{ ($title==='Pelanggan')?'active':''}}">

                                <i class="nav-icon fas fa-users"></i>

                                <p>

                                    pelanggan

                                </p>

                            </a>

                        </li>

                        <li class="nav-item">

                            <a href="{{ route('pengguna.index')}}" class="nav-link {{ ($title==='Pengguna')?'active':''}}">

                                <i class="nav-icon fas fa-users"></i>

                                <p>

                                    Users

                                </p>

                            </a>

                        </li>

                        <li class="nav-item">

                            <a href="penjualan" target="\_blank" class="nav-link {{ ($title==='Penjualan') ? 'active':'' }}">

                                <i class="nav-icon fas fa-cash-register"></i>

                                <p>

                                    Penjualan

                                </p>

                            </a>

                        </li>

                        <li class="nav-item">

                            <a href="#" class="nav-link">

                                <i class="nav-icon fas fa-table"></i>

                                <p>

                                    Laporan

                                    <i class="fas fa-angle-left right"></i>

                                </p>

                            </a>

                            <ul class="nav nav-treeview">

                                <li class="nav-item">

                                    <a href="pages/tables/simple.html" class="nav-link">

                                        <i class="far fa-circle nav-icon"></i>

                                        <p>Harian</p>

                                    </a>

                                </li>

                                <li class="nav-item">

                                    <a href="pages/tables/data.html" class="nav-link">

                                        <i class="far fa-circle nav-icon"></i>

                                        <p>Bulanan</p>

                                    </a>

                                </li>

                                <li class="nav-item">

                                    <a href="pages/tables/jsgrid.html" class="nav-link">

                                        <i class="far fa-circle nav-icon"></i>

                                        <p>Tahunan</p>

                                    </a>

                                </li>

                            </ul>

                        </li>

                    </ul>

                </nav>

                <!-- /.sidebar-menu -->

            </div>

            <!-- /.sidebar -->

        </aside>

        <!-- Content Wrapper. Contains page content -->

        <div class="content-wrapper">

            <!-- Content Header (Page header) -->

            <div class="content-header">

                <div class="container-fluid">

                    <div class="row mb-2">

                        <div class="col-sm-12">

                            <h1 class="m-0">@yield('judulh1')</h1>

                        </div><!-- /.col -->

                    </div><!-- /.row -->

                </div><!-- /.container-fluid -->

            </div>

            <!-- /.content-header -->

            <!-- Main content -->

            <div class="content">

                <div class="container-fluid">

                    <div class="row">

                        @yield('konten')

                    </div>

                    <!-- /.row -->

                </div>

                <!-- /.container-fluid -->

            </div>

            <!-- /.content -->

        </div>

        <!-- /.content-wrapper -->

        <!-- Control Sidebar -->

        <aside class="control-sidebar control-sidebar-dark">

            <!-- Control sidebar content goes here -->

        </aside>

        <!-- /.control-sidebar -->

    </div>

    <!-- ./wrapper -->

    <!-- REQUIRED SCRIPTS -->

    <!-- jQuery -->

    <script src="{{ asset('plugins/jquery/jquery.min.js')}}"></script>

    <!-- Bootstrap -->

    <script src="{{ asset('plugins/bootstrap/js/bootstrap.bundle.min.js')}}"></script>

    @yield('tambahanJS')

    <!-- AdminLTE -->

    <script src="{{ asset('dist/js/adminlte.js')}}"></script>

    <!-- OPTIONAL SCRIPTS -->

    <script src="{{ asset('plugins/chart.js/Chart.min.js')}}"></script>

    <!-- Control Sidebar -->

    <aside class="control-sidebar control-sidebar-dark">

        <!-- Control sidebar content goes here -->

        <!-- Bootstrap 4 -->

        <script src="{{ asset('plugins/bootstrap/js/bootstrap.bundle.min.js')}}"></script>

        <!-- AdminLTE App -->

        <script src="{{ asset('dist/js/adminlte.min.js')}}"></script>

        <!-- AdminLTE for demo purposes -->

        <script src="{{ asset('dist/js/demo.js')}}"></script>

    </aside>

    <!-- /.control-sidebar -->

    </div>

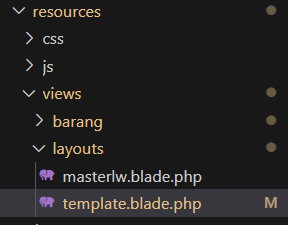
    @yield('tambahScript')

</body>

</html>

Catatan : script di atas bisa didapatkan dari file index.php yang terdapat pada folder Admin LTE, selanjutnya dimodifikasi sehingga menghasilkan tampilan seperti di atas.

1. Selanjutnya copy script berikut ke file **welcome.blade.php** pada folder **resources/views**.



@extends('layouts.template')

@section('judulh1','Admin - Dashboard')

@section('konten')

<div class="container-fluid">

    <!-- =========================================================== -->

    <h5 class="mb-2">Info Box With Custom Shadows <small><i>Using Bootstrap's Shadow Utility</i></small></h5>

    <div class="row">

        <div class="col-md-3 col-sm-6 col-12">

            <div class="info-box shadow-none">

                <span class="info-box-icon bg-info"><i class="far fa-envelope"></i></span>

                <div class="info-box-content">

                    <span class="info-box-text">Shadows</span>

                    <span class="info-box-number">None</span>

                </div>

                <!-- /.info-box-content -->

            </div>

            <!-- /.info-box -->

        </div>

        <!-- /.col -->

        <div class="col-md-3 col-sm-6 col-12">

            <div class="info-box shadow-sm">

                <span class="info-box-icon bg-success"><i class="far fa-flag"></i></span>

                <div class="info-box-content">

                    <span class="info-box-text">Shadows</span>

                    <span class="info-box-number">Small</span>

                </div>

                <!-- /.info-box-content -->

            </div>

            <!-- /.info-box -->

        </div>

        <!-- /.col -->

        <div class="col-md-3 col-sm-6 col-12">

            <div class="info-box shadow">

                <span class="info-box-icon bg-warning"><i class="far fa-copy"></i></span>

                <div class="info-box-content">

                    <span class="info-box-text">Shadows</span>

                    <span class="info-box-number">Regular</span>

                </div>

                <!-- /.info-box-content -->

            </div>

            <!-- /.info-box -->

        </div>

        <!-- /.col -->

        <div class="col-md-3 col-sm-6 col-12">

            <div class="info-box shadow-lg">

                <span class="info-box-icon bg-danger"><i class="far fa-star"></i></span>

                <div class="info-box-content">

                    <span class="info-box-text">Shadows</span>

                    <span class="info-box-number">Large</span>

                </div>

                <!-- /.info-box-content -->

            </div>

            <!-- /.info-box -->

        </div>

        <!-- /.col -->

    </div>

    <!-- /.row -->

    <!-- =========================================================== -->

    <h5 class="mt-4 mb-2">Info Box With <code>bg-gradient-\*</code></h5>

    <div class="row">

        <div class="col-md-3 col-sm-6 col-12">

            <div class="info-box bg-gradient-info">

                <span class="info-box-icon"><i class="far fa-bookmark"></i></span>

                <div class="info-box-content">

                    <span class="info-box-text">Bookmarks</span>

                    <span class="info-box-number">41,410</span>

                    <div class="progress">

                        <div class="progress-bar" style="width: 70%"></div>

                    </div>

                    <span class="progress-description">

                        70% Increase in 30 Days

                    </span>

                </div>

                <!-- /.info-box-content -->

            </div>

            <!-- /.info-box -->

        </div>

        <!-- /.col -->

        <div class="col-md-3 col-sm-6 col-12">

            <div class="info-box bg-gradient-success">

                <span class="info-box-icon"><i class="far fa-thumbs-up"></i></span>

                <div class="info-box-content">

                    <span class="info-box-text">Likes</span>

                    <span class="info-box-number">41,410</span>

                    <div class="progress">

                        <div class="progress-bar" style="width: 70%"></div>

                    </div>

                    <span class="progress-description">

                        70% Increase in 30 Days

                    </span>

                </div>

                <!-- /.info-box-content -->

            </div>

            <!-- /.info-box -->

        </div>

        <!-- /.col -->

        <div class="col-md-3 col-sm-6 col-12">

            <div class="info-box bg-gradient-warning">

                <span class="info-box-icon"><i class="far fa-calendar-alt"></i></span>

                <div class="info-box-content">

                    <span class="info-box-text">Events</span>

                    <span class="info-box-number">41,410</span>

                    <div class="progress">

                        <div class="progress-bar" style="width: 70%"></div>

                    </div>

                    <span class="progress-description">

                        70% Increase in 30 Days

                    </span>

                </div>

                <!-- /.info-box-content -->

            </div>

            <!-- /.info-box -->

        </div>

        <!-- /.col -->

        <div class="col-md-3 col-sm-6 col-12">

            <div class="info-box bg-gradient-danger">

                <span class="info-box-icon"><i class="fas fa-comments"></i></span>

                <div class="info-box-content">

                    <span class="info-box-text">Comments</span>

                    <span class="info-box-number">41,410</span>

                    <div class="progress">

                        <div class="progress-bar" style="width: 70%"></div>

                    </div>

                    <span class="progress-description">

                        70% Increase in 30 Days

                    </span>

                </div>

                <!-- /.info-box-content -->

            </div>

            <!-- /.info-box -->

        </div>

        <!-- /.col -->

    </div>

    <!-- /.row -->

</div><!-- /.container-fluid -->

@endsection

1. Selanjutnya tambahkan script berikut pada routes/web.php.

Route::get('/',function(){

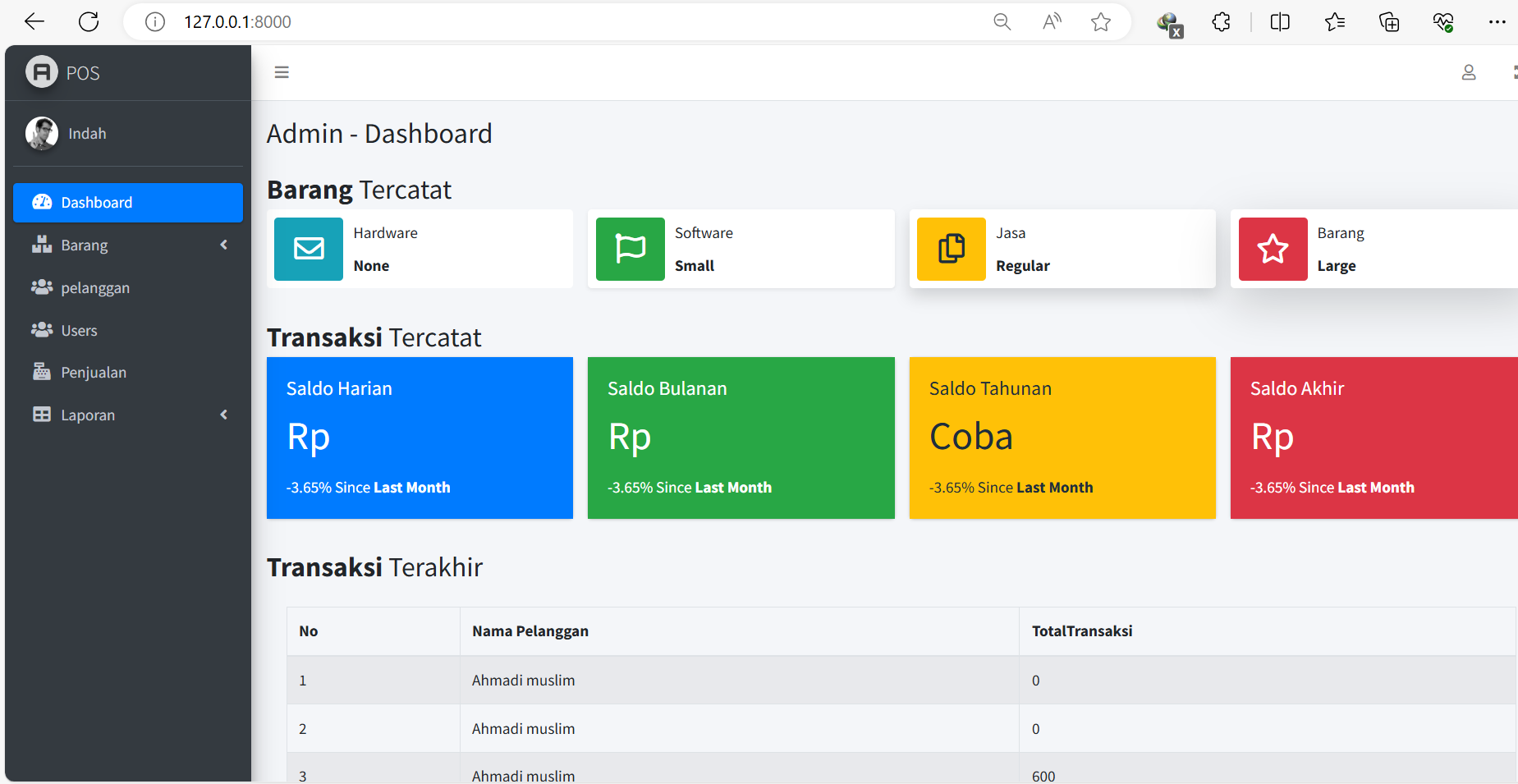
    return view('welcome',[

        "title"=>"Dashboard"

    ]);

});

1. Jalankan server kemudian silahkan buka alamat <http://127.0.0.1:8000/> atau localhost:8000



## Fitur Barang

### **Membuat LayananController dan Modifikasi Model Layanan dan Model Category**

* Buatlah controller layanan dengan cara menjalankan perintah php artisan berikut:

php artisan make:controller BarangController

* Jika sudah berhasil cek kembali **LayananController.php** pada **App\Http\Controller**

### 

### **Membuat Route Layanan dan Method index()**

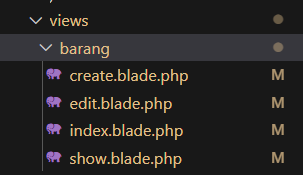
* Buka file **routes\web.php** lalu tambahkan Route berikut:

Route::resource('layanan',BarangController::class;

* Jangan lupa untuk meng-import class **BarangController** ke dalam **routes/web.php**.

use App\Http\Controllers\LayananController;

* Kemudian buat sebuah folder di dalam folder views dengan nama layanan, lalu buat 4 file dengan nama **index.blade.php**, **create.blade.php**, **edit.blade.php**, dan **show.blade.php**.



Kalau diperhatikan pada bagian layanan ini kita mulai menggunakan penamaan standar yang biasanya digunakan dalam project laravel. File **index**, akan menjadi tampilan utama dari layanan, kemudian file **create** adalah tampilan untuk menambahkan layanan, file **edit** untuk merubah/update layanan, sedangkan file **show** untuk menampilkan detail layanan.

* Buka file index.blade.php lalu tambahkan script berikut:

@extends('layouts.template')

@section('tambahanCSS')

<!-- DataTables -->

<link rel="stylesheet" href="plugins/datatables-bs4/css/dataTables.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-responsive/css/responsive.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-buttons/css/buttons.bootstrap4.min.css">

<!-- Toastr -->

<link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.css">

@endsection

@section('judulh1','Admin - Barang')

@section('konten')

<div class="col-md-12">

    <div class="card card-info">

        <div class="card-header">

            <h2 class="card-title">Data Barang</h2>

            <a type="button" class="btn btn-success float-right" href="{{ route('barang.create') }}">

                <i class=" fas fa-plus"></i> Tambah Barang

            </a>

        </div>

        <!-- /.card-header -->

        <div class="card-body">

            <table id="example1" class="table table-bordered table-striped ">

                <thead>

                    <tr>

                        <th>No</th>

                        <th>Barang</th>

                        <th>Stock</th>

                        <th>Harga</th>

                        <th>Aksi</th>

                    </tr>

                </thead>

                <tbody>

                    @foreach($data as $dt)

                    <tr>

                        <td>{{ $loop->iteration }}</td>

                        <td>{{ $dt->barang }}</td>

                        <td>{{ $dt->stock }}</td>

                        <td>{{ $dt->harga }}</td>

                        <td>

                            <div class="btn-group">

                                <form action="{{ route('barang.destroy',$dt->id)}}" method="POST">

                                    @csrf

                                    @method('DELETE')

                                    <button type="submit" class="btn btn-danger">

                                        <i class=" fas fa-trash"></i>

                                    </button>

                                </form>

                                <a type="button" class="btn btn-warning" href="{{ route('barang.edit',$dt->id) }}">

                                    <i class=" fas fa-edit"></i>

                                </a>

                                <a type="button" class="btn btn-success" href="{{ route('barang.show',$dt->id) }}">

                                    <i class=" fas fa-eye"></i>

                                </a>

                            </div>

                        </td>

                    </tr>

                    @endforeach

                </tbody>

            </table>

        </div>

    </div>

</div>

@endsection

* Buka file **template.blade.php** pada folder **views\layouts**. Lalu tambahkan script yang ditandai berikut:

                                <li class="nav-item">

                                    <a href="{{route('barang.index')}}" class="nav-link">

                                        <i class="far fa-circle nav-icon"></i>

                                        <p>Input Barang</p>

                                    </a>

                                </li>

* Buka file **BarangController.php**, lalu tambahkan Method index() di dalam class BarangController.

public function index()

    {

        $barang=Barang::all();

        return view('barang.index',[

            "title"=>"barang",

            "data"=>$barang

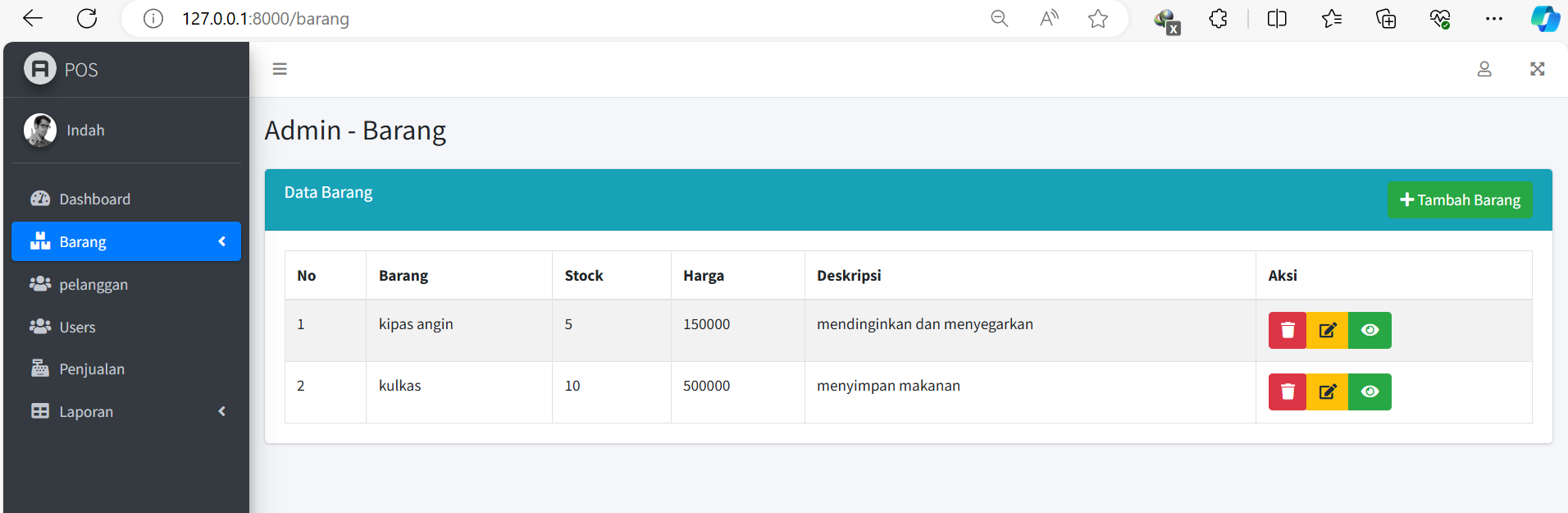
        ]);

    }

* Jangan lupa untuk import model Layanan

use App\Models\Barang;

* Uji coba yang telah dikerjakan.



### **Membuat Method create() dan store()**

* Jika diperhatikan dari manajemen customer sebelumnya, method create() digunakan untuk menampilkan tampilan untuk menambah data sementara method store() digunakan untuk menyimpan data ke dalam database.
* Memperbaiki Controller Barang pada file BarangController.php

<?php

namespace App\Http\Controllers;

use App\Models\Barang;

use Illuminate\Http\Request;

use Illuminate\View\View;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Redirect;

class BarangController extends Controller

{

    //

    public function index()

    {

        $barang=Barang::all();

        return view('barang.index',[

            "title"=>"barang",

            "data"=>$barang

        ]);

    }

    public function create():View

    {

        return view('barang.create')->with([

            "title" => "Tambah Data Barang" ]);

    }

    public function store(Request $request):RedirectResponse

    {

        $request->validate([

            "barang"=>"required",

            "stock"=>"required",

            "harga"=>"nullable",

            "description"=>"required",

        ]);

        Barang::create($request->all());

        return redirect()->route('barang.index')->with('success','Data Barang Berhasil Ditambahkan');

    }

    public function edit(barang $barang):View

    {

        return view('barang.edit',compact('barang'))->with([

            "title"=> "Ubah Data Barang",

        ]);

    }

    public function update(Barang $barang, Request $request):RedirectResponse

    {

        $request->validate([

            "barang"=>"required",

            "stock"=>"required",

            "harga"=>"nullable",

            "description"=>"required"

        ]);

        $barang->update($request->all());

        return redirect()->route('barang.index')->with('updated','Data Barang Berhasil Diubah');

    }

    public function show():View

    {

        $barang=Barang::all();

        return view('barang.show')->with([

            "title"=> "Tampil Data Barang",

            "data"=>$barang

        ]);

    }

    public function destroy($id):RedirectResponse

    {

        Barang::where('id',$id)->delete();

        return redirect()->route('barang.index')->with('deleted','Data Barang Berhasil Dihapus');

    }

}

* Buka file create.blade.php pada folder views\layanan. Tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Barang')

@section('konten')

<div class="col-md-6">

    @if ($errors->any())

    <div class="alert alert-danger">

        <strong>Whoops!</strong> There were some problems with your input.<br><br>

        <ul>

            @foreach ($errors->all() as $error)

            <li>{{ $error }}</li>

            @endforeach

        </ul>

    </div>

    @endif

    <div class="card card-success">

        <div class="card-header">

            <h3 class="card-title">Tambah Data Barang</h3>

        </div>

        <!-- /.card-header -->

        <!-- form start -->

        <form action="{{ route('barang.store') }}" method="POST">

            @csrf

            <div class=" card-body">

                <div class="form-group">

                    <label for="nama">Barang</label>

                    <input type="text" class="form-control" id="barang" name="barang">

                </div>

                <div class="form-group">

                    <label for="stock">Stock</label>

                    <input type="number" class="form-control" id="stock" name="stock">

                </div>

                <div class="form-group">

                    <label for="harga">Harga</label>

                    <input type="number" class="form-control" id="harga" name="harga">

                </div>

                <div class="form-group">

                    <label for="description">Deskripsi</label>

                    <textarea id="description" name="description" class=" form-control" rows="4"></textarea>

                </div>

            </div>

            <!-- /.card-body -->

            <div class="card-footer">

                <button type="submit" class="btn btn-success float-right">Simpan</button>

            </div>

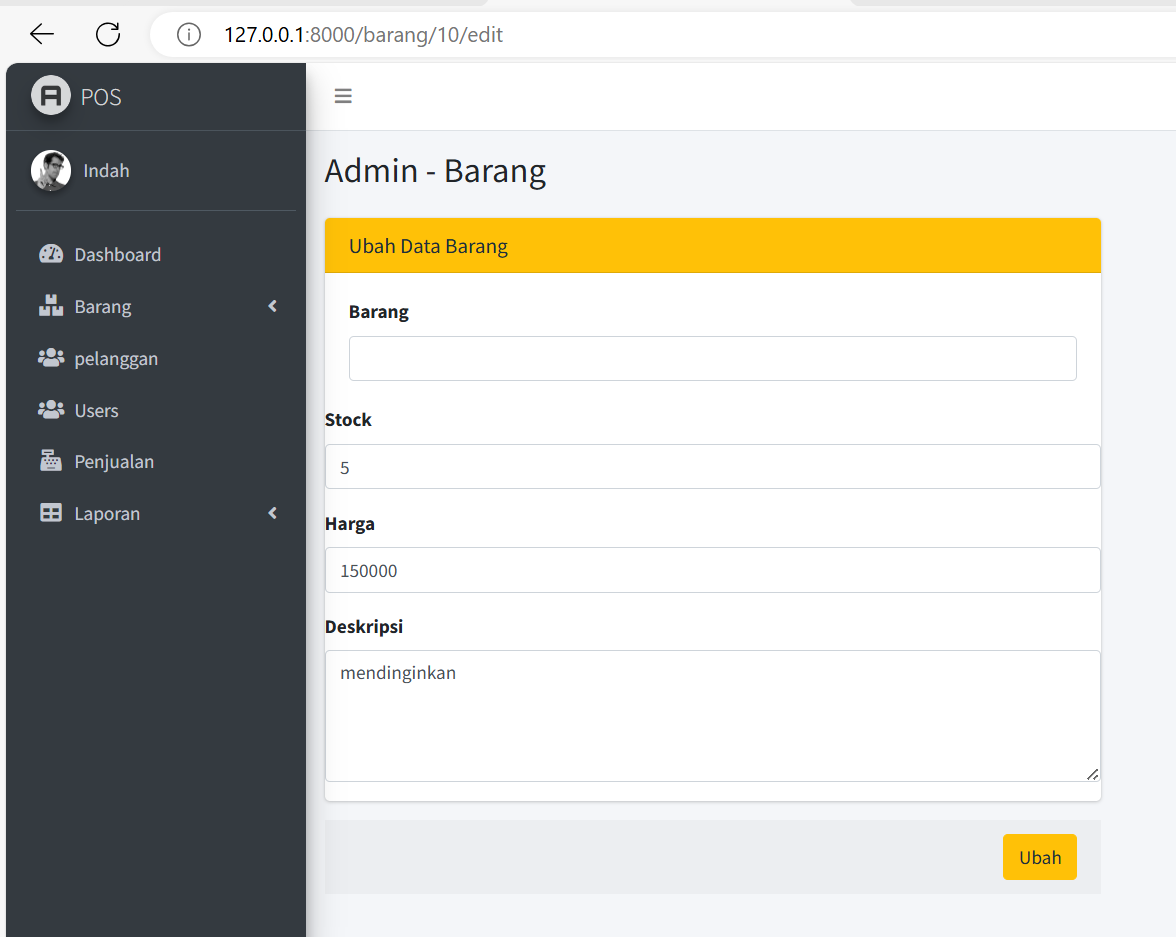
        </form>

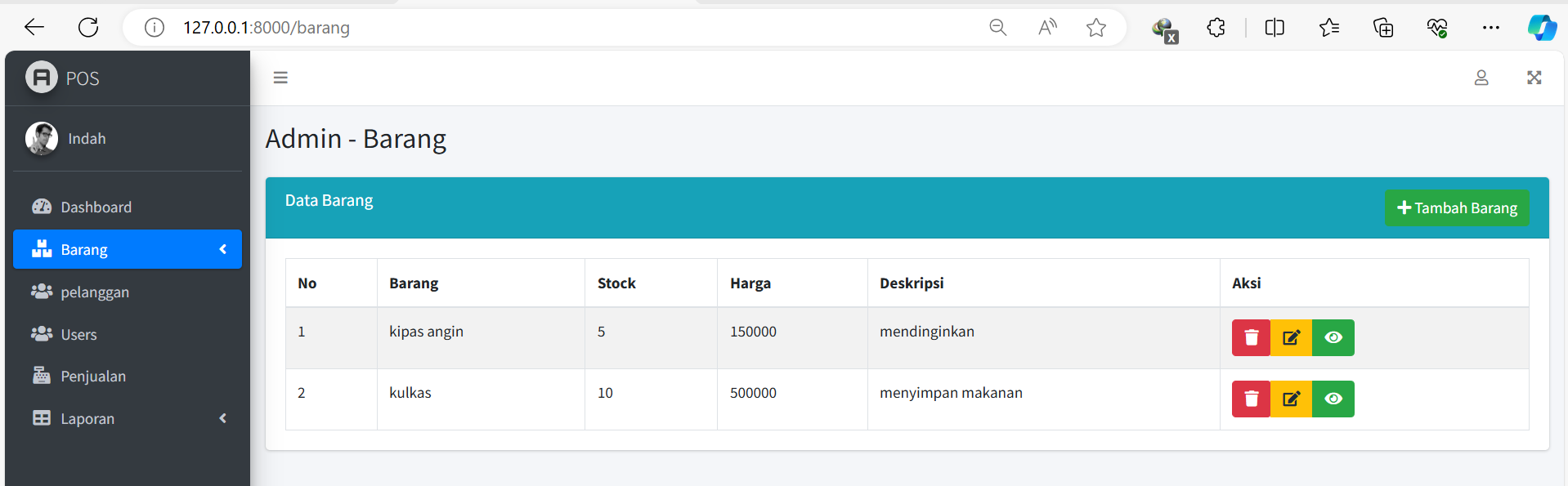
    </div>

</div>

@endsection

* Uji coba penambahan data

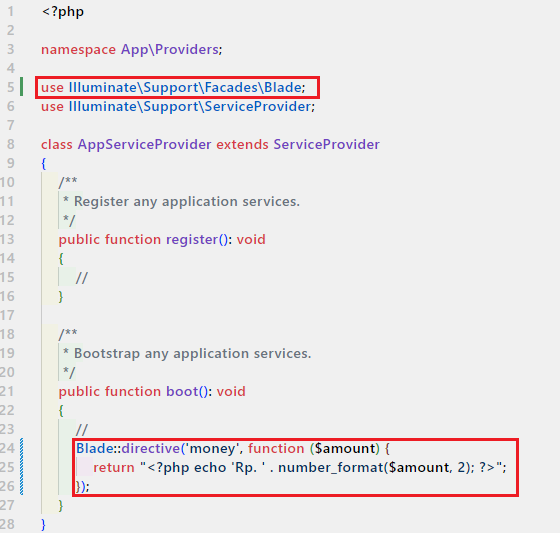




Apakah sudah berhasil ? Jika sudah perhatikan bagian Barang. Selanjutnya perhatikan kolom harga, seharusnya yang sudah dikerjakan tidak muncul format mata uang. Lalu bagaimana caranya agar muncul format uangnya. Untuk melakukan hal tersebut kita harus membuat fungsi baru.

### **Membuat Format Uang**

* Buka file AppServiceProvider.php pada folder app\Providers.
* Tambahkan script berikut:



public function boot(): void

    {

        //

        Blade::directive('money', function ($amount) {

            return "<?php echo 'Rp. ' . number\_format($amount, 2); ?>";

        });

    }

Lalu tambahkan:

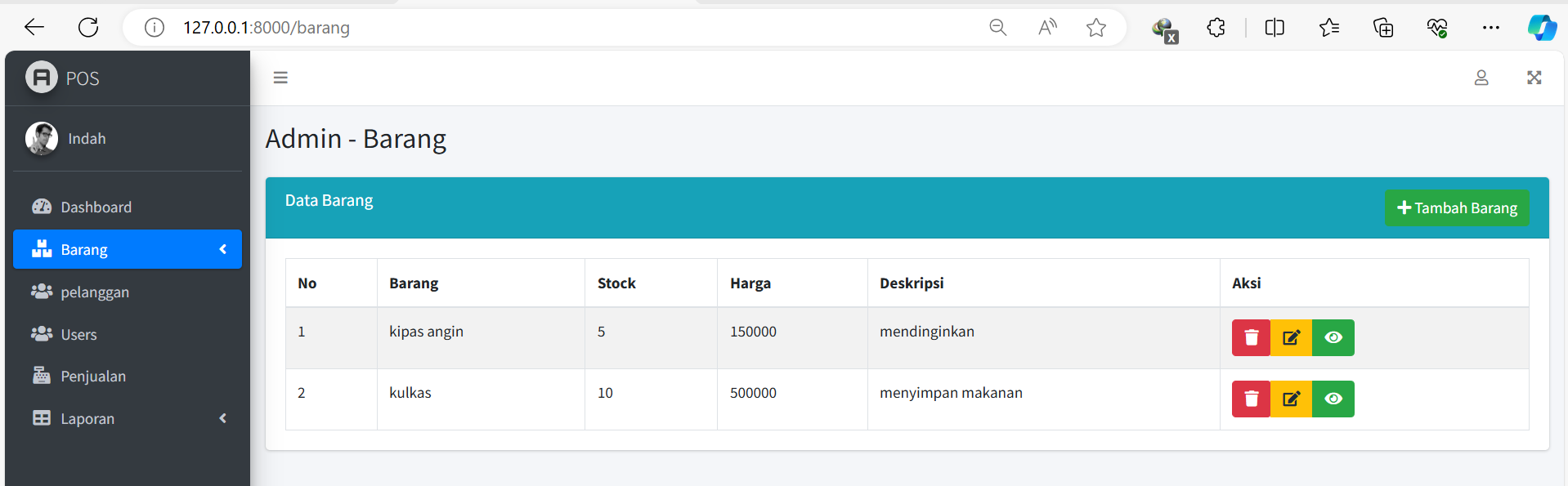
use Illuminate\Support\Facades\Blade;

* Rubah file index.blade.php yang berada di dalam folder views\layanans



<td>@money($dt->harga) </td>

* Buka kembali aplikasi pos di browser, silahkan refresh. Apakah sudah muncul format uang?



### **Membuat Method edit() dan Method update()**

* Method edit digunakan untuk menampilkan tampilan edit layanan sedangkan method update digunakan untuk merubah record pada tabel layanans di database.
* Tambahkan method edit() pada class BarangController.

 public function edit(barang $barang):View

    {

        return view('barang.edit',compact('barang'))->with([

            "title"=> "Ubah Data Barang",

        ]);

    }

* Buka file edit.blade.php lalu tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Barang')

@section('konten')

<div class="col-md-6">

    @if ($errors->any())

    <div class="alert alert-danger">

        <strong>Whoops!</strong> There were some problems with your input.<br><br>

        <ul>

            @foreach ($errors->all() as $error)

            <li>{{ $error }}</li>

            @endforeach

        </ul>

    </div>

    @endif

    <div class="card card-warning">

        <div class="card-header">

            <h3 class="card-title">Ubah Data Barang</h3>

        </div>

        <!-- /.card-header -->

        <!-- form start -->

        <form action="{{ route('barang.update',$barang->id) }}" method="POST">

            @csrf

            @method('PUT')

            <div class="card-body">

                    <label for="nama">Barang</label>

                    <input type="text" class="form-control" id="barang" name="barang" value="{{$barang->nama}}">

                </div>

                <div class="form-group">

                    <label for="stock">Stock</label>

                    <input type="number" class="form-control" id="stock" name="stock" value="{{$barang->stock}}">

                </div>

                <div class="form-group">

                    <label for="harga">Harga</label>

                    <input type="number" class="form-control" id="harga" name="harga" value="{{$barang->harga}}">

                </div>

                <div class=" form-group">

                    <label for="description">Deskripsi</label>

                    <textarea id="description" name="description" class=" form-control"

                        rows="4">{{ $barang->description }}</textarea>

                </div>

            </div>

            <!-- /.card-body -->

            <div class="card-footer">

                <button type="submit" class="btn btn-warning float-right">Ubah</button>

            </div>

        </form>

    </div>

</div>

@endsection

* Buka kembali BarangController lalu tambahkan script berikut pada class LayananController.

public function update(Barang $barang, Request $request):RedirectResponse

    {

        $request->validate([

            "barang"=>"required",

            "stock"=>"required",

            "harga"=>"nullable",

            "description"=>"required"

        ]);

        $barang->update($request->all());

        return redirect()->route('barang.index')->with('updated','Data Barang Berhasil Diubah');

    }

* Selanjutnya uji coba proses update, apakah berjalan lancar atau tidak.

### **Membuat Method show()**

* Tambahkan script berikut pada class LayananController.

public function show():View

    {

        $barang=Barang::all();

        return view('barang.show')->with([

            "title"=> "Tampil Data Barang",

            "data"=>$barang

        ]);

    }

* Buka file show.blade.php pada folder views\layanan, lalu tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Barang')

@section('konten')

<div class="col-md-6">

    @if ($errors->any())

    <div class="alert alert-danger">

        <strong>Whoops!</strong> There were some problems with your input.<br><br>

        <ul>

            @foreach ($errors->all() as $error)

            <li>{{ $error }}</li>

            @endforeach

        </ul>

    </div>

    @endif

    <div class="card card-primary">

        <div class="card-header">

            <h3 class="card-title">Data Barang</h3>

        </div>

        <!-- /.card-header -->

        <div class=" card-body">

            <table>

                <th>Barang</th>

                    <td>:</td>

                    <td>{{ $data[0]->barang }}</td>

                </tr>

                <tr>

                    <th>Stok</th>

                    <td>:</td>

                    <td>{{ $data[0]->Stock}}</td>

                </tr>

                <tr>

                    <th>Harga</th>

                    <td>:</td>

                    <td>@money($data[0]->price)</td>

                </tr>

                <tr>

                    <th>Deskripsi</th>

                    <td>:</td>

                    <td>{{ $data[0]->description}}</td>

                </tr>

            </table>

        </div>

        <!-- /.card-body -->

    </div>

</div>

@endsection

* Uji coba method show.

### **Membuat Method destroy()**

* Tambahkan script berikut pada class BarangController:

 public function destroy($id):RedirectResponse

    {

        Barang::where('id',$id)->delete();

        return redirect()->route('barang.index')->with('deleted','Data Barang Berhasil Dihapus');

    }

}

* Selanjutnya uji coba seluruh method yang telah dibuat.

## Fitur Pelanggan

### **Membuat Controller Pelanggan**

* Buat controller Pelanggan dengan menggunakan php artisan

php artisan make:controller PelangganController

* Tambahkan script berikut pada **PelangganController.php**

<?php

namespace App\Http\Controllers;

use App\Models\Pelanggan;

use Illuminate\Http\Request;

use Illuminate\View\View;

use Illuminate\Http\RedirectResponse;

class PelangganController extends Controller

{

    //

    public function index()

    {

        return view('pelanggan.index',["title"=>"Pelanggan","data"=>Pelanggan::all()

    ]);

    }

    public function create():View

    {

        return view('pelanggan.index')->with(["title"=>"Tambah Data Pelanggan"]);

    }

    public function store(Request $request):RedirectResponse

    {

        $request->validate([

            "nama"=>"required",

            "hp"=>"required",

            "alamat"=>"nullable"

        ]);

        Pelanggan::create($request->all());

        return redirect()->route('pelanggan.index')->with('success','Data Pelanggan Berhasil Ditambahkan');

    }

    public function edit(Pelanggan $pelanggan):View

    {

        return view('pelanggan.edit',compact('pelanggan'))->with(["title"=>"Ubah Data Pelanggan"]);

    }

    public function update(Request $request,Pelanggan $pelanggan):RedirectResponse

    {

        $request->validate([

            "nama"=>"required",

            "hp"=>"required",

            "alamat"=>"nullable"

    ]);

    $pelanggan->update($request->all());

    return redirect()->route('pelanggan.index')

                     ->with('updated','Data Pelanggan Berhasil Diubah');

    }

    public function show(Pelanggan $pelanggan):View

    {

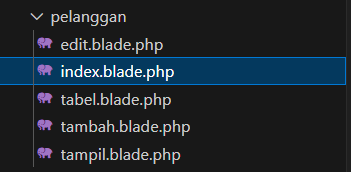
        return view('pelanggan.tampil',compact('pelanggan'))

               ->with(["title"=>"Data Pelanggan"]);

    }

}

* Buat folder baru didalam views dengan nama pelanggan
* Buat file baru di dalam folder pelanggan dengan nama **index.blade.php**

****

* Buka file **views/layouts/template.blade.php**, lalu tambahkan script berikut pada menu pelanggan.

<li class="nav-item">

                            <a href="{{ route('pelanggan.index')}}" class="nav-link {{ ($title==='Pelanggan')?'active':''}}">

                                <i class="nav-icon fas fa-users"></i>

                                <p>

                                    Pelanggan

                                </p>

                            </a>

                        </li>

* Lalu tambahkan script berikut pada file **index.blade.php**:

@extends('layouts.template')

@section('tambahanCSS')

<!-- DataTables -->

<link rel="stylesheet" href="plugins/datatables-bs4/css/dataTables.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-responsive/css/responsive.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-buttons/css/buttons.bootstrap4.min.css">

<!-- Toastr -->

<link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.css">

@endsection

@section('judulh1','Admin - Pelanggan')

@section('judulh3','Pelanggans')

@section('konten')

<div class="col-md-4">

    <div class="card card-success">

        <div class="card-header">

            <h3 class="card-title">Input Pelanggan</h3>

        </div>

        <!-- /.card-header -->

        <!-- form start -->

        <form action="{{ route('pelanggan.store') }}" method="POST">

            @csrf

            <div class=" card-body">

                <div class="form-group">

                    <label for="nama">Nama Pelanggan</label>

                    <input type="text" class="form-control" id="nama" name="nama" placeholder="">

                </div>

                <div class="form-group">

                    <label for="hp">No HP</label>

                    <input type="text" class="form-control" id="hp" name="hp">

                </div>

                <div class="form-group">

                    <label for="alamat">Alamat</label>

                    <input type="text" class="form-control" id="alamat" name="alamat">

                </div>

            </div>

            <!-- /.card-body -->

            <div class="card-footer">

                <button type="submit" class="btn btn-success float-right">Simpan</button>

            </div>

        </form>

    </div>

</div>

<div class="col-md-8">

    <div class="card card-info">

        <div class="card-header">

            <h3 class="card-title">Data Pelanggan</h3>

        </div>

        <!-- /.card-header -->

        <div class="card-body">

            <table id="example1" class="table table-bordered table-striped ">

                <thead>

                    <tr>

                        <th>No</th>

                        <th>Nama</th>

                        <th>No HP</th>

                        <th>Alamat</th>

                        <th>Aksi</th>

                    </tr>

                </thead>

                <tbody>

                    @foreach($data as $dt)

                    <tr>

                        <td>{{ $loop->iteration }}</td>

                        <td>{{ $dt->nama }}</td>

                        <td>{{ $dt->hp }}</td>

                        <td>{{ $dt->alamat }}</td>

                        <td>

                            <div class="btn-group">

                                <a type="button" class="btn btn-warning" href="{{ route('pelanggan.edit',$dt->id) }}">

                                    <i class=" fas fa-edit"></i>

                                </a>

                                <a type="button" class="btn btn-success" href="{{ route('pelanggan.show',$dt->id) }}">

                                    <i class=" fas fa-eye"></i>

                                </a>

                            </div>

                        </td>

                    </tr>

                    @endforeach

                </tbody>

            </table>

        </div>

    </div>

</div>

@endsection

@section('tambahanJS')

<!-- DataTables  & Plugins -->

<script src="plugins/datatables/jquery.dataTables.min.js"></script>

<script src="plugins/datatables-bs4/js/dataTables.bootstrap4.min.js"></script>

<script src="plugins/datatables-responsive/js/dataTables.responsive.min.js"></script>

<script src="plugins/datatables-responsive/js/responsive.bootstrap4.min.js"></script>

<script src="plugins/datatables-buttons/js/dataTables.buttons.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.bootstrap4.min.js"></script>

<script src="plugins/jszip/jszip.min.js"></script>

<script src="plugins/pdfmake/pdfmake.min.js"></script>

<script src="plugins/pdfmake/vfs\_fonts.js"></script>

<script src="plugins/datatables-buttons/js/buttons.html5.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.print.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.colVis.min.js"></script>

<!-- Toastr -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>

@endsection

@section('tambahScript')

<script>

$(function() {

    $("#example1").DataTable({

        "responsive": true,

        "lengthChange": true,

        "autoWidth": false,

        "responsive": true,

    }).buttons().container().appendTo('#example1\_wrapper .col-md-6:eq(0)');

});

@if($message = Session::get('success'))

toastr.success("{{ $message}}");

@elseif($message = Session::get('updated'))

toastr.warning("{{ $message}}");

@endif

</script>

@endsection

### **Membuat Route**

* Tambahkan script berikut pada **route/web.php**

Route::resource('pelanggan',PelangganController::class)->except('destroy';

* Lalu jalankan perintah php artisan route:list untuk melihat daftar method dari pelanggan.

php artisan route:list

* Pada **model Pelanggan** tambahkan variabel $fillable

class Pelanggan extends Model

{

    use HasFactory;

    protected $fillable=['nama','hp','alamat'];

    public function Transaksi():HasMany

    {

        return $this->hasMany(Transaksi::class);

    }

}

* Buat file baru pada Views/Pelanggan dengan nama **editpelanggan.blade.php**, lalu tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Pelanggan')

@section('konten')

<div class="col-md-6">

    @if ($errors->any())

    <div class="alert alert-danger">

        <strong>Whoops!</strong> There were some problems with your input.<br><br>

        <ul>

            @foreach ($errors->all() as $error)

            <li>{{ $error }}</li>

            @endforeach

        </ul>

    </div>

    @endif

    <div class="card card-warning">

        <div class="card-header">

            <h3 class="card-title">Ubah Data Pelanggan</h3>

        </div>

        <!-- /.card-header -->

        <!-- form start -->

        <form action="{{ route('pelanggan.update',$pelanggan->id) }}" method="POST">

            @csrf

            @method('PUT')

            <div class=" card-body">

                <div class="form-group">

                    <label for="nama">Nama Pelanggan</label>

                    <input type="text" class="form-control" id="nama" name="nama" placeholder="" value="{{$pelanggan->nama}}">

                </div>

                <div class="form-group">

                    <label for="hp">No HP</label>

                    <input type="text" class="form-control" id="hp" name="hp" value="{{$pelanggan->hp}}">

                </div>

                <div class="form-group">

                    <label for="alamat">Alamat</label>

                    <input type="text" class="form-control" id="alamat" name="alamat" value="{{$pelanggan->alamat}}">

                </div>

            </div>

            <!-- /.card-body -->

            <div class="card-footer">

                <button type="submit" class="btn btn-warning float-right">Ubah</button>

            </div>

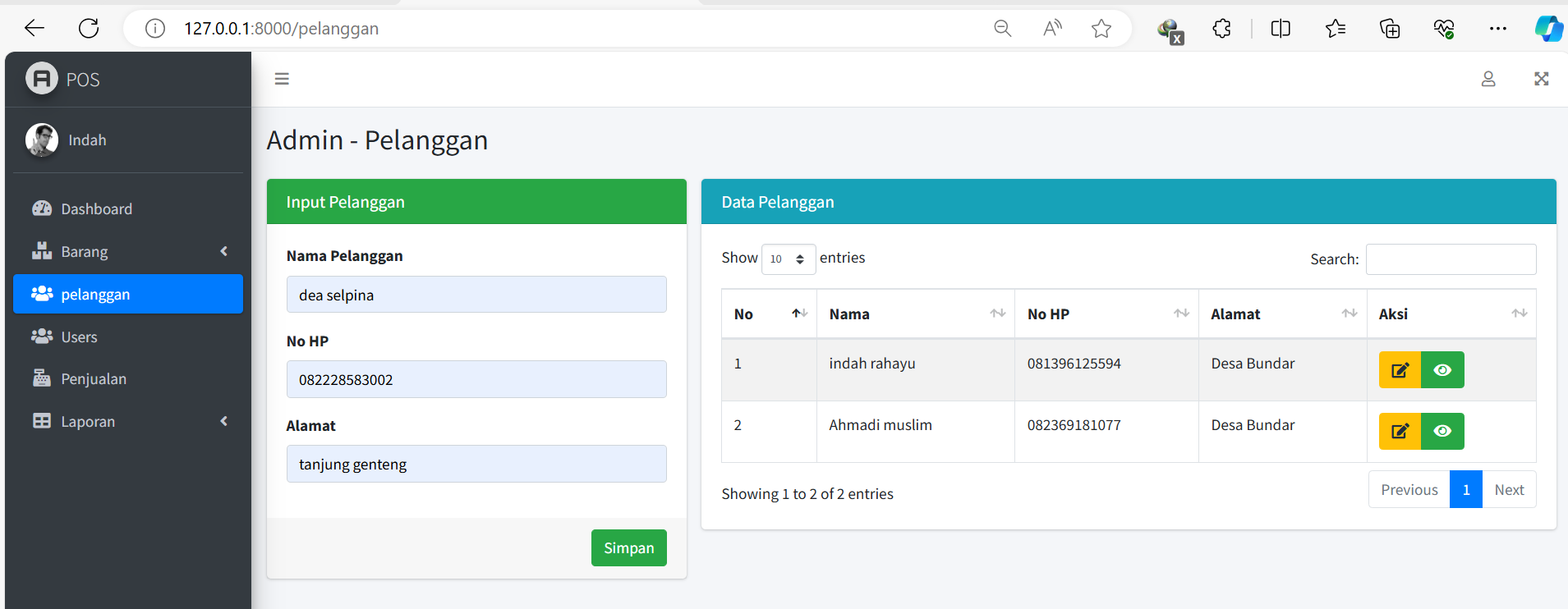
        </form>

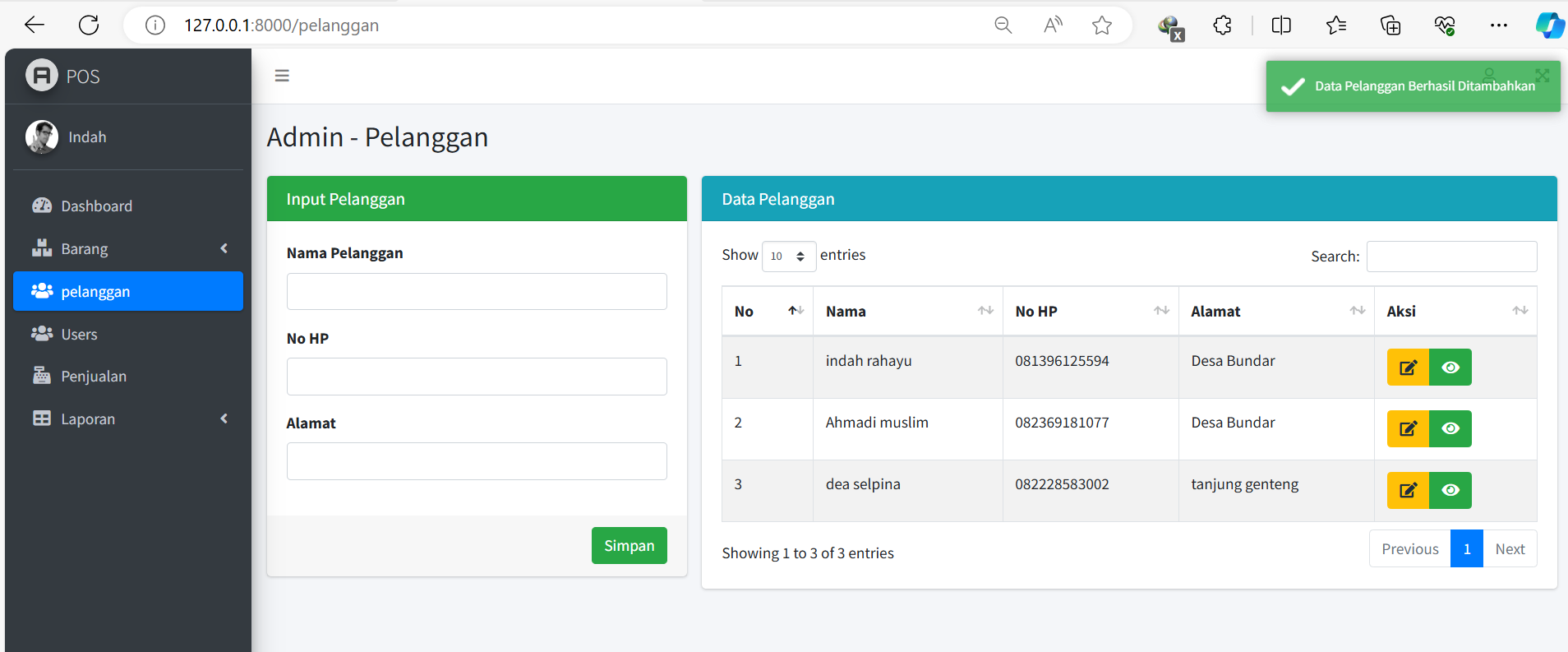
    </div>

</div>

@endsection

* Uji coba keseluruhan method yang telah dibuat
* Hasil: tampilan





* Penjelasan data **NULL**

public function update(Pelanggan $pelanggan, Request $request):RedirectResponse

    {

        $request->validate([

            "nama"=>"required",

        ]);

        if (empty($request['hp'])) {

            $request['hp']='null';

        if (empty($request['alamat']))

            $request['alamat']='null';

        }

Data yang di request akan menjadi Null jika form input tidak diisi atau dikosongkan. Hal ini bertujuan untuk mengantisipasi jika ada pelanggan yang tidak berkenan untuk memberikan data HP dan Alamat karena bersifat privasi

## Fitur User

## Manajemen User

### **Membuat Controller User**

* Membuat Cotroller User dengan prompt artisan pada terminal

php artisan make:controller UserController

* Tambahkan Method index() pada UserController

public function index()

    {

        return view('user.index',[

            "title"=>"Data User",

            "data"=>User::all()

        ]);

    }

* mengimport class model User pada UserController

use App\Models\User;

### **Membuat Route**

* Tambahkan script berikut pada routes/web.php

Route::resource('user',UserController::class)-except('destroy','create','show','update','edit');

* Dan Tambahkan script berikut dibagian atas

use App\Http\Controllers\UserController;

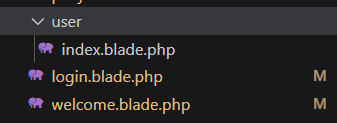
* Lalu cek daftar route menggunakan prompt artisan

php artisan route:list

### 

### **Membuat Tampilan Manajemen User**

* Buat folder baru di dalam views dengan nama user, lalu Buat file baru di dalam folder user dengan nama index.blade.php, lalu tambahkan script berikut pada file index.blade.php



@extends('layouts.template')

@section('tambahanCSS')

<!-- DataTables -->

<link rel="stylesheet" href="plugins/datatables-bs4/css/dataTables.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-responsive/css/responsive.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-buttons/css/buttons.bootstrap4.min.css">

<!-- Toastr -->

<link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.css">

@endsection

@section('judulh1','Admin - Users')

@section('judulh3','Users')

@section('konten')

<div class="col-md-4">

    <div class="card card-success">

        <div class="card-header">

            <h3 class="card-title">Input User</h3>

        </div>

        <!-- /.card-header -->

        <!-- form start -->

        <form action="{{ route('user.store') }}" method="POST">

            @csrf

            <div class=" card-body">

                <div class="form-group">

                    <label for="name">Nama</label>

                    <input type="text" class="form-control" id="name" name="name" placeholder=" Nama Lengkap" required >

                </div>

                <div class="form-group">

                    <label for="email">Email</label>

                    <input type="email" class="form-control @error('email') is-invalid @enderror" id="email" name="email" placeholder="ahmadi@example.com" required >

                    @error('email')

                    <div class="invalid-feedback">

                        {{ $message }}

                    </div>

                    @enderror

                </div>

                <div class="form-group">

                    <label for="password">Password</label>

                    <input type="password" class="form-control" id="password" name="password" required>

                </div>

            <!-- /.card-body -->

            <div class="card-footer">

                <button type="submit" class="btn btn-success float-right">Simpan</button>

            </div>

        </form>

    </div>

</div>

<div class="col-md-8">

    <div class="card card-info">

        <div class="card-header">

            <h3 class="card-title">Data User</h3>

        </div>

        <!-- /.card-header -->

        <div class="card-body">

            <table id="example1" class="table table-bordered table-striped ">

                <thead>

                    <tr>

                        <th>No</th>

                        <th>Nama </th>

                        <th>Email</th>

                    </tr>

                </thead>

                <tbody>

                    @foreach($data as $dt)

                    <tr>

                        <td>{{ $loop->iteration }}</td>

                        <td>{{ $dt->name }}</td>

                        <td>{{ $dt->email }}</td>

                    </tr>

                    @endforeach

                </tbody>

            </table>

        </div>

    </div>

</div>

@endsection

@section('tambahanJS')

<!-- DataTables  & Plugins -->

<script src="plugins/datatables/jquery.dataTables.min.js"></script>

<script src="plugins/datatables-bs4/js/dataTables.bootstrap4.min.js"></script>

<script src="plugins/datatables-responsive/js/dataTables.responsive.min.js"></script>

<script src="plugins/datatables-responsive/js/responsive.bootstrap4.min.js"></script>

<script src="plugins/datatables-buttons/js/dataTables.buttons.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.bootstrap4.min.js"></script>

<script src="plugins/jszip/jszip.min.js"></script>

<script src="plugins/pdfmake/pdfmake.min.js"></script>

<script src="plugins/pdfmake/vfs\_fonts.js"></script>

<script src="plugins/datatables-buttons/js/buttons.html5.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.print.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.colVis.min.js"></script>

<!-- Toastr -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>

@endsection

@section('tambahScript')

<script>

$(function() {

    $("#example1").DataTable({

        "responsive": true,

        "lengthChange": true,

        "autoWidth": false,

        "responsive": true,

    }).buttons().container().appendTo('#example1\_wrapper .col-md-6:eq(0)');

});

@if($message = Session::get('success'))

toastr.success("{{ $message}}");

@elseif($message = Session::get('updated'))

toastr.warning("{{ $message}}");

@endif

</script>

@endsection

* Perbaiki template.blade.php. Dengan menambahkan route pada sidebar user.

<li class="nav-item">

                            <a href="{{ route('user.index')}}" class="nav-link {{ ($title==='User')?'active':''}}">

                                <i class="nav-icon fas fa-user"></i>

                                <p>

                                    User

                                </p>

                            </a>

                        </li>

* Pada manajemen user kita tidak menambahkan menu edit, karena nantinya kita akan membuat menu edit data user tidak di admin, namun di private menu masing-masing user. Sedangkan untuk menu hapus tidak kita buat juga karena User berkaitan dengan tabel order, sehingga jika data user dapat dihapus maka akan terjadi kecacatan data nantinya.

### **Membuat method store()**

* Untuk menyimpan data ke database, tambahkan method store pada UserController

<?php

namespace App\Http\Controllers;

use App\Models\User;

use Illuminate\Http\Request;

use Illuminate\Http\RedirectResponse;

use Illuminate\Support\Facades\Hash;

class UserController extends Controller

{

    //

    public function index()

    {

        return view('user.index',[

            "title"=>"Data User",

            "data"=>User::all()

        ]);

    }

    public function store(Request $request):RedirectResponse

    {

        $request->validate([

            "name"=>"required",

            "email"=>"required",

            "password"=>"required",

        ]);

        $password=Hash::make($request->password);

        $request->merge([

            "password"=>$password

        ]);

        User::create($request->all());

        return redirect()->route('user.index')->with('success','Data User Berhasil Ditambahkan');

    }

}

### **Mengatur model User**

* Untuk menyimpan data ke database, tambahkan script berikut pada model User

<?php

namespace App\Models;

//

use Illuminate\Contracts\Auth\MustVerifyEmail;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Foundation\Auth\User as Authenticatable;

use Illuminate\Notifications\Notifiable;

use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable

{

    use HasApiTokens, HasFactory, Notifiable;

    /\*\*

     \* The attributes that are mass assignable.

     \*

     \* @var array<int, string>

     \*/

    protected $fillable = [

        'name','email','password'

    ];

    /\*\*

     \* The attributes that should be hidden for serialization.

     \*

     \* @var array<int, string>

     \*/

    protected $hidden = [

        'password',

        'remember\_token',

    ];

    /\*\*

     \* The attributes that should be cast.

     \*

     \* @var array<string, string>

     \*/

    protected $casts = [

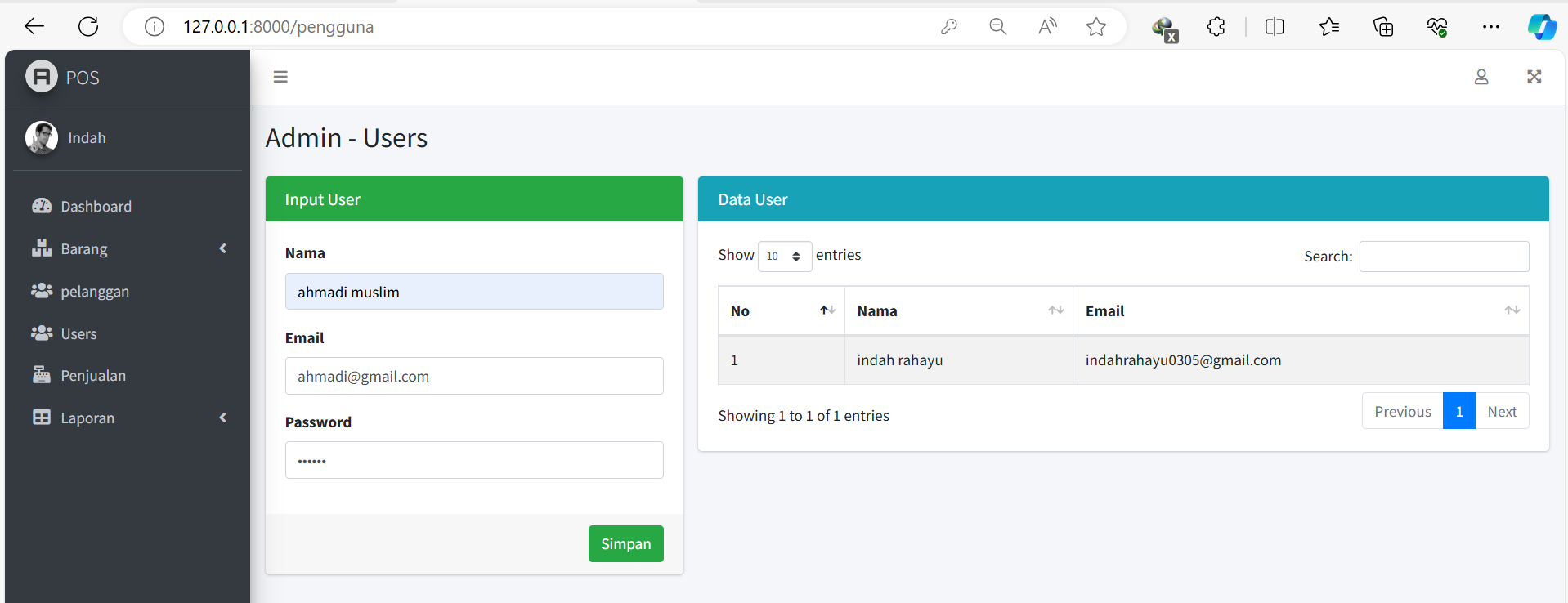
        'email\_verified\_at' => 'datetime',

        'password' => 'hashed',

    ];

}

* Lalu uji coba menyimpan data user



## Autentikasi User

### **Menyiapkan Form Login**

* Buat file baru di dalam folder Views dengan nama login.blade.php.
* Lalu tambahkan script berikut:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Login User</title>

<!-- Google Font: Source Sans Pro -->

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">

<!-- Font Awesome -->

<link rel="stylesheet" href="{{ asset('plugins/fontawesome-free/css/all.min.css') }}">

<!-- icheck bootstrap -->

<link rel="stylesheet" href="{{ asset('plugins/icheck-bootstrap/icheck-bootstrap.min.css') }}">

<!-- Theme style -->

<link rel="stylesheet" href="{{ asset('dist/css/adminlte.min.css') }}">

</head>

<body class="hold-transition login-page">

<div class="login-box">

<!-- /.login-logo -->

<div class="card card-outline card-primary">

<div class="card-header text-center">

<h1 class="h1">Aplikasi<b>POS</b></h1>

</div>

<div class="card-body">

<p class="login-box-msg">Please Login</p>

@if(session()->has('loginError'))

<div class="alert alert-danger alert-dismissible">

<button type="button" class="close" data-dismiss="alert" aria-hidden="true">×</button>

<h5><i class="icon fas fa-ban"></i> Alert!</h5>

{{ session('loginError') }}

</div>

@endif

<form action="/login" method="post">

@csrf

<div class="input-group mb-3">

<input type="email" class="form-control @error('email') is-invalid @enderror" placeholder="Email" id="email" name="email" value="{{ old('email') }}">

<div class="input-group-append">

<div class="input-group-text">

<span class="fas fa-envelope"></span>

</div>

</div>

@error('email')

<div class="invalid-feedback">

{{ $message }}

</div>

@enderror

</div>

<div class="input-group mb-3">

<input type="password" class="form-control" placeholder="Password" id="password" name="password">

<div class="input-group-append">

<div class="input-group-text">

<span class="fas fa-lock"></span>

</div>

</div>

</div>

</div>

<div class="card-footer">

<button type="reset" class="btn btn-warning">Cancel</button>

<button type="submit" class="btn btn-primary float-right">Login</button>

</div>

<!-- /.card-body -->

</div>

</form>

<!-- /.card -->

</div>

<!-- /.login-box -->

<!-- jQuery -->

<script src="{{asset('plugins/jquery/jquery.min.js')}}"></script>

<!-- Bootstrap 4 -->

<script src="{{asset('plugins/bootstrap/js/bootstrap.bundle.min.js')}}"></script>

<!-- AdminLTE App -->

<script src="{{asset('dist/js/adminlte.min.js')}}"></script>

</body>

</html>

### **Menambahkan Controller Login**

* Buat controller baru dengan nama LoginController

php artisan make:controller LoginController

* Tambahkan method loginView pada class LoginController

public function loginView()

    {

        return view('login');

    }

* Tambahkan Method authenticate pada class LoginController

public function authenticate(Request $request):RedirectResponse

    {

        $credentials = $request->validate([

            'email' => ['required'],

            'password' => ['required'],

        ]);

        if (Auth::attempt($credentials)) {

            $request->session()->regenerate();

            return redirect()->intended('/');

        }

        return back()->with('loginError','Login Failed');

    }

* Jangan lupa untuk mengimport 3 class berikut:

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Illuminate\Http\RedirectResponse;

### **Menambahkan Route Login**

* Tambahkan 2 buah route berikut pada routes/web.php

Route::get('login',[LoginController::class,'loginView'])->name('login');

Route tersebut berguna untuk mengarahkan ke method loginView pada controller LoginController ketika mengakses login dengan fungsi get.

Berikutnya,

Route::post('login',[LoginController::class,'authenticate']);

Route tersebut berguna untuk mengarahkan ke method authenticate pada controller LoginController ketika mengakses login dengan fungsi post.

**POST dan GET keduanya adalah method atau fungsi yang digunakan untuk mengirimkan data ke halaman lain.**

* Silahkan ketikkan 127.0.0.1:8000/login maka seharusnya akan tampil form login seperti dibawah berikut:
* Masukkan email dan password yang sudah di daftarkan pada Manajemen User sebelumnya. Jika berhasilnya seharusnya akan mengarah ke Route ( / ) yang jika mengikuti Modul ini akan mengarah pada tampilan [view (dahsboard)](#1g78ehf8jgam).

### **Memperbaiki Route**

Saat ini jika user yang belum login mencoba mengakses route langsung dari browser address tetap akan bisa diakses, hal ini terjadi karena kita belum melindungi akses ke route kita yang lain menggunakan middleware. Middleware seperti namanya yaitu perangkat lunak yang berfungsi sebagai penengah. Dalam konteks laravel khususnya bagian Autentikasi, sebagai penengah antara request yang dilakukan oleh user dan Route yang tersedia. Jadi seandainya ada User atau user ilegal yang mencoba mengakses Route tertentu, maka aksesnya akan diblokir oleh middleware. Cara menggunakan middleware cukup gampang, secara default Laravel sudah menyediakan middleware autentikasi.

* Buka kembali routes/web.php
* Kemudian tambahkan ->middleware('auth'); pada setiap Route yang telah kita buat, kecuali Route yang menangani fungsi login. Berikut perubahan keseluruhan routes/web/php:

Route::get('/',function(){

    return view('welcome',[

        "title"=>"Dashboard"

    ]);

})->middleware('auth');

Route::resource('kategori',CategoryController::class)

->except('show','destroy','create')->middleware('auth');

Route::resource('pelanggan',CustomerController::class)

->except('destroy')->middleware('auth');

Route::resource('produk',ProductController::class)->middleware('auth');

Route::resource('user',UserController::class)->except('destroy','create','show','update','edit')->middleware('auth');

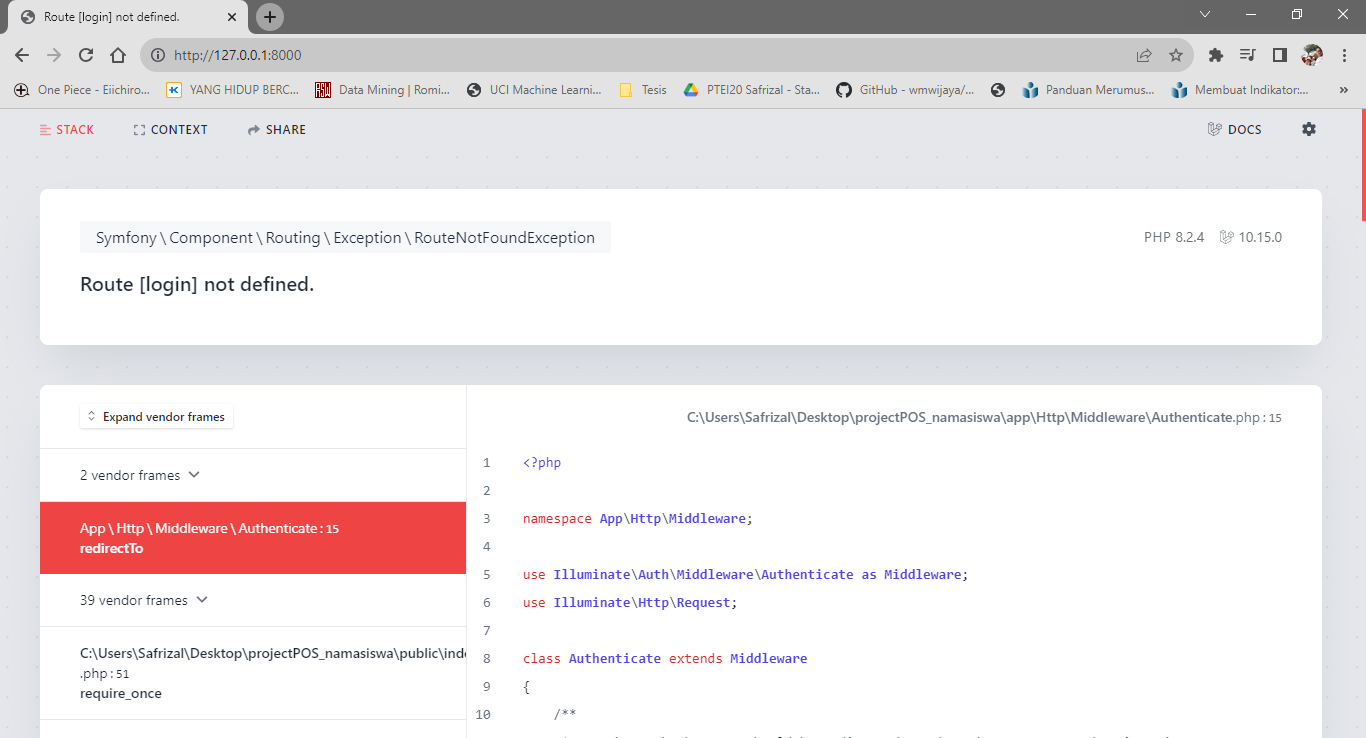
Route::get('cetakReceipt',[CetakController::class,'receipt'])->name('cetakReceipt')->middleware('auth');

Route::get('login',[LoginController::class,'loginView'])->name('login');

Route::post('login',[LoginController::class,'authenticate']);

Route::post('logout',[LoginController::class,'logout'])->middleware('auth');

* Saat ini ketika kita mengakses controller tertentu akan muncul pesan error berikut.



* Hal ini terjadi karena kita belum memberikan nama pada route login kita. Jadi perbaiki kembali routes/web.php. Perbaiki route login yang menuju ke tampilan login menjadi seperti berikut:

Route::get('login',[LoginController::class,'loginView'])->name('login');

* Sekarang seharusnya ketika kita coba mengakses routes tertentu yang kita batasi dengan middleware tanpa login maka akan diarahkan ke halaman login.

### 

### **Menambahkan Method Logout**

Saat ini kita sudah memiliki form login, dan fungsi login yang berjalan. Yang belum kita miliki adalah fungsi logout. Sebelum kita membuat fungsi logout, kita perbaiki dulu tampilan Aplikasi POS yang sedang kita kerjakan. Perbaikan ini bertujuan untuk menambahkan keterangan User yang sedang login dan menambahkan tombol logout.

* Buka template.blade.php Cari bagian Sidebar, lalu perbaiki script sesuai dengan lokasi yang ditandai pada gambar di bawah ini.

<!-- Sidebar -->

            <div class="sidebar">

                <!-- Sidebar user panel (optional) -->

                <div class="user-panel mt-3 pb-3 mb-3 d-flex">

                    <div class="image">

                        <img src="{{ asset('dist/img/user2-160x160.jpg')}}" class="img-circle elevation-2" alt="User Image">

                    </div>

                    <div class="info">

                        <a href="#" class="d-block"> {{ Auth::user()->name }} </a>

                    </div>

                </div>

**Penjelasan:** {{ Auth::user()->name }} berguna untuk menampilkan nama user yang sedang login

* Kemudian cari bagian Right navbar links, tambahkan script <li> berikut di dalam tag <ul class=”navbar-nav ml-auto> </ul>

<li class="nav-item dropdown">

<a class="nav-link" data-toggle="dropdown" href="#">

<i class="far fa-user mr-2"></i>{{ Auth::user()->name }}

<span class="badge badge-warning navbar-badge"></span>

</a>

<div class="dropdown-menu dropdown-menu-lg dropdown-menu-right">

<span class="dropdown-item dropdown-header">User Menu</span>

<div class="dropdown-divider"></div>

<a href="#" class="dropdown-item">

<i class="fas fa-user mr-2"></i> {{ Auth::user()->name }}

<span class="float-right text-muted text-sm"></span>

</a>

<div class="dropdown-divider"></div>

<form action="logout" method="POST">

@csrf

<button type="submit" class="dropdown-item">

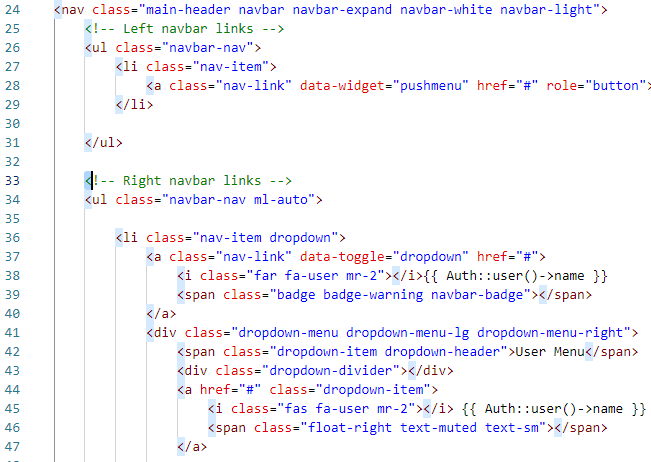
<i class="fas fa-sign-out-alt mr-2"></i>Logout

</button>

</form>

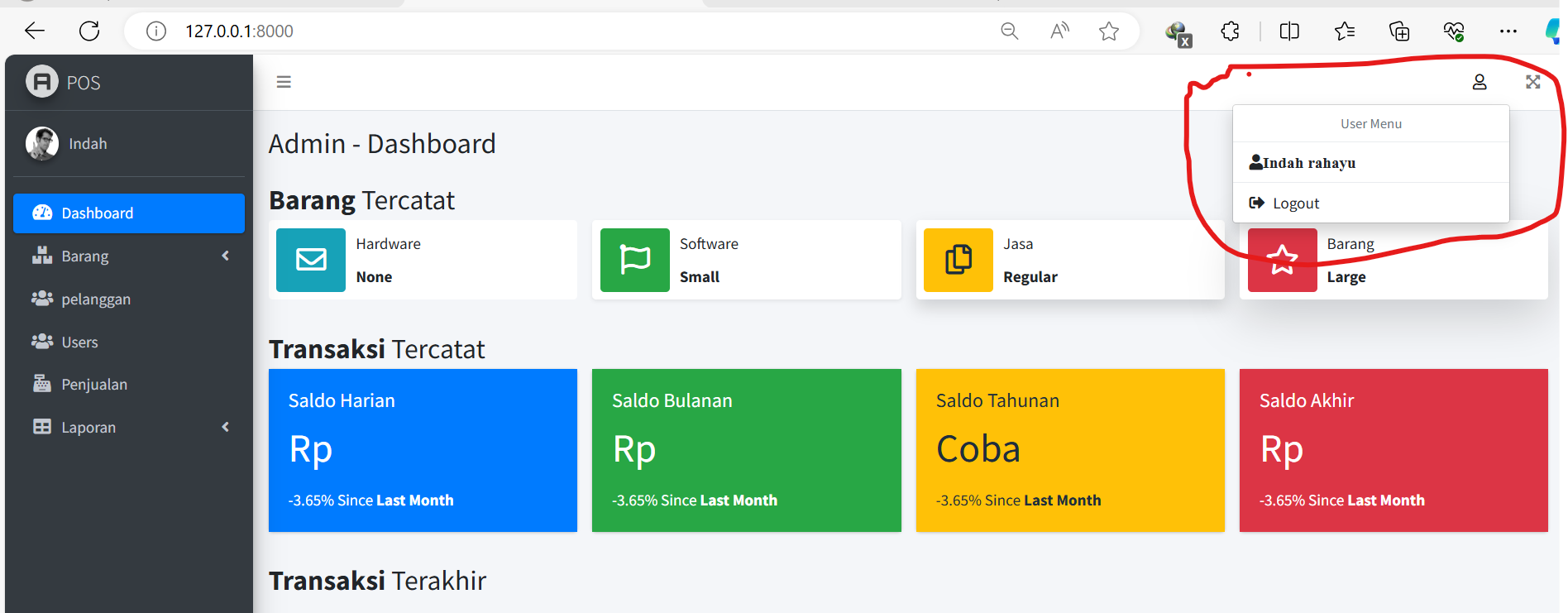
</div>

</li>



**Penjelasan:** jika diperhatikan kembali tombol logout diletakkan didalam sebuah form dengan method POST, tujuannya adalah untuk mencegah orang lain mengeluarkan kita dari aplikasi (logout).

* Jika penempatan script benar maka pada halaman utama akan muncul menu baru seperti berikut:



* Lalu buka kembali controller LoginController lalu tambahkan method logout di dalam class LoginController.

public function logout(Request $request): RedirectResponse

    {

        Auth::logout();

        $request->session()->invalidate();

        $request->session()->regenerateToken();

        return redirect('/login');

    }

* Silahkan uji coba tombol logout, lalu coba akses Routes tanpa melakukan login terlebih dahulu.

## Fitur Transaksi / Penjualan

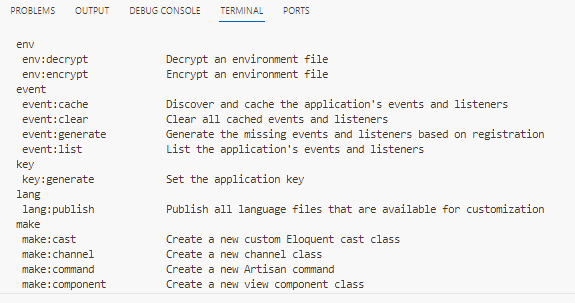
Untuk bagian penjualan, bagian ini agak spesial karena kita harus membuat halaman penjualan berubah tanpa me-refresh halaman, untuk melakukan hal tersebut kita akan gunakan **Livewire** karena livewire adalah framework fullstack untuk laravel yang dapat digunakan untuk membuat antar muka yang dinamis dengan cara yang sederhana tanpa meninggalkan kenyamanan dari laravel.

### **Instalasi Livewire**

1. Buat sebuah file baru didalam layouts dan beri nama **masterlw.blade.php**

php artisan

berikut tampilan jika livewire belum terinstal

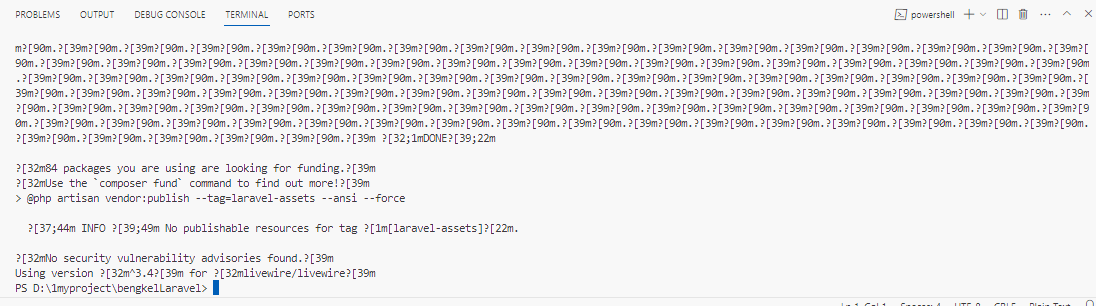


1. Buat 2 buah komponen baru dengan nama penjualan dan transaksis

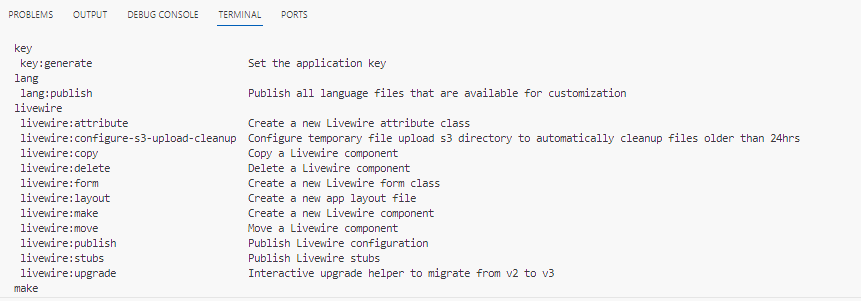


composer require livewire/livewire

berikut tampilan jika instalasi telah selesai



Jika di cek melalui php artisan, maka library livewire telihat sudah terinstal



1. Buat sebuah file baru didalam layouts dan beri nama **masterlw.blade.php**

### **Membuat Komponen**

1. Buat 2 buah komponen baru dengan nama penjualan dan transaksis

php artisan livewire:make penjualan



php artisan livewire:make transaksis

1. Buat sebuah file baru didalam layouts dan beri nama **masterlw.blade.php**



1. Pada file **masterlw.blade.php**, tambahkan script berikut

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Bootstrap demo</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65" crossorigin="anonymous">

<!-- Font Awesome Icons -->

<link rel="stylesheet" href="{{ asset('plugins/fontawesome-free/css/all.min.css') }}">

@livewireStyles

</head>

<body>

<nav class="navbar navbar-expand-lg navbar-dark bg-dark btransaksi-bottom btransaksi-body" data-bs-theme="dark">

<div class="container-fluid">

<a class="navbar-brand" href="#">Navbar</a>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNavAltMarkup">

<div class="navbar-nav ms-auto">

<a class="nav-link active" aria-current="page" href="#">{{ Auth::user()->name }}</a>

<form action="logout" method="POST" class="d-flex">

@csrf

<button type="submit" class="btn btn-outline-secondary"><i class="fas fa-sign-out-alt mr-2"></i>Logout</button>

</form>

</div>

</div>

</div>

</nav>

<div class="container-fluid py-3">

@yield('konten')

</div>

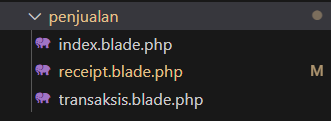
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-kenU1KFdBIe4zVF0s0G1M5b4hcpxyD9F7jL+jjXkk+Q2h455rYXK/7HAuoJl+0I4" crossorigin="anonymous"></script>

@livewireScripts

</body>

</html>

1. Buat 1 buah folder baru didalam folder views dengan nama folder penjualan
2. Di dalam folder penjualan buat 2 buah file baru dengan nama **index.blade.php** dan **transaksis.blade.php**



1. Di dalam file **index.blade.php** tambahkan script berikut:

@extends('layouts.masterlw')

@section('konten')

  @livewire('penjualan')

@endsection

1. Di dalam file **transaksis.blade.php** tambahkan script berikut:

@extends('layouts.masterlw')

@section('konten')

  @livewire('transaksis')

@endsection

1. Sekarang pindah ke folder **views/livewire**, buka file **penjualan.blade.php** tambahkan script berikut.

<div class="row">

<div class="d-flex aligns-items-center justify-content-center">

<div class="col-6">

<div class="card ">

<div class="card-header">

<h5 class="card-title">Invoice</h5>

</div>

<!-- /.card-header -->

<!-- form start -->

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<form wire:submit.prevent="store" method="POST">

@csrf

<div class=" card-body">

<div class="form-group">

<label for="name">Pelanggan</label>

<select class="form-control" wire:model="pelanggan\_id">

<option hidden>--Pilih Pelanggan--</option>

@foreach($data as $dt )

<option value="{{ $dt->id }}">{{ $dt->name }}</option>

@endforeach

</select>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer text-end">

<button type="submit" class="btn btn-success btn-sm">Submit</button>

</div>

</form>

</div>

</div>

</div>

</div>

1. Buka file transaksis.blade.php pada folder views/livewire, lalu tambahkan script berikut:

<div class="row">

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<div class="d-flex aligns-items-center justify-content-center">

<div class="col-lg-6">

<div class="card">

<div class="card-header">Invoice</div>

<div class="card-body">

<table class="table table-sm">

<tr>

<th>Invoice</th>

<th>:</th>

<td>{{ $data->invoice }}</td>

</tr>

<tr>

<th>Pelanggan</th>

<th>:</th>

<td>{{ $data->pelanggan->name}}</td>

</tr>

<tr>

<th>Tanggal</th>

<th>:</th>

<td>{{ $data->created\_at->format('d M Y')}}</td>

</tr>

<tr>

<th>Total</th>

<th>:</th>

<td>@money($total)</td>

</tr>

</table>

</div>

</div>

</div>

<div class="col-lg-6 ps-3">

<div class="card">

<div class="card-header">Layanan </div>

<div class="card-body">

<form wire:submit.prevent="store" method="POST">

@csrf

<div class=" card-body">

<input type="hidden" name="transaksi\_id" wire:model="transaksi\_id" value="{{ $data->id }}">

<div class="form-group">

<label for="layanan\_id">Layanan</label>

<select class="form-control" wire:model="layanan\_id" name="layanan\_id">

<option hidden>--Pilih Layanan--</option>

@foreach($dataLayanan as $dt )

<option value="{{ $dt->id }}">{{ $dt->name }}</option>

@endforeach

</select>

</div>

<div class="form-group">

<label for="qty">Jumlah</label>

<input type="number" class="form-control" wire:model="qty" name="qty">

</div>

</div>

<!-- /.card-body -->

<div class="card-footer text-end">

<button type="submit" class="btn btn-success btn-sm"><i class="fas fa-cart-plus"></i> Tambahkan</button>

</div>

</form>

</div>

</div>

</div>

</div>

<div class="d-flex aligns-items-center justify-content-center pt-3">

<div class="col-lg-12">

<div class="card">

<div class="card-header">Daftar Belanja</div>

<div class="card-body">

<table class="table table-sm">

<thead>

<tr>

<th>No</th>

<th>Nama Layanan</th>

<th>Harga</th>

<th>Jumlah</th>

<th>Sub Total</th>

<th>Hapus</th>

</tr>

</thead>

<tbody>

@foreach($dataDetiltransaksi as $dt)

<tr>

<td>{{ $loop->iteration }}</td>

<td>{{ $dt->layanan->name }}</td>

<td>@money($dt->price)</td>

<td>{{ $dt->qty}}</td>

<td>@money($dt->price \* $dt->qty)</td>

<td><button class="btn btn-sm btn-danger" wire:click="delete({{ $dt->id }})"><i class=" fas fa-trash"></i></a></td>

</tr>

@endforeach

</tbody>

</table>

</div>

<div class="card-footer text-end">

<table class="table table-sm">

<tr>

<th>

<h2>Total</h2>

</th>

<th>

:

</th>

<th><h2>@money($total)</h2>

</th>

</tr>

<tr>

<th>

<h2>Uang</h2>

</th>

<th>

:

</th>

<th>

<input type="number" class="mt-2" wire:model="uang">

</th>

</tr>

<tr>

<th>

<h2>Kembali</h2>

</th>

<th>

:

</th>

<th><h2>

@money($kembali)</h2>

</th>

</tr>

</table>

<button class="btn btn-lg btn-success"><i class="fas fa-print"></i> Cetak</button>

</div>

</div>

</div>

</div>

</div>

### **Memperbaiki seluruh relasi**

1. Relasi pada Model **Pelanggan.php**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

use Illuminate\Database\Eloquent\Relations\HasMany;

class Pelanggan extends Model

{

    use HasFactory;

    protected $fillable=['nama','hp','alamat'];

    public function transaksi():HasMany

    {

        return $this->hasMany(Transaksi::class);

    }

}

1. Relasi pada model Detiltransaksi.php

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

use Illuminate\Database\Eloquent\Relations\BelongsTo;

class Detiltransaksi extends Model

{

    protected $fillable=['transaksi\_id','barang\_id','qty','harga'];

    use HasFactory;

    public function transaksi():BelongsTo

    {

        return $this->belongsTo(Transaksi::class);

    }

    public function layanan():BelongsTo

    {

        return $this->belongsTo(Barang::class);

    }

}

1. Relasi pada Model Transaksi.php

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

use Illuminate\Database\Eloquent\Relations\BelongsTo;

use Illuminate\Database\Eloquent\Relations\HasMany;

class Transaksi extends Model

{

    use HasFactory;

    protected $fillable=['pelanggan\_id','user\_id','invoice','total'];

    public function detiltransaksi():HasMany

    {

        return $this->hasMany(Detiltransaksi::class);

    }

    public function pelanggan():BelongsTo

    {

        return $this->belongsTo(Pelanggan::class);

    }

}

1. Relasi pada Model **Barang.php**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

use Illuminate\Database\Eloquent\Relations\BelongsTo;

use Illuminate\Database\Eloquent\Relations\HasMany;

class Layanan extends Model

{

    use HasFactory;

    protected $fillable=['name','stock','harga','description'];

    public function detiltransaksi():HasMany

    {

        return $this->hasMany(Detiltransaksi::class);

    }

}

### **Memperbaiki komponen livewire**

1. Buka file Penjualan.php yang terdapat pada folder Http\Livewire, kemudian tambahkan beberapa script berikut:

<?php

namespace App\Livewire;

use App\Models\Transaksi;

use Livewire\Component;

use App\Models\Pelanggan;

use Illuminate\Support\Facades\Auth;

class Penjualan extends Component

{

    public $pelanggan\_id;

    public function render()

    {

        return view('livewire.penjualan',[

            'data'=>Pelanggan::transaksiBy('id','desc')->get()

        ]);

    }

    public function store()

    {

        $this->validate([

            'pelanggan\_id'=>'required'

        ]);

        Transaksi::create([

            'invoice'=>$this->invoice(),

            'pelanggan\_id'=>$this->pelanggan\_id,

            'user\_id'=>Auth::user()->id,

            'total'=>'0'

        ]);

        $this->pelanggan\_id=NULL;

        return redirect()->to('transaksi');

    }

    public function invoice()

    {

        $transaksi=Transaksi::transaksiBy('created\_at','DESC');

        if($transaksi->count()>0){

            $transaksi=$transaksi->first();

            $explode=explode('-',$transaksi->invoice);

            return 'INV-'.$explode[1]+1;

        }

        return 'INV-1';

    }

}

1. Pada file Transaksis.php yang terdapat di dalam folder Http\Livewire tambahkan property pada class.

<?php

namespace App\Livewire;

use Exception;

use App\Models\Transaksi;

use App\Models\Layanan;

use Livewire\Component;

use App\Models\Detiltransaksi;

use Illuminate\Support\Facades\Auth;

use Illuminate\Support\Facades\Redirect;

class Transaksis extends Component

{

    public $total;

    public $transaksi\_id;

    public $barang\_id;

    public $qty=1;

    public $uang;

    public $kembali;

    public function render()

    {

        $transaksi=Transaksi::select('\*')->where('user\_id','=',Auth::user()->id)->transaksiBy('id','desc')->first();

        $this->total=$transaksi->total;

        $this->kembali=$this->uang-$this->total;

        return view('livewire.transaksis')

        ->with("data",$transaksi)

        ->with("dataBarang",Barang::where('stock','>','0')->get())

        ->with("dataDetiltransaksi",Detiltransaksi::where('transaksi\_id','=',$transaksi->id)->get());

    }

    public function store()

    {

        $this->validate([

            'layanan\_id'=>'required'

        ]);

        $transaksi=Transaksi::select('\*')->where('user\_id','=',Auth::user()->id)->transaksiBy('id','desc')->first();

        $this->transaksi\_id=$transaksi->id;

        $barang= Barang::where('id','=',$this-> barang\_id)->get();

        $harga=$layanan[0]->price;

        Detiltransaksi::create([

            'transaksi\_id'=>$this->transaksi\_id,

            'layanan\_id'=>$this-> barang\_id,

            'qty'=>$this->qty,

            'harga'=>$harga

        ]);

        $total=$transaksi->total;

        $total=$total+($harga\*$this->qty);

        Transaksi::where('id','=',$this->transaksi\_id)->update([

            'total'=>$total

        ]);

        $this->barang\_id=NULL;

        $this->qty=1;

    }

    public function delete($detiltransaksi\_id)

    {

        $detiltransaksi=Detiltransaksi::find($detiltransaksi\_id);

        $detiltransaksi->delete();

        //update total

        $detiltransaksi=Detiltransaksi::select('\*')->where('transaksi\_id','=',$this->transaksi\_id)->get();

        $total=0;

        foreach($detiltransaksi as $od){

            $total+=$od->qty\*$od->price;

        }

        try{

            Transaksi::where('id','=',$this->transaksi\_id)->update([

                'total'=>$total

            ]);

        }catch(Exception $e){

            dd($e);

        }

    }

    public function receipt($id)

    {

        $detiltransaksi = Detiltransaksi::select('\*')->where('transaksi\_id','=', $id)->get();

        //dd ($detiltransaksi);

        foreach ($detiltransaksi as $od) {

            $stocklama =Barang::select('stock')->where('id','=', $od->barang\_id)->sum('stock');

            $stock = $stocklama - $od->qty;

            try {

                Barang::where('id','=', $od->barang\_id)->update([

                    'stock' => $stock

                ]);

            } catch (Exception $e) {

                dd($e);

            }

        }

        return Redirect::route('cetakReceipt')->with(['id' => $id]);

    }

}

### **Menambahkan route**

1. Buka file web.php pada folder routes
2. Tambahkan route penjualan

Route::get('penjualan',function(){

    return view('penjualan.index',[

        "title"=>"Penjualan"

    ]);

})->name('penjualan')->middleware('auth');

1. Tambahkan route transaksi

Route::get('transaksi',function(){

return view('penjualan.transaksis',[

    "title"=>"Transaksi"

]);

})->middleware('auth');

1. Selanjutnya sebelum melakukan uji coba, buka file template.blade.php pada folder layouts, lalu perbaiki menu penjualan pada bagian side menu

<li class="nav-item">

                            <a href="penjualan" target="\_blank" class="nav-link {{ ($title==='Penjualan') ? 'active':'' }}">

                                <i class="nav-icon fas fa-cash-register"></i>

                                <p>

                                    Penjualan

                                </p>

                            </a>

                        </li>

1. Silahkan lakukan uji coba



### **Update Stok dan Cetak Receipt**

1. Perbaiki script tombol cetak pada file **views/livewire/transaksi.blade.php** seperti pada gambar berikut:

<div class="card-footer text-end">

                            <button type="submit" class="btn btn-success btn-sm"><i class="fas fa-cart-plus"></i> Tambahkan</button>

                        </div>

1. Selanjutnya buka file Transaksis.php pada folder App\Http\Livewire, lalu tambah function receipt berikut:

public function receipt($id)

    {

        $detiltransaksi = Detiltransaksi::select('\*')->where('transaksi\_id','=', $id)->get();

        //dd ($detiltransaksi);

        foreach ($detiltransaksi as $od) {

            $stocklama = Barang::select('stock')->where('id','=', $od->barang\_id)->sum('stock');

            $stock = $stocklama - $od->qty;

            try {

                Barang::where('id','=', $od->barang\_id)->update([

                    'stock' => $stock

                ]);

            } catch (Exception $e) {

                dd($e);

            }

        }

        return Redirect::route('cetakReceipt')->with(['id' => $id]);

    }

1. Buka file routes\web.php, tambahkan route berikut:

Route::get('cetakReceipt',[CetakController::class,'receipt'])->name('cetakReceipt')->middleware('auth');

1. Buat controlerr baru, dengan nama CetakController

php artisan make:Controller CetakController

1. Lalu jangan lupa import class CetakController ke dalam routes/web.php
2. Buka Controller CetakController lalu tambahkan script berikut:

<?php

namespace App\Http\Controllers;

use App\Models\Transaksi;

use Illuminate\View\View;

use App\Models\Detiltransaksi;

use Illuminate\Http\Request;

class CetakController extends Controller

{

    //

    public function receipt():View

    {

        $id=session()->get('id');

        $transaksi=Transaksi::find($id);

        //dd($transaksi)

        $detiltransaksi=Detiltransaksi::where('transaksi\_id',$id)->get();

        return view('penjualan.receipt')->with([

            'dataTransaksi'=>$transaksi,

            'dataDetiltransaksi'=>$detiltransaksi

        ]);

    }

}

1. Buat file baru didalam folder **views/penjualan** dengan nama **receipt.blade.php**
2. Tambahkan script berikut

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Cetak Receipt</title>

    <link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Inconsolata:wght@300;400&display=swap" rel="stylesheet">

    <style>

@page{

    margin:0;

    btransaksi:0;

    size:80mm,100%;

}

body{

    font-size: 9pt;

    font-family: 'Inconsolata', monospace;

    text-align: justify;

}

</style>

</head>

<body>

<p>invoice:{{$dataTransaksi->invoice}}<br>Pelanggan:{{ $dataTransaksi->pelanggan->name}}<br>Cashier:{{ Auth::user()->name}}<br>Tanggal:{{$dataTransaksi->created\_at->format('d M Y H:m')}}</p>

<table>

    <thead>

        <tr>

            <td>Barang</td>

            <td>Qty</td>

            <td>Price</td>

            <td>Amount</td>

        </tr>

    </thead>

@foreach($dataDetiltransaksi as $dod)

<tr>

            <td>{{$dod->barang->name }}</td>

            <td>{{$dod->qty}}</td>

            <td>@money($dod->price)</td>

            <td>@money($dod->harga\*$dod->qty)</td>

        </tr>

@endforeach

<tr>

    <td colspan="3">Total:</td>

    <td>@money($dataTransaksi->total)</td>

</tr>

</table>

<script src="{{ asset('plugins/jquery/jquery.min.js')}}"></script>

<script>

    $(document).ready(function(){

        window.print();

        setInterval(myURL,5000);

    });

    function myURL() {

        document.location.href = "{{route('penjualan') }}";

        // clearInterval(interval);

    }

    </script>

</body>

</html>

1. Lakukan uji coba kembali, pastikan stok terupdate dan semua menu dan tombol berfungsi dengan baik.

## Fitur Laporan

### **Membuat Controller Welcome**

* Buat controller laporan dengan menggunakan php artisan

php artisan make:controller LaporanController

* Tambahkan script berikut pada **LaporanController.php**

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Models\Transaksi;

class LaporanController extends Controller

{

    public function index(Request $request)

    {

        $query = Transaksi::count();

        if ($request->filled('q')) {

            $query = $query->where('id', 'LIKE', '%' . $request->q . '%');

        }

        if ($request->filled('tanggal\_mulai')) {

            $query = $query->where('created\_at', '>=', $request->tanggal\_mulai);

        }

        if ($request->filled('tanggal\_selesai')) {

            $query = $query->where('created\_at', '<=', $request->tanggal\_selesai);

        }

        $transaksi = Transaksi::All();

        $totalpenjualan = Transaksi::sum('total');

        $title = "Laporan Penjualan";

        if ($request->page == 'laporan') {

            return view('laporan.laporan', compact('transaksi', 'totalpenjualan', 'title'));

        }

        return view('laporan.laporan', compact('transaksi', 'totalpenjualan', 'title'));

    }

}

### **Modifikasi Template.blade**

<li class="nav-item">

                            <a href="{{ route('laporan.index')}}" class="nav-link {{ ($title==='Laporan')?'active':''}}">

                                <i class="nav-icon fas fa-file-pdf"></i>

                                <p>

                                    Laporan

                                </p>

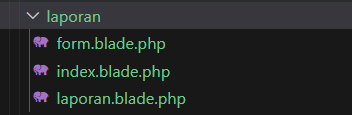
                            </a>

                        </li>

### **Modifikasi Route web**

Route::get('laporan', [LaporanController::class, 'index'])->name('laporan.index');

### **Modifikasi View laporan->laporan.blade**



@extends('layouts.template')

@section('tambahanCSS')

<!-- DataTables -->

<link rel="stylesheet" href="plugins/datatables-bs4/css/dataTables.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-responsive/css/responsive.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-buttons/css/buttons.bootstrap4.min.css">

<!-- Toastr -->

<link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.css">

@endsection

@section('js')

<script>

    $(document).ready(function() {

        $("#cetak").click(function(e) {

            var tanggalMulai = $("#tanggal\_mulai").val();

            var tanggalSelesai = $("#tanggal\_selesai").val();

            var q = $("#q").val();

            params = "?page=laporan&tanggal\_mulai=" + tanggalMulai + "&tanggal\_selesai=" +

                tanggalSelesai + "&q=" + q;

            window.open("{{ route('laporan.index') }}" + params, '\_blank');

        });

    });

</script>

@endsection

@section('konten')

<h1 class="h3 mb-3 text-center">Laporan Penjualan </h1>

<div class="col-md-12">

    <div class="card card-success">

        <div class="card-header">

            <h3 class="card-title">Anda login sebagai : {{ auth()->user()->name }}</h3>

        </div>

        <div class="card-body">

            <table id="example1" class="table table-bordered table-striped ">

                <thead>

                    <tr>

                        <th width="1%">No.</th>

                        <th width="25%">Tanggal</th>

                        <th>petugas</th>

                        <th>pelanggan</th>

                        <th>total</th>

                    </tr>

                </thead>

                <tbody>

                    @foreach ($transaksi as $laporan)

                    <tr>

                        <td>{{ $loop->iteration }}</td>

                        <td>{{ $laporan->created\_at->translatedFormat('d-m-Y') }}</td>

                        <td>{{ $laporan->user->nama }}</td>

                        <td>{{ $laporan->pelanggan->nama }}</td>

                        <td>{{ $laporan->total }}

                        </td>

                    </tr>

                    @endforeach

                </tbody>

                <tfoot>

                    <tr>

                        <td colspan="4" class="text-center fw-bold">TOTAL</td>

                        <td class="text-end">{{ $totalpenjualan }}</td>

                    </tr>

                </tfoot>

            </table>

        </div>

    </div>

</div>

@endsection

</div>

