LAMPIRAN

1. Model

1. Barang

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;
class Barang extends Model
{
protected $table = "barang";

protected $fillable = [
'kode',
'nama',
'slug',
'deskripsi',
'foto1',
'foto2',
'foto3',
'diskon',
'ukuran',
'warna',
'bahan',
'stok',
'kategori_id',
'subkategori_id',
'harga_beli',
'harga_jual',
'harga',
'rating',
'view',
    ];

/**
     * The attributes that should be hidden for
arrays.
     *
     * @var array
     */
protected $hidden = [

    ];
public function kategori()
    {
return $this-
```

```
>belongsTo('App\Kategori','kategori_id','id');
    }
public function subkategori()
    {
return $this-
>belongsTo('App\SubKategori','subkategori_id','id'
);
    }
public function transaksi()
    {
return $this-
>hasMany('App\Transaksi','barang_id','id');
    }


}
```

2. Kategori

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;

class Kategori extends Model
{
protected $table = "kategori";

protected $fillable = [
'nama',
'slug',
'deskripsi',
    ];

public function barang()
    {
return $this->hasMany('App\Barang',
'kategori_id','id');
    }
public function subkategori()
    {
return $this-
>hasMany('App\SubKategori','kategori_id','id');
    }
}
```

3. Konfirmasi

```php
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;
class Konfirmasi extends Model
{
protected $table = "konfirmasi";

protected  $fillable= [
'user_id',
'kode_pemesanan',
'bank',
'no_rek',
'atas_nama',
'jumlah',
'tanggal_transfer',
'foto',
'status'
];
}
```

4. SubKategori

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class SubKategori extends Model
{

protected $table = "subkategori";

protected $fillable = [
'kategori_id',
'nama',
'slug',
'deskripsi',
    ];
public function kategori()
    {
return $this->belongsTo('App\Kategori',
'kategori_id','id');
    }
}
```

5. Transaksi

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;

class Transaksi extends Model
{
protected $table = "transaksi";
protected  $fillable= [
'user_id',
'barang_id',
'jml',
'total_harga',
'tanggal_pesan',
'kode_pemesanan',
'metode_bayar',
'status',
'warna',
'bahan',

    ];


public function user()
    {
return $this->belongsTo('App\User', 'user_id'
,'id');
    }

public function barang()
    {
return $this->belongsTo('App\Barang', 'barang_id'
,'id');
    }
}
```

6. User

```
<?php

namespace App;

use Illuminate\Foundation\Auth\User as
Authenticatable;

class User extends Authenticatable
```

```
{
/**
     * The attributes that are mass assignable.
     *
     * @var array
     */
protected $table ="users";
protected $fillable = [
'nama',
'username',
'email',
'password',
'foto',
'tlp',
'alamat',
'kota',
'provinsi_id',
'kode_pos',
'level',
'status_email'
];

/**
     * The attributes that should be hidden for
arrays.

     */
protected $hidden = [
'password', 'remember_token',
    ];

public function tansaksi()
    {
return $this->hasMany('App\Transaksi', 'user_id'
,'id');
    }
}
```

2. Controller

7. AdminController

```
<?php
namespace App\Http\Controllers;
use App\Barang;
use App\Konfirmasi;
use App\Transaksi;
use App\User;
```

```
use Illuminate\Http\Request;

use App\Http\Requests;

class AdminController extends Controller
{

public function __construct()
    {
$this->middleware('isAdmin');
    }

public function beranda()
    {

$barang = Barang::all()->count();
$users = User::all()->count();
$transaksi = Transaksi::all()->count();
return view('admin.beranda', compact('barang',
'users', 'transaksi'));
    }

public function users()
    {

$users = User::all()->sortBy('nama');
$no = 1;
return view('admin.users.index', compact('users',
'no'));
    }

public function transaksi($data)
    {
if ($data == "sudah-membayar") {
$transaksi = Transaksi::where('status', 'Lunas')-
>get();
        } elseif ($data == "transaksi-sukses") {
$transaksi = Transaksi::where('status', 'Lunas &
Selesai')->get();
        } elseif ($data == "belum-membayar") {
$transaksi = Transaksi::where('status', 'Belum
Dibayar')->get();

        } else {
$transaksi = Transaksi::all();
        }
```

```
$no = 1;
$data_nav = $data;
return view('admin.transaksi.index',
compact('transaksi', 'no', 'data_nav'));
    }

public
    function konfirmasi($data)
    {
if ($data == "sudah-diverifikasi") {
$konfirmasi = Konfirmasi::where('status', 'Sudah
Diverifikasi')->get();
        } else if ($data == "belum-diverifikasi")
{
$konfirmasi = Konfirmasi::where('status', 'Belum
Diverifikasi')->get();

        } else {
$konfirmasi = Konfirmasi::all();
        }

$no = 1;
$data_nav = $data;
return view('admin.konfirmasi.index',
compact('konfirmasi', 'no', 'data_nav'));
    }

public
    function verifikasi($kode)
    {

$transaksi = Transaksi::where('kode_pemesanan',
$kode)->first();
if ($transaksi) {
$transaksi->update(['status' =>'Lunas']);
$konfirmasi = Konfirmasi::where('kode_pemesanan',
$kode)->first();

if ($konfirmasi) {
$konfirmasi->update(['status' =>'Sudah
Diverifikasi']);
            }

return back()->with('success', 'Proses vrifikasi
berhasil');
        } else {
            abort('404');
```

```
        }
    }

public
    function kirim($kode)
    {
$transaksi = Transaksi::where('kode_pemesanan',
$kode)->first();

if ($transaksi) {
$transaksi->update(['status' =>'Lunas &
Selesai']);
return back()->with('success', 'Barang siap
dikirim / diambil');
        } else {
            abort('404');
        }

    }

public function laporan(Request $request)
    {

if (isset($request->tgl_mulai) or isset($request-
>tgl_sampai)) {

$transaksi = Transaksi::where('status', '!=',
'Belum Dibayar')
                ->whereBetween('tanggal_pesan',
array($request->tgl_mulai, $request->tgl_sampai))
                ->get()-
>sortByDesc('tanggal_pesan');

        } else {
$transaksi = Transaksi::where('status', '!=',
'Belum Dibayar')
                ->get()-
>sortByDesc('tanggal_pesan');
        }

$no = 1;

return view('admin.laporan.index',
compact('transaksi', 'no'));
    }
}
```

8. **Barang Controller**

```
<?php

namespace App\Http\Controllers;

use App\Barang;
use App\Http\Requests\CreateBarangRequest;
use App\Http\Requests\UpdateBarangRequest;
use App\Kategori;
use App\SubKategori;
use App\User;
use Illuminate\Http\Request;

use App\Http\Requests;
use Illuminate\Support\Facades\Input;
use Intervention\Image\ImageManagerStatic as
Image;

class BarangController extends Controller
{
public function __construct()
    {
$this->middleware('isAdmin');
    }

public function index()
    {
$barang = Barang::all()-
>sortByDesc('updated_at');
$no = 1;

return view('admin.barang.index',
compact('barang', 'no'));
    }


/**
     * Show the form for creating a new
resource.
     *
     * @return \Illuminate\Http\Response
     */
public function create()
    {

$kategori = Kategori::all()->sortBy('nama')-
```

```
>lists('nama', 'id');
$subkategori = SubKategori::all()-
>sortBy('nama')->lists('nama', 'id');

return view('admin.barang.create',
compact('kategori', 'subkategori'));
    }

/**
     * Store a newly created resource in
storage.
     *
     * @param  \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
public function store(CreateBarangRequest
$request)
    {


if ($request->harga_jual <= $request-
>harga_beli) {
return back()->with('error', 'Harga jual harus
lebih besar dari harga beli')->withInput();
        } else {
//tentukan harga diskon

$diskon = $request->harga_jual * $request-
>diskon / 100;
$harga = $request->harga_jual - $diskon;

$ukuran1 = filesize($request->file('foto1')) /
1000;
$ukuran2 = filesize($request->file('foto2')) /
1000;
$ukuran3 = filesize($request->file('foto3')) /
1000;

if ($ukuran1 >= 501 OR $ukuran2 >= 501 OR
$ukuran3 >= 501) {

return back()->with('error', 'Gagal Upload.
Pastikan ukuran foto dari barang maksimal 500
kilo byte')->withInput();
            } else {
```

```
$image1 = $request->file('foto1');
$image2 = $request->file('foto2');
$image3 = $request->file('foto3');
$filename1 = $request->kode . '-1.' . $image1-
>getClientOriginalExtension();
$filename2 = $request->kode . '-2.' . $image2-
>getClientOriginalExtension();
$filename3 = $request->kode . '-3.' . $image3-
>getClientOriginalExtension();

$path1 = 'assets/images/barang/' . $filename1;
$path2 = 'assets/images/barang/' . $filename2;
$path3 = 'assets/images/barang/' . $filename3;
                Image::make($image1-
>getRealPath())->save($path1, 50);
                Image::make($image2-
>getRealPath())->save($path2, 50);
                Image::make($image3-
>getRealPath())->save($path3, 50);

$path4 = 'assets/images/barang/thumbnail/' .
$filename1;
$path5 = 'assets/images/barang/thumbnail/' .
$filename2;
$path6 = 'assets/images/barang/thumbnail/' .
$filename3;
                Image::make($image1-
>getRealPath())->fit(500)->save($path4, 50);
                Image::make($image2-
>getRealPath())->fit(500)->save($path5, 50);
                Image::make($image3-
>getRealPath())->fit(500)->save($path6, 50);


                Barang::create([
'kode' =>$request->kode,
'nama' =>$request->nama,
'slug' => str_slug($request->nama),
'deskripsi' =>$request->deskripsi,
'harga_beli' =>$request->harga_beli,
'harga_jual' =>$request->harga_jual,
'ukuran' =>$request->ukuran,
'stok' =>$request->stok,
'diskon' =>$request->diskon,
'harga' =>$harga,
'kategori_id' =>$request->kategori_id,
'subkategori_id' =>$request->subkategori_id,
```

```
'foto1' =>$filename1,
'foto2' =>$filename2,
'foto3' =>$filename3,

                ]);


return back()->with('success', 'Data barang
berhasil ditambahkan');
            }
        }
    }

/**
     * Display the specified resource.
     *
     * @param  int $id
     * @return \Illuminate\Http\Response
     */
public function show($id)
    {
$barang = Barang::findOrFail($id);


return view('admin.barang.single',
compact('barang'));
    }

public function edit($id)
    {

$barang = Barang::findOrFail($id);

$kategori = Kategori::all()->sortBy('nama')-
>lists('nama', 'id');
$subkategori = SubKategori::all()-
>sortBy('nama')->lists('nama', 'id');
return view('admin.barang.edit',
compact('kategori', 'subkategori', 'barang'));
    }
public function update(UpdateBarangRequest
$request, $id)
    {
if ($request->harga_jual <= $request-
>harga_beli) {
return back()->with('error', 'Harga jual harus
lebih besar dari harga beli')->withInput();
```

```
        } else {
//tentukan harga diskon

$diskon = $request->harga_jual * $request-
>diskon / 100;
$harga = $request->harga_jual - $diskon;


$barang = Barang::findOrFail($id);


if ($request->hasFile('foto1') or $request-
>hasFile('foto2') or $request->hasFile('foto3'))
{

$ukuran1 = filesize($request->file('foto1')) /
1000;
$ukuran2 = filesize($request->file('foto2')) /
1000;
$ukuran3 = filesize($request->file('foto3')) /
1000;

if ($ukuran1 >= 501 OR $ukuran2 >= 501 OR
$ukuran3 >= 501) {

return back()->with('error', 'Gagal Upload.
Pastikan ukuran foto dari barang maksimal 500
kilo byte')->withInput();
                } else {

$image1 = $request->file('foto1');
$image2 = $request->file('foto2');
$image3 = $request->file('foto3');
$filename1 = $request->kode . '-1.' . $image1-
>getClientOriginalExtension();
$filename2 = $request->kode . '-2.' . $image2-
>getClientOriginalExtension();
$filename3 = $request->kode . '-3.' . $image3-
>getClientOriginalExtension();

$path1 = 'assets/images/barang/' . $filename1;
$path2 = 'assets/images/barang/' . $filename2;
$path3 = 'assets/images/barang/' . $filename3;
                    Image::make($image1-
>getRealPath())->save($path1, 50);
                    Image::make($image2-
>getRealPath())->save($path2, 50);
```

```
                    Image::make($image3-
>getRealPath())->save($path3, 50);

$path4 = 'assets/images/barang/thumbnail/' .
$filename1;
$path5 = 'assets/images/barang/thumbnail/' .
$filename2;
$path6 = 'assets/images/barang/thumbnail/' .
$filename3;
                    Image::make($image1-
>getRealPath())->fit(500)->save($path4, 50);
                    Image::make($image2-
>getRealPath())->fit(500)->save($path5, 50);
                    Image::make($image3-
>getRealPath())->fit(500)->save($path6, 50);

                }


            } else {

$filename1 = $barang->foto1;
$filename2 = $barang->foto2;
$filename3 = $barang->foto3;
            }

$barang->update([
'kode' =>$request->kode,
'nama' =>$request->nama,
'slug' => str_slug($request->nama),
'deskripsi' =>$request->deskripsi,
'harga_beli' =>$request->harga_beli,
'harga_jual' =>$request->harga_jual,
'ukuran' =>$request->ukuran,
'stok' =>$request->stok,
'diskon' =>$request->diskon,
'harga' =>$harga,
'kategori_id' =>$request->kategori_id,
'subkategori_id' =>$request->subkategori_id,
'foto1' =>$filename1,
'foto2' =>$filename2,
'foto3' =>$filename3,

            ]);


return redirect('admin/barang')->with('success',
```

```
'Data barang berhasil diubah menjadi ' .
$request->nama . '');

        }
    }

/**
     * Remove the specified resource from
storage.
     *
     * @param  int $id
     * @return \Illuminate\Http\Response
     */
public function destroy($id)
    {
$barang = Barang::findOrFail($id);

if ($barang->foto1 != "default.png") {

unlink('assets/images/barang/' . $barang-
>foto1);
unlink('assets/images/barang/' . $barang-
>foto2);
unlink('assets/images/barang/' . $barang-
>foto3);
        }

$barang->delete();

return redirect('admin/barang')->with('success',
'Data barang berhasildihapus');

    }

}
```

9. **Cart Controller**

```
<?php

namespace App\Http\Controllers;

use App\Barang;
use App\Transaksi;
use App\User;
use Carbon\Carbon;
use Gloudemans\Shoppingcart\Facades\Cart;
```

```
use Illuminate\Http\Request;

use App\Http\Requests;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Cache;
use Krucas\Notification\Facades\Notification;

class CartController extends Controller
{

/*public function __construct()
    {
        $this->middleware('isUser');
    }*/

public function get_data(Request $request, $id)
    {

if (empty($request->jml_barang)) {
$jml = 1;
        } else {
$jml = $request->jml_barang;
        }
if (empty($request->warna)) {
$warna = 'Cokelat';
        } else {
$warna = $request->warna;
        }

if (empty($request->bahan)) {
$bahan = 'Kayu Jati';
        } else {
$bahan = $request->bahan;
        }

$barang = Barang::findOrFail($id);

$id = $barang->id;
$name = $barang->nama;
$qty = $jml;
$price = $barang->harga;
$foto1 = $barang->foto1;
$slug = $barang->slug;

$data = array('id' =>$id,
'name' =>$name,
'qty' =>$qty,
```

```
'price' =>$price,
'options' =>array(
'slug' =>$slug,
'foto' =>$foto1,
'bahan' =>$bahan,
'warna' =>$warna
));

        Cart::add($data);


return back()->with('success', 'Barang berhasil
ditambahkan pada keranjang belanja anda');
    }

public function delete_data($id)
    {
        Cart::remove($id);
return back();
    }

public function destroy()
    {
        Cart::destroy();

return back()->with('success', 'Keranjang belanja
telah dikosongkan');
    }

public function update_data($id, $qty)
    {
$rowId = $id;
$jml = $qty - 1;

        Cart::update($rowId, $jml); // Will
update the quantity
return back()->with('success', 'Barang berhasil
dihapus dari keranjang belanja anda');
    }

public function checkout(Request $request)
    {
$kode = str_random(8);
$cart_content = Cart::content(1);

foreach ($cart_content as $cart) {
```

```
/*$barang = Barang::find($cart->id);*/
Transaksi::create([
'user_id' => Auth::user()->id,
'barang_id' =>$cart->id,
'warna' =>$cart->options->warna,
'bahan' =>$cart->options->bahan,
'jml' =>$cart->qty,
'total_harga' =>$cart->qty * $cart->price,
'tanggal_pesan' =>$today = Carbon::now(),
'kode_pemesanan' =>$kode,
'metode_bayar' =>"Transfer",
'status' =>'Belum Dibayar',
            ]);

/* $barang->update([
                'stok' => $barang->stok - $cart->qty
            ]);*/
}
            Cart::destroy();

return redirect('checkout/pembayaran/'.$kode)
                ->with('success', 'Barang berhasil dipesan. Tunggu konfirmasi dari kami')
                ->with('info', 'Silahkan melakukan pembayaran sesuai petunjuk berikut');
        }
    }
```

10. **Checkout Controller**

```
<?php

namespace App\Http\Controllers;

use App\Provinsi;
use App\Transaksi;
use App\User;
use Gloudemans\Shoppingcart\Facades\Cart;
use Illuminate\Http\Request;

use App\Http\Requests;
use Illuminate\Support\Facades\Auth;

class CheckoutController extends Controller
{
```

```
public function pesanan()
    {
$cart_content = Cart::content();
if (Cart::count() <= 0) {

return redirect('keranjang-belanja');

        } else {
return view('users.checkout.pesanan',
compact('cart_content'));
        }
    }

public function biodata()
    {
$user = User::findOrFail(Auth::user()->id);
return view('users.checkout.biodata',
compact('user'));
    }

public function post_biodata(Request $request)
    {


$this->validate($request, [
'nama' =>'required',
'tlp' =>'required',
'alamat' =>'required',


        ]);

$user = User::findOrFail(Auth::user()->id);
$user->update([
'nama' =>$request->nama,
'tlp' =>$request->tlp,
'alamat' =>$request->alamat,

        ]);

return redirect('cart/checkout');
    }

public function pembayaran($kode)
    {
```

```
$pesanan = Transaksi::where('kode_pemesanan',
$kode)->get();

if ($pesanan) {
return view('users.checkout.cetak',
compact('pesanan', 'kode'));
        } else {
return abort('404');
        }
    }

public function batalkan($kode)
    {
        Transaksi::where('kode_pemesanan',
$kode)->delete();

return redirect('user/riwayat-pemesanan')-
>with('success', 'Pesanan anda berhasil di
batalkan');

    }

public function cetak_nota($kode)
    {

$pesanan = Transaksi::where('kode_pemesanan',
$kode)->get();

if ($pesanan) {
return view('users.checkout.cetak_nota',
compact('pesanan', 'kode'));
        } else {
return abort('404');
        }
    }
}
```

11. **Home Controller**

```
<?php

namespace App\Http\Controllers;

use App\Http\Requests;
use Illuminate\Http\Request;

class HomeController extends Controller
```

```
{
/**
     * Create a new controller instance.
     *
     * @return void
     */
public function __construct()
    {
$this->middleware('auth');
    }

/**
     * Show the application dashboard.
     *
     * @return \Illuminate\Http\Response
     */
public function index()
    {
return view('home');
    }
}
```

12. **KategoriController**

```
<?php

namespace App\Http\Controllers;

use App\Barang;
use App\Galeri;
use App\Kategori;
use App\Pegawai;
use App\Osis;
use App\Periode;
use App\SubKategori;
use Illuminate\Http\Request;

use App\Http\Requests;
use Illuminate\Support\Facades\Input;
use Illuminate\Support\Facades\Validator;
use Intervention\Image\ImageManagerStatic as
Image;

class KategoriController extends Controller
{
public function __construct()
    {
```

```
$this->middleware('isAdmin');
    }
public function index()
    {
$subkategori = SubKategori::all();

$kategori = Kategori::with('subkategori')->get();
$kategori_id= Kategori::lists('nama','id');
$no = 1;

return view('admin.kategori.index',
compact('kategori', 'no',
'subkategori','kategori_id'));
    }
public function store(Request $request)
    {

$validator = Validator::make($request->all(), [

'nama' =>'required|unique:kategori',
'deskripsi' =>'required'
]);

if ($validator->fails()) {
return back()
                ->withErrors($validator)
                ->withInput()
                ->with('tambah_kategori',
'error');

        } else {

            Kategori::create([
'nama'=>$request->nama,
'slug'=>str_slug($request->nama),
'deskripsi'=>$request->deskripsi
]);

return back()->with('success', 'Kategori berhasil
ditambahkan');

        }
    }
public function update(Request $request, $id)
    {
$validator = Validator::make($request->all(), [
```

```
'nama' =>'required|unique:kategori,nama,'.$id,
'deskripsi' =>'required'
]);

if ($validator->fails()) {
return back()
                ->withErrors($validator)
                ->withInput()
                ->with('edit_kategori', 'error');
        } else {

$kategori = Kategori::findOrFail($id);
$kategori->update([
'nama'=>$request->nama,
'slug'=>str_slug($request->nama),
'deskripsi'=>$request->deskripsi
]);

return back()->with('success', 'Kategori berhasil
diubah');
        }

    }
public function destroy($id)
        {
$kategori = Kategori::findOrFail($id);
$cek = Barang::where('kategori_id',$id)->count();
if ($cek>=1) {
return back()->with('error', 'Gagal menghapus.
Terdapat barang yang menggunakan kategori ini');;
            } else {
$kategori->delete();
return back()->with('success', 'Kategori berhasil
dihapus');
            }
        }
    }
```

13. **Pendaftaran Controller**

```
<?php

namespace App\Http\Controllers;

use App\Provinsi;
use App\User;
use Illuminate\Http\Request;
```

```
use App\Http\Requests;
use Illuminate\Support\Facades\Auth;

class PendaftaranController extends Controller
{
public function __construct()
    {
$this->middleware('isUser');
    }
public function siswa()
    {
$user = User::findOrFail(Auth::user()->id);
$provinsi = Provinsi::all()->sortBy('name')-
>lists('name', 'name');

return view('users.users.form.siswa',
compact('provinsi', 'user'));
    }
public function update_siswa(Request $request)
    {
//dd($request->all());
$this->validate($request, [
'nama' =>'required',
'tempat_lahir' =>'required',
'tanggal_lahir' =>'required',
'jenis_kelamin' =>'required',
'agama' =>'required',
'alamat' =>'required',
'kota' =>'required',
'provinsi' =>'required',
'no_telp' =>'required',
'no_hp' =>'required',
        ]);

$user = User::findOrFail(Auth::user()->id);
$user->update($request->all());
        flash()->success('Data users berhasil
disimpan');
return redirect('user/users/orang-tua');
    }

public function orang_tua()
    {
$user = User::findOrFail(Auth::user()->id);
return view('users.users.form.orang_tua',
compact('user'));
    }
```

```
public function update_orang_tua(Request
$request)
    {
//dd($request->all());
$this->validate($request, [

'nama_ayah' =>'required',
'pekerjaan_ayah' =>'required',
'telp_ayah' =>'required',
'nama_ibu' =>'required',
'pekerjaan_ibu' =>'required',
'telp_ibu' =>'required',
'alamat_ortu' =>'required',
'kota_ortu' =>'required',
'nama_wali' =>'required',
'alamat_wali' =>'required',
'pekerjaan_wali' =>'required',
'telp_wali' =>'required',

        ]);

$user = User::findOrFail(Auth::user()->id);
$user->update($request->all());

        flash()->success('Data users berhasil
disimpan');
return redirect('user/users/asal-sekolah  ');
    }

public function asal_sekolah()
    {
$user = User::findOrFail(Auth::user()->id);
return view('users.users.form.asal_sekolah',
compact('user'));
    }
public function update_asal_sekolah(Request
$request)
    {
//dd($request->all());
$this->validate($request, [

'nomor_ijazah' =>'required',
'tanggal_ijazah' =>'required',
'nomor_stl' =>'required',
'tanggal_stl' =>'required',
'asal_smp' =>'required',
'alamat_smp' =>'required',
```

```
'kota_smp' =>'required',
'prestasi' =>'required',
        ]);
$user = User::findOrFail(Auth::user()->id);
$user->update($request->all());
        flash()->success('Data users berhasil
disimpan');
return redirect('user/users/program-keahlian
');
    }
public function program_keahlian()
    {
$user = User::findOrFail(Auth::user()->id);
return view('users.users.form.program_keahlian',
compact('user'));
    }
public function update_program_keahlian(Request
$request)
    {
//dd($request->all());
$this->validate($request, [

'pilihan_1' =>'required',
'pilihan_2' =>'required',

        ]);

$user = User::findOrFail(Auth::user()->id);
$user->update($request->all());
        flash()->success('Seluruh data telah
berhasil disimpan');
return redirect('user/users/daftar');
    }

public function daftar_final()
    {

$user = User::findOrFail(Auth::user()->id);
return view('users.users.form.daftar',
compact('user'));
    }
public function update_daftar(Request $request)
    {
//dd($request->all());

$user = User::findOrFail(Auth::user()->id);
$user->update($request->all());
```

```
        flash()->success('Pendaftaran
berhasil!');
return redirect('user/users/daftar');
    }
}
```

14. **Sub kategori Controller**

```
<?php
namespace App\Http\Controllers;
use App\Barang;
use App\Galeri;
use App\Kategori;
use App\Pegawai;
use App\Osis;
use App\Periode;
use App\SubKategori;
use Illuminate\Http\Request;
use App\Http\Requests;
use Illuminate\Support\Facades\Input;
use Illuminate\Support\Facades\Validator;
use Intervention\Image\ImageManagerStatic as
Image;

class SubkategoriController extends Controller
{
public function __construct()
    {
$this->middleware('isAdmin');
    }
public function index()
    {
    }
public function store(Request $request)
    {

$validator = Validator::make($request->all(), [

'nama' =>'required|unique:subkategori',
'kategori_id' =>'required',
'deskripsi' =>'required'
]);

if ($validator->fails()) {
return back()
                ->withErrors($validator)
```

```
                ->withInput()
                ->with('tambah_subkategori',
'error');

        } else {

            SubKategori::create([
'kategori_id'=>$request->kategori_id,
'nama'=>$request->nama,
'slug'=>str_slug($request->nama).".html",
'deskripsi'=>$request->deskripsi
]);
return back()->with('success', 'Sub Kategori
berhasil ditambahkan');;
        }
    }
public function update(Request $request)
    {
$validator = Validator::make($request->all(), [
'nama'
=>'required|unique:subkategori,nama,'.$request-
>id,
'kategori_id' =>'required',
'deskripsi' =>'required'
]);
if ($validator->fails()) {
return back()
                ->withErrors($validator)
                ->withInput()
                ->with('edit_subkategori',
'error');
        } else {

$kategori = SubKategori::findOrFail($request-
>id);
$kategori->update([
'kategori_id'=>$request->kategori_id,
'nama'=>$request->nama,
'slug'=>str_slug($request->nama).".html",
'deskripsi'=>$request->deskripsi
]);
return back()->with('success', 'Sub Kategori
berhasil diubah');
        }
    }
public function destroy(Request $request)
        {
```

```
$subkategori = SubKategori::findOrFail($request-
>id);
$cek = Barang::where('subkategori_id',$request-
>id)->count();
if ($cek>=1) {
return back()->with('error', 'Gagal menghapus.
Terdapat barang yang menggunakan subkategori
ini');
            } else {
$subkategori->delete();
return back()->with('success', 'Sub Kategori
berhasil dihapus');
            }
        }
    }
```

15. **UserController**

```
<?php
namespace App\Http\Controllers;
use App\Transaksi;
use App\User;
use Illuminate\Http\Request;
use App\Http\Requests;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;
use Intervention\Image\Facades\Image;

class UserController extends Controller
{
public function __construct()
    {
$this->middleware('isUser');
    }
public function riwayat_pemesanan()
    {
$riwayat_pemesanan =
Transaksi::groupBy('kode_pemesanan')-
>orderBy('tanggal_pesan', 'desc')-
>where('user_id', Auth::user()->id)-
>paginate(10);
$jumlah_transaksi = $riwayat_pemesanan->count();
return view('users.riwayat_pemesanan',
compact('riwayat_pemesanan',
'jumlah_transaksi'));
    }
```

```
public function profil()
    {
return view('users.user_profil');
    }
public function ubah_foto(Request $request)
    {
$id = Auth::user()->id;
$this->validate($request, [

'foto' =>'required',
        ]);

$user = User::findOrFail($id);

if ($request->hasFile('foto')) {

$size = filesize($request->file('foto'));

$ukuran = $size / 1000;

if ($ukuran <= 500) {

$image = $request->file('foto');

$filename = $request->nis . '.' . $image-
>getClientOriginalExtension();
$path = 'assets/images/pengguna/' . $filename;
                Image::make($image-
>getRealPath())->fit(2000)->save($path, 50);

$user->update(['foto' =>$filename]);
return back()->with('success','Foto berhasil
dirubah');
            } else {
return back()->with('error','Gagal upload, foto
terlalu besar. maksimal ukuran gambar ialah
500kb');
            }
        }
    }
public function ubah_akun(Request $request)
    {
$id = Auth::user()->id;
$this->validate($request, [
'nama' =>'required',
'username' =>'required|unique:users,username,' .
$id,
```

```
'email' =>'required|email|unique:users,email,' .
$id,
        ]);

$user = User::findOrFail($id);
$user->update($request->all());
return back()->with('success','Data akun berhasil
diubah');

    }
public function ubah_password(Request $request)
    {
$id = Auth::user()->id;
$this->validate($request, [

'password_lama' =>'required',
'password' =>'required|min:6|confirmed'
]);
$user = User::findOrFail($id);
if (Hash::check($request->password_lama, $user-
>password)) {
$user->update(['password' => bcrypt($request-
>password)]);
return back()->with('success','Password berhasil
dirubah');
        } else {
return back()->with('error','Password lama anda
tidak sesuai');
        }
    }
}
```

16. **Web Controller**

```
<?php namespace App\Http\Controllers;
use App\Barang;
use App\Berita;
use App\Galeri;
use App\Http\Requests;
use App\Http\Controllers\Controller;
use App\Konfirmasi;
use App\Provinsi;
use App\SubKategori;
use App\Transaksi;
use App\User;
use Gloudemans\Shoppingcart\Facades\Cart;
use Illuminate\Http\Request;
```

```
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Mail;
use Intervention\Image\Facades\Image;

class WebController extends Controller
{
public function beranda()
    {
        $beranda = 'beranda';
        $barang = Barang::OrderBy('updated_at',
'desc')->paginate('8');
        $diskon = Barang::all()-
>sortByDesc('diskon')->take('4');
        $populer = Barang::all()-
>sortByDesc('view')->take('4');
        $murah = Barang::all()->sortBy('harga')-
>take('4');
        $subkategori =
SubKategori::OrderBy('nama', 'asc')->get();
return view('users.beranda', compact('beranda',
'no', 'barang', 'populer', 'murah', 'diskon',
'subkategori'));
    }
public function daftar_barang($kategori,
$subkategori)
    {
        $judul = SubKategori::where('slug',
$subkategori)->first();

if ($judul) {
            $barang =
Barang::where('subkategori_id', $judul->id)-
>orderBy('nama', 'asc')->paginate(8);

        } else {
            $barang = Barang::with('kategori')-
>paginate(4)->where('kategori.slug', $kategori)-
>sortBy('barang.nama');
        }

        $jml = $barang->count();

        $most_populer = Transaksi::all()-
>groupBy('barang_id')->min()->first();
return view('users.daftar_barang',
compact('barang', 'kategori', 'judul', 'jml',
'most_populer'));
```

```
    }
public function detail($slug)
    {

        $barang = Barang::where('slug', $slug)-
>first();
if ($barang) {
            $barang->update([
'view' => $barang->view + 1
            ]);
            $barang_lain =
Barang::where('subkategori_id', $barang-
>subkategori_id)->orderBy('nama', 'view')->get()-
>take('8');
return view('users.detail', compact('barang',
'barang_lain'));
        } else {
return abort(404);
        }
    }

public function cart()
    {
        $cart_content = Cart::content();
return view('users.cart',
compact('cart_content'));
    }


public function konfirmasi(Request $request)
    {
        $kode = $request->kode;

return view('users.konfirmasi', compact('kode'));
    }

public function post_konfirmasi(Request $request)
    {

        $this->validate($request, [
'kode_pemesanan' =>'required',
'bank' =>'required',
'nomor_rekening' =>'required',
'atas_nama' =>'required',
'jumlah' =>'required',
'tanggal_transfer' =>'required',
'foto' =>'required|image',
```

```
        ]);

        $cek = Transaksi::where('kode_pemesanan',
$request->kode_pemesanan)->first();

if ($cek) {

            $size = filesize($request-
>file('foto'));
            $ukuran = $size / 1000;

if ($ukuran <= 500) {

                $image = $request->file('foto');

                $filename = $request-
>kode_pemesanan . '.' . $image-
>getClientOriginalExtension();
                $path =
'assets/images/konfirmasi/' . $filename;
                Image::make($image-
>getRealPath())->save($path, 50);

                $cek2 =
Konfirmasi::where('kode_pemesanan', $request-
>kode_pemesanan)->first();

if ($cek2) {
                    $cek2->update([
'user_id' => Auth::user()->id,
'kode_pemesanan' => $request->kode_pemesanan,
'bank' => $request->bank,
'no_rek' => $request->nomor_rekening,
'atas_nama' => $request->atas_nama,
'jumlah' => $request->jumlah,
'tanggal_transfer' => $request->tanggal_transfer,
'foto' => $filename
                    ]);
                } else {
                    Konfirmasi::create([
'user_id' => Auth::user()->id,
'kode_pemesanan' => $request->kode_pemesanan,
'bank' => $request->bank,
'no_rek' => $request->nomor_rekening,
```

```
'atas_nama' => $request->atas_nama,
'jumlah' => $request->jumlah,
'tanggal_transfer' => $request->tanggal_transfer,
'foto' => $filename
                    ]);
                }

return back()->with('success', 'Upload Berhasil. Kami akan segera mengkonfirmasi pembayaran anda');

            } else {
return back()->withInput()->with('error', 'Gagal upload, foto terlalu besar. maksimal ukuran gambar ialah 500kb');
            }

        } else {

return back()->withInput()->with('error', 'Kode tidak ditemukan. Cek kode pemesanan anda !');
        }

    }


public function contact()
    {
return view('users.contact');
    }

public function register()
    {
if (Auth::check()) {
return redirect('/');
        } else {
return view('users.register');
        }

    }

public function post_register(Request $request)
    {
        $this->validate($request, [
'nama' =>'required',
'username' =>'required|unique:users',
'email' =>'required|unique:users',
```

```
'password' =>'required|min:6|confirmed'
]);

        User::create([
'nama' => $request->nama,
'username' => $request->username,
'email' => $request->email,
'password' => bcrypt($request->password)
        ]);

return redirect('user/login')->with('success',
'Pendaftaran berhasil. Silahkan masuk!');

    }


public function login()
    {
if (Auth::check()) {
return redirect('/');
        } else {
return view('users.login');
        }

    }

public function post_login(Request $request)
    {
        $this->validate($request, [
'email' =>'required',
'password' =>'required|min:6'
]);


if (Auth::attempt(['email' => $request->email,
'password' => $request->password]) OR
Auth::attempt(['username' => $request->email,
'password' => $request->password])) {

if (Auth::user()->level == "admin") {
return redirect('admin');
            } else {
return redirect()->intended('/')->with('success',
'Selamat Datang' . Auth::user()->nama . '');
            }

        } else {
```

```
return back()->with('error', 'Login gagal. Cek
Username dan Password anda');

        }


    }


public function lupa_password()
    {

return view('users.lupa_password',
compact('form'));
    }

public function reset_password(Request $request)
    {

        $this->validate($request, [
'email' =>'required|email',
        ]);

        $users = User::where('email', $request-
>email)->first();
if (is_null($users)) {
            flash()->error('<p style="text-align:
center">Email tidak ditemukan... <br> Coba email
lain</p>');
return back();

        } else {
            $nama = $users->nama;
            $kode = str_random(6);
            $users->password = bcrypt($kode);
            $users->update();


            Mail::send('emails.password_reset',
['name' => $nama, 'kode' => $kode], function
($message) use ($request, $nama) {
                $message->to($request->email,
$nama)
```

```
                        -
>from('info.garvan@gmail.com')
                        ->subject('Reset Password');
                });

                flash()->success('<p style="text-
align: center"> Password berhasil di Reset. Cek
email untuk meilhat password! </p>');
return redirect('user/lupa-password');
            }
        }
```

1. **Routing**

```
//Admin Routes

Route::get('admin', 'AdminController@beranda');
Route::get('admin/laporan',
'AdminController@laporan');
Route::get('admin/transaksi/{data}',
'AdminController@transaksi');
Route::post('admin/transaksi/kirim/{kode}',
'AdminController@kirim');
Route::get('admin/konfirmasi/{data}',
'AdminController@konfirmasi');
Route::get('admin/verifikasi/{kode}',
'AdminController@verifikasi');
Route::get('admin/users', 'AdminController@users');
Route::resource('admin/barang', 'BarangController');
Route::resource('admin/kategori',
'KategoriController', ['except' => [
'show', 'edit', 'create'
]]);
```

```
Route::post('admin/subkategori',
'SubkategoriController@store');
Route::post('admin/subkategori/update',
'SubkategoriController@update');
Route::post('admin/subkategori/delete',
'SubkategoriController@destroy');

//User Routes
Route::get('/', 'WebController@beranda');
Route::get('/detail/{slug}', 'WebController@detail');
Route::get('/cari/{kategori}/{subkategori}',
'WebController@daftar_barang');
Route::get('keranjang-belanja',
'WebController@cart');

Route::get('checkout/pesanan',
'CheckoutController@pesanan')->middleware('isCart');
Route::get('checkout/biodata',
'CheckoutController@biodata')->middleware('isCart');
Route::post('checkout/biodata',
'CheckoutController@post_biodata')-
>middleware('isCart');
Route::get('checkout/pembayaran/{kode}',
'CheckoutController@pembayaran')-
>middleware('isUser');
Route::get('checkout/batalkan/{kode}',
'CheckoutController@batalkan')->middleware('isUser');


Route::get('konfirmasi-pembayaran',
```

```
'WebController@konfirmasi')->middleware('isUser');
Route::post('konfirmasi-pembayaran',
'WebController@post_konfirmasi')-
>middleware('isUser');
Route::get('hubungi-kami', 'WebController@contact');


Route::get('user/lupa-password',
'WebController@lupa_password');
Route::post('user/lupa-password',
'WebController@reset_password');
Route::get('user/login', 'WebController@login');
Route::get('user/register',
'WebController@register');
Route::post('user/post_login',
'WebController@post_login');
Route::post('user/post_register',
'WebController@post_register');


Route::get('user/profil/{user}',
'UserController@profil');
Route::post('user/pengaturan/akun',
'UserController@ubah_akun');
Route::post('user/pengaturan/foto',
'UserController@ubah_foto');
Route::post('user/pengaturan/password',
'UserController@ubah_password');
Route::get('user/riwayat-pemesanan',
'UserController@riwayat_pemesanan');
```

```
Route::get('cart/destroy' ,'CartController@destroy');
Route::get('cart/checkout',
'CartController@checkout')->middleware('isUser');
Route::get('cart/{id}', 'CartController@get_data');
Route::get('cart/delete/{id}'
,'CartController@delete_data');
Route::get('cart/update/{id}/{qty}'
,'CartController@update_data');
```

2.

```
Route::auth();

Route::get('/home', 'HomeController@index');
```