LAMPIRAN

- **CONTROLLERS**

ADMINCONTROLLERS.PHP

```
<?php
namespace App\Http\Controllers;
use Inertia\Inertia;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Facades\Redirect;
use Illuminate\Support\Str;
use Illuminate\Http\Request;
use Illuminate\Validation\Rule;
use Illuminate\Support\Facades\Auth;
use App\Models\User;
use Spatie\Permission\Models\Role;
use Illuminate\Http\RedirectResponse;
use Illuminate\Support\Facades\Hash;
class AdminController extends Controller
{
    public function index()
    {
        $user = User::with('roles')->get();
        return Inertia::render('Admin/index', [
            'user' => $user
        ]);
    }
    public function destroy(User $user)
    {
        $user->delete();
        return Redirect::route('admin.index')-
>with('success', 'Pengguna berhasil di hapus');
    }
    public function showCheckEmailForm()
    {
        return Inertia::render('Admin/CheckEmail');
    }
    // Memvalidasi email dan menampilkan
halaman ganti password
    public function checkEmail(Request $request)
    {
        $request->validate([
            'email' => 'required|email',
        ]);
        $user = User::where('email', $request-
>input('email'))->first();
        if ($user) {
            return
Inertia::render('Admin/Gantipassword', [
                'userId' => $user->id,
            ]);
        }
        return redirect()->back()-
>withErrors(['email' => 'Email tidak
ditemukan']);
    }
    public function updatePassword(Request
$request)
    {
        $request->validate([
            'user_id' => 'required|exists:users,id',
            'password' => 'required|confirmed|min:8',
        ]);
        $user = User::find($request-
>input('user_id'));
        if ($user) {
            $user->update([
                'password' => bcrypt($request-
>input('password')),
            ]);
            return redirect()->route('admin.index')-
>with('success', 'Password berhasil diubah');
        }
        return redirect()->back()-
>withErrors(['user_id' => 'User tidak valid']);
    }
}
```

CARCONTROLLER.PHP

```
<?php
namespace App\Http\Controllers;
use App\Models\Cart;
use App\Models\CartItem;
use App\Models\Produk;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\DB;
class CartController extends Controller
{
    public function index()
    {
        $cart = Cart::with('items.produk')-
>where('user_id', Auth::id())->first();
        return inertia('Cart/Index', [
            'cart' => $cart,
        ]);
    }
    public function store(Request $request)
    {
        $request->validate([
            'produk_id' =>
'required|exists:produks,id',
            'quantity' => 'required|integer|min:1',
        ]);
        DB::beginTransaction(); // <- Mulai
transaksi database
        try {
            $produk = Produk::lockForUpdate()-
>findOrFail($request->produk_id); // <- Kunci
baris supaya aman dari race condition
            if ($produk->stok < $request->quantity) {
                return redirect()->back()-
>withErrors(['quantity' => 'Stok tidak
mencukupi']);
            }
            $cart = Cart::firstOrCreate(['user_id' =>
Auth::id()]);
            $item = $cart->items()-
>where('produk_id', $produk->id)->first();
            if ($item) {
                $item->quantity += $request-
>quantity;
                $item->subtotal = $item->quantity *
$produk->harga; // Subtotal diperbarui
                $item->save();
            } else {
                $cart->items()->create([
                    'produk_id' => $produk->id,
                    'quantity' => $request->quantity,
                    'subtotal' => $produk->harga *
$request->quantity,
```

```
            ]);
        }
        // Update stok produk
        $produk->stok -= $request->quantity;
        $produk->save();
        DB::commit(); // <- Commit transaksi
        return redirect()->back()->with('success',
'Produk telah ditambahkan ke keranjang.');
    } catch (\Exception $e) {
        DB::rollBack(); // <- Rollback kalau error
        return redirect()->back()->with('error',
'Terjadi kesalahan. Silakan coba lagi.');
    }
  }
  public function update(Request $request, $id)
  {
    $request->validate([
      'quantity' => 'required|integer|min:1',
    ]);
    $item = CartItem::with('produk')-
>findOrFail($id);
    $item->quantity = $request->quantity;
    $item->total = $item->produk->harga *
$request->quantity;
    $item->save();
    return redirect()->back()->with('success',
'Jumlah produk diperbarui');
  }
  public function destroy($id)
  {
    DB::beginTransaction();
    try {
      $item = CartItem::findOrFail($id);
      $produk = Produk::lockForUpdate()-
>findOrFail($item->produk_id); // Kunci data
produk
      // Kembalikan stok produk
      $produk->stok += $item->quantity;
      $produk->save();

      $item->delete(); // Baru hapus cart item
setelah stok dikembalikan
      DB::commit();
      return redirect()->back()->with('success',
'Produk dihapus dari keranjang dan stok
dikembalikan.');
    } catch (\Exception $e) {
      DB::rollBack();
      return redirect()->back()->with('error',
'Terjadi kesalahan. Silakan coba lagi.');
    }
  }
}
```

CATEGORYCONTROLLER.PHP

```
<?php
namespace App\Http\Controllers;
use App\Models\Category;
use App\Models\Produk;
use Illuminate\Http\Request;
use Inertia\Inertia;
use Illuminate\Support\Facades\Redirect;
use Illuminate\Support\Facades\Storage;
class CategoryController extends Controller
{
  public function index()
  {
    $category = Category::all();
    return Inertia::render('category/index',
['category' => $category]);
  }
  public function create()
  {
    return Inertia::render('category/create');
  }
  public function store(Request $request)
  {
    $request->validate([
      'kategori' => 'required|string|max:255',
    ]);

    Category::create([
      'kategori' => $request->kategori,
    ]);
    return Redirect::route('category.index')-
>with('success', 'Kategori berhasil
ditambahkan.');
  }
  public function show(Category $category)
  {
    $Produk = Produk::where('category_id',
$category->id)
      ->with('category')
      ->get()
      ->map(function ($item) {
        $item->foto = $item->foto ?
Storage::url($item->foto) : null;
        return $item;
      });
    return Inertia::render('Produk/index', [
      'category' => $category,
      'Produk' => $Produk,
    ]);
  }
  public function edit(Category $category)
  {
    return Inertia::render('category/update', [
      'category' => $category,
    ]);
  }
  public function update(Request $request,
Category $category)
  {
    $request->validate([
      'kategori' => 'required|string|max:255',
    ]);
    $category->update([
      'kategori' => $request->kategori,
    ]);

    return Redirect::route('category.index')-
>with('success', 'Kategori berhasil diperbarui.');
  }
  public function destroy(Category $category)
  {
    $category->delete();
    return Redirect::route('category.index')-
>with('success', 'Kategori berhasil dihapus.');
```

```
    }
}
```

CONTROLLER.PHP

```
<?php
namespace App\Http\Controllers;
abstract class Controller
{
    //
}
```

PEMINJAMANCONTROLLER.PHP

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Models\Pesanan;
use Inertia\Inertia;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Redirect;
use Carbon\Carbon;
class PeminjamanController extends Controller
{
    public function index()
    {
        $user = Auth::user();

        $Pesanan = Pesanan::when(!$user-
>hasRole('admin'), function ($query) use ($user)
{
            return $query->where('user_id', $user-
>id);
        })->get()->map(function ($item) {
            $item->ktp = $item->ktp ?
Storage::url($item->ktp) : null;
            return $item;
        });
        return Inertia::render('Peminjaman/index',
['Pesanan' => $Pesanan]);
    }
    public function show($pesananId)
    {
        // Ambil pesanan berdasarkan ID
        $pesanan =
Pesanan::with('order_items.produk')-
>findOrFail($pesananId);
        // Kirim data pesanan dan order_items ke
view Inertia
        return Inertia::render('Peminjaman/show', [
            'pesanan' => $pesanan,
            'order_items' => $pesanan->order_items,
        ]);
    }
    public function verifikasipesanan(Request
$request, Pesanan $Pesanan)
    {
        $Pesanan->status = 'selesai';
        $Pesanan->save();
        return redirect()-
>route('Peminjaman.index')->with('success',
'Pesanan ini telah selesai.');
    }
public function destroy(Pesanan $Pesanan)
{
    // Hapus file KTP jika ada
    if ($Pesanan->ktp &&
Storage::exists($Pesanan->ktp)) {
        Storage::delete($Pesanan->ktp);
    }
    // Hapus relasi order_items jika ada
    if ($Pesanan->relationLoaded('order_items')) {
        $Pesanan->order_items()->delete();
    } else {
        // Pastikan order_items tetap dihapus
walaupun belum diload
        $Pesanan->load('order_items')-
>order_items->each->delete();
    }
    // Hapus data pesanan
    $Pesanan->delete();
    return Redirect::route('Peminjaman.index')-
>with('success', 'Pesanan berhasil dihapus.');
}
public function laporan(Request $request)
{
    $startDate = $request->input('start_date');
    $endDate = $request->input('end_date');
    $bulan = $request->input('bulan');
    $tahun = $request->input('tahun');
    $pesanan = Pesanan::query()
        ->when($startDate && $endDate, function
($query) use ($startDate, $endDate) {
            $query->whereBetween('peminjaman', [
                Carbon::parse($startDate)-
>startOfDay(),
                Carbon::parse($endDate)->endOfDay()
            ]);
        })
        ->when($bulan && $tahun, function
($query) use ($bulan, $tahun) {
            $query->whereMonth('peminjaman',
$bulan)
                ->whereYear('peminjaman', $tahun);
        })
        ->orderBy('peminjaman', 'desc')
        ->get();
    return Inertia::render('Peminjaman/Laporan', [
        'Pesanan' => $pesanan,
        'filters' => [
            'start_date' => $startDate,
            'end_date' => $endDate,
            'bulan' => $bulan,
            'tahun' => $tahun,
        ]
    ]);
}
}
```

PESANANCONTROLLER.PHP

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Models\Pesanan;
use App\Models\Cart;
use App\Models\CartItem;
use App\Models\order_item;
use Midtrans\Snap;
use Midtrans\Config;
```

```
use Illuminate\Support\Str;
use Midtrans\Notification;
use Illuminate\Support\Facades\Log;
use Illuminate\Support\Facades\Redirect;
class PesananController extends Controller
{
    public function buatPesanan(Request $request)
    {
        $request->validate([
            'total' => 'required|numeric|min:1000',
            'peminjaman' => 'required|date',
            'ktp' =>
'nullable|image|mimes:jpeg,png,jpg,gif,svg,pdf|m
ax:10048',
            'pengembalian' =>
'required|date|after:peminjaman',
        ]);
        $fotoName = null;
        if ($request->hasFile('ktp')) {
            $fotoName = $request->file('ktp')-
>store('public/jaminan');
        }

        // Midtrans Config
        Config::$serverKey =
config('midtrans.server_key');
        Config::$isProduction =
config('midtrans.is_production');
        Config::$isSanitized =
config('midtrans.is_sanitized');
        Config::$is3ds = config('midtrans.is_3ds');
        $user = auth()->user();
        if (!$user) {
            return response()->json(['message' =>
'User belum login'], 401);
        }
        $orderId = 'ORDER-' .
Str::upper(Str::random(10));
        // Buat pesanan awal
        $pesanan = Pesanan::create([
            'user_id' => $user->id,
            'order_id' => $orderId,
            'total' => 0, // awalnya 0
            'ktp' => $fotoName,
            'peminjaman' => $request->peminjaman,
            'pengembalian' => $request-
>pengembalian,
            'status' => 'belum bayar',
            'pembayaran' => 'Non tunai'
        ])
        // Cari cart user
        $cart = Cart::where('user_id', $user->id)-
>first();
        if (!$cart) {
            return response()->json(['message' =>
'Keranjang kosong'], 400);
        }
        // Ambil semua cart_items
        $cartItems = CartItem::where('cart_id',
$cart->id)->get();
        foreach ($cartItems as $item) {
            order_item::create([
                'pesanan_id' => $pesanan->id,
                'produk_id' => $item->produk_id,
                'quantity' => $item->quantity,
            ]);
        }
        // Update total setelah semua item dipindah
        $pesanan->update([
            'total' => $pesanan->calculateTotal(),
        ]);
        $params = [
            'transaction_details' => [
                'order_id' => $orderId,
                'gross_amount' => $pesanan->total,
            ],
            'customer_details' => [
                'first_name' => $user->name ?? 'User',
                'email' => $user->email ??
'dummy@email.com',
            ],
        ];
        try {
            $snapToken =
Snap::getSnapToken($params);
            return response()->json([
                'snap_token' => $snapToken,
                'pesanan_id' => $pesanan->id,
            ]);
        } catch (\Exception $e) {
            Log::error('Midtrans error: ' . $e-
>getMessage());
            return response()->json([
                'message' => 'Gagal membuat token
pembayaran.',
                'error' => $e->getMessage(),
            ], 500);
        }
    }
public function buatPesananCash(Request
$request)
{
    $request->validate([
        'total' => 'required|numeric|min:1000',
        'peminjaman' => 'required|date',
        'ktp' =>
'nullable|image|mimes:jpeg,png,jpg,gif,svg,pdf|m
ax:10048',
        'pengembalian' =>
'required|date|after:peminjaman',
    ]);
    $fotoName = null;
    if ($request->hasFile('ktp')) {
        $fotoName = $request->file('ktp')-
>store('public/jaminan');
    }
    $user = auth()->user();
    if (!$user) {
        return response()->json(['message' => 'User
belum login'], 401);
    }
    $orderId = 'CASH-' .
Str::upper(Str::random(10));
    $pesanan = Pesanan::create([
        'user_id' => $user->id,
        'order_id' => $orderId,
        'total' => 0, // nanti dihitung
        'ktp' => $fotoName,
```

```
            'peminjaman' => $request->peminjaman,
            'pengembalian' => $request->pengembalian,
            'status' => 'menunggu konfirmasi',
            'pembayaran' => 'tunai'
        ]);
        $cart = Cart::where('user_id', $user->id)->first();
        if (!$cart) {
            return response()->json(['message' => 'Keranjang kosong'], 400);
        }
        $cartItems = CartItem::where('cart_id', $cart->id)->get();
        foreach ($cartItems as $item) {
            order_item::create([
                'pesanan_id' => $pesanan->id,
                'produk_id' => $item->produk_id,
                'quantity' => $item->quantity,
            ]);
        }
        // Hitung total akhir
        $pesanan->update([
            'total' => $pesanan->calculateTotal(),
        ]);
        return response()->json([
            'message' => 'Pesanan cash berhasil diajukan.',
            'pesanan_id' => $pesanan->id,
        ]);
    }
    public function konfirmasiCash($id)
    {
        $pesanan = Pesanan::findOrFail($id);
        if ($pesanan->status === 'menunggu konfirmasi') {
            $pesanan->status = 'peminjaman';
            $pesanan->save();
            return back()->with('success', 'Pesanan berhasil dikonfirmasi.');
        }
        return back()->with('error', 'Pesanan tidak valid untuk dikonfirmasi.');
    }
        public function updateStatus($id)
        {
            $pesanan = Pesanan::findOrFail($id);
            // Pastikan hanya bisa diupdate jika status masih "belum bayar"
            if ($pesanan->status !== 'belum bayar') {
                return response()->json(['message' => 'Status tidak bisa diupdate.'], 400);
            }
            $pesanan->status = 'peminjaman';
            $pesanan->save();
            return response()->json(['message' => 'Status pesanan berhasil diupdate.']);
        }
    }
```

PRODUKCONTROLLER.PHP

```
<?php
namespace App\Http\Controllers;
use App\Models\Produk;
use App\Models\Category;
use Illuminate\Http\Request;
use Inertia\Inertia;
use Illuminate\Support\Facades\Redirect;
use Illuminate\Support\Facades\Storage;
class ProdukController extends Controller
{
    // public function index()
    // {
    //     $Produk = Produk::with('category')->get()->map(function ($item) {
    //         $item->foto = Storage::url($item->foto);
    //         return $item;
    //     });
    //     return Inertia::render('Produk/index', ['Produk' => $Produk]);
    // }
    public function create()
    {
        $categories = Category::all();
        return Inertia::render('Produk/create', [
            'categories' => $categories,
        ]);
    }
    public function store(Request $request)
    {
        $request->validate([
            'nama_barang' => 'required',
            'ukuran' => 'required|array|min:1',
            'dekripsi'=>'required',
            'ukuran.*' => 'in:S,M,L,XL,XXL', // opsional: batasi pilihan ukuran yang valid
            'harga' => 'required',
            'stok' => 'required',
            'foto' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:10048',
            'status' => 'required',
            'category_id' => 'required|exists:categories,id',
        ]);
        $fotoName = null;
        if ($request->hasFile('foto')) {
            $fotoName = $request->file('foto')->store('public/produk');
        }
        Produk::create([
            'nama_barang' => $request->input('nama_barang'),
            'ukuran' => $request->input('ukuran'), // langsung array, Laravel otomatis simpan sebagai JSON
            'harga' => $request->input('harga'),
            'dekripsi'=> $request->input('dekripsi'),
            'stok' => $request->input('stok'),
            'foto' => $fotoName,
            'status' => 'tersedia',
            'category_id' => $request->input('category_id'),
        ]);
        return Redirect::route('category.index')->with('success', 'Produk berhasil di tambah');
    }
    public function show(Produk $Produk)
```

```
    {
        return Inertia::render('Produk/show',
['Produk' => $Produk]);
    }
    public function edit(Produk $Produk)
    {
        $categories = Category::all();
        return Inertia::render('Produk/update', [
            'Produk' => $Produk,
            'categories' => $categories,
        ]);
    }
    public function update(Request $request,
Produk $Produk)
    {
        $request->validate([
            'nama_barang' => 'required',
            'ukuran' => 'required|array', // ✅ validasi
array
            'dekripsi'=>'required',
            'ukuran.*' => 'string', // ✅ validasi setiap
elemen array
            'harga' => 'required',
            'stok' => 'required',
            'foto' =>
'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:
20048',
            'category_id' =>
'required|exists:categories,id',
        ]);
        if ($request->hasFile('foto')) {
            if ($Produk->foto) {
                Storage::delete($Produk->foto);
            }
            $fotoName = $request->file('foto')-
>store('public/produk');
            $Produk->foto = $fotoName;
        }
        $Produk->update([
            'nama_barang' => $request-
>input('nama_barang'),
            'ukuran' => $request->input('ukuran'), //
✅ array langsung disimpan
            'harga' => $request->input('harga'),
            'dekripsi'=> $request->input('dekripsi'),
            'stok' => $request->input('stok'),
            'category_id' => $request-
>input('category_id'),
        ]);
        return Redirect::route('category.index')-
>with('success', 'Produk berhasil di update');
    public function destroy(Produk $Produk)
    {
        Storage::delete($Produk->foto);
        $Produk->delete();
        return Redirect::route('category.index')-
>with('success', 'Produk berhasil di hapus');
    }
}
```

PROFILCONTROLLER.PHP

```
<?php
namespace App\Http\Controllers;

use App\Http\Requests\ProfileUpdateRequest;
use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Redirect;
use Inertia\Inertia;
use Inertia\Response;
class ProfileController extends Controller
{
    /**
     * Display the user's profile form.
     */
    public function edit(Request $request):
Response
    {
        return Inertia::render('Profile/Edit', [
            'mustVerifyEmail' => $request->user()
instanceof MustVerifyEmail,
            'status' => session('status'),
        ]);
    }
    /**
     * Update the user's profile information.
     */
    public function update(ProfileUpdateRequest
$request): RedirectResponse
    {
        $request->user()->fill($request-
>validated());
        if ($request->user()->isDirty('email')) {
            $request->user()->email_verified_at =
null;
        }
        $request->user()->save();
        return Redirect::route('profile.edit');
    }
    /**
     * Delete the user's account.
     */
    public function destroy(Request $request):
RedirectResponse
    {
        $request->validate([
            'password' => ['required',
'current_password'],
        ]);
        $user = $request->user();
        Auth::logout();
        $user->delete();
        $request->session()->invalidate();
        $request->session()->regenerateToken();
        return Redirect::to('/');
    }
}
```

- **MODELS**

CART.PHP

```
<?php
namespace App\Models;
use
Illuminate\Database\Eloquent\Factories\HasFact
ory;
use Illuminate\Database\Eloquent\Model;
class Cart extends Model
{
    use HasFactory;
    protected $fillable = ['user_id', 'total'];
    public function user()
    {
        return $this->belongsTo(User::class);
    }
    public function items()
    {
        return $this->hasMany(CartItem::class);
    }
    public function calculateTotal()
    {
        return $this->items->sum(function ($item) {
            return $item->produk->harga * $item-
>quantity;
        });
    }
}
```

CARTITEM

```
<?php
namespace App\Models;
use
Illuminate\Database\Eloquent\Factories\HasFact
ory;
use Illuminate\Database\Eloquent\Model;
class CartItem extends Model
{
    use HasFactory;
    protected $fillable = ['cart_id', 'produk_id',
'quantity'];
    public function cart()
    {
        return $this->belongsTo(Cart::class);
    }
    public function produk()
    {
        return $this->belongsTo(Produk::class);
    }
    protected static function booted()
    {
        static::saved(function ($item) {
            $item->cart->update(['total' => $item-
>cart->calculateTotal()]);
        });
        static::deleted(function ($item) {
            $item->cart->update(['total' => $item-
>cart->calculateTotal()]);
        });
    }
}
```

CATEGORY

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
class Category extends Model
{
    protected $fillable = ['kategori'];

    public function produk()
    {
        return $this->hasMany(Category::class);
    }
}
```

ORDER_ITEM.PHP

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
class order_item extends Model
{
    protected $fillable = ['pesanan_id', 'produk_id',
'quantity'];
    public function pesanan()
    {
        return $this->belongsTo(Pesanan::class);
    }
    public function produk()
    {
        return $this->belongsTo(Produk::class);
    }
    protected static function booted()
    {
        static::saved(function ($item) {
            $item->pesanan->update(['total' =>
$item->pesanan->calculateTotal()]);
        });
        static::deleted(function ($item) {
            $item->pesanan->update(['total' =>
$item->pesanan->calculateTotal()]);
        });
    }
}
```

PESANAN.PHP

```
<?php
namespace App\Models;
use
Illuminate\Database\Eloquent\Factories\HasFact
ory;
use Illuminate\Database\Eloquent\Model;
use Carbon\Carbon;
class Pesanan extends Model
{
    use HasFactory;
    protected $table = 'pesanans';
    protected $fillable = [
        'user_id',
        'order_id',
        'total',
        'status',
        'ktp',
        'pembayaran',
        'peminjaman',
        'pengembalian'
```

```
    ];
    public function user()
    {
        return $this->belongsTo(User::class);
    }
    public function order_items()
    {
        return $this->hasMany(order_item::class);
    }
    public function calculateTotal()
    {
        $durasiHari = 1; // Default 1 hari
        if ($this->peminjaman && $this-
>pengembalian) {
            $peminjaman = Carbon::parse($this-
>peminjaman);
            $pengembalian = Carbon::parse($this-
>pengembalian);
            $durasiHari = $peminjaman-
>diffInDays($pengembalian);
            $durasiHari = max(1, $durasiHari); //
minimal 1 hari
        }
        $totalProduk = $this->order_items-
>sum(function ($item) {
            return $item->produk->harga * $item-
>quantity;
        });

        return $totalProduk * $durasiHari;
    }
}
```

PRODUK.PHP

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
use App\status;
class Produk extends Model
{
    protected $fillable = [
        'nama_barang',
        'ukuran',
        'stok',
        'dekripsi',
        'status',
        'harga',
        'foto',
        'category_id'
    ];
    public function category()
    {
        return $this->belongsTo(Category::class);
    }
        /**
     * Get the attributes that should be cast.
     *
     * @return array<string, string>
     */
    protected function casts(): array
    {
        return [
            'status' => status::class,
        ];
    }
    protected $casts = [
        'ukuran' => 'array',
    ];
}
```

USER.PHP

```
<?php
namespace App\Models;

// use
Illuminate\Contracts\Auth\MustVerifyEmail;
use
Illuminate\Database\Eloquent\Factories\HasFact
ory;
use Illuminate\Foundation\Auth\User as
Authenticatable;
use Illuminate\Notifications\Notifiable;
use Spatie\Permission\Traits\HasRoles;
class User extends Authenticatable
{
    use HasFactory, Notifiable,HasRoles;
    /**
     * The attributes that are mass assignable.
     *
     * @var array<int, string>
     */
    protected $fillable = [
        'name',
        'no',
        'email',
        'password',
    ];
    /**
     * The attributes that should be hidden for
serialization.
     *
     * @var array<int, string>
     */
    protected $hidden = [
        'password',
        'remember_token',
    ];
    /**
     * Get the attributes that should be cast.
     *
     * @return array<string, string>
     */
    protected function casts(): array
    {
        return [
            'email_verified_at' => 'datetime',
            'password' => 'hashed',
        ];
    }
    public function cart()
{
    return $this-
>hasOne(\App\Models\Cart::class);
}
}
```

- **MIGRATIONS**

0001_01_01_000000_CREATE_USERS_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('no');
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
        Schema::create('password_reset_tokens', function (Blueprint $table) {
            $table->string('email')->primary();
            $table->string('token');
            $table->timestamp('created_at')->nullable();
        });

        Schema::create('sessions', function (Blueprint $table) {
            $table->string('id')->primary();
            $table->foreignId('user_id')->nullable()->index();
            $table->string('ip_address', 45)->nullable();
            $table->text('user_agent')->nullable();
            $table->longText('payload');
            $table->integer('last_activity')->index();
        });
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('users');
        Schema::dropIfExists('password_reset_tokens');
        Schema::dropIfExists('sessions');
    }
};
```

0001_01_01_000001_CREATE_CACHE_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('cache', function (Blueprint $table) {
            $table->string('key')->primary();
            $table->mediumText('value');
            $table->integer('expiration');
        });
        Schema::create('cache_locks', function (Blueprint $table) {
            $table->string('key')->primary();
            $table->string('owner');
            $table->integer('expiration');
        });
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('cache');
        Schema::dropIfExists('cache_locks');
    }
};
```

0001_01_01_000002_CREATE_JOBS_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('jobs', function (Blueprint $table) {
            $table->id();
            $table->string('queue')->index();
            $table->longText('payload');
            $table->unsignedTinyInteger('attempts');
            $table->unsignedInteger('reserved_at')->nullable();
            $table->unsignedInteger('available_at');
            $table->unsignedInteger('created_at');
        });
        Schema::create('job_batches', function (Blueprint $table) {
            $table->string('id')->primary();
            $table->string('name');
            $table->integer('total_jobs');
            $table->integer('pending_jobs');
            $table->integer('failed_jobs');
            $table->longText('failed_job_ids');
            $table->mediumText('options')->nullable();
```

```
            $table->integer('cancelled_at')-
>nullable();
            $table->integer('created_at');
            $table->integer('finished_at')->nullable();
        });
        Schema::create('failed_jobs', function
(Blueprint $table) {
            $table->id();
            $table->string('uuid')->unique();
            $table->text('connection');
            $table->text('queue');
            $table->longText('payload');
            $table->longText('exception');
            $table->timestamp('failed_at')-
>useCurrent();
        });
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('jobs');
        Schema::dropIfExists('job_batches');
        Schema::dropIfExists('failed_jobs');
    }
};
```

2025_04_09_163316_CREATE_PRODUKS_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('categories', function
(Blueprint $table) {
            $table->id();
            $table->string('kategori');
            $table->timestamps();
        });
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('categories');
    }
};
```

2025_04_10_125138_CREATE_PESANANS_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('pesanans', function
(Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('user_id'); //
relasi ke user
            $table->string('order_id')->unique(); //
untuk Midtrans
            $table->integer('total'); // total harga
            $table->date('peminjaman');
            $table->date('pengembalian');
            $table->string('pembayaran');
            $table->string('ktp')->nullable();
            $table->enum('status', ['belum bayar',
'menunggu konfirmasi','peminjaman', 'selesai'])-
>default('belum bayar');
            $table->timestamps();
            $table->foreign('user_id')-
>references('id')->on('users')-
>onDelete('cascade');
        });
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('pesanans');
    }
};
```

2025_04_11_182448_CREATE_CATEGORIES_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('produks', function
(Blueprint $table) {
            $table->id();
            $table->string('nama_barang');
            $table->json('ukuran');
            $table->integer('stok');
            $table->text('dekripsi');
            $table->enum('status', ['tersedia',
'tidaktersedia'])->default('tersedia');
            $table->string('harga');
            $table->string('foto')->nullable();
            $table->foreignId('category_id')-
>constrained('categories')->onDelete('cascade');
            $table->timestamps();
        });
```

```
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('produks');
    }
};
```

2025_04_13_203835_CREATE_PERMISSION_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        $teams = config('permission.teams');
        $tableNames =
config('permission.table_names');
        $columnNames =
config('permission.column_names');
        $pivotRole =
$columnNames['role_pivot_key'] ?? 'role_id';
        $pivotPermission =
$columnNames['permission_pivot_key'] ??
'permission_id';
        if (empty($tableNames)) {
            throw new \Exception('Error:
config/permission.php not loaded. Run [php
artisan config:clear] and try again.');
        }
        if ($teams &&
empty($columnNames['team_foreign_key'] ??
null)) {
            throw new \Exception('Error:
team_foreign_key on config/permission.php not
loaded. Run [php artisan config:clear] and try
again.');
        }
        Schema::create($tableNames['permissions'],
static function (Blueprint $table) {
            // $table->engine('InnoDB');
            $table->bigIncrements('id'); // permission
id
            $table->string('name');       // For
MyISAM use string('name', 225); // (or 166 for
InnoDB with Redundant/Compact row format)
            $table->string('guard_name'); // For
MyISAM use string('guard_name', 25);
            $table->timestamps();
            $table->unique(['name', 'guard_name']);
        });
        Schema::create($tableNames['roles'], static
function (Blueprint $table) use ($teams,
$columnNames) {
            // $table->engine('InnoDB');
            $table->bigIncrements('id'); // role id
            if ($teams || config('permission.testing')) {
// permission.testing is a fix for sqlite testing
                $table-
>unsignedBigInteger($columnNames['team_fore
ign_key'])->nullable();
                $table-
>index($columnNames['team_foreign_key'],
'roles_team_foreign_key_index');
            }
            $table->string('name');       // For
MyISAM use string('name', 225); // (or 166 for
InnoDB with Redundant/Compact row format)
            $table->string('guard_name'); // For
MyISAM use string('guard_name', 25);
            $table->timestamps();
            if ($teams || config('permission.testing')) {
                $table-
>unique([$columnNames['team_foreign_key'],
'name', 'guard_name']);
            } else {
                $table->unique(['name',
'guard_name']);
            }
        });
        Schema::create($tableNames['model_has_p
ermissions'], static function (Blueprint $table)
use ($tableNames, $columnNames,
$pivotPermission, $teams) {
            $table-
>unsignedBigInteger($pivotPermission);
            $table->string('model_type');
            $table-
>unsignedBigInteger($columnNames['model_mo
rph_key']);
            $table-
>index([$columnNames['model_morph_key'],
'model_type'],
'model_has_permissions_model_id_model_type_
index');
            $table->foreign($pivotPermission)
                ->references('id') // permission id
                ->on($tableNames['permissions'])
                ->onDelete('cascade');
            if ($teams) {
                $table-
>unsignedBigInteger($columnNames['team_fore
ign_key']);
                $table-
>index($columnNames['team_foreign_key'],
'model_has_permissions_team_foreign_key_inde
x');
                $table-
>primary([$columnNames['team_foreign_key'],
$pivotPermission,
$columnNames['model_morph_key'],
'model_type'],
                    'model_has_permissions_permission
_model_type_primary');
            } else {
                $table->primary([$pivotPermission,
$columnNames['model_morph_key'],
'model_type'],
                    'model_has_permissions_permission
_model_type_primary');
```

```
        }
    });
    Schema::create($tableNames['model_has_roles'], static function (Blueprint $table) use ($tableNames, $columnNames, $pivotRole, $teams) {
        $table->unsignedBigInteger($pivotRole);
        $table->string('model_type');
        $table->unsignedBigInteger($columnNames['model_morph_key']);
        $table->index([$columnNames['model_morph_key'], 'model_type'], 'model_has_roles_model_id_model_type_index');
        $table->foreign($pivotRole)
            ->references('id') // role id
            ->on($tableNames['roles'])
            ->onDelete('cascade');
        if ($teams) {
            $table->unsignedBigInteger($columnNames['team_foreign_key']);
            $table->index($columnNames['team_foreign_key'], 'model_has_roles_team_foreign_key_index');
            $table->primary([$columnNames['team_foreign_key'], $pivotRole, $columnNames['model_morph_key'], 'model_type'],
                'model_has_roles_role_model_type_primary');
        } else {
            $table->primary([$pivotRole, $columnNames['model_morph_key'], 'model_type'],
                'model_has_roles_role_model_type_primary');
        }
    });
    Schema::create($tableNames['role_has_permissions'], static function (Blueprint $table) use ($tableNames, $pivotRole, $pivotPermission) {
        $table->unsignedBigInteger($pivotPermission);
        $table->unsignedBigInteger($pivotRole);
        $table->foreign($pivotPermission)
            ->references('id') // permission id
            ->on($tableNames['permissions'])
            ->onDelete('cascade');
        $table->foreign($pivotRole)
            ->references('id') // role id
            ->on($tableNames['roles'])
            ->onDelete('cascade');
        $table->primary([$pivotPermission, $pivotRole], 'role_has_permissions_permission_id_role_id_primary');
    });
    app('cache')
        ->store(config('permission.cache.store') != 'default' ? config('permission.cache.store') : null)
        ->forget(config('permission.cache.key'));
  }
  /**
   * Reverse the migrations.
   */
  public function down(): void
  {
    $tableNames = config('permission.table_names');
    if (empty($tableNames)) {
        throw new \Exception('Error: config/permission.php not found and defaults could not be merged. Please publish the package configuration before proceeding, or drop the tables manually.');
    }
    Schema::drop($tableNames['role_has_permissions']);
    Schema::drop($tableNames['model_has_roles']);
    Schema::drop($tableNames['model_has_permissions']);
    Schema::drop($tableNames['roles']);
    Schema::drop($tableNames['permissions']);
  }
};
```

2025_04_23_161957_CREATE_CARTS_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
  /**
   * Run the migrations.
   */
  public function up(): void
  {
    Schema::create('carts', function (Blueprint $table) {
        $table->id();
        $table->foreignId('user_id')->constrained()->onDelete('cascade');
        $table->integer('total')->default(0); // total harga semua item di keranjang
        $table->timestamps();
    });
  }
  /**
   * Reverse the migrations.
   */
  public function down(): void
  {
    Schema::dropIfExists('carts');
  }
};
```

2025_04_23_162019_CREATE_ITEMS_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('cart_items', function (Blueprint $table) {
            $table->id();
            $table->foreignId('cart_id')->constrained()->onDelete('cascade');
            $table->foreignId('produk_id')->constrained('produks')->onDelete('cascade');
            $table->integer('quantity')->default(1);
            $table->integer('subtotal')->default(0); // harga produk * quantity
            $table->timestamps();
        });
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('cart_items');
    }
};
```

2025_04_29_014237_CREATE_ORDERS_ITEMS_TABLE.PHP

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('order_items', function (Blueprint $table) {
            $table->id();
            $table->foreignId('pesanan_id')->constrained()->onDelete('cascade');
            $table->foreignId('produk_id')->constrained('produks')->onDelete('cascade');
            $table->integer('quantity')->default(1);
            $table->integer('subtotal')->default(0); // harga produk * quantity
            $table->timestamps();
        });
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('order_items');
    }
};
```

- **SEEDERS**

DATABASESEEDER.PHP

```
<?php
namespace Database\Seeders;
use App\Models\User;
// use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;
use Database\Seeders\RolesSeeder;
class DatabaseSeeder extends Seeder
{
    /**
     * Seed the application's database.
     */
    public function run(): void
    {
        // User::factory(10)->create();

        // User::factory()->create([
        //     'name' => 'Test User',
        //     'email' => 'test@example.com',
        // ]);
        $this->call([
            RolesSeeder::class,
        ]);
    }
}
```

ROLESSEEDER.PHP

```
<?php
namespace Database\Seeders;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;
use Spatie\Permission\Models\Role;
use App\Models\User;
class RolesSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        Role::create(['name' => 'admin']);
        Role::create(['name' => 'user']);
        // Create admin user
        $admin = User::create([
            'name' => 'admin',
            'no' => '087782283819',
            'email' => 'admin@example.com',
            'password' => bcrypt('password'),
        ]);
        $admin->assignRole('admin');
    }
}
```

- **RESOURCES**
- **JS**
- **COMPONENTS**

3D-PIN.TSX

```
"use client";
import React, { useState } from "react";
import { motion } from "framer-motion";
import { cn } from "@/types/cn";
export const PinContainer = ({
  children,
  title,
  href,
  className,
  containerClassName,
}: {
  children: React.ReactNode;
  title?: string;
  href?: string;
  className?: string;
  containerClassName?: string;
}) => {
  const [transform, setTransform] = useState(
    "translate(-50%,-50%) rotateX(0deg)"
  );
  const onMouseEnter = () => {
    setTransform("translate(-50%,-50%)
rotateX(40deg) scale(0.8)");
  };
  const onMouseLeave = () => {
    setTransform("translate(-50%,-50%)
rotateX(0deg) scale(1)");
  };
  return (
    <div
      className={cn(
        "relative group/pin z-50 cursor-
pointer",
        containerClassName
      )}
      onMouseEnter={onMouseEnter}
      onMouseLeave={onMouseLeave}
    >
      <div
        style={{
          perspective: "1000px",
          transform: "rotateX(70deg)
translateZ(0deg)",
        }}
        className="absolute left-1/2 top-1/2
ml-[0.09375rem] mt-4 -translate-x-1/2 -
translate-y-1/2"
      >
        <div
          style={{
            transform: transform,
          }}
          className="absolute left-1/2 p-4 top-
1/2 flex justify-start items-start rounded-2xl
shadow-[0_8px_16px_rgb(0_0_0/0.4)] bg-
black border border-white/[0.1] group-
hover/pin:border-white/[0.2] transition
duration-700 overflow-hidden"
        >
          <div className={cn("relative z-50",
className)}>{children}</div>
        </div>
      </div>
      <PinPerspective title={title} href={href}
/>
    </div>
  );
};
export const PinPerspective = ({
  title,
  href,
}: {
  title?: string;
  href?: string;
}) => {
  return (
    <motion.div className="pointer-events-
auto w-96 h-80 flex items-center justify-
center opacity-0 group-hover/pin:opacity-
100 z-[60] transition duration-500">
      <div className="w-full h-full -mt-7
flex-none inset-0">
        <div className="absolute top-0 inset-
x-0 flex justify-center">
          <a
            href={href}
            target={"_blank"}
            className="relative flex space-x-2
items-center z-20 rounded-full bg-zinc-950
py-0.5 px-4 ring-1 ring-white/10"
          >
            <span className="relative z-20
text-white text-xs font-bold inline-block py-
0.5">
              {title}
            </span>
            <span className="absolute -bottom-
0 left-[1.125rem] h-px w-[calc(100%-
2.25rem)] bg-gradient-to-r from-emerald-
400/0 via-emerald-400/90 to-emerald-400/0
transition-opacity duration-500 group-
hover/btn:opacity-40"></span>
          </a>
        </div>
        <div
          style={{
            perspective: "1000px",
            transform: "rotateX(70deg)
translateZ(0)",
          }}
          className="absolute left-1/2 top-1/2
ml-[0.09375rem] mt-4 -translate-x-1/2 -
translate-y-1/2"
        >
          <>
            <motion.div
              initial={{
                opacity: 0,
                scale: 0,
                x: "-50%",
                y: "-50%",
              }}
              animate={{
```

```
            opacity: [0, 1, 0.5, 0],
            scale: 1,
            z: 0,
          }}
          transition={{
            duration: 6,
            repeat: Infinity,
            delay: 0,
          }}
          className="absolute left-1/2 top-
1/2 h-[11.25rem] w-[11.25rem] rounded-
[50%] bg-sky-500/[0.08] shadow-
[0_8px_16px_rgb(0_0_0/0.4)]"
        ></motion.div>
        <motion.div
          initial={{
            opacity: 0,
            scale: 0,
            x: "-50%",
            y: "-50%",
          }}
          animate={{
            opacity: [0, 1, 0.5, 0],
            scale: 1,
            z: 0,
          }}
          transition={{
            duration: 6,
            repeat: Infinity,
            delay: 2,
          }}
          className="absolute left-1/2 top-
1/2 h-[11.25rem] w-[11.25rem] rounded-
[50%] bg-sky-500/[0.08] shadow-
[0_8px_16px_rgb(0_0_0/0.4)]"
        ></motion.div>
        <motion.div
          initial={{
            opacity: 0,
            scale: 0,
            x: "-50%",
            y: "-50%",
          }}
          animate={{
            opacity: [0, 1, 0.5, 0],
            scale: 1,
            z: 0,
          }}
          transition={{
            duration: 6,
            repeat: Infinity,
            delay: 4,
          }}
          className="absolute left-1/2 top-
1/2 h-[11.25rem] w-[11.25rem] rounded-
[50%] bg-sky-500/[0.08] shadow-
[0_8px_16px_rgb(0_0_0/0.4)]"
        ></motion.div>
      </>
    </div>
    <>
      <motion.div className="absolute
right-1/2 bottom-1/2 bg-gradient-to-b from-
transparent to-cyan-500 translate-y-[14px]
w-px h-20 group-hover/pin:h-40 blur-[2px]"
/>
      <motion.div className="absolute
right-1/2 bottom-1/2 bg-gradient-to-b from-
transparent to-cyan-500 translate-y-[14px]
w-px h-20 group-hover/pin:h-40" />
      <motion.div className="absolute
right-1/2 translate-x-[1.5px] bottom-1/2 bg-
cyan-600 translate-y-[14px] w-[4px] h-[4px]
rounded-full z-40 blur-[3px]" />
      <motion.div className="absolute
right-1/2 translate-x-[0.5px] bottom-1/2 bg-
cyan-300 translate-y-[14px] w-[2px] h-[2px]
rounded-full z-40" />
    </>
  </div>
 </motion.div>
);
};
```

APPLICATIONLOGO.JSX

```
export default function
ApplicationLogo(props) {
  return (
    <img src="/image/baju.jpeg"
className=' ' alt="" />
  );
}
```

BUTTON.JSX

```
export default function Button({ className
= '', disabled, children, ...props }) {
  return (
    <button
      {...props}
      className={
        `p-2 px-5 rounded-lg font-medium
text-center text-white dark:text-black border
border-gray-300 dark:border-gray-900  ${
          disabled && 'opacity-25'
        } ` + className
      }
      disabled={disabled}
    >
      {children}
    </button>
  );
}
```

CHECKBOX.JSX

```
export default function Checkbox({
className = '', ...props }) {
  return (
    <input
      {...props}
      type="checkbox"
      className={
        'rounded dark:bg-gray-900 border-
gray-300 dark:border-gray-700 text-indigo-
600 shadow-sm focus:ring-indigo-500
dark:focus:ring-indigo-600 dark:focus:ring-
offset-gray-800 ' +
          className
      }
```

```
      />
    );
}
```

DANGERBUTTON.JSX

```
export default function DangerButton({
className = '', disabled, children, ...props })
{
   return (
      <button
         {...props}
         className={
            `inline-flex items-center px-4 py-2
bg-red-600 border border-transparent
rounded-md font-semibold text-xs text-white
uppercase tracking-widest hover:bg-red-500
active:bg-red-700 focus:outline-none
focus:ring-2 focus:ring-red-500 focus:ring-
offset-2 dark:focus:ring-offset-gray-800
transition ease-in-out duration-150 ${
                disabled && 'opacity-25'
            } ` + className
         }
         disabled={disabled}
      >
         {children}
      </button>
   );
}
```

DARKMODE.JSX

```
import React, { useState, useEffect } from
"react";
import { IconMoonStars, IconSunset2 }
from '@tabler/icons-react';
export function DarkMode() {
  // atur default dengan true atau false
  const [darkMode, setDarkMode] =
useState(false);
  const toggleDarkMode = () => {
    setDarkMode(!darkMode);
  };
  useEffect(() => {
    if (darkMode) {
      document.documentElement.classList.ad
d("dark");
    } else {
      document.documentElement.classList.re
move("dark");
    }
  }, [darkMode]);
  return (
    <button
      className="text-gray-600 dark:text-
white hover:text-gray-900 dark:hover:text-
white px-2 py-2 transition duration-300
ease-in-out transform hover:scale-105"
      onClick={toggleDarkMode}
    >
      {darkMode ? <IconSunset2
className="h-8 w-8" /> : <IconMoonStars
className="h-8 w-8" />}
    </button>
  );
}
```

DROPDOWN.JSX

```
import { useState, createContext,
useContext, Fragment } from 'react';
import { Link } from '@inertiajs/react';
import { Transition } from
'@headlessui/react';
const DropDownContext = createContext();
const Dropdown = ({ children }) => {
    const [open, setOpen] = useState(false);
    const toggleOpen = () => {
        setOpen((previousState) =>
!previousState);
    };
    return (
        <DropDownContext.Provider value={{
open, setOpen, toggleOpen }}>
            <div
className="relative">{children}</div>
        </DropDownContext.Provider>
    );
};
const Trigger = ({ children }) => {
    const { open, setOpen, toggleOpen } =
useContext(DropDownContext);
    return (
        <>
            <div
onClick={toggleOpen}>{children}</div>

            {open && <div className="fixed
inset-0 z-40" onClick={() =>
setOpen(false)}></div>}
        </>
    );
};
const Content = ({ align = 'right', width =
'48', contentClasses = 'py-1 bg-white
dark:bg-gray-700', children }) => {
    const { open, setOpen } =
useContext(DropDownContext);
    let alignmentClasses = 'origin-top';

    if (align === 'left') {
        alignmentClasses = 'ltr:origin-top-left
rtl:origin-top-right start-0';
    } else if (align === 'right') {
        alignmentClasses = 'ltr:origin-top-right
rtl:origin-top-left end-0';
    }
    let widthClasses = '';
    if (width === '48') {
        widthClasses = 'w-48';
    }
    return (
        <>
            <Transition
                as={Fragment}
                show={open}
                enter="transition ease-out
duration-200"
                enterFrom="opacity-0 scale-95"
                enterTo="opacity-100 scale-100"
```

```
            leave="transition ease-in duration-
75"
            leaveFrom="opacity-100 scale-
100"
            leaveTo="opacity-0 scale-95"
          >
            <div
              className={`absolute z-50 mt-
2 rounded-md shadow-lg
${alignmentClasses} ${widthClasses}`}
              onClick={() => setOpen(false)}
            >
              <div className={`rounded-md
ring-1 ring-black ring-opacity-5 ` +
contentClasses}>{children}</div>
            </div>
          </Transition>
        </>
    );
};
const DropdownLink = ({ className = '',
children, ...props }) => {
    return (
        <Link
            {...props}
            className={
              'block w-full px-4 py-2 text-start
text-sm leading-5 text-gray-700 dark:text-
gray-300 hover:bg-gray-100 dark:hover:bg-
gray-800 focus:outline-none focus:bg-gray-
100 dark:focus:bg-gray-800 transition
duration-150 ease-in-out ' +
              className
            }
        >
            {children}
        </Link>
    );
};
Dropdown.Trigger = Trigger;
Dropdown.Content = Content;
Dropdown.Link = DropdownLink;
export default Dropdown;
```

FLIPWORDS.TSX

```
"use client";
import React, { useCallback, useEffect,
useRef, useState } from "react";
import { AnimatePresence, motion,
LayoutGroup } from "framer-motion";
import { cn } from "@/types/cn";
export const FlipWords = ({
  words,
  duration = 3000,
  className,
}: {
  words: string[];
  duration?: number;
  className?: string;
}) => {
  const [currentWord, setCurrentWord] =
useState(words[0]);
  const [isAnimating, setIsAnimating] =
useState<boolean>(false);

  // thanks for the fix Julian -
https://github.com/Julian-AT
  const startAnimation = useCallback(() => {
    const word =
words[words.indexOf(currentWord) + 1] ||
words[0];
    setCurrentWord(word);
    setIsAnimating(true);
  }, [currentWord, words]);
  useEffect(() => {
    if (!isAnimating)
      setTimeout(() => {
        startAnimation();
      }, duration);
  }, [isAnimating, duration, startAnimation]);
  return (
    <AnimatePresence
      onExitComplete={() => {
        setIsAnimating(false);
      }}
    >
      <motion.div
        initial={{
          opacity: 0,
          y: 10,
        }}
        animate={{
          opacity: 1,
          y: 0,
        }}
        transition={{
          duration: 0.4,
          ease: "easeInOut",
          type: "spring",
          stiffness: 100,
          damping: 10,
        }}
        exit={{
          opacity: 0,
          y: -40,
          x: 40,
          filter: "blur(8px)",
          scale: 2,
          position: "absolute",
        }}
        className={cn(
          "z-10 inline-block relative text-left
text-neutral-900 dark:text-neutral-100 px-2",
          className
        )}
        key={currentWord}
      >
        {currentWord.split("").map((letter,
index) => (
          <motion.span
            key={currentWord + index}
            initial={{ opacity: 0, y: 10, filter:
"blur(8px)" }}
            animate={{ opacity: 1, y: 0, filter:
"blur(0px)" }}
            transition={{
              delay: index * 0.08,
              duration: 0.4,
            }}
```

```
          className="inline-block"
        >
          {letter}
        </motion.span>
      ))}
    </motion.div>
  </AnimatePresence>
 );
};
```

INPUTERROR.JSX

```
export default function InputError({
message, className = '', ...props }) {
  return message ? (
    <p {...props} className={'text-sm
text-red-600 dark:text-red-400 ' +
className}>
      {message}
    </p>
  ) : null;
}
```

INPUTLABEL.JSX

```
export default function InputLabel({ value,
className = '', children, ...props }) {
  return (
    <label {...props} className={`block
font-medium text-sm text-gray-700
dark:text-gray-300 ` + className}>
      {value ? value : children}
    </label>
  );
}
```

MODAL.JSX

```
import { Fragment } from 'react';
import { Dialog, Transition } from
'@headlessui/react';
export default function Modal({ children,
show = false, maxWidth = '2xl', closeable =
true, onClose = () => {} }) {
  const close = () => {
    if (closeable) {
      onClose();
    }
  };
  const maxWidthClass = {
    sm: 'sm:max-w-sm',
    md: 'sm:max-w-md',
    lg: 'sm:max-w-lg',
    xl: 'sm:max-w-xl',
    '2xl': 'sm:max-w-2xl',
  }[maxWidth];
  return (
    <Transition show={show}
as={Fragment} leave="duration-200">
      <Dialog
        as="div"
        id="modal"
        className="fixed inset-0 flex
overflow-y-auto px-4 py-6 sm:px-0 items-
center z-50 transform transition-all"
        onClose={close}
      >
        <Transition.Child
          as={Fragment}
          enter="ease-out duration-300"
          enterFrom="opacity-0"
          enterTo="opacity-100"
          leave="ease-in duration-200"
          leaveFrom="opacity-100"
          leaveTo="opacity-0"
        >
          <div className="absolute
inset-0 bg-gray-500/75 dark:bg-gray-
900/75" />
        </Transition.Child>
        <Transition.Child
          as={Fragment}
          enter="ease-out duration-300"
          enterFrom="opacity-0 translate-
y-4 sm:translate-y-0 sm:scale-95"
          enterTo="opacity-100 translate-
y-0 sm:scale-100"
          leave="ease-in duration-200"
          leaveFrom="opacity-100
translate-y-0 sm:scale-100"
          leaveTo="opacity-0 translate-y-
4 sm:translate-y-0 sm:scale-95"
        >
          <Dialog.Panel
            className={`mb-6 bg-white
dark:bg-gray-800 rounded-lg overflow-
hidden shadow-xl transform transition-all
sm:w-full sm:mx-auto ${maxWidthClass}`}
          >
            {children}
          </Dialog.Panel>
        </Transition.Child>
      </Dialog>
    </Transition>
  );
}
```

NAVBAR.JSX

```
import React from "react";
import { Link, usePage } from
'@inertiajs/react';
import { DarkMode } from
"@/Components/DarkMode";
import Dropdown from
'@/Components/Dropdown';
import {
  IconHome,
  IconLayoutDashboard,
  IconUserCircle,
  IconLogout,
  IconShoppingBag,
  IconShoppingCart,
  IconCash
} from '@tabler/icons-react';
import { div } from "framer-motion/client";
export default function Navbar() {
  const { auth } = usePage().props;
  return (
    <nav className="bg-green-700 text-
white fixed w-full z-20 top-0 left-0 shadow-
md">
```

```
    <div className="max-w-screen-xl mx-auto px-4 py-3 flex items-center justify-between">
      {/* Brand Logo */}
      <Link href="/" className="text-xl font-bold text-white">
        Penyewaan Baju Adat
      </Link>
      {/* Navigasi Tengah */}
      <div className="space-x-6 hidden md:flex">
        <Link href="/" className="text-white hover:text-green-200 text-lg font-medium">
          Beranda
        </Link>
        <Link href="#tentang" className="text-white hover:text-green-200 text-lg font-medium">
          Tentang Kami
        </Link>
        <Link href="#produk" className="text-white hover:text-green-200 text-lg font-medium">
          Produk
        </Link>
        <Link href="#kontak" className="text-white hover:text-green-200 text-lg font-medium">
          Kontak
        </Link>
      </div>
      {/* Bagian Kanan */}
      <div className="flex items-center space-x-4">
        {/* <DarkMode /> */}
        {auth.user ? (
<div className=" flex ">
<Link href="/Peminjaman" className="text-xl font-bold text-white">
        <IconCash/>
      </Link>
            <Link href="/cart" className="text-xl font-bold text-white">
              <IconShoppingCart/>
            </Link>
</div>
        ):(
          <div className=""></div>
        )}
        {auth.user ? (
          <Dropdown>
            <Dropdown.Trigger>
              <span className="inline-flex rounded-md">
                <button
                  type="button"
                  className="inline-flex items-center px-3 py-2 text-sm font-medium rounded-md bg-white text-green-700 hover:bg-green-100 transition"
                >
                  {auth.user.name}
                  <svg
                    className="ms-2 -me-0.5 h-4 w-4"
                    xmlns="http://www.w3.org/2000/svg"
                    viewBox="0 0 20 20"
                    fill="currentColor"
                  >
                    <path
                      fillRule="evenodd"
                      d="M5.293 7.293a1 1 0 011.414 0L10 10.586l3.293-3.293a1 1 0 111.414 1.414l-4 4a1 1 0 01-1.414 0l-4-4a1 1 0 010-1.414z"
                      clipRule="evenodd"
                    />
                  </svg>
                </button>
              </span>
            </Dropdown.Trigger>

            <Dropdown.Content
              className="bg-white text-green-700 shadow-lg rounded-md">
              <Dropdown.Link href="/" className="flex items-center gap-2">
                <IconHome className="h-5 w-5" /> Home
              </Dropdown.Link>
              <Dropdown.Link href="#tentang" className="flex items-center gap-2">
                <IconLayoutDashboard
                  className="h-5 w-5" /> Tentang
              </Dropdown.Link>
              <Dropdown.Link href="#kontak" className="flex items-center gap-2 border-b">
                <IconLayoutDashboard
                  className="h-5 w-5" /> Kontak
              </Dropdown.Link>
              <Dropdown.Link
                href={route('dashboard')} className="flex items-center gap-2">
                <IconLayoutDashboard
                  className="h-5 w-5" /> Dashboard
              </Dropdown.Link>
              <Dropdown.Link
                href={route('category.index')}
                className="flex items-center gap-2">
                <IconShoppingBag
                  className="h-5 w-5" /> Produk
              </Dropdown.Link>
              <Dropdown.Link
                href={route('Peminjaman.laporan')}
                className="flex items-center gap-2">
                <IconLayoutDashboard
                  className="h-5 w-5" /> Laporan Peminjaman
              </Dropdown.Link>
              {auth.roles.includes("admin") && (
              <Dropdown.Link
                href={route('admin.index')}
                className="flex items-center gap-2">
```

```
                <IconUserCircle className="h-5
w-5" /> Admin
              </Dropdown.Link>
            )}
              <Dropdown.Link
href={route('profile.edit')} className="flex
items-center gap-2">
                <IconUserCircle className="h-
5 w-5" /> Profile
              </Dropdown.Link>
              <Dropdown.Link
href={route('logout')} method="post"
as="button" className="flex items-center
gap-2">
                <IconLogout className="h-5 w-
5" /> Log Out
              </Dropdown.Link>
            </Dropdown.Content>
          </Dropdown>
        ) : (
          <>
            <Link
              href={route('login')}
              className="px-4 py-2 rounded-
md bg-white text-green-700 hover:bg-green-
100 font-semibold transition"
            >
              Login
            </Link>
            <Link
              href={route('register')}
              className="px-4 py-2 rounded-
md bg-white text-green-700 hover:bg-green-
100 font-semibold transition"
            >
              Register
            </Link>
          </>
        )}
      </div>
    </div>
  </nav>
 );
}
```

NAVLINK.JSX

```
import { Link } from '@inertiajs/react';
export default function NavLink({ active =
false, classNam e = '', children, ...props }) {
  return (
    <Link
      {...props}
      className={
        'inline-flex items-center px-1 pt-1
border-b-2 text-sm font-medium leading-5
transition duration-150 ease-in-out
focus:outline-none ' +
          (active
            ? 'border-indigo-400
dark:border-indigo-600 text-gray-900
dark:text-gray-100 focus:border-indigo-700 '
            : 'border-transparent text-gray-
500 dark:text-gray-400 hover:text-gray-700
dark:hover:text-gray-300 hover:border-gray-
300 dark:hover:border-gray-700 focus:text-
gray-700 dark:focus:text-gray-300
focus:border-gray-300 dark:focus:border-
gray-700 ') +
          className
      }
    >
      {children}
    </Link>
  );
}
```

PRIMARYBUTTON.JSX

```
export default function PrimaryButton({
className = '', disabled, children, ...props })
{
  return (
    <button
      {...props}
      className={
        `inline-flex items-center px-4 py-2
bg-gray-800 dark:bg-gray-200 border
border-transparent rounded-md font-
semibold text-xs text-white dark:text-gray-
800 uppercase tracking-widest hover:bg-
gray-700 dark:hover:bg-white focus:bg-
gray-700 dark:focus:bg-white active:bg-
gray-900 dark:active:bg-gray-300
focus:outline-none focus:ring-2 focus:ring-
indigo-500 focus:ring-offset-2
dark:focus:ring-offset-gray-800 transition
ease-in-out duration-150 ${
          disabled && 'opacity-25'
        } ` + className
      }
      disabled={disabled}
    >
      {children}
    </button>
  );
}
```

RESPONSIVENAVLINK.JSX

```
import { Link } from '@inertiajs/react';
export default function
ResponsiveNavLink({ active = false,
className = '', children, ...props }) {
  return (
    <Link
      {...props}
      className={`w-full flex items-start
ps-3 pe-4 py-2 border-l-4 ${
        active
          ? 'border-indigo-400
dark:border-indigo-600 text-indigo-700
dark:text-indigo-300 bg-indigo-50 dark:bg-
indigo-900/50 focus:text-indigo-800
dark:focus:text-indigo-200 focus:bg-indigo-
100 dark:focus:bg-indigo-900 focus:border-
indigo-700 dark:focus:border-indigo-300'
          : 'border-transparent text-gray-
600 dark:text-gray-400 hover:text-gray-800
dark:hover:text-gray-200 hover:bg-gray-50
dark:hover:bg-gray-700 hover:border-gray-
```

```
300 dark:hover:border-gray-600 focus:text-
gray-800 dark:focus:text-gray-200 focus:bg-
gray-50 dark:focus:bg-gray-700
focus:border-gray-300 dark:focus:border-
gray-600'
        } text-base font-medium
focus:outline-none transition duration-150
ease-in-out ${className}`}
      >
        {children}
      </Link>
  );
}
```

SECONDARYBUTTON.JSX

```
export default function SecondaryButton({
type = 'button', className = '', disabled,
children, ...props }) {
  return (
    <button
      {...props}
      type={type}
      className={
        `inline-flex items-center px-4 py-2
bg-white dark:bg-gray-800 border border-
gray-300 dark:border-gray-500 rounded-md
font-semibold text-xs text-gray-700
dark:text-gray-300 uppercase tracking-
widest shadow-sm hover:bg-gray-50
dark:hover:bg-gray-700 focus:outline-none
focus:ring-2 focus:ring-indigo-500
focus:ring-offset-2 dark:focus:ring-offset-
gray-800 disabled:opacity-25 transition
ease-in-out duration-150 ${
          disabled && 'opacity-25'
        } ` + className
      }
      disabled={disabled}
    >
      {children}
    </button>
  );
}
```

TEXTINPUT.JSX

```
import { forwardRef, useEffect, useRef }
from 'react';
export default forwardRef(function
TextInput({ type = 'text', className = '',
isFocused = false, ...props }, ref) {
  const input = ref ? ref : useRef();
  useEffect(() => {
    if (isFocused) {
      input.current.focus();
    }
  }, []);
  return (
    <input
      {...props}
      type={type}
      className={
        'border-gray-300 dark:border-
gray-700 dark:bg-gray-900 dark:text-gray-
300 focus:border-indigo-500
dark:focus:border-indigo-600 focus:ring-
indigo-500 dark:focus:ring-indigo-600
rounded-md shadow-sm ' +
          className
      }
      ref={input}
    />
  );
});
```

- **LAYOUTS**

AUTHENTICATEDLAYOUT.JSX

```
import React from "react";
import {useState} from 'react';
import Navbar from
"@/Components/Navbar";
import Swal from "sweetalert2";
import { usePage } from "@inertiajs/react";
export default function
Authenticated({  children }) {
  const { flash } = usePage().props;
  const [isLoaded, setIsLoaded] =
useState(false);
// izin push notifikasi
  React.useEffect(() => {
    if (flash?.success) {
      Swal.fire(
        'Berhasil!',
        flash.success,
        'success'
      );
    }
    if (flash?.error) {
      Swal.fire(
        'gagal!',
        flash.error,
        'error'
      );
    }
  }, [flash]);
  return (
    <>
     <div className={` bg-white dark:bg-
black min-h-screen max-w-full p-3`}>
      <Navbar/>
      <main>{children}</main>
    </div>
    </>
  );
}
```

GUESTLAYOUT.JSX

```
import React from "react";
import {useState} from 'react';
import Navbar from
"@/Components/Navbar";
import Swal from "sweetalert2";
import { usePage } from "@inertiajs/react";
export default function
Authenticated({  children }) {
  const { flash } = usePage().props;
  const [isLoaded, setIsLoaded] =
useState(false);
// izin push notifikasi
```

```
    React.useEffect(() => {
        if (flash?.success) {
            Swal.fire(
                'Berhasil!',
                flash.success,
                'success'
            );
        }
        if (flash?.error) {
            Swal.fire(
                'gagal!',
                flash.error,
                'error'
            );
        }
    }, [flash]);
    return (
        <>
         <div className={` bg-white dark:bg-
black min-h-screen max-w-full p-3`}>
            <Navbar/>
            <main>{children}</main>
        </div>
        </>
    );
}
```

PUBLICLAYOUT.JSX

```
import React from "react";
import Navbar from
"@/Components/Navbar";
export default function PublicLayout({
children }) {
  return (
    <>
      <div className={` bg-white dark:bg-
black min-h-screen max-w-full p-3`}>
        <Navbar />
          <main>{children}</main>
        </div>
    </>
  );
}
```

- **PAGES**
- **ADMIN**

CHECKEMAIL.JSX

```
import { useState } from "react";
import GuestLayout from
"@/Layouts/GuestLayout";
import InputError from
"@/Components/InputError";
import InputLabel from
"@/Components/InputLabel";
import PrimaryButton from
"@/Components/PrimaryButton";
import TextInput from
"@/Components/TextInput";
import { router, usePage } from
"@inertiajs/react";
import { IconBrandWhatsapp } from
"@tabler/icons-react";
export default function CheckEmail() {
    const { admin_contact } =
usePage().props;
    const [email, setEmail] = useState("");
    const [errors, setErrors] = useState({});
    const handleSubmit = (e) => {
        e.preventDefault();
        router.post(
            "/check-email",
            { email },
            {
                onError: (err) => setErrors(err),
            }
        );
    };
    const whatsappAdminNumber =
admin_contact ?
`+62${admin_contact.phone}` :
'+6281234567890';
    return (
        <GuestLayout>
            {/* Title */}
            <h1 className="text-2xl font-bold
text-center text-gray-800 dark:text-white
mb-4">
                Periksa Email Anda
            </h1>
            {/* Description */}
            <div className="text-gray-600
dark:text-gray-300 text-center mb-6">
                Masukkan email Anda untuk
menerima tautan reset password. Ini akan
membantu kami memastikan akun Anda
tetap aman.
            </div>
            {/* Form */}
            <form onSubmit={handleSubmit}
className="space-y-6">
                <div>
                    <InputLabel htmlFor="email"
value="Email" />
                    <TextInput
                        id="email"
                        type="email"
                        name="email"
                        value={email}
                        placeholder="Masukkan
alamat email Anda"
                        className="mt-1 block w-
full rounded-md border-gray-300
focus:border-blue-500 focus:ring focus:ring-
blue-200 focus:ring-opacity-50"
                        isFocused={true}
                        onChange={(e) =>
setEmail(e.target.value)}
                    />
                    <InputError
message={errors.email} className="mt-2"
/>
                </div>
                <div className="flex justify-
center">
                    <PrimaryButton
className="w-full bg-blue-600 hover:bg-
blue-700 text-white text-center font-bold py-
```

```
2 px-4 rounded-md transition duration-
300">
                    Kirim Link Reset Password
                </PrimaryButton>
            </div>
        </form>
        {/* Information Section */}
        <div className="mt-6 text-sm text-
gray-500 dark:text-gray-400 text-center">
            <div>
                Mengapa kami meminta email
Anda? <br />
                Kami menggunakan alamat
email Anda untuk mengirimkan tautan aman
yang memungkinkan Anda mengganti kata
sandi. Jangan khawatir, email Anda aman
bersama kami.
            </div>
        </div>

        {/* WhatsApp Button */}
        <div className="mt-6 text-center">
            <div className="text-gray-600
dark:text-gray-300 mb-2">
                Jika Anda lupa email, klik
tombol di bawah untuk menghubungi admin.
            </div>
            <a
                href={`https://wa.me/${whatsap
pAdminNumber}?text=Masukkan%20nama
%20dan%20email%20Anda.`}
                target="_blank"
                rel="noopener noreferrer"
                className="inline-flex items-
center gap-2 bg-green-500 hover:bg-green-
600 text-white font-semibold py-2 px-4
rounded-md transition duration-300"
            >
                <IconBrandWhatsapp
size={24} />
                Hubungi Admin via WhatsApp
            </a>
        </div>
        {/* Back to Login */}
        <div className="mt-6 text-center">
            <a
                href="/login"
                className="text-blue-600
hover:text-blue-700 font-semibold transition
duration-300"
            >
                Kembali ke Halaman Login
            </a>
        </div>
    </GuestLayout>
  );
}
```

GANTIPASSWORD.JSX

```
import { useEffect, useState } from "react";
import GuestLayout from
"@/Layouts/GuestLayout";
import InputError from
"@/Components/InputError";
import InputLabel from
"@/Components/InputLabel";
import PrimaryButton from
"@/Components/PrimaryButton";
import TextInput from
"@/Components/TextInput";
import { router } from "@inertiajs/react";
export default function Gantipassword({
userId }) {
  const [password, setPassword] =
useState('');
  const [passwordConfirmation,
setPasswordConfirmation] = useState('');
  const [errors, setErrors] = useState({});
  const [showPassword, setShowPassword]
= useState(false);
  const handleSubmit = (e) => {
    e.preventDefault();
    router.post('/update-password', {
      user_id: userId,
      password,
      password_confirmation:
passwordConfirmation,
    }, {
      onError: (err) => setErrors(err),
    });
  };
  return (
    <GuestLayout>
        {/* Header */}
        <h2 className="text-3xl font-
bold text-gray-800 dark:text-white text-
center mb-4">
          Ganti Password
        </h2>
        <p className="text-sm text-
gray-600 dark:text-gray-300 mb-6 text-
center">
          Demi keamanan akun Anda,
harap ganti password Anda secara berkala
          atau jika Anda merasa akun
Anda tidak aman.
        </p>
        {/* Form */}
        <form
onSubmit={handleSubmit}
className="space-y-6">
          {/* Password */}
          <div className="relative">
            <InputLabel
htmlFor="password" value="Password
Baru" />
            <TextInput
              id="password"
              type={showPassword ?
"text" : "password"}
              name="password"
              value={password}
              className="mt-1 block
w-full border border-gray-300 rounded-lg
px-4 py-2"
              onChange={(e) =>
setPassword(e.target.value)}
```

```
                        placeholder="Masukkan
password baru"
                    />
                    {/* Toggle Password
Visibility */}
                    <button
                        type="button"
                        className="absolute
right-3 top-1/2 transform -translate-y-1/2
text-gray-500"
                        onClick={() =>
setShowPassword(!showPassword)}
                    >
                        {showPassword ?
"Hide" : "Show"}
                    </button>
                    <InputError
message={errors.password}
className="mt-2" />
                </div>
                {/* Confirm Password */}
                <div className="relative">
                    <InputLabel
                        htmlFor="password_con
firmation"
                        value="Konfirmasi
Password Baru"
                    />
                    <TextInput
                        id="password_confirmat
ion"
                        type={showPassword ?
"text" : "password"}
                        name="password_confir
mation"
                        value={passwordConfir
mation}
                        className="mt-1 block
w-full border border-gray-300 rounded-lg
px-4 py-2"
                        onChange={(e) =>
setPasswordConfirmation(e.target.value)}
                        placeholder="Ulangi
password baru"
                    />
                    {/* Toggle Password
Visibility */}
                    <button
                        type="button"
                        className="absolute
right-3 top-1/2 transform -translate-y-1/2
text-gray-500"
                        onClick={() =>
setShowPassword(!showPassword)}
                    >
                        {showPassword ?
"Hide" : "Show"}
                    </button>
                    <InputError
                        message={errors.passwo
rd_confirmation}
                        className="mt-2"
                    />
                </div>
                {/* Submit Button */}
                <div className="flex
justify-center">
                    <PrimaryButton
className="w-full bg-blue-600 hover:bg-
blue-700 text-white py-2 px-4 rounded-lg
shadow-lg transition duration-300">
                        Simpan Password
                    </PrimaryButton>
                </div>
            </form>

        </GuestLayout>
    );
}
```

INDEX.JSX

```
import React, { useState } from "react";
import { useForm, router } from
"@inertiajs/react";
import AuthenticatedLayout from
"@/Layouts/AuthenticatedLayout";
import Swal from "sweetalert2";
import { IconUsers, IconKey, IconTrash }
from "@tabler/icons-react";
import image from "/public/image/user.png";
import InputError from
"@/Components/InputError";
import InputLabel from
"@/Components/InputLabel";
import PrimaryButton from
"@/Components/PrimaryButton";
import TextInput from
"@/Components/TextInput";
export default function Index({ user, success
}) {
    const [password, setPassword] =
useState('');
    const [errors, setErrors] = useState({});
    const [passwordConfirmation,
setPasswordConfirmation] = useState('');
    const [selectedUser, setSelectedUser] =
useState(null);
    const [showPassword, setShowPassword]
= useState(false);
    const handleDelete = (id) => {
        Swal.fire({
            title: "Apakah Anda yakin?",
            text: "Data pengguna akan dihapus
secara permanen!",
            icon: "warning",
            showCancelButton: true,
            confirmButtonColor: "#d33",
            cancelButtonColor: "#3085d6",
            confirmButtonText: "Ya, hapus!",
        }).then((result) => {
            if (result.isConfirmed) {
                router.delete(route("admin.destroy
", id), {
                    onSuccess: () => {
                        Swal.fire("Terhapus!", "Data
pengguna berhasil dihapus.", "success");
                    },
                    onError: () => {
```

```
                    Swal.fire("Error!", "Terjadi
kesalahan saat menghapus pengguna.",
"error");
                },
            });
        }
    });
  };
  const handleSubmit = (e) => {
    e.preventDefault();
    if (!selectedUser?.id) {
      setErrors({ user_id: "Silakan pilih
pengguna terlebih dahulu" });
      return;
    }
    router.post('/update-password', {
      user_id: selectedUser.id,
      password,
      password_confirmation:
passwordConfirmation,
    }, {
      onError: (err) => setErrors(err),
      onSuccess: () => {
        setSelectedUser(null);
        setPassword('');
        setPasswordConfirmation('');
        Swal.fire("Berhasil!", "Password
berhasil diperbarui.", "success");
      }
    });
  };
  return (
    <AuthenticatedLayout>
      <div className="container mx-auto
px-4 py-8 text-gray-900 dark:text-gray-
100">
        <section className="bg-gradient-
to-tr from-green-600 to-emerald-600 text-
white py-12 rounded-xl shadow-lg mb-8
text-center">
          <h1 className="text-4xl font-
bold mb-3">Selamat Datang di Manajemen
Pengguna</h1>
          <p className="text-lg opacity-
90">Kelola semua pengguna dengan mudah
dan efisien.</p>
        </section>
        {success && (
          <div className="mb-6 bg-
green-100 dark:bg-green-800 text-green-800
dark:text-green-100 p-4 rounded-lg text-
center shadow-md">
            {success}
          </div>
        )}
        <div className="grid grid-cols-1
sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4
gap-6">
          {user.map((u) => (
            <div key={u.id}
className="bg-white dark:bg-gray-800
rounded-2xl shadow-lg p-5 flex flex-col
items-center text-center transition
hover:shadow-2xl">
              <img src={image}
alt={u.name} className="w-24 h-24
rounded-full mb-4 shadow" />
              <h2 className="text-lg
font-semibold">{u.name}</h2>
              <p className="text-gray-
500 dark:text-gray-300 text-sm mb-
4">{u.email}</p>
              <button
                onClick={() =>
setSelectedUser(u)}
                className="bg-indigo-
600 hover:bg-indigo-700 transition text-
white py-2 px-4 rounded-lg shadow flex
items-center justify-center w-full mb-2"
              >
                <IconKey
className="mr-2" /> Ganti Password
              </button>
              <button
                onClick={() =>
handleDelete(u.id)}
                className="bg-red-600
hover:bg-red-700 transition text-white py-2
px-4 rounded-lg shadow flex items-center
justify-center w-full"
              >
                <IconTrash
className="mr-2" /> Hapus
              </button>
            </div>
          ))}
        </div>
        {/* Modal Ganti Password */}
        {selectedUser && (
          <div className="fixed inset-0
z-50 flex items-center justify-center bg-
black bg-opacity-60">
            <div className="bg-white
dark:bg-gray-900 p-6 rounded-xl w-full
max-w-md mx-4 shadow-lg relative">
              <h2 className="text-xl
font-semibold mb-6 text-center">
                Ganti Password <span
className="text-indigo-
500">{selectedUser.name}</span>
              </h2>
              <form
onSubmit={handleSubmit}
className="space-y-5">
                {/* Password */}
                <div
className="relative">
                  <InputLabel
htmlFor="password" value="Password
Baru" />
                  <TextInput
                    id="password"
                    type={showPasswo
rd ? "text" : "password"}
                    name="password"
                    value={password}
                    className="mt-1
block w-full"
```

```
                                onChange={(e) =>
setPassword(e.target.value)}
                                placeholder="Masu
kkan password baru"
                            />
                            <button
                                type="button"
                                className="absolu
te right-3 top-8 text-sm text-gray-600
dark:text-gray-300"
                                onClick={() =>
setShowPassword(!showPassword)}
                            >
                                {showPassword ?
"Sembunyikan" : "Lihat"}
                            </button>
                            <InputError
message={errors.password}
className="mt-2" />
                        </div>
                        {/* Confirm Password
*/}
                        <div
className="relative">
                            <InputLabel
htmlFor="password_confirmation"
value="Konfirmasi Password Baru" />
                            <TextInput
                                id="password_conf
irmation"
                                type={showPasswo
rd ? "text" : "password"}
                                name="password_c
onfirmation"
                                value={passwordC
onfirmation}
                                className="mt-1
block w-full"
                                onChange={(e) =>
setPasswordConfirmation(e.target.value)}
                                placeholder="Ulan
gi password baru"
                            />
                            <button
                                type="button"
                                className="absolu
te right-3 top-8 text-sm text-gray-600
dark:text-gray-300"
                                onClick={() =>
setShowPassword(!showPassword)}
                            >
                                {showPassword ?
"Sembunyikan" : "Lihat"}
                            </button>
                            <InputError
message={errors.password_confirmation}
className="mt-2" />
                        </div>
                        {/* Submit */}
                        <PrimaryButton
className="w-full bg-blue-600 hover:bg-
blue-700 py-2 px-4 rounded-lg shadow-lg
text-white transition">
                            Simpan Password
                        </PrimaryButton>
                        {/* Close Modal */}
                        <div className="text-
center mt-4">
                            <button onClick={()
=> setSelectedUser(null)} className="text-
sm text-gray-500 hover:text-gray-800
dark:text-gray-400 dark:hover:text-white
transition">
                                Batal
                            </button>
                        </div>
                    </form>
                </div>
            </div>
        )}
    </div>
</AuthenticatedLayout>

);
}
```

- **AUTH CONFIRMPASSWORD.JSX**

```
import { useEffect } from 'react';
import GuestLayout from
'@/Layouts/GuestLayout';
import InputError from
'@/Components/InputError';
import InputLabel from
'@/Components/InputLabel';
import PrimaryButton from
'@/Components/PrimaryButton';
import TextInput from
'@/Components/TextInput';
import { Head, useForm } from
'@inertiajs/react';
export default function ConfirmPassword() {
    const { data, setData, post, processing,
errors, reset } = useForm({
        password: '',
    });
    useEffect(() => {
        return () => {
            reset('password');
        };
    }, []);
    const submit = (e) => {
        e.preventDefault();
        post(route('password.confirm'));
    };
    return (
        <GuestLayout>
            <Head title="Confirm Password" />
            <div className="mb-4 text-sm text-
gray-600 dark:text-gray-400">
                This is a secure area of the
application. Please confirm your password
before continuing.
            </div>

            <form onSubmit={submit}>
                <div className="mt-4">
```

```
                <InputLabel
htmlFor="password" value="Password" />
                <TextInput
                    id="password"
                    type="password"
                    name="password"
                    value={data.password}
                    className="mt-1 block w-
full"
                    isFocused={true}
                    onChange={(e) =>
setData('password', e.target.value)}
                />
                <InputError
message={errors.password}
className="mt-2" />
            </div>
            <div className="flex items-
center justify-end mt-4">
                <PrimaryButton
className="ms-4" disabled={processing}>
                    Confirm
                </PrimaryButton>
            </div>
        </form>
    </GuestLayout>
  );
}
```

FORGOTPASSWORD.JSX

```
import GuestLayout from
'@/Layouts/GuestLayout';
import InputError from
'@/Components/InputError';
import PrimaryButton from
'@/Components/PrimaryButton';
import TextInput from
'@/Components/TextInput';
import { Head, useForm } from
'@inertiajs/react';

export default function ForgotPassword({
status }) {
  const { data, setData, post, processing,
errors } = useForm({
    email: '',
  });
  const submit = (e) => {
    e.preventDefault();
    post(route('password.email'));
  };
  return (
    <GuestLayout>
      <Head title="Forgot Password" />
      <div className="mb-4 text-sm text-
gray-600 dark:text-gray-400">
        Forgot your password? No
problem. Just let us know your email
address and we will email you a password
        reset link that will allow you to
choose a new one.
      </div>
      {status && <div className="mb-4
font-medium text-sm text-green-600
dark:text-green-400">{status}</div>}
      <form onSubmit={submit}>
        <TextInput
          id="email"
          type="email"
          name="email"
          value={data.email}
          className="mt-1 block w-full"
          isFocused={true}
          onChange={(e) =>
setData('email', e.target.value)}
        />
        <InputError
message={errors.email} className="mt-2"
/>
        <div className="flex items-
center justify-end mt-4">
          <PrimaryButton
className="ms-4" disabled={processing}>
            Email Password Reset Link
          </PrimaryButton>
        </div>
      </form>
    </GuestLayout>
  );
}
```

LOGIN.JSX

```
import { useEffect } from 'react';
import Checkbox from
'@/Components/Checkbox';
import GuestLayout from
'@/Layouts/GuestLayout';
import InputError from
'@/Components/InputError';
import InputLabel from
'@/Components/InputLabel';
import PrimaryButton from
'@/Components/PrimaryButton';
import TextInput from
'@/Components/TextInput';
import { Head, Link, useForm } from
'@inertiajs/react';
export default function Login({ status,
canResetPassword }) {
  const { data, setData, post, processing,
errors, reset } = useForm({
    email: '',
    password: '',
    remember: false,
  });
  useEffect(() => {
    return () => {
      reset('password');
    };
  }, []);
  const submit = (e) => {
    e.preventDefault();
    post(route('login'));
  };
  return (
    <GuestLayout>
```

```
            <div className="">
                <h2 className="text-2xl font-
bold text-green-700 mb-6 text-
center">Masuk ke Akun Anda</h2>
                {status && <div className="mb-
4 font-medium text-sm text-green-
600">{status}</div>}
                <form onSubmit={submit}>
                    <div className="mb-4">
                        <InputLabel htmlFor="email"
value="Email" />
                        <TextInput
                            id="email"
                            type="email"
                            name="email"
                            value={data.email}
                            className="mt-1 block w-
full"
                            autoComplete="username"
                            isFocused={true}
                            onChange={(e) =>
setData('email', e.target.value)}
                        />
                        <InputError
message={errors.email} className="mt-2"
/>
                    </div>
                    <div className="mb-4">
                        <InputLabel
htmlFor="password" value="Password" />
                        <TextInput
                            id="password"
                            type="password"
                            name="password"
                            value={data.password}
                            className="mt-1 block w-
full"
                            autoComplete="current-
password"
                            onChange={(e) =>
setData('password', e.target.value)}
                        />

                        <InputError
message={errors.password}
className="mt-2" />
                    </div>
                    <div className="flex items-
center justify-between mb-4">
                        <label className="flex
items-center">
                            <Checkbox
                                name="remember"
                                checked={data.remembe
r}
                                onChange={(e) =>
setData('remember', e.target.checked)}
                            />
                            <span className="ml-2
text-sm text-gray-700">Ingat saya</span>
                        </label>
                        <Link
                            href={route('show.check.e
mail')}
                            className="text-sm text-
green-600 hover:underline"
                        >
                            Lupa password?
                        </Link>
                    </div>
                    <div className="flex items-
center justify-between">
                        {canResetPassword && (
                            <Link
                                href={route('register')}
                                className="text-sm
text-green-600 hover:underline"
                            >
                                Belum punya akun?
                            </Link>
                        )}
                        <PrimaryButton
                            className="bg-green-700
hover:bg-green-800 focus:ring-green-500"
                            disabled={processing}
                        >
                            Masuk
                        </PrimaryButton>
                    </div>
                </form>
            </div>
        </GuestLayout>

    );
}
```

REGISTER.JSX

```
import { useEffect } from 'react';
import GuestLayout from
'@/Layouts/GuestLayout';
import InputError from
'@/Components/InputError';
import InputLabel from
'@/Components/InputLabel';
import PrimaryButton from
'@/Components/PrimaryButton';
import TextInput from
'@/Components/TextInput';
import { Head, Link, useForm } from
'@inertiajs/react';
export default function Register() {
    const { data, setData, post, processing,
errors, reset } = useForm({
        name: '',
        no: '',
        email: '',
        password: '',
        password_confirmation: '',
    });
    useEffect(() => {
        return () => {
            reset('password',
'password_confirmation');
        };
    }, []);
    const submit = (e) => {
        e.preventDefault();
        post(route('register'));
```

```
    };

    // Menambahkan "+62" ke input nomor WhatsApp secara otomatis
    const handleNoWaChange = (e) => {
        const input = e.target.value.replace(/[^0-9]/g, ''); // hanya angka
        if (!input.startsWith("62")) {
            setData('no', '62' + input.replace(/^0+/, '')); // hilangkan 0 di depan
        } else {
            setData('no', input);
        }
    };
    return (
        <GuestLayout>
<div className="">
            <h2 className="text-2xl font-bold text-green-700 mb-6 text-center">Daftar Akun Baru</h2>
            <form onSubmit={submit}>
                <div className="mb-4">
                    <InputLabel htmlFor="name" value="Nama Lengkap" />
                    <TextInput
                        id="name"
                        name="name"
                        value={data.name}
                        className="mt-1 block w-full"
                        autoComplete="name"
                        isFocused={true}
                        onChange={(e) => setData('name', e.target.value)}
                        required
                    />
                    <InputError message={errors.name} className="mt-2" />
                </div>
                <div className="mb-4">
                    <InputLabel htmlFor="no" value="Nomor WhatsApp" />
                    <TextInput
                        id="no"
                        name="no"
                        value={data.no}
                        className="mt-1 block w-full"
                        autoComplete="tel"
                        onChange={handleNoWaChange}
                        required
                    />
                    <p className="text-xs text-gray-500 mt-1">
                        Nomor WhatsApp harus dimulai dengan <strong>62</strong> (contoh: <code>6281234567890</code>) tanpa spasi atau tanda baca.
                    </p>
                    <InputError message={errors.no} className="mt-2" />
                </div>

                <div className="mb-4">
                    <InputLabel htmlFor="email" value="Email" />
                    <TextInput
                        id="email"
                        type="email"
                        name="email"
                        value={data.email}
                        className="mt-1 block w-full"
                        autoComplete="username"
                        onChange={(e) => setData('email', e.target.value)}
                        required
                    />
                    <InputError message={errors.email} className="mt-2" />
                </div>
                <div className="mb-4">
                    <InputLabel htmlFor="password" value="Password" />
                    <TextInput
                        id="password"
                        type="password"
                        name="password"
                        value={data.password}
                        className="mt-1 block w-full"
                        autoComplete="new-password"
                        onChange={(e) => setData('password', e.target.value)}
                        required
                    />
                    <InputError message={errors.password} className="mt-2" />
                </div>
                <div className="mb-6">
                    <InputLabel htmlFor="password_confirmation" value="Konfirmasi Password" />
                    <TextInput
                        id="password_confirmation"
                        type="password"
                        name="password_confirmation"
                        value={data.password_confirmation}
                        className="mt-1 block w-full"
                        autoComplete="new-password"
                        onChange={(e) => setData('password_confirmation', e.target.value)}
                        required
                    />
```

```
                <InputError
message={errors.password_confirmation}
className="mt-2" />
            </div>
            <div className="flex items-
center justify-between">
                <Link
                    href={route('login')}
                    className="text-sm text-
green-600 hover:underline"
                >
                    Sudah punya akun?
                </Link>
                <PrimaryButton
                    className="bg-green-700
hover:bg-green-800 focus:ring-green-500"
                    disabled={processing}
                >
                    Daftar
                </PrimaryButton>
            </div>
        </form>
    </div>
    </GuestLayout>

  );
}
```

RESETPASSWORD.JSX

```
import { useEffect } from 'react';
import GuestLayout from
'@/Layouts/GuestLayout';
import InputError from
'@/Components/InputError';
import InputLabel from
'@/Components/InputLabel';
import PrimaryButton from
'@/Components/PrimaryButton';
import TextInput from
'@/Components/TextInput';
import { Head, useForm } from
'@inertiajs/react';
export default function ResetPassword({
token, email }) {
  const { data, setData, post, processing,
errors, reset } = useForm({
    token: token,
    email: email,
    password: '',
    password_confirmation: '',
  });

  useEffect(() => {
    return () => {
      reset('password',
'password_confirmation');
    };
  }, []);
  const submit = (e) => {
    e.preventDefault();
    post(route('password.store'));
  };
  return (
    <GuestLayout>
      <Head title="Reset Password" />
      <form onSubmit={submit}>
        <div>
          <InputLabel htmlFor="email"
value="Email" />
          <TextInput
            id="email"
            type="email"
            name="email"
            value={data.email}
            className="mt-1 block w-
full"
            autoComplete="username"
            onChange={(e) =>
setData('email', e.target.value)}
          />
          <InputError
message={errors.email} className="mt-2"
/>
        </div>
        <div className="mt-4">
          <InputLabel
htmlFor="password" value="Password" />
          <TextInput
            id="password"
            type="password"
            name="password"
            value={data.password}
            className="mt-1 block w-
full"
            autoComplete="new-
password"
            isFocused={true}
            onChange={(e) =>
setData('password', e.target.value)}
          />
          <InputError
message={errors.password}
className="mt-2" />
        </div>
        <div className="mt-4">
          <InputLabel
htmlFor="password_confirmation"
value="Confirm Password" />
          <TextInput
            type="password"
            id="password_confirmation"
            name="password_confirmati
on"
            value={data.password_confir
mation}
            className="mt-1 block w-
full"
            autoComplete="new-
password"
            onChange={(e) =>
setData('password_confirmation',
e.target.value)}
          />
          <InputError
message={errors.password_confirmation}
className="mt-2" />
        </div>
```

```
            <div className="flex items-
center justify-end mt-4">
                <PrimaryButton
className="ms-4" disabled={processing}>
                    Reset Password
                </PrimaryButton>
            </div>
        </form>
    </GuestLayout>
  );
}
```

VERIYEMAIL.JSX

```
import GuestLayout from
'@/Layouts/GuestLayout';
import PrimaryButton from
'@/Components/PrimaryButton';
import { Head, Link, useForm } from
'@inertiajs/react';
export default function VerifyEmail({ status
}) {
  const { post, processing } = useForm({});
  const submit = (e) => {
    e.preventDefault();
    post(route('verification.send'));
  };
  return (
    <GuestLayout>
      <Head title="Email Verification" />
      <div className="mb-4 text-sm text-
gray-600 dark:text-gray-400">
        Thanks for signing up! Before
getting started, could you verify your email
address by clicking on the
        link we just emailed to you? If you
didn't receive the email, we will gladly send
you another.
      </div>
      {status === 'verification-link-sent'
&& (
        <div className="mb-4 font-
medium text-sm text-green-600 dark:text-
green-400">
          A new verification link has been
sent to the email address you provided
during registration.
        </div>
      )}
      <form onSubmit={submit}>
        <div className="mt-4 flex items-
center justify-between">
          <PrimaryButton
disabled={processing}>Resend Verification
Email</PrimaryButton>
          <Link
            href={route('logout')}
            method="post"
            as="button"
            className="underline text-
sm text-gray-600 dark:text-gray-400
hover:text-gray-900 dark:hover:text-gray-
100 rounded-md focus:outline-none
focus:ring-2 focus:ring-offset-2 focus:ring-
indigo-500 dark:focus:ring-offset-gray-800"
          >
            Log Out
          </Link>
        </div>
      </form>
    </GuestLayout>
  );
}
```

- **CART**

CHECKOUTORDER.JSX

```
import React, { useState } from 'react';
import { useForm } from '@inertiajs/react';
export default function CheckoutOrder({
produk }) {
  const [quantity, setQuantity] = useState(1);
  const { data, setData, post, processing,
errors } = useForm({
    produk_id: produk.id,
    quantity: 1,
  });
  const handleChange = (e) => {
    const value = Math.max(1,
parseInt(e.target.value) || 1); // Minimal 1
    setQuantity(value);
    setData('quantity', value);
  };
  const handleSubmit = (e) => {
    e.preventDefault();
    post(route('cart.store')); // Pastikan nama
route sesuai di Laravel
  };

  const total = quantity * produk.harga;
  return (
    <form onSubmit={handleSubmit}
className="mt-6 space-y-4">
      <div>
        <label htmlFor="quantity"
className="block text-sm font-medium
text-gray-700">
          Jumlah Pesanan
        </label>
        <input
          type="number"
          id="quantity"
          name="quantity"
          min={1}
          value={quantity}
          onChange={handleChange}
          className="mt-1 w-24 border
border-gray-300 rounded px-3 py-1
focus:outline-none focus:ring-2 focus:ring-
green-400"
        />
        {errors.quantity && <p
className="text-sm text-red-500 mt-
1">{errors.quantity}</p>}
      </div>
      <div>
```

```
      <p className="text-sm text-gray-600">
        Total Harga: {' '}
        <span className="font-semibold text-green-600">
          Rp {total.toLocaleString("id-ID")}
        </span>
      </p>
    </div>
    <button
      type="submit"
      disabled={processing}
      className="bg-green-600 hover:bg-green-700 text-white font-semibold px-6 py-2 rounded-lg transition"
    >
      {processing ? 'Memproses...' : 'Pesan Sekarang'}
    </button>
  </form>
 );
}
```

INDEX.JSX

```
import React from 'react';
import { Head, router } from '@inertiajs/react';
import PesananPage from '../PesananPage';
export default function Index({ cart }) {
 const items = cart?.items || [];
 const handleDelete = (id) => {
  if (confirm('Apakah Anda yakin ingin menghapus item ini dari keranjang?')) {
    router.delete(route('cart.destroy', id));
  }
 };
 const totalHarga = items.reduce((sum, item) => sum + item.subtotal, 0);
 return (
  <>
    <Head title="Keranjang Belanja" />
    <div className="max-w-5xl mx-auto py-10 px-4 space-y-8">
      {/* Header */}
      <div className="flex items-center justify-between">
        <h1 className="text-3xl font-bold text-green-700">🛒 Keranjang Belanja</h1>
        <button
          onClick={() => router.visit('/dashboard')}
          className="bg-gray-200 hover:bg-gray-300 text-gray-800 px-4 py-2 rounded shadow"
        >
          ← Kembali ke Dashboard
        </button>
      </div>

      {/* Konten */}
      {items.length === 0 ? (
        <div className="text-center py-16">
          <p className="text-gray-500 text-lg">Keranjang kamu masih kosong.</p>
        </div>
      ) : (
        <div className="space-y-6">
          {/* List Item */}
          {items.map(item => (
            <div
              key={item.id}
              className="flex items-start justify-between p-4 border rounded-lg shadow-sm bg-white"
            >
              <div>
                <h2 className="text-lg font-semibold text-gray-800">{item.produk.nama_barang}</h2>
                <p className="text-sm text-gray-600 mt-1">
                  Jumlah: <strong>{item.quantity}</strong> × Rp {item.produk.harga.toLocaleString('id-ID')}
                </p>
                <p className="text-sm text-green-700 font-semibold mt-1">
                  Subtotal: Rp {item.subtotal.toLocaleString('id-ID')}
                </p>
              </div>
              <button
                onClick={() => handleDelete(item.id)}
                className="bg-red-500 hover:bg-red-600 text-white px-4 py-2 rounded shadow transition"
              >
                Hapus
              </button>
            </div>
          ))}
          {/* Total + Checkout */}
          <div className="p-6 border-t border-gray-300 bg-gray-50 rounded-lg shadow-sm">
            <h2 className="text-xl font-bold text-green-800 mb-4">
              Total Belanja: Rp {totalHarga.toLocaleString('id-ID')}
            </h2>

            {/* Komponen Checkout */}
            <PesananPage total={totalHarga} />
          </div>
        </div>
      )}
    </div>
  </>
 );
}
```

- **CATEGORY**

CREATE.JSX

```
import React from 'react';
```

```
import { Head, useForm, Link } from
'@inertiajs/react';
import { IconArrowLeft } from '@tabler/icons-
react';
export default function Create() {
  const { data, setData, post, progress, errors } =
useForm({
    kategori: '',
  });
  const handleSubmit = (e) => {
    e.preventDefault();
    post(route('category.store'));
  };
  return (
    <>
      <Head title="Tambah Kategori" />
      <div className="max-w-2xl mx-auto mt-10
px-4">
        <div className="flex items-center justify-
between mb-6">
          <h1 className="text-3xl font-bold text-
green-700">Tambah Kategori</h1>
          <Link
            href={route('category.index')}
            className="flex items-center text-sm
text-green-600 hover:underline hover:text-green-
800 transition"
          >
            <IconArrowLeft className="w-5 h-5
mr-1" /> Kembali ke Dashboard
          </Link>
        </div>
        <div className="bg-white shadow-md
rounded-2xl p-6 border border-green-200">
          <form onSubmit={handleSubmit}
className="space-y-5">
            <div>
              <label className="block text-sm font-
medium text-gray-700 mb-1">
                Nama Kategori
              </label>
              <input
                type="text"
                className="w-full px-4 py-2 border
border-gray-300 rounded-lg focus:outline-none
focus:ring-2 focus:ring-green-400"
                value={data.kategori}
                onChange={(e) => setData('kategori',
e.target.value)}
                placeholder="Masukkan nama
kategori"
              />
              {errors.kategori && (
                <p className="text-red-500 text-sm
mt-1">{errors.kategori}</p>
              )}
            </div>
            <div className="pt-2">
              <button
                type="submit"
                className="bg-green-600 hover:bg-
green-700 text-white px-6 py-2 rounded-lg
transition font-semibold"
              >
                Simpan Kategori
              </button>
            </div>
            {progress && (
              <div className="text-sm text-gray-
600">
                Upload: {progress.percentage}%
              </div>
            )}
          </form>
        </div>
      </div>
    </>
  );
}
```

INDEX.JSX

```
import { useState } from "react";
import { Head, usePage, Link, router } from
'@inertiajs/react';
import { IconCategory, IconTrash, IconEdit,
IconEye,IconArrowLeft } from '@tabler/icons-
react';
import Swal from "sweetalert2";
import { div } from "framer-motion/client";
export default function Index({ category }) {
    const { auth } = usePage().props;
  const [isDeleting, setIsDeleting] =
useState(false);
  const Delete = (id) => {
    Swal.fire({
      title: "Apakah Anda yakin?",
      text: "Anda tidak akan dapat mengembalikan
ini!",
      icon: "warning",
      showCancelButton: true,
      confirmButtonColor: "#308d46",
      cancelButtonColor: "#d33",
      confirmButtonText: "Ya, hapus saja!",
    }).then((result) => {
      if (result.isConfirmed) {
        setIsDeleting(true);
        router.delete(route("category.destroy", id), {
          onSuccess: () => {
            setIsDeleting(false);
            Swal.fire("Dihapus!", "Kategori telah
dihapus.", "success");
          },
          onError: () => {
            setIsDeleting(false);
            Swal.fire("Gagal!", "Terjadi kesalahan
saat menghapus kategori.", "error");
          },
        });
      }
    });
  };
  return (
    <>
      <Head title="Daftar Kategori" />
      <div className="max-w-6xl mx-auto mt-10
p-4">
        <div className="flex justify-between
items-center mb-6">
```

```
      <div>
        <h1 className="text-3xl md:text-4xl font-bold text-green-700 mb-1">Kategori Baju Adat</h1>
        <p className="text-gray-600">Jelajahi berbagai kategori pakaian adat dari seluruh Bugis</p>
      </div>
      <div className="flex flex-row gap-5">
      {auth.user ? (
      <Link
        href={route('dashboard')}
        className="bg-white border border-green-600 text-green-700 px-4 py-2 rounded-lg hover:bg-green-100 transition flex items-center"
      >
        <IconArrowLeft className="w-5 h-5 mr-2" /> Kembali
      </Link>
      ):( <div className=""></div> )}
      {auth.roles.includes("admin") && (
          <Link
          href={route("category.create")}
          className="bg-green-600 text-white px-4 py-2 rounded-lg hover:bg-green-700 transition flex items-center"
        >
          <IconEdit className="w-5 h-5 mr-2" /> Tambah
        </Link>
      )}
      </div>
    </div>
    <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-6">
      {category && category.length > 0 ? (
       category.map((item, index) => (
        <div
          key={index}
          className="bg-white rounded-2xl shadow-md hover:shadow-lg transition duration-300 p-6 border border-green-200"
        >
          <div className="flex items-center gap-4 mb-4">
            <div className="w-14 h-14 bg-green-600 text-white flex items-center justify-center rounded-full">
              <IconCategory className="w-7 h-7" />
            </div>
            <div>
              <h2 className="text-xl font-semibold text-green-800">{item.kategori}</h2>
            </div>
          </div>
          <div className="flex justify-between items-center border-t pt-4 mt-4 text-sm">
            {auth.roles.includes("admin") && (
                <Link
                href={route("category.edit", item.id)}
                className="text-green-700 hover:underline flex items-center"
              >
                <IconEdit className="w-4 h-4 mr-1" /> Edit
              </Link>
            )}
            <Link
              href={route("Produk.index", item.id)}
              className="text-blue-600 hover:underline flex items-center"
            >
              <IconEye className="w-4 h-4 mr-1" /> Lihat
            </Link>
            {auth.roles.includes("admin") && (
                <button
                onClick={() => Delete(item.id)}
                disabled={isDeleting}
                className="text-red-600 hover:underline flex items-center"
              >
                <IconTrash className="w-4 h-4 mr-1" /> Hapus
              </button>
            )}
          </div>
        </div>
       ))
      ) : (
        <p className="text-gray-500 text-center col-span-full">Belum ada kategori.</p>
      )}
    </div>
  </div>
  </>
 );
}
```

UPDATE.JSX

```
import React from 'react';
import { Head, useForm, Link } from '@inertiajs/react';
import { IconArrowLeft } from '@tabler/icons-react';

export default function Update({ category }) {
 const { data, setData, post, progress, errors } = useForm({
   kategori: category.kategori || '',
 });
 const handleSubmit = (e) => {
  e.preventDefault();
  post(route('category.update', category.id), {
   preserveScroll: true,
  });
 };
 return (
  <>
   <Head title="Edit Kategori" />
   <div className="max-w-2xl mx-auto mt-10 px-4">
    <div className="flex items-center justify-between mb-6">
```

```
        <h1 className="text-3xl font-bold text-
green-700">Edit Kategori</h1>
        <Link
         href={route('category.index')}
          className="flex items-center text-sm
text-green-600 hover:text-green-800
hover:underline transition"
        >
          <IconArrowLeft className="w-5 h-5
mr-1" /> Kembali ke Dashboard
        </Link>
       </div>
       <div className="bg-white shadow-md
rounded-2xl p-6 border border-green-200">
        <form onSubmit={handleSubmit}
className="space-y-5">
          <div>
           <label className="block text-sm font-
medium text-gray-700 mb-1">
             Nama Kategori
           </label>
           <input
             type="text"
             className="w-full px-4 py-2 border
border-gray-300 rounded-lg focus:outline-none
focus:ring-2 focus:ring-green-400"
             value={data.kategori}
             onChange={(e) => setData('kategori',
e.target.value)}
             placeholder="Masukkan nama
kategori"
           />
           {errors.kategori && (
             <p className="text-red-500 text-sm
mt-1">{errors.kategori}</p>
           )}
          </div>
          <div className="pt-2">
           <button
             type="submit"
             className="bg-green-600 hover:bg-
green-700 text-white px-6 py-2 rounded-lg
transition font-semibold"
           >
             Update Kategori
           </button>
          </div>
          {progress && (
           <div className="text-sm text-gray-
600">
             Upload: {progress.percentage}%
           </div>
          )}
        </form>
       </div>
     </div>
    </>
  );
}
```

- **PEMINJAMAN**

INDEX.JSX

```
import { useState } from "react";
import { Head, usePage, Link, router } from
'@inertiajs/react';
import { IconUser, IconBrandWhatsapp,
IconArrowLeft, IconShoppingBag } from
'@tabler/icons-react';
import Swal from "sweetalert2";
export default function Index({ Pesanan }) {
  const { auth } = usePage().props;
  const [activeTab, setActiveTab] =
useState('menunggu konfirmasi');
  const [kondisiPesanan, setKondisiPesanan] =
useState(() => {
    // Ambil dari localStorage saat komponen
pertama kali render
    const stored =
localStorage.getItem('kondisiPesanan');
    return stored ? JSON.parse(stored) : {};
  });
  const handleKonfirmasiCash = (id) => {
    Swal.fire({
      title: 'Konfirmasi Pesanan Cash',
      text: 'Apakah kamu yakin ingin
mengkonfirmasi pesanan ini?',
      icon: 'question',
      showCancelButton: true,
      confirmButtonText: 'Ya, Konfirmasi',
      cancelButtonText: 'Batal',
      confirmButtonColor: '#16a34a',
      cancelButtonColor: '#d33',
    }).then((result) => {
      if (result.isConfirmed) {
       router.post(route('pesanan.konfirmasiCash',
id), {}, {
         onSuccess: () => {
           Swal.fire({
             icon: 'success',
             title: 'Berhasil',
             text: 'Pesanan telah dikonfirmasi!',
           });
         },
         onError: () => {
           Swal.fire({
             icon: 'error',
             title: 'Gagal',
             text: 'Terjadi kesalahan saat
mengkonfirmasi pesanan.',
           });
         }
       });
      }
    });
  };
  const handleSubmit = (id) => {
    Swal.fire({
      title: 'Kondisi Baju Saat Dikembalikan',
      input: 'radio',
      inputOptions: {
        normal: 'Baju Normal',
        rusak_ringan: 'Baju Setengah Rusak',
        rusak_parah: 'Baju Rusak Parah',
      },
```

```
      inputValidator: (value) => {
        if (!value) {
          return 'Pilih salah satu kondisi baju!';
        }
      },
      showCancelButton: true,
      confirmButtonText: 'Simpan',
      cancelButtonText: 'Batal',
      confirmButtonColor: '#16a34a',
      cancelButtonColor: '#d33',
    }).then((result) => {
      if (result.isConfirmed) {
        const kondisi = result.value;
        if (kondisi === 'normal') {
          // Kirim ke backend untuk update status
          router.post(route("Peminjaman.selesai", id), {}, {
            onSuccess: () => {
              Swal.fire({
                icon: 'success',
                title: 'Pesanan Selesai',
                text: 'Pesanan berhasil ditandai sebagai selesai.',
              });
              // Hapus dari localStorage karena sudah selesai
              setKondisiPesanan(prev => {
                const updated = { ...prev, [id]: 'selesai' };
                localStorage.setItem('kondisiPesanan', JSON.stringify(updated));
                return updated;
              });
              // Optional: bersihkan dari storage (agar status lokal hilang)
              const stored = JSON.parse(localStorage.getItem('kondisiPesanan')) || {};
              delete stored[id];
              localStorage.setItem('kondisiPesanan', JSON.stringify(stored));
            },
            onError: () => {
              Swal.fire({
                icon: 'error',
                title: 'Gagal',
                text: 'Terjadi kesalahan saat menyelesaikan pesanan.',
              });
            }
          });
        } else {
          // Simpan ke localStorage dan update state
          Swal.fire({
            icon: 'info',
            title: 'Perlu Tindakan Admin',
            text: 'Pesanan belum bisa diselesaikan karena kondisi baju tidak normal.',
          });
          setKondisiPesanan(prev => {
            const updated = { ...prev, [id]: kondisi };
            localStorage.setItem('kondisiPesanan', JSON.stringify(updated));
            return updated;
          });
        }
      }
    });
  };
  const handleDelete = (id) => {
    Swal.fire({
      title: 'Hapus Pesanan?',
      text: 'Data pesanan beserta jaminan (KTP) akan dihapus. Lanjutkan?',
      icon: 'warning',
      showCancelButton: true,
      confirmButtonColor: '#d33',
      cancelButtonColor: '#3085d6',
      confirmButtonText: 'Ya, Hapus!',
      cancelButtonText: 'Batal'
    }).then((result) => {
      if (result.isConfirmed) {
        router.delete(route('Peminjaman.delete', id), {
          onSuccess: () => {
            Swal.fire('Terhapus!', 'Pesanan telah dihapus.', 'success');
          },
          onError: () => {
            Swal.fire('Gagal!', 'Gagal menghapus pesanan.', 'error');
          }
        });
      }
    });
  };
  const isAdmin = auth.user.roles.some(role => role.name === 'admin');
  const whatsappAdminNumber = isAdmin ? `+62${auth.user.no}` : '+6281234567890'; // Ganti dengan nomor default
  return (
    <>
      <Head title="Peminjaman Baju" />
      <div className="bg-gradient-to-br from-blue-50 to-white min-h-screen py-10">
        <div className="max-w-6xl mx-auto px-4">
          {/* Tombol Kembali */}
          <Link
            href={route('dashboard')}
            className="flex items-center text-blue-600 hover:text-blue-800 mb-4"
          >
            <IconArrowLeft className="w-5 h-5 mr-2" />
            Kembali ke Dashboard
          </Link>
          <div className="bg-white rounded-2xl shadow-lg p-8">
            <h1 className="text-3xl font-bold text-blue-700 mb-4">Halaman Peminjaman Baju</h1>
            <p className="text-gray-700 leading-relaxed mb-6">
              Selamat datang di halaman peminjaman baju! Di sini, kamu dapat melihat status pesanan baju yang telah kamu pinjam.
```

```
            Kami menyediakan layanan peminjaman baju untuk berbagai keperluan seperti wisuda, acara formal, photoshoot, dan lain-lain.
            Hubungi admin melalui WhatsApp jika kamu membutuhkan bantuan pengantaran atau informasi lebih lanjut.
          </p>
          <div className="bg-yellow-50 border-l-4 border-yellow-400 p-5 rounded-lg shadow">
            <h2 className="text-lg font-semibold text-yellow-700 mb-2">🔒 Keamanan Data KTP</h2>
            <p className="text-sm text-gray-700 leading-relaxed">
              Kami menjamin keamanan dan kerahasiaan data pribadi Anda, termasuk foto KTP yang digunakan sebagai jaminan peminjaman.
              Data tersebut tidak akan disalahgunakan dan hanya digunakan untuk keperluan administrasi selama proses peminjaman berlangsung.
            </p>
          </div>
          <div className="bg-red-50 border-l-4 border-red-400 p-5 rounded-lg shadow mt-5">
            <h2 className="text-lg font-semibold text-red-700 mb-2">💸 Denda Keterlambatan</h2>
            <p className="text-sm text-gray-700 leading-relaxed">
              Apabila terjadi keterlambatan dalam pengembalian baju, maka akan dikenakan denda sebesar <strong>Rp 30.000 per hari</strong>.
              Pastikan untuk mengembalikan baju tepat waktu agar terhindar dari biaya tambahan.
            </p>
          </div>
          {/* Informasi Pengambilan Barang */}
          <div className="mb-4 mt-5">
            <p className="text-sm text-gray-700 leading-relaxed">
              Untuk pengambilan barang, Anda dapat <strong>menghubungi admin</strong> melalui WhatsApp atau <strong>langsung datang ke toko</strong> kami.
              Pastikan untuk menyertakan nama produk saat menghubungi admin untuk memudahkan proses peminjaman.
            </p>
          </div>
          {/* Tombol WA */}
          <div className="mb-6">
            <p className="text-sm text-gray-600 mb-2">Hubungi Admin:</p>
            <a
              href={`https://wa.me/${whatsappAdminNumber}?text=Halo,%20saya%20ingin%20bertanya%20mengenai%20peminjaman%20baju%20dengan%20nomor%20pesanan:.`}
              target="_blank"
              rel="noopener noreferrer"
              className="inline-flex items-center bg-green-500 hover:bg-green-600 text-white px-4 py-2 rounded-lg transition"
            >
              <IconBrandWhatsapp className="w-5 h-5 mr-2" />
              Chat WhatsApp
            </a>
          </div>
          <div className="flex gap-2 mb-6">
            {['menunggu konfirmasi', 'peminjaman', 'selesai'].map((status) => (
              <button
                key={status}
                onClick={() => setActiveTab(status)}
                className={`px-4 py-2 rounded-lg text-sm font-medium transition ${activeTab === status
                  ? 'bg-blue-600 text-white'
                  : 'bg-gray-200 hover:bg-gray-300 text-gray-700'
                }`}
              >
                {status.charAt(0).toUpperCase() + status.slice(1)}
              </button>
            ))}
          </div>
          {/* List Pesanan */}
          <div className="grid grid-cols-1 md:grid-cols-2 xl:grid-cols-3 gap-6">
            {Pesanan && Pesanan.length > 0 ? (
              Pesanan.filter(item => item.status === activeTab).map((item, index) => (
                <div
                  key={index}
                  className="bg-gray-100 hover:bg-white border border-gray-200 rounded-xl shadow-md p-5 transition duration-300 ease-in-out"
                >
                  <div className="flex items-center mb-4">
                    <div className="w-12 h-12 bg-blue-600 text-white flex items-center justify-center rounded-full">
                      <IconShoppingBag className="w-6 h-6" />
                    </div>
                    <div className="ml-3">
                      <h2 className="text-md font-bold text-gray-800">{item.order_id}</h2>
                    </div>
                  </div>
                  {/* Foto Jaminan */}
                  {item.ktp && (
                    <div className="mb-3">
                      <p className="text-gray-700 text-sm mb-1">Foto Jaminan:</p>
                      <img
                        src={item.ktp}
                        alt="Foto Jaminan"
                        className="w-full h-48 object-cover rounded-lg border"
```

```
                    />
                  </div>
                )}
                <p className="text-gray-700 text-
sm mb-1">Total: <strong>Rp
{item.total.toLocaleString('id-
ID')}</strong></p>
                <p className="text-gray-700 text-
sm mb-1">Status: {item.status}</p>
                <p className="text-gray-700 text-
sm mb-1">Jenis pembayaran:
{item.pembayaran}</p>
                {item.status !== 'selesai' &&
kondisiPesanan[item.id] && (
                  <p className="text-sm text-blue-
600 mb-1">
                    Kondisi Baju: <strong>
                      {
                        kondisiPesanan[item.id] ===
'normal'
                          ? 'Baju Normal'
                          : kondisiPesanan[item.id] ===
'rusak_ringan'
                            ? 'Baju Setengah Rusak'
                            : 'Baju Rusak Parah'
                      }
                    </strong>
                  </p>
                )}
                <p className="text-gray-700 text-
sm mb-1">Tanggal Peminjaman:
{item.peminjaman}</p>
                <p className="text-gray-700 text-
sm mb-3">Tanggal Pengembalian:
{item.pengembalian}</p>
                {isAdmin && (
                  <button
                    onClick={() =>
handleDelete(item.id)}
                    className="bg-red-600 hover:bg-
red-700 text-white rounded-lg px-4 py-2 mb-5
text-sm"
                  >
                    Hapus Pesanan
                  </button>
                )}
                {isAdmin && item.status ===
'menunggu konfirmasi' && (
                  <button
                    onClick={() =>
handleKonfirmasiCash(item.id)}
                    className="bg-green-600
hover:bg-green-700 text-white rounded-lg px-4
py-2 text-sm mb-2"
                  >
                    Konfirmasi Pesanan Cash
                  </button>
                )}
                {(item.status === 'peminjaman' &&
kondisiPesanan[item.id] !== 'selesai') ? (
                  isAdmin ? (
                    isAdmin ? (
                      <div className="flex flex-col
gap-2">
                        <button
                          onClick={() =>
handleSubmit(item.id)}
                          className="bg-blue-600
hover:bg-blue-700 text-white rounded-lg px-4
py-2 text-sm"
                        >
                          Tandai Selesai
                        </button>
                        <Link
                          href={route('order-
items.index', item.id)}
                          className="bg-gray-500
hover:bg-gray-600 text-white text-center
rounded-lg px-4 py-2 text-sm"
                        >
                          Detail Pesanan
                        </Link>
                        {kondisiPesanan[item.id] &&
kondisiPesanan[item.id] !== 'selesai' && (
                          <p className="text-red-600
text-sm mt-1 italic">
                            Status baju:
{kondisiPesanan[item.id].replace('_', ' ')}
                          </p>
                        )}
                      </div>
                    ) : (
                      <div className="flex flex-col
gap-2">
                        <p className="text-gray-600
font-semibold mt-2">Pesanan sedang
berjalan</p>
                        <Link
                          href={route('order-
items.index', item.id)}
                          className="bg-gray-500
hover:bg-gray-600 text-white text-center
rounded-lg px-4 py-2 text-sm"
                        >
                          Detail Pesanan
                        </Link>
                      </div>
                    )
                  ) : (
                    <div className="flex flex-col
gap-2">
                      <p className="text-gray-600
font-semibold mt-2">Pesanan sedang
berjalan</p>
                      <Link
                        href={route('order-items.index',
item.id)}
                        className="bg-gray-500
hover:bg-gray-600 text-white text-center
rounded-lg px-4 py-2 text-sm"
                      >
                        Detail Pesanan
                      </Link>
                    </div>
                  )
                ) : (
                  <div className="flex flex-col gap-
2">
```

```
                    <p className="text-green-600
font-semibold mt-2">Pesanan Sudah
Diselesaikan</p>
                    <Link
                      href={route('order-items.index',
item.id)}
                      className="bg-gray-500
hover:bg-gray-600 text-white text-center
rounded-lg px-4 py-2 text-sm"
                    >
                      Detail Pesanan
                    </Link>
                  </div>
                )}
              </div>
            ))
          ) : (
            <p className="text-gray-500 col-
span-full text-center">
              Belum ada pesanan yang tercatat.
            </p>
          )}
        </div>
      </div>
    </div>
  </>
 );
}
```

LAPORAN.JSX

```
import React from 'react';
import { Head, Link, useForm } from
'@inertiajs/react';
import AuthenticatedLayout from
'@/Layouts/AuthenticatedLayout';
import { IconArrowLeft } from '@tabler/icons-
react';
export default function Laporan({ Pesanan,
filters }) {
 const { data, setData, get } = useForm({
  start_date: filters.start_date || '',
  end_date: filters.end_date || '',
  bulan: filters.bulan || '',
  tahun: filters.tahun || '',
 });
 const handleFilter = (e) => {
  e.preventDefault();
  const filterType =
e.nativeEvent.submitter.value;
  if (filterType === 'tanggal') {
   get(route('Peminjaman.laporan', {
    start_date: data.start_date,
    end_date: data.end_date,
   }));
  } else if (filterType === 'bulan') {
   get(route('Peminjaman.laporan', {
    bulan: data.bulan,
    tahun: data.tahun,
   }));
  }
 };
 return (
  <AuthenticatedLayout>
   <Head title="Laporan Peminjaman" />
   <div className="py-10 px-6 max-w-6xl mt-
10 mx-auto">
    <div className="mb-6">
     <Link href={route('Peminjaman.index')}
className="flex items-center text-blue-600
hover:text-blue-800">
      <IconArrowLeft className="w-5 h-5
mr-2" />
      Kembali ke Peminjaman
     </Link>
    </div>

    <h1 className="text-3xl font-bold text-
gray-800 mb-6">📊 Laporan Peminjaman</h1>
    <form onSubmit={handleFilter}
className="flex flex-wrap gap-4 mb-8 bg-white
p-4 rounded-lg shadow">
     {/* Filter Tanggal */}
     <div>
      <label className="block text-sm font-
medium text-gray-700">Dari Tanggal</label>
      <input
       type="date"
       value={data.start_date}
       onChange={(e) => setData('start_date',
e.target.value)}
       disabled={data.bulan || data.tahun}
       className="mt-1 block w-full border-
gray-300 rounded-md shadow-sm
disabled:opacity-50"
      />
     </div>
     <div>
      <label className="block text-sm font-
medium text-gray-700">Sampai Tanggal</label>
      <input
       type="date"
       value={data.end_date}
       onChange={(e) => setData('end_date',
e.target.value)}
       disabled={data.bulan || data.tahun}
       className="mt-1 block w-full border-
gray-300 rounded-md shadow-sm
disabled:opacity-50"
      />
     </div>
     <div className="flex items-end">
      <button
       type="submit"
       name="filter"
       value="tanggal"
       className="bg-blue-600 hover:bg-blue-
700 text-white px-4 py-2 rounded-lg
disabled:opacity-50"
       disabled={!data.start_date ||
!data.end_date || data.bulan || data.tahun}
      >
       Filter Tanggal
      </button>
     </div>
     {/* Filter Bulan/Tahun */}
     <div>
```

```
          <label className="block text-sm font-
medium text-gray-700">Bulan</label>
          <select
            value={data.bulan}
            onChange={(e) => setData('bulan',
e.target.value)}
            disabled={data.start_date ||
data.end_date}
            className="mt-1 block w-full border-
gray-300 rounded-md shadow-sm
disabled:opacity-50"
          >
            <option value="">Semua</option>
            {Array.from({ length: 12 }, (_, i) => (
              <option key={i + 1} value={i + 1}>
                {new Date(0, i).toLocaleString('id-
ID', { month: 'long' })}
              </option>
            ))}
          </select>
        </div>
        <div>
          <label className="block text-sm font-
medium text-gray-700">Tahun</label>
          <select
            value={data.tahun}
            onChange={(e) => setData('tahun',
e.target.value)}
            disabled={data.start_date ||
data.end_date}
            className="mt-1 block w-full border-
gray-300 rounded-md shadow-sm
disabled:opacity-50"
          >
            <option value="">Semua</option>
            {Array.from({ length: 5 }, (_, i) => {
              const year = new Date().getFullYear() -
i;
              return (
                <option key={year} value={year}>
                  {year}
                </option>
              );
            })}
          </select>
        </div>
        <div className="flex items-end">
          <button
            type="submit"
            name="filter"
            value="bulan"
            className="bg-green-600 hover:bg-
green-700 text-white px-4 py-2 rounded-lg
disabled:opacity-50"
            disabled={!data.bulan || !data.tahun ||
data.start_date || data.end_date}
          >
            Filter Bulan
          </button>
        </div>
        {/* Tombol Hapus Filter */}
        {(data.start_date || data.end_date ||
data.bulan || data.tahun) && (
          <div className="flex items-end">
            <button
              type="button"
onClick={() => {
  setData({
    start_date: '',
    end_date: '',
    bulan: '',
    tahun: '',
  });
  // Kembali ke URL tanpa filter (benar-benar
bersih)
  window.location.href =
route('Peminjaman.laporan');
}}
              className="bg-red-500 hover:bg-red-
600 text-white px-4 py-2 rounded-lg"
            >
              Hapus Filter
            </button>
          </div>
        )}
      </form>
      {/* Tabel Laporan */}
      <div className="overflow-x-auto bg-white
rounded-lg shadow">
        <table className="min-w-full table-auto
text-left border">
          <thead className="bg-gray-100 border-
b">
            <tr>
              <th className="px-4 py-2">No</th>
              <th className="px-4 py-2">Kode
Pesanan</th>
              <th className="px-4 py-2">Tanggal
transaksi</th>
              <th className="px-4 py-
2">Total</th>
              <th className="px-4 py-2">jenis
pembayaran</th>
              <th className="px-4 py-
2">Status</th>
            </tr>
          </thead>
          <tbody>
            {Pesanan.length > 0 ? (
              Pesanan.map((item, index) => (
                <tr key={item.id}
className="border-b hover:bg-gray-50">
                  <td className="px-4 py-2">{index
+ 1}</td>
                  <td className="px-4 py-2 font-
semibold text-blue-600">{item.order_id}</td>
                  <td className="px-4 py-2">{new
Date(item.peminjaman).toLocaleDateString('id-
ID')}</td>
                  <td className="px-4 py-2">Rp
{item.total?.toLocaleString('id-ID')}</td>
                  <td className="px-4 py-2
capitalize">{item.pembayaran}</td>
                  <td className="px-4 py-2
capitalize">{item.status}</td>
                </tr>
              ))
            ) : (
```

```
            <tr>
              <td colSpan="5" className="text-
center px-4 py-6 text-gray-500">
                Tidak ada data pemesanan pada
periode ini.
              </td>
            </tr>
          )}
        </tbody>
      </table>
    </div>
  </div>
 </AuthenticatedLayout>
);
}
```

SHOW.JSX

```
import { Head, usePage, router, Link } from
'@inertiajs/react';
import { IconArrowLeft } from '@tabler/icons-
react';
export default function Show() {
 const { pesanan, order_items } =
usePage().props;
 // Hitung durasi peminjaman (pengembalian -
peminjaman - 1)
 const pinjamDate = new
Date(pesanan.peminjaman);
 const kembaliDate = new
Date(pesanan.pengembalian);
 const durasiHari = Math.max(
  Math.floor((kembaliDate - pinjamDate) /
(1000 * 60 * 60 * 24)) ,
  0
 );
 return (
  <>
   <Head title={`Detail Pesanan
${pesanan.order_id}`} />

   <div className="bg-gradient-to-br from-
blue-50 to-white min-h-screen py-10">
     <div className="max-w-6xl mx-auto px-
4">
      {/* Tombol Kembali */}
      <Link
       href={route('Peminjaman.index')}
       className="flex items-center text-blue-
600 hover:text-blue-800 mb-4"
      >
       <IconArrowLeft className="w-5 h-5
mr-2" />
       Kembali ke Daftar Peminjaman
      </Link>
      <div className="bg-white rounded-2xl
shadow-lg p-8">
       <h1 className="text-3xl font-bold text-
blue-700 mb-6">
        Detail Pesanan {pesanan.order_id}
       </h1>
       {/* Info Pesanan */}
       <div className="grid grid-cols-1
md:grid-cols-2 gap-6 mb-8">
        <div>
         <p className="text-gray-
700"><strong>Status:</strong>
{pesanan.status}</p>
         <p className="text-gray-
700"><strong>Jenis pembayaran:</strong>
{pesanan.pembayaran}</p>
         <p className="text-gray-
700"><strong>Tanggal Peminjaman:</strong>
{pesanan.peminjaman}</p>
         <p className="text-gray-
700"><strong>Tanggal Pengembalian:</strong>
{pesanan.pengembalian}</p>
         <p className="text-gray-
700"><strong>Durasi Peminjaman:</strong>
{durasiHari} hari</p>
        </div>
        <div>
         <p className="text-gray-
700"><strong>Total Harga:</strong> Rp
{pesanan.total.toLocaleString('id-ID')}</p>
         {pesanan.ktp && (
          <div className="mt-2">
           <p className="text-gray-700 mb-
1"><strong>Foto KTP:</strong></p>
           <img
            src={`/storage/${pesanan.ktp.repla
ce('public/', '')}`}
            alt="Foto KTP"
            className="w-72 h-40 object-
cover rounded-lg border"
           />
          </div>
         )}
        </div>
       </div>
       {/* Tabel Order Items */}
       <div className="overflow-x-auto">
        <table className="min-w-full table-
auto border-collapse border border-gray-300">
         <thead>
          <tr className="bg-gray-100">
           <th className="border border-gray-
300 p-2">No</th>
           <th className="border border-gray-
300 p-2">Produk</th>
           <th className="border border-gray-
300 p-2">Jumlah</th>
           <th className="border border-gray-
300 p-2">Harga Satuan</th>
           <th className="border border-gray-
300 p-2">Total Harga</th>
          </tr>
         </thead>
         <tbody>
          {order_items.length === 0 ? (
           <tr>
            <td colSpan="5"
className="border border-gray-300 p-4 text-
center text-gray-500">
             Tidak ada item dalam pesanan ini.
            </td>
           </tr>
          ) : (
           order_items.map((item, index) => (
```

```
                <tr key={item.id}>
                  <td className="border border-
gray-300 p-2 text-center">{index + 1}</td>
                  <td className="border border-
gray-300 p-2">{item.produk.nama_barang}</td>
                  <td className="border border-
gray-300 p-2 text-center">{item.quantity}</td>
                  <td className="border border-
gray-300 p-2 text-center">Rp
{item.produk.harga.toLocaleString('id-ID')}</td>
                  <td className="border border-
gray-300 p-2 text-center">Rp
{(item.produk.harga *
item.quantity).toLocaleString('id-ID')}</td>
                </tr>
              ))
            )}
          </tbody>
        </table>
      </div>
      {/* Tombol Kembali */}
      <div className="mt-6">
        <button
          onClick={() =>
router.visit(route('Peminjaman.index'))}
          className="px-4 py-2 bg-blue-600
hover:bg-blue-700 text-white rounded-lg
transition"
        >
          Kembali ke Daftar Pesanan
        </button>
      </div>
    </div>
  </div>
 </>
 );
}
```

- **PRODUK**

CREATE.JSX

```
import React from 'react';
import { Head, useForm, Link } from
'@inertiajs/react';
import { IconArrowLeft } from '@tabler/icons-
react';
export default function Create({ categories }) {
 const { data, setData, post, progress, errors } =
useForm({
  nama_barang: '',
  ukuran: [], // <-- array sekarang
  harga: '',
  dekripsi: '',
  stok: '',
  foto: null,
  status: 'tersedia',
  category_id: '',
 });
 const handleSubmit = (e) => {
  e.preventDefault();
  post(route('Produk.store'));
 };
 const handleUkuranChange = (value) => {
  if (data.ukuran.includes(value)) {
   setData('ukuran', data.ukuran.filter((item) =>
item !== value));
  } else {
   setData('ukuran', [...data.ukuran, value]);
  }
 };
 return (
  <>
   <Head title="Tambah Produk" />

   <div className="max-w-2xl mx-auto mt-12
px-6 py-8 bg-white shadow-lg rounded-2xl
border border-green-100">
    {/* Header */}
    <div className="mb-6 flex items-center
justify-between">
     <h1 className="text-3xl font-bold text-
green-700">Tambah Produk</h1>
     <Link
      href={route('category.index')}
      className="flex items-center text-sm
text-green-600 hover:text-green-800 transition"
     >
      <IconArrowLeft className="w-5 h-5
mr-1" />
      Kembali ke Dashboard
     </Link>
    </div>
    {/* Form */}
    <form onSubmit={handleSubmit}
encType="multipart/form-data"
className="space-y-5">
     {/* Nama Barang */}
     <div>
      <label className="block text-sm font-
semibold mb-1 text-gray-700">Nama
Barang</label>
      <input
       type="text"
       className="w-full border border-green-
300 rounded-md p-2 focus:ring-2 focus:ring-
green-400 focus:outline-none"
       value={data.nama_barang}
       onChange={(e) =>
setData('nama_barang', e.target.value)}
      />
      {errors.nama_barang && <p
className="text-red-500 text-sm mt-
1">{errors.nama_barang}</p>}
     </div>
     <div>
      <label className="block text-sm font-
semibold mb-1 text-gray-700">Dekripsi
Barang</label>
      <textarea
       type="text"
       className="w-full h-56 border border-
green-300 rounded-md p-2 focus:ring-2
focus:ring-green-400 focus:outline-none"
       value={data.dekripsi}
       onChange={(e) => setData('dekripsi',
e.target.value)}
      />
```

```
          {errors.dekripsi && <p className="text-
red-500 text-sm mt-1">{errors.dekripsi}</p>}
        </div>
        <div>
          <label className="block text-sm font-
semibold mb-1 text-gray-700">Stok
Barang</label>
          <input
            type="number"
            className="w-full border border-green-
300 rounded-md p-2 focus:ring-2 focus:ring-
green-400 focus:outline-none"
            value={data.nama_stok}
            onChange={(e) => setData('stok',
e.target.value)}
          />
          {errors.stok && <p className="text-red-
500 text-sm mt-1">{errors.stok}</p>}
        </div>
        {/* Ukuran */}
        <div>
          <label className="block text-sm font-
semibold mb-1 text-gray-700">Ukuran</label>
          <div className="flex gap-4 flex-wrap">
            {['S', 'M', 'L', 'XL', 'XXL'].map((ukuran)
=> (
              <label key={ukuran}
className="inline-flex items-center space-x-2">
                <input
                  type="checkbox"
                  value={ukuran}
                  checked={data.ukuran.includes(ukur
an)}
                  onChange={() =>
handleUkuranChange(ukuran)}
                  className="text-green-600
focus:ring-green-500 border-green-300 rounded"
                />
                <span className="text-sm text-gray-
700">{ukuran}</span>
              </label>
            ))}
          </div>
          {errors.ukuran && <p className="text-
red-500 text-sm mt-1">{errors.ukuran}</p>}
        </div>
        {/* Harga */}
        <div>
          <label className="block text-sm font-
semibold mb-1 text-gray-700">Harga</label>
          <input
            type="number"
            className="w-full border border-green-
300 rounded-md p-2 focus:ring-2 focus:ring-
green-400 focus:outline-none"
            value={data.harga}
            onChange={(e) => setData('harga',
e.target.value)}
          />
          {errors.harga && <p className="text-
red-500 text-sm mt-1">{errors.harga}</p>}
        </div>
        {/* Kategori */}
        <div>
          <label className="block text-sm font-
semibold mb-1 text-gray-700">Kategori</label>
          <select
            className="w-full border border-green-
300 rounded-md p-2 focus:ring-2 focus:ring-
green-400 focus:outline-none"
            value={data.category_id}
            onChange={(e) =>
setData('category_id', e.target.value)}
          >
            <option value="">-- Pilih Kategori --
</option>
            {categories.map((cat) => (
              <option key={cat.id} value={cat.id}>
                {cat.kategori}
              </option>
            ))}
          </select>
          {errors.category_id && <p
className="text-red-500 text-sm mt-
1">{errors.category_id}</p>}
        </div>
        {/* Foto */}
        <div>
          <label className="block text-sm font-
semibold mb-1 text-gray-700">Foto
Produk</label>
          <input
            type="file"
            className="w-full border border-green-
300 rounded-md p-2 focus:outline-none"
            onChange={(e) => setData('foto',
e.target.files[0])}
          />
          {errors.foto && <p className="text-red-
500 text-sm mt-1">{errors.foto}</p>}
          {progress && (
            <div className="w-full bg-gray-200
rounded mt-2">
              <div
                className="bg-green-500 text-xs
text-white p-1 text-center rounded"
                style={{ width:
`${progress.percentage}%` }}
              >
                {progress.percentage}%
              </div>
            </div>
          )}
        </div>

        {/* Tombol Simpan */}
        <div className="pt-2">
          <button
            type="submit"
            className="w-full bg-green-600
hover:bg-green-700 text-white font-semibold py-
2 rounded-lg transition duration-300"
          >
            Simpan Produk
          </button>
        </div>
      </form>
    </div>
```

```
    </>
  );
}
```

INDEX.JSX

```
import { useState } from "react";
import { Head, Link, router, usePage } from
'@inertiajs/react';
import {
  IconUser,
  IconTrash,
  IconEdit,
  IconArrowLeft,
  IconEye,
  IconPlus,
  IconShoppingBag
} from '@tabler/icons-react';
import Swal from "sweetalert2";
export default function Index({ Produk }) {
  const { auth } = usePage().props;
  const [isDeleting, setIsDeleting] =
useState(false);
  const [search, setSearch] = useState("");

  const Delete = (id) => {
    Swal.fire({
      title: "Apakah Anda yakin?",
      text: "Anda tidak akan dapat mengembalikan
ini!",
      icon: "warning",
      showCancelButton: true,
      confirmButtonColor: "#16a34a",
      cancelButtonColor: "#d33",
      confirmButtonText: "Ya, hapus saja!",
    }).then((result) => {
      if (result.isConfirmed) {
        setIsDeleting(true);
        router.delete(route("Produk.delete", id), {
          onSuccess: () => {
            setIsDeleting(false);
            Swal.fire("Dihapus!", "Produk telah
dihapus.", "success");
          },
          onError: () => {
            setIsDeleting(false);
            Swal.fire("Gagal!", "Terjadi kesalahan
saat menghapus produk.", "error");
          },
        });
      }
    });
  };
  const filteredProduk = Produk.filter((item) =>
    item.nama_barang.toLowerCase().includes(sea
rch.toLowerCase())
  );
  return (
    <>
      <Head title="Daftar Produk" />
      <div className="max-w-7xl mx-auto mt-10
px-4 sm:px-6 lg:px-8">
        {/* Header */}
        <div className="flex flex-col sm:flex-row
sm:items-center sm:justify-between mb-6 gap-
4">
          <div>
            <h1 className="text-3xl font-bold text-
green-700">Daftar Produk</h1>
            <p className="text-sm text-gray-
500">Kelola produk yang tersedia dalam
sistem.</p>
          </div>

          <div className="flex flex-wrap gap-3">
            <Link
              href={route('category.index')}
              className="inline-flex items-center
text-sm text-green-600 hover:text-green-800
hover:underline transition"
            >
              <IconArrowLeft className="w-5 h-5
mr-1" /> Kembali ke Dashboard
            </Link>
            {auth.roles.includes("admin") && (
              <Link
                href={route("Produk.create")}
                className="bg-green-600 text-white
px-4 py-2 rounded-md flex items-center
hover:bg-green-700 transition"
              >
                <IconPlus className="w-5 h-5 mr-2"
/> Tambah Produk
              </Link>
            )}
          </div>
        </div>
        {/* Search Bar */}
        <div className="mb-6">
          <input
            type="text"
            placeholder="Cari produk berdasarkan
nama..."
            value={search}
            onChange={(e) =>
setSearch(e.target.value)}
            className="w-full px-4 py-2 border
border-green-300 rounded-md shadow-sm
focus:outline-none focus:ring-2 focus:ring-green-
400"
          />
        </div>
        <div className="grid grid-cols-1 sm:grid-
cols-2 lg:grid-cols-3 xl:grid-cols-4 gap-6">
  {filteredProduk.length > 0 ? (
    filteredProduk.map((item, index) => (
      <div
        key={index}
        className="bg-white border border-gray-
100 shadow-md hover:shadow-xl transition
duration-300 rounded-2xl overflow-hidden flex
flex-col"
      >
        {/* Gambar Produk */}
        <img
          src={item.foto}
          alt={item.nama_barang}
```

```
          className="w-full h-48 object-cover"
        />
        {/* Konten */}
        <div className="p-4 flex flex-col flex-
grow justify-between">
          <div>
            {/* Nama Produk */}
            <h2 className="text-lg font-bold text-
gray-800 mb-1">
              {item.nama_barang}
            </h2>
            {/* Harga */}
            <p className="text-green-600 font-
semibold text-base">
              {Number(item.harga).toLocaleString("id
-ID", {
                style: "currency",
                currency: "IDR",
                minimumFractionDigits: 0,
              })}
            </p>
            {/* Status, Stok, dan Ukuran */}
            <div className="mt-2 flex flex-wrap
items-center text-sm gap-2">
              <span className="bg-green-100 text-
green-700 px-2 py-0.5 rounded-full">
                {item.status}
              </span>
              <span className="bg-blue-100 text-
blue-700 px-2 py-0.5 rounded-full">
                Stok: {item.stok}
              </span>
              {/* Ukuran */}
              {item.ukuran && item.ukuran.map((uk,
index) => (
                <span
                  key={index}
                  className="inline-block bg-gray-100
text-gray-700 text-xs px-2 py-1 rounded"
                >
                  Ukuran: {uk}
                </span>
              ))}
            </div>
            {/* Deskripsi Produk */}
            <p className="text-sm text-gray-600
mt-3 line-clamp-3">
              {item.deskripsi ?? "Produk berkualitas
tinggi dengan harga terjangkau. Segera miliki
sekarang juga!"}
            </p>
          </div>
          {/* Aksi */}
          <div className="flex justify-between
items-center mt-4 border-t pt-4 text-sm">
            {auth.roles.includes("admin") && (
              <Link
                href={route("Produk.edit", item.id)}
                className="text-green-600 hover:text-
green-800 flex items-center"
              >
                <IconEdit className="w-4 h-4 mr-1"
/> Edit
              </Link>
            )}
            <Link
              href={route("Produk.show", item.id)}
              className="text-blue-600 hover:text-
blue-800 flex items-center"
            >
              <IconEye className="w-4 h-4 mr-1" />
Detail
            </Link>
            {auth.roles.includes("admin") && (
              <button
                onClick={() => Delete(item.id)}
                disabled={isDeleting}
                className="text-red-500 hover:text-
red-700 flex items-center"
              >
                <IconTrash className="w-4 h-4 mr-1"
/> Hapus
              </button>
            )}
          </div>
        </div>
      </div>
    ))
  ) : (
    <div className="col-span-full text-center
text-gray-500">
      Produk tidak ditemukan.
    </div>
  )}
</div>
      </div>
    </>
  );
}
```

SHOW.JSX

```
import React from 'react';
import { useState } from 'react';
import { Head, Link, usePage } from
'@inertiajs/react';
import CheckoutOrder from
'../Cart/CheckoutOrder';
import { IconArrowLeft } from '@tabler/icons-
react';
import Swal from "sweetalert2";
export default function show({ Produk }) {
  const { flash } = usePage().props;
  const [isLoaded, setIsLoaded] = useState(false);
  // izin push notifikasi
  React.useEffect(() => {
    if (flash?.success) {
      Swal.fire(
        'Berhasil!',
        flash.success,
        'success'
      );
    }

    if (flash?.error) {
      Swal.fire(
        'gagal!',
        flash.error,
        'error'
```

```
    );
  }
 }, [flash]);
 return (
  <>
   <Head title="Detail Produk" />
   <div className="max-w-6xl mx-auto mt-14 px-6 py-10 bg-white rounded-3xl shadow-xl border border-green-200">
    {/* Header */}
    <div className="mb-10 flex items-center justify-between">
     <h1 className="text-3xl font-extrabold text-green-700 tracking-wide">
      Detail Produk
     </h1>
     <Link
      href={route('category.index')}
      className="flex items-center text-sm text-green-600 hover:text-green-800 transition"
     >
      <IconArrowLeft className="w-5 h-5 mr-1" />
      Kembali ke Daftar Produk
     </Link>
    </div>
    {/* Detail Produk */}
    {Produk ? (
     <div className="grid grid-cols-1 md:grid-cols-2 gap-10 items-start">
      {/* Gambar Produk */}
      <div className="group relative overflow-hidden rounded-xl shadow-md">
       <img
        src={`/storage/${Produk.foto.replace("public/", "")}`}
        alt={Produk.nama_barang}
        className="w-full h-[350px] object-cover rounded-xl transform transition-transform duration-500 group-hover:scale-105"
       />
      </div>
      {/* Deskripsi Produk */}
      <div className="space-y-5">
       {/* Nama Barang */}
       <h2 className="text-2xl font-bold text-gray-800">
        {Produk.nama_barang}
       </h2>
       {/* Harga */}
       <p className="text-3xl font-extrabold text-green-600">
        {Number(Produk.harga).toLocaleString("id-ID", {
         style: "currency",
         currency: "IDR",
         minimumFractionDigits: 0,
        })}
       </p>
       {/* Status dan Stok */}
       <div className="flex items-center gap-3 flex-wrap">
        <span className="inline-block bg-green-100 text-green-700 text-xs font-semibold px-3 py-1 rounded-full">
         {Produk.status}
        </span>
        <span className="inline-block bg-blue-100 text-blue-700 text-xs font-semibold px-3 py-1 rounded-full">
         Stok: {Produk.stok}
        </span>
       </div>
       {/* Ukuran */}
       <div className="flex flex-wrap items-center gap-2 text-sm mt-2">
        <span className="font-semibold text-gray-600">Ukuran:</span>
        {Array.isArray(Produk.ukuran)
         ? Produk.ukuran.map((uk, index) => (
          <span
           key={index}
           className="bg-gray-100 text-gray-700 text-xs px-2 py-1 rounded"
          >
           {uk}
          </span>
         ))
         : Produk.ukuran?.split(',').map((uk, index) => (
          <span
           key={index}
           className="bg-gray-100 text-gray-700 text-xs px-2 py-1 rounded"
          >
           {uk.trim()}
          </span>
         ))}
       </div>
       {/* Deskripsi */}
       <p className="text-sm text-gray-700 mt-2 leading-relaxed">
        {Produk.deskripsi ??
         "Produk unggulan kami yang dirancang dengan bahan berkualitas, nyaman digunakan, dan cocok untuk kebutuhan sehari-hari Anda."}
       </p>
{Produk.dekripsi.split('\n').map((line, index) => (
  <p key={index} className="text-sm text-gray-700 mt-2 leading-relaxed break-words">
    {line}
  </p>
))}
       {/* Komponen Pemesanan */}
       <div className="pt-4">
        <CheckoutOrder produk={Produk} />
       </div>
      </div>
     </div>
    ) : (
     <p className="text-red-500 text-center mt-8">
      Produk tidak ditemukan.
     </p>
    )}
```

```
    </div>
   </>
  );
}
```

UPDATE.JSX

```
import React from 'react';
import { Head, useForm, Link } from
'@inertiajs/react';

export default function update({ Produk,
categories }) {
// Gunakan useForm dan inisialisasi ukuran
sebagai array
const { data, setData, post, progress, errors } =
useForm({
  nama_barang: Produk.nama_barang || '',
  dekripsi: Produk.dekripsi || '',
  ukuran: Produk.ukuran || [''], // ✅ array default
  harga: Produk.harga || '',
  stok: Produk.stok || '',
  foto: null,
  category_id: Produk.category_id || '',
});

// Handler untuk menambah field ukuran
const addUkuran = () => {
  setData('ukuran', [...data.ukuran, '']);
};

// Handler untuk hapus field ukuran
const removeUkuran = (index) => {
  const updated = [...data.ukuran];
  updated.splice(index, 1);
  setData('ukuran', updated);
};

// Handler untuk ubah nilai ukuran tertentu
const handleUkuranChange = (value, index) => {
  const updated = [...data.ukuran];
  updated[index] = value;
  setData('ukuran', updated);
};

  const handleSubmit = (e) => {
    e.preventDefault();
    post(route('Produk.update', Produk.id), {
      preserveScroll: true,
    });
  };

  return (
    <>
      <Head title="Edit Produk" />

      <div className="max-w-2xl mx-auto mt-12
p-8 bg-white shadow-xl rounded-2xl border
border-gray-100">
        {/* Tombol Kembali */}
        <div className="mb-6 flex justify-between
items-center">
          <h1 className="text-3xl font-extrabold
text-gray-800">Edit Produk</h1>
          <Link
            href={route('dashboard')}
            className="text-blue-600 hover:text-
blue-800 font-semibold transition duration-200"
          >
            ← Kembali ke Dashboard
          </Link>
        </div>

        <form
          onSubmit={handleSubmit}
          encType="multipart/form-data"
          className="space-y-6"
        >
          {/* Nama Barang */}
          <div>
            <label className="block font-semibold
mb-1">Nama Barang</label>
            <input
              type="text"
              className="w-full border border-gray-
300 rounded-lg p-3 focus:outline-none
focus:ring-2 focus:ring-blue-400"
              value={data.nama_barang}
              onChange={(e) =>
setData('nama_barang', e.target.value)}
            />
            {errors.nama_barang && <p
className="text-red-500 text-
sm">{errors.nama_barang}</p>}
          </div>
          <div>
            <label className="block font-semibold
mb-1">Deksripsi Barang</label>
            <textarea
              type="text"
              className="w-full h-56  border border-
gray-300 rounded-lg p-3 focus:outline-none
focus:ring-2 focus:ring-blue-400"
              value={data.dekripsi}
              onChange={(e) => setData('dekripsi',
e.target.value)}
            />
            {errors.dekripsi && <p className="text-
red-500 text-sm">{errors.dekripsi}</p>}
          </div>
          <div>
            <label className="block font-semibold
mb-1">Stok Barang</label>
            <input
              type="number"
              className="w-full border border-gray-
300 rounded-lg p-3 focus:outline-none
focus:ring-2 focus:ring-blue-400"
              value={data.stok}
              onChange={(e) => setData('stok',
e.target.value)}
            />
            {errors.stok && <p className="text-red-
500 text-sm">{errors.stok}</p>}
          </div>

          {/* Kategori */}
          <div>
```

```
          <label className="block font-semibold
mb-1">Kategori</label>
          <select
            className="w-full border border-gray-
300 rounded-lg p-3 focus:outline-none
focus:ring-2 focus:ring-blue-400"
            value={data.category_id}
            onChange={(e) =>
setData('category_id', e.target.value)}
          >
            <option value="">Pilih
Kategori</option>
            {categories.map((category) => (
              <option key={category.id}
value={category.id}>
                {category.kategori}
              </option>
            ))}
          </select>
          {errors.category_id && (
            <p className="text-red-500 text-
sm">{errors.category_id}</p>
          )}
        </div>
{/* Ukuran */}
<div>
  <label className="block font-semibold mb-
1">Ukuran</label>
  {data.ukuran.map((uk, index) => (
    <div key={index} className="flex items-
center mb-2">
      <input
        type="text"
        className="w-full border border-gray-300
rounded-lg p-3 mr-2 focus:outline-none
focus:ring-2 focus:ring-blue-400"
        value={uk}
        onChange={(e) =>
handleUkuranChange(e.target.value, index)}
      />
      {data.ukuran.length > 1 && (
        <button
          type="button"
          onClick={() => removeUkuran(index)}
          className="text-red-600 hover:text-red-
800 font-bold"
        >
          Hapus
        </button>
      )}
    </div>
  ))}
  <button
    type="button"
    onClick={addUkuran}
    className="mt-2 text-sm text-blue-600
hover:text-blue-800"
  >
    + Tambah Ukuran
  </button>
  {errors.ukuran && <p className="text-red-
500 text-sm">{errors.ukuran}</p>}
</div>
        {/* Harga */}
        <div>
          <label className="block font-semibold
mb-1">Harga</label>
          <input
            type="number"
            className="w-full border border-gray-
300 rounded-lg p-3 focus:outline-none
focus:ring-2 focus:ring-blue-400"
            value={data.harga}
            onChange={(e) => setData('harga',
e.target.value)}
          />
          {errors.harga && <p className="text-
red-500 text-sm">{errors.harga}</p>}
        </div>
        {/* Foto Baru */}
        <div>
          <label className="block font-semibold
mb-1">Foto Baru (Opsional)</label>
          <input
            type="file"
            className="w-full border border-gray-
300 rounded-lg p-3"
            onChange={(e) => setData('foto',
e.target.files[0])}
          />
          {errors.foto && <p className="text-red-
500 text-sm">{errors.foto}</p>}
          {progress && (
            <div className="w-full bg-gray-200
rounded mt-2">
              <div
                className="bg-blue-500 text-white
text-xs p-1 text-center rounded"
                style={{ width:
`${progress.percentage}%` }}
              >
                {progress.percentage}%
              </div>
            </div>
          )}
        </div>
        {/* Foto Saat Ini */}
        {Produk.foto && (
          <div>
            <label className="block font-semibold
mb-1">Foto Saat Ini:</label>
            <img
              src={`/storage/${Produk.foto.replace('p
ublic/', '')}`}
              alt="Foto Produk"
              className="w-48 h-auto mt-2
rounded-lg shadow-lg"
            />
          </div>
        )}
        {/* Tombol Update */}
        <div className="pt-4">
          <button
            type="submit"
            className="w-full bg-green-600
hover:bg-green-700 text-white font-bold py-3
rounded-lg transition duration-300"
          >
```

```
            Update Produk
          </button>
        </div>
      </form>
    </div>
  </>
 );
}
```

- **PROFILE**

DASHBOARD.JSX

```
import AuthenticatedLayout from
'@/Layouts/AuthenticatedLayout';
import { FlipWords } from
"@/Components/FlipWords";
import { Player } from '@lottiefiles/react-lottie-
player';
export default function Dashboard({ auth }) {
  const words = ["Menyewa", "Menjelajah",
"Menampilkan"];
  return (
    <AuthenticatedLayout user={auth.user}>
      <div className="min-h-screen bg-green-50
py-16 px-4">
        <div className="max-w-5xl mx-auto text-
center">
          {/* Lottie Animation */}
          <Player
            autoplay
            loop
            src="/image/anim.json" // Ganti jika
kamu punya animasi lain
            style={{ height: '250px', width: '250px',
margin: '0 auto' }}
          />
          <h1 className="text-3xl md:text-5xl font-
bold text-green-700 mb-4 mt-6">
            Selamat Datang di Dashboard
          </h1>
          <p className="text-xl text-gray-700">
            Di halaman ini Anda bisa <FlipWords
words={words} /> berbagai <span
className="font-semibold">baju adat
Bugis</span>.
          </p>
        </div>
        {/* Tentang Baju Adat */}
        <div className="mt-16 max-w-4xl mx-
auto bg-white shadow-lg rounded-2xl p-8 border
border-green-200">
          <h2 className="text-2xl font-semibold
text-green-700 mb-4">Apa Itu Baju Adat?</h2>
          <p className="text-gray-700 leading-
relaxed">
            Baju adat adalah pakaian tradisional khas
suatu daerah di Indonesia yang mencerminkan
budaya dan kearifan lokal.
            Dengan menyewa baju adat, Anda ikut
melestarikan kekayaan budaya bangsa dan tampil
memukau dalam setiap acara.
          </p>
        </div>
      </div>
    </AuthenticatedLayout>
  );
}
```

PENDAFTARAN.JSX

```
import { useEffect } from "react";
export default function Pendaftaran() {
  useEffect(() => {
    window.TelegramLoginWidget = {
      dataOnauth: (user) => {
        console.log("User Data:", user);
        fetch("/telegram-login", {
          method: "POST",
          headers: {
            "Content-Type":
"application/json",
          },
          body: JSON.stringify(user),
        })
          .then((response) => response.json())
          .then((data) => {
            if (data.success) {
              window.location.href = "/"; //
Redirect jika login sukses
            } else {
              alert("Login gagal. Silakan
coba lagi.");
            }
          });
      },
    };
  }, []);
  return (
    <div className="flex flex-col items-center
justify-center min-h-screen">
      <h2 className="text-2xl font-semibold
mb-4">Login dengan Telegram</h2>
      <script
        async
        src="https://telegram.org/js/telegram-
widget.js?22"
        data-telegram-login="Palkytbot"
        data-size="large"
        data-auth-
url="https://penjualan.test/telegram-login"
        data-request-access="write"
      ></script>
    </div>
  );
}
```

PESANANPAGE.JSX

```
import React, { useState, useEffect } from 'react';
import { usePage } from '@inertiajs/react';
import axios from 'axios';
const PesananPage = () => {
  const { cart } = usePage().props;
  const baseTotal = cart?.total || 0;
  const [peminjaman, setPeminjaman] =
useState('');
  const [pengembalian, setPengembalian] =
useState('');
  const [durasi, setDurasi] = useState(1);
  const [total, setTotal] = useState(baseTotal);
```

```
  const [pesananId, setPesananId] =
useState(null);
  const [ktp, setKtp] = useState(null);
  const [preview, setPreview] = useState(null);
  useEffect(() => {
    if (peminjaman && pengembalian) {
      const tglPinjam = new Date(peminjaman);
      const tglKembali = new Date(pengembalian);
      const selisihWaktu = tglKembali - tglPinjam;
      const selisihHari = Math.max(1,
Math.ceil(selisihWaktu / (1000 * 60 * 60 * 24)));
      setDurasi(selisihHari);
      setTotal(baseTotal * selisihHari);
    } else {
      setDurasi(1);
      setTotal(baseTotal);
    }
  }, [peminjaman, pengembalian, baseTotal]);
  const handleFileChange = (e) => {
    const file = e.target.files[0];
    if (file) {
      setKtp(file);
      setPreview(URL.createObjectURL(file));
    }
  };
  const validateForm = () => {
    if (!peminjaman || !pengembalian || !ktp || total
<= 0) {
      alert('Mohon lengkapi tanggal & upload KTP
dengan total yang valid!');
      return false;
    }
    return true;
  };
  const handleBayar = async () => {
    if (!validateForm()) return;
    const formData = new FormData();
    formData.append('total', total);
    formData.append('peminjaman', peminjaman);
    formData.append('pengembalian',
pengembalian);
    formData.append('ktp', ktp);
    try {
      const response = await axios.post('/pesanan',
formData, {
        headers: { 'Content-Type': 'multipart/form-
data' },
      });
      const snapToken = response.data.snap_token;
      const pesananIdBaru =
response.data.pesanan_id;
      setPesananId(pesananIdBaru);

      window.snap.pay(snapToken, {
        onSuccess: async () => {
          alert('Pembayaran berhasil!');
          await
axios.post(`/pesanan/${pesananIdBaru}/status`);
          await axios.post('/cart/clear');
          window.location.href = '/Peminjaman';
        },
        onPending: () => alert('Menunggu
pembayaran!'),
        onError: () => {
          alert('Pembayaran gagal!');
          handleDeletePesanan(pesananIdBaru);
        },
        onClose: () => {
          alert('Kamu menutup popup sebelum
selesai');
          handleDeletePesanan(pesananIdBaru);
        },
      });
    } catch (error) {
      if (error.response?.status === 422) {
        alert('Pastikan semua data valid & total
minimal Rp1.000');
      }
    }
  };
  const handleAjukanCash = async () => {
    if (!validateForm()) return;

    const formData = new FormData();
    formData.append('total', total);
    formData.append('peminjaman', peminjaman);
    formData.append('pengembalian',
pengembalian);
    formData.append('ktp', ktp);
    try {
      const res = await axios.post('/pesanan/cash',
formData, {
        headers: { 'Content-Type': 'multipart/form-
data' },
      });
      alert('Pengajuan pesanan cash berhasil!
Menunggu konfirmasi admin.');
      await axios.post('/cart/clear');
      window.location.href = '/Peminjaman';
    } catch (err) {
      alert('Gagal mengajukan pesanan cash.
Pastikan data sudah benar.');
    }
  };
  const handleDeletePesanan = async (id) => {
    try {
      await axios.delete(`/pesanan/${id}/delete`);
      alert('Pesanan gagal dibayar dan telah
dihapus.');
    } catch {
      alert('Gagal menghapus pesanan setelah error
pembayaran.');
    }
  };
  return (
    <div className="p-6 mt-4 bg-gray-50 border
border-gray-200 rounded-xl shadow-sm">
      <h2 className="text-xl font-semibold mb-4
text-gray-800">Pesan Produk Ini</h2>
      <div className="space-y-4">
        <div>
          <label className="block text-sm font-
medium text-gray-700 mb-1">Tanggal
Peminjaman:</label>
          <input
            type="date"
            value={peminjaman}
```

```
          onChange={(e) =>
setPeminjaman(e.target.value)}
          className="w-full p-2 border border-
gray-300 rounded"
         />
        </div>
        <div>
         <label className="block text-sm font-
medium text-gray-700 mb-1">Tanggal
Pengembalian:</label>
         <input
          type="date"
          value={pengembalian}
          onChange={(e) =>
setPengembalian(e.target.value)}
          className="w-full p-2 border border-
gray-300 rounded"
         />
        </div>
        <div>
         <label className="block text-sm font-
medium text-gray-700 mb-1">Upload Foto
KTP:</label>
         <input
          type="file"
          accept="image/*,.pdf"
          onChange={handleFileChange}
          className="w-72 lg:w-full p-2 border
border-gray-300 rounded"
         />
         {preview && (
          <img
           src={preview}
           alt="Preview KTP"
           className="mt-2 w-96 lg:max-w-lg
rounded shadow border"
          />
         )}
        </div>
        <div className="text-sm text-gray-700
space-y-1">
         <p><strong>Durasi Peminjaman:</strong>
{durasi} hari</p>
         <p><strong>Total Harga:</strong> Rp
{total.toLocaleString('id-ID')}</p>
         <p className="text-xs text-red-500
italic">* Orderan yg melebihi Rp2.000.000 harus
konsultasi ke penjual</p>
        </div>
        <div className="flex flex-col lg:flex-row
gap-3 mt-4">
         <button
          onClick={handleBayar}
          className="flex-1 bg-blue-600
hover:bg-blue-700 text-white px-4 py-2 rounded"
         >
          Bayar Sekarang (Online)
         </button>
         <button
          onClick={handleAjukanCash}
          className="flex-1 bg-green-600
hover:bg-green-700 text-white px-4 py-2
rounded"
         >
          Ajukan Pesanan Cash
         </button>
        </div>
       </div>
      </div>
    );
};
export default PesananPage;
// langsung bayar
// import React, { useState, useEffect } from
'react';
// import axios from 'axios';
// const PesananPage = ({ harga }) => {
//   const [peminjaman, setPeminjaman] =
useState('');
//   const [pengembalian, setPengembalian] =
useState('');
//   const [jumlahHari, setJumlahHari] =
useState(0);
//   const [total, setTotal] = useState(0);
//   const [pesananId, setPesananId] =
useState(null);
//   const [ktp, setKtp] = useState(null); // input
file
//   const [preview, setPreview] = useState(null);
// untuk preview gambar
//   useEffect(() => {
//     if (peminjaman && pengembalian) {
//       const date1 = new Date(peminjaman);
//       const date2 = new Date(pengembalian);
//       const diffTime = date2.getTime() -
date1.getTime();
//       const diffDays = Math.ceil(diffTime /
(1000 * 60 * 60 * 24));

//       if (diffDays > 0) {
//         setJumlahHari(diffDays);
//         setTotal(diffDays * harga);
//       } else {
//         setJumlahHari(0);
//         setTotal(0);
//       }
//     }
//   }, [peminjaman, pengembalian, harga]);
//   const handleBayar = async () => {
//     if (!ktp) {
//       alert('Silakan upload foto KTP terlebih
dahulu!');
//       return;
//     }
//     const formData = new FormData();
//     formData.append('total', total);
//     formData.append('peminjaman',
peminjaman);
//     formData.append('pengembalian',
pengembalian);
//     formData.append('ktp', ktp);
//     try {
//       const response = await
axios.post('/pesanan', formData, {
//         headers: { 'Content-Type':
'multipart/form-data' },
//       });
```

```
//       const snapToken =
response.data.snap_token;
//       const pesananIdBaru =
response.data.pesanan_id;
//       setPesananId(pesananIdBaru);
//       window.snap.pay(snapToken, {
//         onSuccess: async function () {
//           alert('Pembayaran berhasil!');
//           try {
//             await
axios.post(`/pesanan/${pesananIdBaru}/status`);
//             alert('Status pesanan berhasil diupdate
ke "peminjaman"');
//             window.location.href = '/Peminjaman';
//           } catch {
//             alert('Pembayaran berhasil, tapi gagal
update status.');
//           }
//         },
//         onPending: function () {
//           alert('Menunggu pembayaran!');
//         },
//         onError: function () {
//           alert('Pembayaran gagal!');
//           handleDeletePesanan(pesananIdBaru);
//         },
//         onClose: function () {
//           alert('Kamu menutup popup sebelum
selesai');
//           handleDeletePesanan(pesananIdBaru);
//         },
//       });
//     } catch (error) {
//       if (error.response?.status === 422) {
//         alert('Pastikan semua data valid & total
minimal Rp1.000');
//       }
//     }
//   };

//   const handleDeletePesanan = async (id) => {
//     try {
//       await axios.delete(`/pesanan/${id}/delete`);
//       alert('Pesanan gagal dibayar dan telah
dihapus.');
//     } catch {
//       alert('Gagal menghapus pesanan setelah
error pembayaran.');
//     }
//   };
//   const handleFileChange = (e) => {
//     const file = e.target.files[0];
//     if (file) {
//       setKtp(file);
//       setPreview(URL.createObjectURL(file));
//     }
//   };
//   return (
//     <div className="p-6 mt-4 bg-gray-50
border border-gray-200 rounded-xl shadow-sm">
//       <h2 className="text-xl font-semibold mb-
4 text-gray-800">Pesan Produk Ini</h2>
//       <div className="space-y-4">
//         {/* Tanggal Peminjaman */}
//         <div>
//           <label className="block text-sm font-
medium text-gray-700 mb-1">
//             Tanggal Peminjaman:
//           </label>
//           <input
//             type="date"
//             value={peminjaman}
//             onChange={(e) =>
setPeminjaman(e.target.value)}
//             className="w-full p-2 border border-
gray-300 rounded focus:ring-2 focus:ring-green-
500 focus:outline-none"
//           />
//         </div>

//         {/* Tanggal Pengembalian */}
//         <div>
//           <label className="block text-sm font-
medium text-gray-700 mb-1">
//             Tanggal Pengembalian:
//           </label>
//           <input
//             type="date"
//             value={pengembalian}
//             onChange={(e) =>
setPengembalian(e.target.value)}
//             className="w-full p-2 border border-
gray-300 rounded focus:ring-2 focus:ring-green-
500 focus:outline-none"
//           />
//         </div>
// {/* Upload KTP */}
// <div>
//   <label className="block text-sm font-
medium text-gray-700 mb-1">
//     Upload Foto KTP:
//   </label>
//   <input
//     type="file"
//     accept="image/*,.pdf"
//     onChange={handleFileChange}
//     className="w-full p-2 border border-gray-
300 rounded"
//   />
//   {preview && (
//     <img
//       src={preview}
//       alt="Preview KTP"
//       className="mt-2 max-w-sm rounded
shadow border"
//     />
//   )}
//   <p className="text-sm text-gray-600 mt-2">
//     <strong>Mengapa perlu mengunggah
KTP?</strong><br />
//     Dokumen KTP diperlukan sebagai bentuk
<strong>jaminan keamanan</strong> dan
<strong>verifikasi identitas</strong> peminjam.
//     Ini untuk memastikan bahwa transaksi
dilakukan oleh pihak yang sah serta untuk
menghindari penyalahgunaan layanan.
//     Data ini hanya digunakan untuk keperluan
verifikasi dan akan dijaga kerahasiaannya.
```

```
//   </p>
// </div>
//         {/* Ringkasan dan tombol bayar */}
//         {jumlahHari > 0 && (
//           <div className="text-sm text-gray-700
space-y-1">
//             <p><strong>Lama
Peminjaman:</strong> {jumlahHari} hari</p>
//             <p><strong>Total Harga:</strong> Rp
{total.toLocaleString('id-ID')}</p>
//             <p className="text-xs text-red-500
italic">* Orderan yg melebihi Rp2.000.000 harus
konsultasi ke penjual</p>
//           </div>
//         )}

//         <button
//           onClick={handleBayar}
//           className={`w-full text-white px-4 py-
2 rounded transition ${
//             jumlahHari > 0 ? 'bg-blue-600
hover:bg-blue-700' : 'bg-gray-400 cursor-not-
allowed'
//           }`}
//           disabled={jumlahHari <= 0}
//         >
//           Bayar Sekarang
//         </button>
//       </div>
//     </div>
//   );
// };
// export default PesananPage;
```

WELCOME.JSX

```
import React from "react";
import  PublicLayout  from
"@/Layouts/PublicLayout";
import Category from "./category/index"
import { router, usePage } from
"@inertiajs/react";
import { Player } from '@lottiefiles/react-lottie-
player';
import { BarChart, Bar, XAxis, YAxis, Tooltip,
ResponsiveContainer, CartesianGrid, Legend }
from 'recharts';
export default function LandingPage({category})
{
const { admin_contact, topProduk } =
usePage().props;
  const whatsappAdminNumber = admin_contact
? `+62${admin_contact.phone}` :
'+6281234567890';
  return (
    <PublicLayout>
       <div className="bg-green-50 min-h-
screen">
      {/* Hero Section */}
<header className="bg-green-700 text-white
py-24 px-6 text-center">
  <div className="flex flex-col items-center
justify-center">
    <Player
      autoplay
      loop
      src="/image/animasi.json" // Ganti dengan
URL/JSON animasimu
      style={{ height: '250px', width: '250px' }}
    />
    <h1 className="text-4xl md:text-5xl font-
bold mb-4">Sewa Baju Adat Bugis</h1>
    <p className="text-lg md:text-xl pb-
5">Tampil Anggun dan Berbudaya dalam
Balutan Baju Adat</p>
    <a href="/login" className="mt-6 px-6 py-3
bg-white text-green-700 font-semibold rounded-
full shadow hover:bg-green-100 transition">
      Pesan Sekarang
    </a>
  </div>
</header>
      {/* About Section */}
      <section className="py-16 px-6 max-w-5xl
mx-auto text-center">
        <h2 className="text-3xl font-bold text-
green-700 mb-6">Kenapa Memilih Kami?</h2>
        <div className="grid md:grid-cols-3 gap-
8">
          <div className="bg-white p-6 rounded-lg
shadow">
            <h3 className="text-xl font-semibold
mb-2 text-green-700">Koleksi Lengkap</h3>
            <p>Kami menyediakan berbagai jenis
baju adat dari seluruh Indonesia, siap untuk
berbagai acara.</p>
          </div>
          <div className="bg-white p-6 rounded-lg
shadow">
            <h3 className="text-xl font-semibold
mb-2 text-green-700">Harga Terjangkau</h3>
            <p>Paket sewa yang hemat dan fleksibel
sesuai kebutuhan Anda.</p>
          </div>
          <div className="bg-white p-6 rounded-lg
shadow">
            <h3 className="text-xl font-semibold
mb-2 text-green-700">Layanan Profesional</h3>
            <p>Tim kami siap membantu Anda
memilih dan menyesuaikan baju adat yang
pas.</p>
          </div>
        </div>
      </section>
<section className="py-16 px-6 max-w-5xl mx-
auto text-center">
  <h2 className="text-3xl font-bold text-green-
700 mb-8">☑ Statistik Produk Terlaris</h2>
  <div className="bg-white rounded-xl shadow
p-6 mb-12">
    <ResponsiveContainer width="100%"
height={350}>
      <BarChart data={topProduk}>
        <CartesianGrid strokeDasharray="3 3" />
        <XAxis dataKey="nama_barang" tick={{
fontSize: 12 }} />
        <YAxis />
        <Tooltip />
```

```
        <Legend />
        <Bar dataKey="total" name="Jumlah
Peminjaman" fill="#16a34a" />
      </BarChart>
    </ResponsiveContainer>
  </div>
  <h2 className="text-3xl font-bold text-green-
700 mb-8">🔥 Produk Terlaris</h2>
  <div className="grid md:grid-cols-3 gap-6">
    {topProduk?.map((produk) => (
     <div key={produk.id} className="bg-white
rounded-xl shadow hover:shadow-md transition
p-4">
       <img
         src={`/storage/${produk.foto.replace("publ
ic/", "")}`}
         alt={produk.nama_barang}
         className="w-full h-48 object-cover
rounded-lg mb-3"
       />
       <h3 className="text-lg font-semibold text-
gray-800">{produk.nama_barang}</h3>
       <p className="text-sm text-gray-600">Rp
{produk.harga?.toLocaleString('id-ID')}</p>
       <p className="text-xs text-gray-400 mt-1
italic">
         Telah dipinjam {produk.total}x
       </p>
     </div>
    ))}
  </div>
</section>
     <Category id="produk" category={category}
/>
     <section id="tentang" className="py-16 px-
6 bg-white text-center max-w-5xl mx-auto">
     <h2 className="text-3xl font-bold text-
green-700 mb-6">Tentang Kami</h2>
     <p className="text-lg leading-relaxed text-
gray-700">
       Kami adalah penyedia layanan penyewaan
baju adat terpercaya yang telah melayani
berbagai kebutuhan acara formal maupun non-
formal, seperti pernikahan, upacara adat,
photoshoot, dan kegiatan sekolah.
       Dengan koleksi lengkap dan layanan yang
profesional, kami berkomitmen untuk menjaga
dan melestarikan kekayaan budaya Indonesia
melalui balutan busana tradisional yang anggun
dan autentik.
     </p>
    </section>
     {/* Contact Section */}
     <section  id="kontak" className="py-16 px-
6 max-w-3xl mx-auto text-center">
       <h2 className="text-3xl font-bold text-
green-700 mb-4">Hubungi Kami</h2>
       <p className="mb-6">Ingin informasi
lebih lanjut atau konsultasi? Hubungi kami
melalui WhatsApp atau kunjungi toko kami
langsung.</p>
       <a                href={`https://wa.me/${what
sappAdminNumber}?text=Masukkan%20nama%
20dan%20email%20Anda.`}
                target="_blank"
                rel="noopener noreferrer"
                className="px-6 py-3 bg-green-
700 text-white rounded-full font-semibold
hover:bg-green-800 transition">
         Hubungi via WhatsApp
       </a>
     </section>
     {/* Footer */}
     <footer className="bg-green-700 text-white
py-6 text-center">
       <p>&copy; {new Date().getFullYear()}
Sewa Baju Adat Bugis. All rights reserved.</p>
     </footer>
    </div>
    </PublicLayout>
  );
}
```

- **ROUTES**

AUTH.PHP

```
<?php
use
App\Http\Controllers\Auth\AuthenticatedSession
Controller;
use
App\Http\Controllers\Auth\ConfirmablePasswor
dController;
use
App\Http\Controllers\Auth\EmailVerificationNot
ificationController;
use
App\Http\Controllers\Auth\EmailVerificationPro
mptController;
use
App\Http\Controllers\Auth\NewPasswordContro
ller;
use
App\Http\Controllers\Auth\PasswordController;
use
App\Http\Controllers\Auth\PasswordResetLinkC
ontroller;
use
App\Http\Controllers\Auth\RegisteredUserContr
oller;
use
App\Http\Controllers\Auth\VerifyEmailControlle
r;
use Illuminate\Support\Facades\Route;

Route::middleware('guest')->group(function () {
    Route::get('register',
[RegisteredUserController::class, 'create'])
                ->name('register');
    Route::post('register',
[RegisteredUserController::class, 'store']);
    Route::get('login',
[AuthenticatedSessionController::class, 'create'])
                ->name('login');
    Route::post('login',
[AuthenticatedSessionController::class, 'store']);
    Route::get('forgot-password',
[PasswordResetLinkController::class, 'create'])
                ->name('password.request');
```

```
    Route::post('forgot-password',
[PasswordResetLinkController::class, 'store'])
                ->name('password.email');
    Route::get('reset-password/{token}',
[NewPasswordController::class, 'create'])
                ->name('password.reset');
    Route::post('reset-password',
[NewPasswordController::class, 'store'])
                ->name('password.store');
});
Route::middleware('auth')->group(function () {
    Route::get('verify-email',
EmailVerificationPromptController::class)
                ->name('verification.notice');
    Route::get('verify-email/{id}/{hash}',
VerifyEmailController::class)
                ->middleware(['signed', 'throttle:6,1'])
                ->name('verification.verify');
    Route::post('email/verification-notification',
[EmailVerificationNotificationController::class,
'store'])
                ->middleware('throttle:6,1')
                ->name('verification.send');
    Route::get('confirm-password',
[ConfirmablePasswordController::class, 'show'])
                ->name('password.confirm');
    Route::post('confirm-password',
[ConfirmablePasswordController::class, 'store']);
    Route::put('password',
[PasswordController::class, 'update'])-
>name('password.update');
    Route::post('logout',
[AuthenticatedSessionController::class,
'destroy'])
                ->name('logout');
});
```

CONSOLE.PHP

```
<?php
use Illuminate\Foundation\Inspiring;
use Illuminate\Support\Facades\Artisan;
Artisan::command('inspire', function () {
    $this->comment(Inspiring::quote());
})->purpose('Display an inspiring quote')-
>hourly();
```

WEB.PHP

```
<?php
use App\Http\Controllers\AdminController;
use App\Http\Controllers\CategoryController;
use App\Http\Controllers\ProfileController;
use App\Http\Controllers\ProdukController;
use App\Http\Controllers\PesananController;
use App\Http\Controllers\PeminjamanController;
use App\Http\Controllers\CartController;
use Illuminate\Foundation\Application;
use Illuminate\Support\Facades\Route;
use App\Models\User;
use App\Models\Pesanan;
use Inertia\Inertia;
use Illuminate\Support\Facades\Auth;
use App\Models\Cart;
Route::get('/pengembalian', function () {
    $today = now()->toDateString(); // Format Y-
m-d
    $pesananHariIni =
Pesanan::whereDate('pengembalian', $today)-
>get();
    if ($pesananHariIni->isEmpty()) {
        return 'Tidak ada pesanan yang
pengembaliannya hari ini.';
    }
    $pesanan = $pesananHariIni->first(); // Ambil
satu untuk uji coba
    $user = $pesanan->user;
    if (!$user || !$user->no) {
        return 'User tidak ditemukan atau nomor
WA kosong.';
    }
    $pesan = "📅 Tanggal peminjaman Anda telah
berakhir.\nMohon segera kembalikan produk
Anda atau akan dikenakan denda
Rp30.000/hari.";
    $curl = curl_init();
    curl_setopt_array($curl, [
        CURLOPT_URL =>
'https://api.fonnte.com/send',
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_CUSTOMREQUEST =>
'POST',
        CURLOPT_POSTFIELDS => [
            'target' => $user->no,
            'message' => $pesan,
        ],
        CURLOPT_HTTPHEADER => [
            'Authorization:
XfPe249h65VgJUmCjPPj', // Ganti token sesuai
milikmu
        ],
    ]);

    $response = curl_exec($curl);
    $error = curl_error($curl);
    curl_close($curl);
    return $response;
    // return dd([
    //     'nomor' => $user->no,
    //     'nama' => $user->name ?? 'Tanpa Nama',
    //     'response' => $response,
    //     'error' => $error,
    //     'tanggal_hari_ini' => $today,
    //     'pengembalian' => $pesanan-
>pengembalian,
    // ]);
});
// Route::get('/kirim-tes-fonnte', function () {
//     $curl = curl_init();
//     curl_setopt_array($curl, array(
//       CURLOPT_URL =>
'https://api.fonnte.com/send',
//       CURLOPT_RETURNTRANSFER =>
true,
//       CURLOPT_CUSTOMREQUEST =>
'POST',
//       CURLOPT_POSTFIELDS => array(
```

```
//         'target' => '6287782283819', // Nomor
tujuan
//         'message' => 'Halo, ini pesan dari Laravel
via Fonnte!',
//       ),
//       CURLOPT_HTTPHEADER => array(
//         'Authorization:XfPe249h65VgJUmCjPPj',
//       ),
//     ));
//     $response = curl_exec($curl);
//     curl_close($curl);
//     return $response;
// });
Route::get('/', function () {
    return Inertia::render('Welcome', [
        'canLogin' => Route::has('login'),
        'canRegister' => Route::has('register'),
        'laravelVersion' => Application::VERSION,
        'phpVersion' => PHP_VERSION,
    ]);
});
Route::get('/dashboard', function () {
    return Inertia::render('Dashboard');
})->middleware(['auth', 'verified'])-
>name('dashboard');
Route::middleware('auth')->group(function () {
    Route::get('/profile', [ProfileController::class,
'edit'])->name('profile.edit');
    Route::patch('/profile',
[ProfileController::class, 'update'])-
>name('profile.update');
    Route::delete('/profile',
[ProfileController::class, 'destroy'])-
>name('profile.destroy');
    Route::post('/pesanan',
[PesananController::class, 'buatPesanan']);
    Route::post('/pesanan/{id}/status',
[PesananController::class, 'updateStatus']);
    Route::post('/pesanan/cash',
[PesananController::class, 'buatPesananCash'])-
>name('pesanan.cash');
    Route::post('/pesanan/{id}/konfirmasi-cash',
[PesananController::class, 'konfirmasiCash'])-
>name('pesanan.konfirmasiCash');
    Route::get('/cart', [CartController::class,
'index'])->name('cart.index');
    Route::post('/cart', [CartController::class,
'store'])->name('cart.store');
    Route::put('/cart/{id}', [CartController::class,
'update'])->name('cart.update');
    Route::delete('/cart/{id}',
[CartController::class, 'destroy'])-
>name('cart.destroy');
});

Route::get('/category',
[CategoryController::class, 'index'])-
>name('category.index');
Route::get('/category/create',
[CategoryController::class, 'create'])-
>name('category.create');
Route::post('/category',
[CategoryController::class, 'store'])-
>name('category.store');
Route::get('/category/{category}',
[CategoryController::class, 'show'])-
>name('Produk.index');
Route::get('/category/{category}/edit',
[CategoryController::class, 'edit'])-
>name('category.edit');
Route::post('/category/{category}',
[CategoryController::class, 'update'])-
>name('category.update');
Route::delete('/category/{category}',
[CategoryController::class, 'destroy'])-
>name('category.destroy');
Route::get('/Produk/create',
[ProdukController::class, 'create'])-
>name('Produk.create');
Route::post('/Produk/store',
[ProdukController::class, 'store'])-
>name('Produk.store');
Route::get('/Produk/{Produk}/update',
[ProdukController::class, 'edit'])-
>name('Produk.edit');
Route::post('/Produk/{Produk}/update',
[ProdukController::class, 'update'])-
>name('Produk.update');
Route::delete('/Produk/{Produk}/delete',
[ProdukController::class, 'destroy'])-
>name('Produk.delete');
Route::get('/Produk/{Produk}/show',
[ProdukController::class, 'show'])-
>name('Produk.show');
Route::get('/Peminjaman',
[PeminjamanController::class, 'index'])-
>name('Peminjaman.index');
Route::post('/Peminjaman/{pesanan}/selesai',
[PeminjamanController::class,
'verifikasipesanan'])-
>name('Peminjaman.selesai');
Route::delete('/Peminjaman/{Pesanan}/delete',
[PeminjamanController::class, 'destroy'])-
>name('Peminjaman.delete');
Route::get('/pesanan/{pesananId}/order-items',
[PeminjamanController::class, 'show'])
    ->name('order-items.index');
Route::get('/laporan-peminjaman',
[PeminjamanController::class, 'laporan'])-
>name('Peminjaman.laporan');
Route::middleware('auth')->post('/cart/clear',
function () {
    $userId = Auth::id();
    if (!$userId) {
        return response()->json(['message' =>
'Unauthorized'], 401);
    }
    Cart::where('user_id', $userId)->delete();
    return response()->json(['message' => 'Cart
cleared successfully']);
});
Route::get('/lupa-password',
[AdminController::class,
'showCheckEmailForm'])-
>name('show.check.email');
Route::post('/check-email',
[AdminController::class, 'checkEmail'])-
>name('check.email');
```

```
Route::post('/update-password',
[AdminController::class, 'updatePassword'])-
>name('update.password');
Route::get('/admin', [AdminController::class,
'index'])->name('admin.index');
Route::delete('/admin/{user}',
[AdminController::class, 'destroy'])-
>name('admin.destroy');
require __DIR__ . '/auth.php';
```

KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa : Besse Syulfiani	**Pembimbing I : Andi Wafiah, S.Kom., M.Kom**
NIM : 221280036	**Pembimbing II : Mugahffir Yunus. ST., MT**
Judul Skripsi : Perencanaan Aplikasi Sistem Manajemen Toko Online Penyewaan Baju Untuk Meningkatkan Konversi Penjualan	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 Lengkapi aturan dan syarat sewa baju bodo.	1.	Konsultasi 1 Perbaiki tata bahasa di latar belakang !!!	
Konsultasi 2 Use case diagram di lengkapi	1.	Konsultasi 2 - Buatkan kerangka pikir - Perbaikan use care - perbaikan penulisan judul	
Konsultasi 3 - Batasan masalah - Tempat penelitian. - Penggunaan bahasa asing di cetak miring.	1	Konsultasi 3 - Perbaikan kerangka pikir - Menggati jenis penelitian dan lokasi dan waktu - buatkan perancangan antarmuka	
Konsultasi 4 - Daftar Pustaka disesuaikan dgn Pedoman Penulisan. dan diurut abjad.	1	Konsultasi 4 - Revisi antarmuka perancang aplikasi.	
Konsultasi 5 ACC Proposal	1.	Konsultasi 5 Acc Proposal	

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton A4 berwarna Hijau muda dan dicetak timbal balik

KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

SKRIPSI

Mahasiswa : BESSE SYULFIANI	**Pembimbing I : Andi Wafiah, S.Kom., M.Kom**
NIM : 221280036	**Pembimbing II : Mughaffir Tunus. ST., MT**
Judul Skripsi : PERANCANGAN APLIKASI SISTEM PENYEWAAN BAJU ADAT	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 Revisi Aplikasi	4.	Konsultasi 1 - Revisi Aplikasi	
Konsultasi 2 keterangan size setiap ukuran baju	4	Konsultasi 2 Revisi Aplikasi	
Konsultasi 3 Revisi Aplikasi	4.	Konsultasi 3 Revisi Aplikasi	
Konsultasi 4 Acc	4.	Konsultasi 4 Format Penulisan	
Konsultasi 5 - Lengkapi daftar pustaka. - Revisi Aplikasi	4.	Konsultasi 5 Perjelas Gambar.	

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton A4 berwarna Hijau muda dan dicetak timbal balik