

Lampiran



KARTU MONITORING BIMBINGAN
MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa : Muhammad shabir ismail	Pembimbing I : Hj. A. Irmayani P,ST,MT
NIM : 221280050	Pembimbing II : Wahyuddin, S.Kom., M.Kom
Judul Skripsi : Rancang bangun sistem pembuatan Curriculum vitae (CV) otomatis berbasis progressive web apps (PWA)	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 Pembantu DF	5/10/24 Arj	Konsultasi 1 - studi penulisan penulisan - format penulisan & perbaikan	
Konsultasi 2 Bab 1 System Dgn DF	Arj	Konsultasi 2 - margin - spasi pada dokumen	
Konsultasi 3 Pawella	Arj	Konsultasi 3 - Daftar pustaka & perbaikan	
Konsultasi 4 Penulisan dan Petunjuk Penulisan	Arj	Konsultasi 4 Ace gaminar proposal	
Konsultasi 5 Ace & proprose	Arj	Konsultasi 5	

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa di setiap 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

Lanjutan ...

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 6		Konsultasi 6	
Konsultasi 7		Konsultasi 7	
Konsultasi 8		Konsultasi 8	
Konsultasi 9		Konsultasi 9	
Konsultasi 10		Konsultasi 10	

Mengetahui
Ketua Program Studi

Marlina
Marlina, S.Kom.,M.Kom.
NBM. 1162 680

Parepare,

Mahasiswa

Muhammad shabir ismail
NIM. 221280050

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa saat setiap konsultasi dan diserahkan kepada Pembimbing
3. Kartu ini wajib diampelas 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 (50% dicetak timbul baik)

Lanjutan ...

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 6 <i>Xce 4/4/21</i>	<i>10/2/21</i>	Konsultasi 6	
Konsultasi 7		Konsultasi 7	
Konsultasi 8		Konsultasi 8	
Konsultasi 9		Konsultasi 9	
Konsultasi 10		Konsultasi 10	

Parepare,

Mengetahui
Ketua Program Studi

Marlina
Marlina, S.Kom., M.Kom.
NBM. 1162 680

Mahasiswa

Muhammad shabir ismail
NIM. 221280050

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/jalan singkat
4. Kartu ini dicetak di atas kertas karton A4 berwarna Hijau muda dan dicetak timbul baik



KARTU MONITORING BIMBINGAN
 MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
 FAKULTAS TEKNIK
 UNIVERSITAS MUHAMMADIYAH PAREPARE

SKRIPSI

Mahasiswa : Muhammad shabir ismail	Pembimbing I : Hj. A. Irmayani Pawelloi, ST, MT
NIM : 221280050	Pembimbing II : Wahyuddin, S.Kom, M.Kom
Judul Skripsi : rancang bangun sistem pembuatan Curriculum vitae (CV) otomatis berbasis progressive web apps (PWA)	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 ~ perbaiki let head 9br (jane)	7/1/25 d/1	Konsultasi 1 - detail perbaiki - kata asing dihindari	f
Konsultasi 2 ~ perbaiki kt asy (image)	8/1	Konsultasi 2 - perbaiki detail	f
Konsultasi 3 ~ perbaiki asy	8/1	Konsultasi 3 Acc	f
Konsultasi 4 Acc / perbaiki Habr	7/1/25 7/1	Konsultasi 4 perbaiki	f
Konsultasi 5 - Absrak - Pembahasan	18/2/25 d/1	Konsultasi 5	

Lanjut ke halaman sebelah...

Perhatian :
 1. Mahasiswa wajib konsultasi minimal 5 kali
 2. Kartu ini wajib dibawa oleh mahasiswa saat datang konsultasi dan ditanda dengan Pembimbing
 3. Kartu ini wajib diserahkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/jalan skripsi
 4. Kartu ini dicetak di atas kertas karton A4 berwarna Hijau Muda dan tidak boleh rusak

Migrasi Database:

```
<?php
```

```
use
```

```
Illuminate\Support\Facades\Schema;
```

```
use
```

```
Illuminate\Database\Migrations\Migration;
```

```
return new class extends Migration
```

```
{
```

```
use
```

```
Illuminate\Database\Schema\Blueprint;
```

```
/**
```

```
* Run the migrations.
```

```

        */
        public function up(): void
        {
            Schema::create('users', function
            (Blueprint $table) {
                $table->id();

                $table->string('name');

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

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

                $table-
                >timestamp('email_verified_at')-
                >nullable();

                $table->string('password');

                $table->rememberToken();

                $table->timestamps();

            });

            Schema::create('password_reset_t
            okens', 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_re
            set_tokens');

            Schema::dropIfExists('sessions');

        }
    };
}

```

<?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('biodatas', function (Blueprint $table) {
            $table->id();

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

            $table->unsignedBigInteger('user_id');

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

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

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

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

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

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

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

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

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

```

```

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

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

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

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

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

            $table->timestamps();

        });
    }

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

Model:

<?php

namespace App\Models;

// use
Illuminate\Contracts\Auth\MustVerifyEmail;

use
Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Foundation\Auth\User as

```

```

        Authenticatable;

use Illuminate\Notifications\Notifiable;

use 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',

        'email',

        'password',

        'google_id'
    ];


    /**
     * 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 Biodatas()
    // {
    //     //
    //     return $this->hasMany(Biodata::class);
    // }
}
<?php

namespace App\Models;

use

```



```

        Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Biodata extends Model
{
    use HasFactory;

    protected $fillable = [

        'nama',

        'asal',

        'email',

        'phone',

        'linkedin',

        'lokasi',

        'profil_singkat',

        'skill',

        'pengalaman',

        'pendidikan',

        'sertifikat',

        'bahasa',

        'foto',

        'user_id',

        'jenis_CV',

        'slug'
    ];

    public function user()
    {

```

```

        return $this->belongsTo(User::class,
        'user_id', 'id');
    }
}

Controller:

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Models\User;
use Illuminate\Auth\Events\Registered;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;
use Illuminate\Validation\Rules;
use Spatie\Permission\Models\Role;
use Inertia\Inertia;
use Inertia\Response;

class RegisteredUserController extends Controller
{
    /**
     * Display the registration view.
     */

    public function create(): Response

```

```
{
    return Inertia::render('Auth/Register');
}
```

```
/**
 * Handle an incoming registration request.
 *
 * @throws
 * \Illuminate\Validation\ValidationException
 */
```

```
public function store(Request $request):
    RedirectResponse
{
    $request->validate([
        'name' => 'required|string|max:255',
        'email' =>
            'required|string|lowercase|email|max:
            255|unique:'.User::class,
        'password' => ['required', 'confirmed',
            Rules\Password::defaults()],
    ]);

    $user = User::create([
        'name' => $request->name,
        'email' => $request->email,
        'password' => Hash::make($request->
            password),
    ]);

    $role = Role::findByName('user');
```

```
$user->assignRole($role);
event(new Registered($user));

Auth::login($user);
```

```
return redirect(route('biodata.index',
    absolute: false));
```

```
}
}
```

```
<?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',
    ]);
}

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 = 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']);
    }
}

```

```

        $user->update([
            'password' => bcrypt($request->input('password')),
        ]);

        return redirect()->route('admin.index')->with('success', 'Password berhasil diubah');
    }

    return redirect()->back()->withErrors(['user_id' => 'User tidak valid']);
}

}

<?php

namespace App\Http\Controllers;

use App\Models\Biodata;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Storage;

use Inertia\Inertia;

use Illuminate\Support\Facades\Redirect;

use Illuminate\Support\Str;

class BiodataController extends Controller
{
    public function index()
    {
        $user->update([
            'password' => bcrypt($request->input('password')),
        ]);

        return redirect()->route('admin.index')->with('success', 'Password berhasil diubah');
    }

    return redirect()->back()->withErrors(['user_id' => 'User tidak valid']);
}

}

} elseif ($user->hasRole('user')) {

    $Biodata = Biodata::with('user')->get()->map(function ($item) {

        $item->foto = Storage::url($item->foto);

        return $item;

    });

    $Biodata = Biodata::where('user_id', $user->id)->get()->map(function ($item) {

        $item->foto = Storage::url($item->foto);

        return $item;

    });

    return Inertia::render('Biodata/index', ['Biodata' => $Biodata]);

}

public function create()
{
    return Inertia::render('Biodata/create');
}

```

```

}

public function store(Request $request)
{
    $request->validate([
        'nama' => 'required',
        'asal' => 'nullable',
        'email' => 'nullable',
        'phone' => 'nullable',
        'linkedin' => 'nullable',
        'lokasi' => 'nullable',
        'profil_singkat' => 'nullable',
        'skill' => 'nullable',
        'pengalaman' => 'nullable|string',
        'pendidikan' => 'nullable',
        'sertifikat' => 'nullable',
        'bahasa' => 'nullable',
        'jenis_CV' => 'nullable',
        'foto' =>
            'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:20048',
        'slug' =>
            'nullable|unique:Biodatas,slug',
    ]);

    $fotoName = null;
    if ($request->hasFile('foto')) {
        $fotoName = $request->file('foto')->store('public/Biodata');
    }
}

```

```

        $slug = Str::slug($request->input('nama'));

        $originalSlug = $slug;
        $counter = 1;

        while (Biodata::where('slug', $slug)->exists()) {
            $slug = $originalSlug . '-' . $counter++;
        }

        Biodata::create([
            'nama' => $request->input('nama'),
            'asal' => $request->input('asal'),
            'email' => $request->input('email'),
            'phone' => $request->input('phone'),
            'linkedin' => $request->input('linkedin'),
            'lokasi' => $request->input('lokasi'),
            'profil_singkat' => $request->input('profil_singkat'),
            'skill' => $request->input('skill'),
            'pengalaman' => $request->input('pengalaman'),
            'pendidikan' => $request->input('pendidikan'),
            'sertifikat' => $request->input('sertifikat'),
            'bahasa' => $request->input('bahasa'),
        ]);
    }
}

```

```

        'jenis_CV' => $request-
        >input('jenis_CV'),

        'foto' => $fotoName,

        'slug' => $slug,

        'user_id' => auth()->id(),

    ]);

    return Redirect::route('biodata.index')-
    >with('success', 'Biodata berhasil
    ditambah');
}

```

```

public function edit(Biodata $Biodata)

{

    return Inertia::render('Biodata/edit',
    ['Biodata' => $Biodata]);

}

```

```

public function update(Request $request,
    Biodata $Biodata)

{

    $request->validate([

        'nama' => 'required',

        'asal' => 'required',

        'email' => 'required',

        'phone' => 'nullable',

        'linkedin' => 'required',

        'lokasi' => 'required',

        'profil_singkat' => 'required',
    ]

```

```

        'skill' => 'required',

        'pengalaman' => 'required|string',

        'pendidikan' => 'required',

        'sertifikat' => 'required',

        'bahasa' => 'required',

        'jenis_CV' => 'required',

        'foto' =>
        'nullable|image|mimes:jpeg,png,jpg,g
        if,svg|max:20048',

        'slug' =>
        'nullable|unique:Biodatas,slug',
    ]

```

```

    ]);

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

        if ($Biodata->foto) {

            Storage::delete($Biodata->foto);

        }

        $fotoName = $request->file('foto')-
        >store('public/Biodata');

        $Biodata->foto = $fotoName;

    }

    $Biodata->update([

        'nama' => $request->input('nama'),

        'asal' => $request->input('asal'),

        'email' => $request->input('email'),

        'phone' => $request->input('phone'),

        'linkedin' => $request-
        >input('linkedin'),

        'lokasi' => $request->input('lokasi'),

        'profil_singkat' => $request-

```

```

        >input('profil_singkat'),

        'skill' => $request->input('skill'),

        'pengalaman' => $request->input('pengalaman'),

        'pendidikan' => $request->input('pendidikan'),

        'sertifikat' => $request->input('sertifikat'),

        'bahasa' => $request->input('bahasa'),

        'jenis_CV' => $request->input('jenis_CV'),

    ]);

    return Redirect::route('biodata.index')->with('success', 'Biodata berhasil di ubah');
}

public function show(Biodata $Biodata)
{
    return Inertia::render('Biodata/show', ['Biodata' => $Biodata]);
}

public function showlayout(Biodata $Biodata)
{
    return Inertia::render('Biodata/layout', ['Biodata' => $Biodata]);
}

public function destroy(Biodata $Biodata)
{
    if ($Biodata->foto) {
        Storage::delete($Biodata->foto);
    }

    $Biodata->delete();

    return Redirect::route('biodata.index')->with('success', 'Biodata berhasil di hapus');
}
}

<?php

namespace App\Http\Controllers;

use App\Models\Biodata;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Storage;
use Inertia\Inertia;
use Illuminate\Support\Facades\Redirect;

class EkportController extends Controller
{
    public function exportPdf($id, $viewType)
    {
        // Ambil data user

        $Biodata = Biodata::where('id', $id)->first();

```

```

if (!$Biodata) {

    return redirect()->back()->with('error',
        'Data tidak ditemukan.');
```

}

```

// Load view PDF berdasarkan pilihan
user

$viewName = 'template1';

if (in_array($viewType, ['template2',
    'template3', 'template4', 'template5',
    'template6', 'template7',
    'template8','template9','template10','te
    mplate11'])) {

    $viewName = $viewType;
}

// Return response dengan header untuk
tampilan preview

//      return
view("Template/$viewName",['Bioda
ta' => $Biodata]);

      return
Inertia::render("Template/$viewNam
e", [

    'Biodata' => $Biodata

]);

}

}

```

<?php

```

namespace App\Http\Controllers;

```

```

use App\Models\User;

use Illuminate\Http\Request;

use Laravel\Socialite\Facades\Socialite;

use Spatie\Permission\Models\Role;

use Illuminate\Support\Str;

class SocialiteController extends Controller
{

    public function redirect()
    {

        return Socialite::driver('google')-
            >redirect();

    }

    public function callback()
    {

        $userFromGoogle =
        Socialite::driver('google')-
            >stateless()->user();

        $userFromDb =
        User::where('google_id',
            $userFromGoogle->getId()->first();

        if (!$userFromDb) {

            $userFromDb = new User();

            $userFromDb->email =
            $userFromGoogle->getEmail();

            $userFromDb->google_id =
            $userFromGoogle->getId();

```



```

        $userFromDb->name =
$userFromGoogle->getName();

        $userFromDb->password =
bcrypt(Str::random(16));

$userFromDb->save();

$role = Role::findByName('user');

$userFromDb->assignRole($role);

auth('web')->login($userFromDb);
session()->regenerate();

if ($userFromDb->hasRole('user')) {
    return redirect()-
>route('biodata.index');
}

return redirect('/');
}

auth('web')->login($userFromDb);
session()->regenerate();

if ($userFromDb->hasRole('anggota')) {
    return redirect()-
>route('biodata.index');
}

return redirect('/');

```

Route:

```

<?php

use
App\Http\Controllers\ProfileController;

use
App\Http\Controllers\EkportController;

use
App\Http\Controllers\BiodataController;

use
App\Http\Controllers\AdminController;

use Illuminate\Foundation\Application;
use Illuminate\Support\Facades\Route;
use Inertia\Inertia;

Route::get('/', function () {
    return Inertia::render('Welcome', [
        'canLogin' => Route::has('login'),
        'canRegister' =>
Route::has('register'),
        'laravelVersion' =>
Application::VERSION,
        'phpVersion' => PHP_VERSION,
    ]);
})->name('home');

Route::get('/template1', function () {

```

```

        return
        Inertia::render('Template/template1');
    })->name('template1');

    Route::get('/template2', function () {
        return
        Inertia::render('Template/template2');
    })->name('template2');

    Route::get('/template3', function () {
        return
        Inertia::render('Template/template3');
    })->name('template3');

    Route::get('/template4', function () {
        return
        Inertia::render('Template/template4');
    })->name('template4');

    Route::get('/template5', function () {
        return
        Inertia::render('Template/template5');
    })->name('template5');

    Route::get('/template6', function () {
        return
        Inertia::render('Template/template6');
    })->name('template6');

    Route::get('/template7', function () {
        return
        Inertia::render('Template/template7');
    })->name('template7');

    Route::get('/template8', function () {
        return
        Inertia::render('Template/template8');
    })->name('template8');

    Route::get('/template9', function () {
        return
        Inertia::render('Template/template9');
    })->name('template9');

```

```

    })->name('template9');

    Route::get('/template10', function () {
        return
        Inertia::render('Template/template10');
    })->name('template10');

    Route::get('/template11', function () {
        return
        Inertia::render('Template/template11');
    })->name('template11');

    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('/tampilan-
    cv/{Biodata:slug}/show',
    [BiodataController::class, 'showlayout'])->
    >name('layout');

    Route::middleware(['auth', 'role:admin'])->
    group(function () {

        Route::get('/admin',
        [AdminController::class, 'index'])->
        >name('admin.index');

        Route::delete('/admin/{user}',
        [AdminController::class, 'destroy'])->
        >name('admin.destroy');

    });

```

```

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::get('/Biodata',
[BiodataController::class, 'index'])-
>name('biodata.index');

    Route::get('/Biodata/create',
[BiodataController::class, 'create'])-
>name('biodata.create');

    Route::post('/Biodata/store',
[BiodataController::class, 'store'])-
>name('biodata.store');

    Route::get('/Biodata/{Biodata:slug}/update',
[BiodataController::class, 'edit'])-
>name('biodata.edit');

    Route::post('/Biodata/{Biodata:slug}/update',
[BiodataController::class, 'update'])-
>name('biodata.update');

    Route::delete('/Biodata/{Biodata:slug}/delete',
[BiodataController::class,
'destroy']->name('biodata.delete');

    Route::get('/Biodata/{Biodata:slug}/show',
[BiodataController::class, 'show'])-
>name('biodata.show');

    Route::get('/biodata/{id}/export/{viewType}',
[EkportController::class,
'exportPdf']->name('export.pdf');

});

```

```
require __DIR__.'auth.php';
```

Views:

```

import { useEffect, useState } 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"));
};

const [showPassword,
setShowPassword] = useState(false);

return (
    <GuestLayout>
        <Head title="Log in" />

        {status && (
            <div className="mb-4 font-
medium text-sm text-green-600">
                {status}
            </div>
        )}

        <form onSubmit={submit}
className="space-y-6">
            <div>
                <InputLabel
htmlFor="email" value="Email" />
                <TextInput
                    id="email"
                    type="email"

```

```

                    name="email"
                    value={data.email}
                    className="mt-1 block
w-full"
                    autoComplete="usernam
e"
                    isFocused={true}
                    onChange={(e) =>
setData("email", e.target.value)}
                />
                <InputError
message={errors.email}
className="mt-2" />
            </div>

            <div className="relative">
                <InputLabel
htmlFor="password" value="Password"
/>

                <TextInput
                    id="password"
                    type={showPassword ?
"text" : "password"}
                    name="password"
                    value={data.password}
                    className="mt-1 block
w-full"
                    autoComplete="current-
password"
                    onChange={(e) =>
setData("password", e.target.value)}
                />
                <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>

    <div className="flex items-
center justify-between">

        {/* <label className="flex
items-center">

            <Checkbox

                name="remember"

                checked={data.remem
ber}

                onChange={(e) =>
setData("remember", e.target.checked)}

            />

            <span className="ms-2
text-sm text-gray-600 dark:text-gray-
400">

                Ingta

            </span>

        </label> */}

        <Link

            href={route("register")}

```

```

        className="underline
text-sm text-gray-600 hover:text-gray-
900"

        >

        Belum punya akun?

    </Link>

    <Link

        href={route("show.check.
email")}

        className="underline
text-sm text-gray-600 hover:text-gray-
900"

        >

        Lupa password?

    </Link>

</div>

    <div className="flex items-
center justify-center">

        <PrimaryButton

            className="w-full text-
center"

            disabled={processing}

            >

                Log in

            </PrimaryButton>

        </div>

        {/* Tombol Masuk dengan
Google */}

        <div className="text-center
mt-4">

```

```

        <p className="text-gray-
500 text-sm mb-2">
            atau lanjutan dengan
        </p>
        <a
            href={route("redirect")}
            className="flex items-
center justify-center gap-2 bg-red-600
text-white py-2 px-4 rounded-lg
hover:bg-red-700 transition duration-
300"
        >
            <svg
                className="w-5 h-5"
                xmlns="http://www.w3.
org/2000/svg"
                viewBox="0 0 48 48"
            >
                <path
                    fill="#4285F4"
                    d="M24 9.5c3.39 0
6.42 1.17 8.8 3.11 6.6-6.6C35.02 2.02
29.87 0 24 0 14.93 0 7.28 5.48 3.75
13.43 7.88 6.11C13.07 14.11 18.06 9.5
24 9.5z"
                />
                <path
                    fill="#34A853"
                    d="M46.62 24.5c0-
1.47-.13-2.9-.38-4.28H24v8.1h12.8c-.56
2.78-2.1 5.12-4.3 6.69 7.04 5.46C43.47
37.54 46.62 31.56 46.62 24.5z"
                />
                <path
                    fill="#FBBC05"

```

```

                    d="M11.63
28.05A14.48 14.48 0 0 1 9.5 24c0-
1.41 24-2.76.68-4.01 7.88-6.1C1.46
16.27 0 20.03 0 24c0 3.97 1.46 7.73 4.3
10.11 7.33-6.06z"
                />
            <path
                fill="#EA4335"
                d="M24 48c6.48 0
11.91-2.13 15.88-5.77 7.04-5.46c-2
1.37-4.57 2.23-7.6 2.23-5.94 0-10.93-
4.6-12.37-10.73 7.33 6.06C10.08 43.63
16.49 48 24 48z"
            />
        </svg>
        Sign in with Google
    </a>
</div>
</form>
</GuestLayout>
);
}

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 { Head, Link, useForm } from
"@inertiajs/react";

export default function Register() {

  const { data, setData, post,
processing, errors, reset } = useForm({

    name: "",
    email: "",
    password: "",
    password_confirmation: "",
  });

  useEffect(() => {

    return () => {

      reset("password",
"password_confirmation");

    };

  }, []);

  const submit = (e) => {

    e.preventDefault();

    post(route("register"));

  };

  const [showPassword,
setShowPassword] = useState(false);

  return (

    <GuestLayout>

      <Head title="Register" />

      <form onSubmit={submit}>

        <div>

```

```

      <InputLabel
htmlFor="name" value="Name" />

      <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="mt-4">

      <InputLabel
htmlFor="email" value="Email" />

      <TextInput

        id="email"

        type="email"

        name="email"

        value={data.email}

        className="mt-1 block

w-full"

```

```

        autoComplete="username"
e"
        onChange={(e) =>
setData("email", e.target.value)}
        required
    />

    <InputError
message={errors.email}
className="mt-2" />

    </div>

    <div className="relative">

        <InputLabel
htmlFor="password" value="Password"
/>

        <TextInput
            id="password"

            type={showPassword ?
"text" : "password"} // Mengatur tipe
input berdasarkan state showPassword

            name="password"
            value={data.password}
            className="mt-1 block
w-full"

            autoComplete="current-
password"

            onChange={(e) =>
setData("password", e.target.value)}
        />

        {/* Tombol untuk toggle
melihat password */}

        <button
            type="button"

```

```

        className="absolute
right-3 top-1/2 transform -translate-y-1/2
text-gray-500"

        onClick={() =>
setShowPassword(!showPassword)} //
Toggle nilai showPassword
    >

        {showPassword ? "Hide"
: "Show"}

    </button>

    <InputError
        message={errors.passwo
rd}

        className="mt-2"
    />
</div>

<div className="relative">

    <InputLabel
        htmlFor="password_confir
mation"

        value="Confirm
Password"
    />

    <TextInput
        id="password_confirmati
on"

        type={showPassword ?
"text" : "password"}

        name="password_confir
mation"

        value={data.password_c
onfirmation}

```



```

        className="mt-1 block
w-full"
        autoComplete="new-
password"
        onChange={(e) =>
            setData("password_co
nfirmation", e.target.value)
        }
        required
    />

```

```

<button
    type="button"
    className="absolute
right-3 top-1/2 transform -translate-y-1/2
text-gray-500"
    onClick={() =>
        setShowPassword(!showPassword)} //
        Toggle nilai showPassword
    >
        {showPassword ? "Hide"
: "Show"}
</button>

```

```

<InputError
    message={errors.passwo
rd_confirmation}
    className="mt-2"
    />
</div>

```

```

<div className="flex items-
center justify-end mt-4">
    <Link

```

```

        href={route("login")}
        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"
    >
        Sudah punya akun?
</Link>

```

```

<PrimaryButton
    className="ms-4"
    disabled={processing}>
        Register
</PrimaryButton>
</div>

```

```

        {/* Tombol Masuk
dengan Google */}
        <div
            className="text-center mt-4">
                <p className="text-gray-
500 text-sm mb-2">
                    atau lanjutan dengan
                </p>
                <a
                    href={route("redirect")}
                    className="flex items-
center justify-center gap-2 bg-red-600
text-white py-2 px-4 rounded-lg
hover:bg-red-700 transition duration-
300"
                >
                    <svg

```

```

        className="w-5 h-5"
        xmlns="http://www.w3.
org/2000/svg"
        viewBox="0 0 48 48"
    >
        <path
            fill="#4285F4"
            d="M24 9.5c3.39 0
6.42 1.17 8.8 3.116.6-6.6C35.02 2.02
29.87 0 24 0 14.93 0 7.28 5.48 3.75
13.4317.88 6.11C13.07 14.11 18.06 9.5
24 9.5z"
        />
        <path
            fill="#34A853"
            d="M46.62 24.5c0-
1.47-.13-2.9-.38-4.28H24v8.1h12.8c-.56
2.78-2.1 5.12-4.3 6.6917.04 5.46C43.47
37.54 46.62 31.56 46.62 24.5z"
        />
        <path
            fill="#FBBC05"
            d="M11.63
28.05A14.48 14.48 0 0 1 9.5 24c0-
1.41.24-2.76.68-4.011-7.88-6.1C1.46
16.27 0 20.03 0 24c0 3.97 1.46 7.73 4.3
10.1117.33-6.06z"
        />
        <path
            fill="#EA4335"
            d="M24 48c6.48 0
11.91-2.13 15.88-5.771-7.04-5.46c-2
1.37-4.57 2.23-7.6 2.23-5.94 0-10.93-
4.6-12.37-10.731-7.33 6.06C10.08 43.63
16.49 48 24 48z"
        />

```

```

    </svg>
    Sign in with Google
  </a>
</div>
</form>
</GuestLayout>
);
}

import React, { useState, useEffect }
from "react";

import { Head, useForm } from
"@inertiajs/react";

import "react-quill/dist/quill.snow.css";
import ReactQuill from "react-quill";

import InputError from
"@/Components/InputError";

import InputLabel from
"@/Components/InputLabel";

import PrimaryButton from
"@/Components/PrimaryButton";

import AuthenticatedLayout from
"@/Layouts/AuthenticatedLayout";

import indonesia from "territory-
indonesia";

import parse from "html-react-parser";

const Create = () => {
  const { data, setData, post,
processing, errors } = useForm({
    nama: "",
    asal: "",
    email: "",

```

```

    phone: "",
    linkedin: "",
    lokasi: "",
    profil_singkat: "",
    skill: "",
    pengalaman: "",
    pendidikan: "",
    sertifikat: "",
    bahasa: "",
    jenis_CV: "",
    foto: null,
  });

  const [provinsiList, setProvinsiList] =
    useState([]);

  const [kotaList, setKotaList] =
    useState([]);

  const [kecamatanList,
    setKecamatanList] = useState([]);

  const [kelurahanList,
    setKelurahanList] = useState([]);

  useEffect(() => {

    const getProvinsi = async () => {

      await indonesia

        .getAllProvinces()

        .then((res) =>
          setProvinsiList(res));

    };

    getProvinsi();

  }, []);

```

```

    const getKota = async (provinsid) =>
    {

      await indonesia

        .getRegenciesOfProvinceId(provi
          nsid)

        .then((res) => setKotaList(res));

    };

    const getKecamatan = async (kotaid)
    => {

      await indonesia

        .getDistrictsOfRegencyId(kotaid)

        .then((res) =>
          setKecamatanList(res));

    };

    const getKelurahan = async
    (kecamatanId) => {

      await indonesia

        .getVillagesOfDistrictId(kecamatanId)

        .then((res) =>
          setKelurahanList(res));

    };

    const handleChangeProvinsi = (e) =>
    {

      getKota(e.target.value);

      setData("provinsi",
        e.target.options[e.target.selectedIndex].t
          ext);

    };

    const handleChangeKota = (e) => {

```

```

        getKecamatan(e.target.value);

        setData("kota",
e.target.options[e.target.selectedIndex].t
ext);

    };

    const handleChangeKecamatan = (e)
=> {

        getKelurahan(e.target.value);

        setData("kecamatan",
e.target.options[e.target.selectedIndex].t
ext);

    };

    const handleChangeKelurahan = (e)
=> {

        const kelurahan =
e.target.options[e.target.selectedIndex].t
ext;

        setData("kelurahan", kelurahan);

        const lokasi = `${data.provinsi},
${data.kota}, ${data.kecamatan},
${kelurahan}`;

        setData("lokasi", lokasi);

    };

    const handleChange = (e) => {

        if (e.target.name === "foto") {

            setData("foto", e.target.files[0] ||
null);

        } else {

            setData(e.target.name,
e.target.value || "");

        }
    }

```

```

    };

    const handleQuillChange = (name,
content) => {

        setData(name, content);

    };

    const modules = {

        toolbar: [

            [{ header: [1, 2, 3, false] }],

            [{ font: [] }],

            [{ size: ["small", false, "large",
"huge"] }],

            ["bold", "italic", "underline",
"strike"],

            [{ list: "ordered" }, { list: "bullet" }],

            [{ script: "sub" }, { script: "super"
}],

            [{ indent: "-1" }, { indent: "+1" }],

            [{ align: [] }],

            ["link", "image"],

            ["blockquote", "code-block"],

            [{ color: [] }, { background: [] }],

            ["clean"],

        ],

    };

    const formats = [

        "header",

        "font",

        "size",

        "bold",

```

```

        "italic",
        "underline",
        "strike",
        "list",
        "bullet",
        "script",
        "indent",
        "align",
        "link",
        "image",
        "blockquote",
        "code-block",
        "color",
        "background",
    ];

    const submit = (e) => {
        e.preventDefault();
        post(route("biodata.store"));
    };

    const quillFields = [
        { label: "Profil Singkat", name:
"profil_singkat" },
        {
            label: "Skill",
            name: "skill",
            deskripsi:
                "Catatan: <br/> Tulis basic
                untuk skill dasar <br/> Tulis intermediate
                untuk skill lanjutan <br/> Tulis advance
                untuk skill yg dikuasai sepenuhnya <br/>
                ",
        },
        {
            label: "Pengalaman kerja", name:
"pengalaman" },
        { label: "Riwayat Pendidikan",
            name: "pendidikan" },
        { label: "Sertifikat", name:
"sertifikat" },
        { label: "Bahasa yg dikuasai",
            name: "bahasa" },
    ];

    return (
        <AuthenticatedLayout>

        <div className="mt-20 flex justify-
        center items-center text-black dark:text-
        white">

            <div className="bg-gray-200
            dark:bg-gray-800 border border-gray-
            800 dark:border-gray-600 rounded-xl p-
            5 w-full max-w-3xl">

                <div>

                    <Head title="Create
                    Biodata" />

                    { /* Penjelasan Form */ }

                    <div className="mb-6 p-4
                    bg-yellow-100 dark:bg-yellow-700
                    rounded-md">

                        <h2 className="font-
                        bold text-lg">Informasi Pengisian</h2>

                        <p className="text-
                        sm">

                            <span
                            className="font-semibold">Kolom
                            dengan tanda (*) wajib diisi:</span>

                            </p>

```

```

        <ul className="list-disc
list-inside text-sm">
            <li>Nama lengkap
        *</li>
            <li>Alamat Detail *</li>
            <li>Profil Singkat *</li>
            <li>Alamat (provinsi,
kota, kecamatan, kelurahan) *</li>
            <li>Jenis CV *</li>
            <li>Pengalaman Kerja
        *</li>
            <li>Riwayat
Pendidikan *</li>
            <li>Foto Diri *</li>
        </ul>
        <p className="text-sm
mt-2">
            <span
className="font-semibold">Kolom
lainnya bersifat opsional dan boleh
dikosongkan.</span>
        </p>
    </div>

    <form onSubmit={submit}>
        { /* Input Text */ }
        {[
            { label: "Nama
lengkap", name: "nama", type: "text",
required: true },
            { label: "Email", name:
"email", type: "text", required: false },
            { label: "Nomor
WhatsApp atau Nomor Handphone",
name: "phone", type: "number",
required: false },

```

```

            { label: "LinkedIn atau
Sosial Media Lain", name: "linkedin",
type: "text", required: false },
            { label: "Alamat Detail",
name: "asal", type: "text", required: true
        },
        ].map((field, index) => (
            <div className="mt-
4" key={index}>
                <InputLabel
                    htmlFor={field.na
me}
                    value={` ${field.lab
el}${field.required ? " *" : ""}`}
                    className={field.
required ? "font-bold text-red-500" : ""}
                />
                <input
                    id={field.name}
                    name={field.name
                }
                    type={field.type}
                    value={data[field.
name]}
                    className="mt-1
block w-full border-gray-300 bg-white
dark:bg-gray-600 rounded-md shadow-
sm focus:ring focus:ring-indigo-200"
                    onChange={handl
eChange}
                />
                <InputError
                    message={errors[field.name]}
                    className="mt-2" />
            </div>
        )))
    )

```

```

        { /* File Input - Foto Diri */ }
        <div className="mt-4">
            <InputLabel
                htmlFor="foto"
                value="Foto Diri *"
                className="font-
bold text-red-500"
            />
            <input
                id="foto"
                name="foto"
                type="file"
                className="mt-1
block w-full"
                onChange={handle
Change}
                accept="image/*"
            />
            <InputError
                message={errors.foto} className="mt-
2" />
        </div>

        { /* Alamat (Provinsi,
Kota, Kecamatan, Kelurahan) */ }
        <div className="mt-4">
            <InputLabel
                htmlFor="lokasi"
                value="Alamat
(Provinsi, Kota, Kecamatan, Kelurahan)
*"
                className="font-
bold text-red-500"
            />
            <select
                defaultValue={data.
provinsi}
                onChange={handle
ChangeProvinsi}
                className="mt-4
block w-full bg-gray-200 dark:bg-gray-
800 border-gray-600 rounded-md
shadow-sm focus:ring focus:ring-indigo-
200"
            >
                {provinsiList.map((pr
ovinsi) => (
                    <option
                        key={provinsi.id}
                        value={provinsi.id}>{provinsi.name}</opt
ion>
                )))}
            </select>

            <select
                defaultValue={data.k
ota}
                onChange={handle
ChangeKota}
                className="mt-4
block w-full bg-gray-200 dark:bg-gray-
800 border-gray-600 rounded-md
shadow-sm focus:ring focus:ring-indigo-
200"
            >
                {kotaList.map((kota)
=> (
                    <option
                        key={kota.id}
                        value={kota.id}>{kota.name}</option>

```

```

        )))
    </select>

    <select
        defaultValue={data.kecamatan}

        onChange={handle
        ChangeKecamatan}

        className="mt-4
        block w-full bg-gray-200 dark:bg-gray-
        800 border-gray-600 rounded-md
        shadow-sm focus:ring focus:ring-indigo-
        200"

        >

        {kecamatanList.map
        ((kecamatan) => (

            <option
            key={kecamatan.id}
            value={kecamatan.id}>{kecamatan.nam
            e}</option>

            )))
    </select>

    <select
        defaultValue={data.kelurahan}

        onChange={handle
        ChangeKelurahan}

        className="mt-4
        block w-full bg-gray-200 dark:bg-gray-
        800 border-gray-600 rounded-md
        shadow-sm focus:ring focus:ring-indigo-
        200"

        >

        {kelurahanList.map(
        (kelurahan) => (

```

```

            <option
            key={kelurahan.id}
            value={kelurahan.id}>{kelurahan.name}
            </option>

            )))
        </select>
    </div>

    {/* Jenis CV */}

    <div className="mt-4">

        <InputLabel
        htmlFor="jenis_CV" value="Jenis CV *"
        className="font-bold text-red-500" />

        <select

            id="jenis_CV"

            name="jenis_CV"

            value={data.jenis_C
            V}

            onChange={handle
            Change}

            className="mt-1
            block w-full bg-gray-200 dark:bg-gray-
            800 border-gray-600 rounded-md
            shadow-sm focus:ring focus:ring-indigo-
            200"

            >

            <option
            value="">Pilih Jenis CV</option>

            <option value="CV
            Kombinasi">CV Kombinasi</option>

            <option value="CV
            Fungsional">CV Fungsional</option>

            <option value="CV
            Kronologis">CV Kronologis</option>

            </select>

```



```

<InputError
message={errors.jenis_CV}
className="mt-2" />

</div>

{/* Quill Fields */}

{quillFields.map((field,
index) => (

<div className="mt-
4" key={index}>

<InputLabel

htmlFor={field.name}

value={` ${field.label}
${"profil_singkat", "pengalaman",
"pendidikan"}.includes(field.name) ? " *"
: ""}`}

className={`["pro
fil_singkat", "pengalaman",
"pendidikan"}.includes(field.name) ?
"font-bold text-red-500" : ""}`}

/>

{/* Render
deskripsi jika ada */}

{field.deskripsi

&& (

<div
className="text-sm text-gray-500 mt-
2">

{/*
Gunakan parse untuk merender
deskripsi HTML */}

{parse(field
d.deskripsi)}}

</div>

)}}

<ReactQuill

```

```

theme="snow"

value={data[field.
name]}

onChange={({cont
ent) => handleQuillChange(field.name,
content)}}

modules={module
s}

formats={formats}

placeholder={`Tuli
s ${field.label} di sini...`}

className="mt-1
block h-96 w-full bg-white dark:bg-gray-
900 rounded border dark:border-gray-
600"

/>

<InputError
message={errors[field.name]}
className="mt-2" />

</div>

)))

{/* Tombol Submit */}

<div className="flex
items-center justify-end mt-4 pt-20">

<PrimaryButton

className="ms-4"
disabled={processing}>

Simpan

</PrimaryButton>

</div>

</form>

</div>

</div>

</div>

</div>

```

```

    </AuthenticatedLayout>

    );
};

export default Create;

import React, { useState, useEffect }
from "react";

import { Head, usePage, router } from
"@inertiajs/react";

import InputError from
"@/Components/InputError";

import InputLabel from
"@/Components/InputLabel";

import PrimaryButton from
"@/Components/PrimaryButton";

import TextInput from
"@/Components/TextInput";

import AuthenticatedLayout from
"@/Layouts/AuthenticatedLayout";

import "react-quill/dist/quill.snow.css";

import ReactQuill from "react-quill";

import indonesia from "territory-
indonesia";

import parse from "html-react-parser";

const edit = ({ Biodata }) => {

    const [productData, setProductData]
= useState({

        nama: "",

        asal: "",

        email: "",

        phone: "",

        linkedin: "",

        domisili: "",

```

```

        lokasi: "",

        profil_singkat: "",

        skill: "",

        pengalaman: "",

        pendidikan: "",

        sertifikat: "",

        bahasa: "",

        jenis_CV: "",

        foto: null,

    });

    const { errors, processing } =
usePage().props;

    const [provinsiList, setProvinsiList] =
useState([]);

    const [kotaList, setKotaList] =
useState([]);

    const [kecamatanList,
setKecamatanList] = useState([]);

    const [kelurahanList,
setKelurahanList] = useState([]);

    useEffect(() => {

        if (Biodata) {

            setProductData({

                nama: Biodata.nama || "",

                asal: Biodata.asal || "",

                email: Biodata.email || "",

                phone: Biodata.phone || "",

                domisili: Biodata.domisili || "",

                linkedin: Biodata.linkedin || "",

```

```

        lokasi: Biodata.lokasi || "",
        profil_singkat:
Biodata.profil_singkat || "",
        skill: Biodata.skill || "",
        pengalaman:
Biodata.pengalaman || "",
        pendidikan:
Biodata.pendidikan || "",
        sertifikat: Biodata.sertifikat ||
"",
        bahasa: Biodata.bahasa || "",
        jenis_CV: Biodata.jenis_CV ||
"",
        foto: Biodata.foto || null,
    });
}
}, [Biodata]);

useEffect(() => {
    const getProvinsi = async () => {
        await indonesia
            .getAllProvinces()
            .then((res) =>
setProvinsiList(res));
    };
    getProvinsi();
}, []);

const getKota = async (provinsid) =>
{
    await indonesia
        .getRegenciesOfProvinceId(provi
nsid)

```

```

        .then((res) => setKotaList(res));
    };

    const getKecamatan = async (kotaid)
=> {
        await indonesia
            .getDistrictsOfRegencyId(kotaid)
            .then((res) =>
setKecamatanList(res));
    };

    const getKelurahan = async
(kecamatanId) => {
        await indonesia
            .getVillagesOfDistrictId(kecamat
anId)
            .then((res) =>
setKelurahanList(res));
    };

    const handleChangeProvinsi = (e) =>
{
        getKota(e.target.value);
        setProductData((prevState) => ({
            ...prevState,
            provinsi:
e.target.options[e.target.selectedIndex].t
ext,
        }));
    };

    const handleChangeKota = (e) => {
        getKecamatan(e.target.value);
    };

```

```

        setProductData((prevState) => ({
            ...prevState,
            kota:
e.target.options[e.target.selectedIndex].t
ext,
        }));
    };

```

```

const handleChangeKecamatan = (e)
=> {
    getKelurahan(e.target.value);
    setProductData((prevState) => ({
        ...prevState,
        kecamatan:
e.target.options[e.target.selectedIndex].t
ext,
    }));
};

```

```

const handleChangeKelurahan = (e)
=> {
    const kelurahan =
e.target.options[e.target.selectedIndex].t
ext;
    setProductData((prevState) => ({
        ...prevState,
        kelurahan: kelurahan,
        lokasi: `${prevState.provinsi},
${prevState.kota},
${prevState.kecamatan}, ${kelurahan}`,
    }));
};

```

```

const handleChange = (e) => {

```

```

    const { name, value, files } =
e.target;
    setProductData({
        ...productData,
        [name]: files ? files[0] : value,
    });
};

```

```

    // Mengatur perubahan untuk
    ReactQuill
    const handleQuillChange = (name,
content) => {
        setProductData({
            ...productData,
            [name]: content,
        });
    };
};

```

```

const modules = {
    toolbar: [
        [{ header: [1, 2, 3, false] }], //
Heading level 1, 2, 3
        [{ font: [] }], // Font family
        [{ size: ["small", false, "large",
"huge" ] }], // Font size
        ["bold", "italic", "underline",
"strike"], // Formatting options
        [{ list: "ordered" }, { list: "bullet" }],
// Ordered/Unordered list
        [{ script: "sub" }, { script: "super"
}], // Subscript/Superscript
        [{ indent: "-1" }, { indent: "+1" }], //
Indentation
        [{ align: [] }], // Text alignment

```

```

        ["link", "image"], // Link and
Image
        ["blockquote", "code-block"], //
Blockquote and Code Block
        [{ color: [] }, { background: [] }], //
Text and background color
        ["clean"], // Remove formatting
    ],
};

```

```
const formats = [
```

```

    "header",
    "font",
    "size",
    "bold",
    "italic",
    "underline",
    "strike",
    "list",
    "bullet",
    "script",
    "indent",
    "align",
    "link",
    "image",
    "blockquote",
    "code-block",
    "color",
    "background",

```

```
];
```

```

const submit = (e) => {
    e.preventDefault();
    const formData = new FormData();
    for (const key in productData) {
        formData.append(key,
productData[key]);
    }
    router.post(`/Biodata/${Biodata.slug
}/update`, formData);
};

```

```
const quillFields = [
```

```

    {
        label: "Profil Singkat",
        name: "profil_singkat",
        autoComplete: "profil_singkat",
    },

```

```

    {
        label: "Skill",
        name: "skill",
        deskripsi:

```

```

            "Catatan: <br/> Tulis basic
untuk skill dasar <br/> Tulis intermediate
untuk skill lanjutan <br/> Tulis advance
untuk skill yg dikuasai sepenuhnya <br/>
",

```

```
        autoComplete: "skill"
```

```
    },
```

```

        { label: "Pengalaman kerja", name:
"pengalaman", autoComplete:
"pengalaman" },

```

```

        { label: "Riwayat Pendidikan",
name: "pendidikan", autoComplete:
"pendidikan" },

```

```

        { label: "Sertifikat", name:
"sertifikat", autoComplete: "sertifikat" },

        { label: "Bahasa yg dikuasai",
name: "bahasa", autoComplete:
"bahasa" },

    ];

    return (
        <AuthenticatedLayout>
            <div className=" mt-20 flex
justify-center items-center text-black
dark:text-white">
                <div className="bg-gray-200
dark:bg-gray-800 border border-gray-
800 dark:border-gray-600 rounded-xl p-
5 w-full max-w-3xl">
                    <div>
                        <Head title="Update
Product" />
                        <form
onSubmit={submit}>
                            {[
                                {
                                    label: "Nama",
                                    name: "nama",
                                    type: "text",
                                    autoComplete:
"nama",
                                },
                                {
                                    label: "Email",
                                    name: "email",
                                    type: "email",
                                    autoComplete:
"email",
                                },
                                {
                                    label: "Nomor
WhatsApp atau nomor Handphone",
                                    name: "phone",
                                    type: "number",
                                    autoComplete:
"phone",
                                },
                                {
                                    label: "Linkedln
atau Sosial media lain",
                                    name: "linkedin",
                                    type: "text",
                                    autoComplete:
"linkedin",
                                },
                                {
                                    label: "foto diri",
                                    name: "foto",
                                    type: "file",
                                    autoComplete:
"foto",
                                },
                                {
                                    label: "Alamat",
                                    name: "asal",
                                    type: "text",
                                    autoComplete:
"asal",
                                },
                            ]}
                        </form>
                    </div>
                </div>
            </div>
        )
    )

```

```

                ].map((field, index) =>
                    className="m
(
                    t-2"
                    <div
                        />
                    className="mt-4" key={index}>
                        </div>
                        <InputLabel
                            )))
                            htmlFor={field.
                                name}
                                value={field.lab
                                    <div>
                                        <InputLabel
                                            htmlFor="Alamat Detail" value="Alamat
                                                Detail" />
                                            <select
                                                <select
                                                    defaultValue={pro
                                                        ductData.provinsi}
                                                        onChange={handl
                                                            eChangeProvinsi}
                                                            className="mt-4
                                                                block w-full bg-gray-200 dark:bg-gray-
                                                                    800 border-gray-600 text-gray-800
                                                                        dark:text-gray-200 rounded-md shadow-
                                                                            sm focus:ring focus:ring-indigo-200"
                                                                            ?
                                                                            >
                                                                            {provinsiList.map(
                                                                                (provinsi) => (
                                                                                    <option
                                                                                        key={provins
                                                                                            i.id}
                                                                                            value={provi
                                                                                                nsi.id}
                                                                                                >
                                                                                                {provinsi.na
                                                                                                    me}
                                                                                    </option>
                                                                                )))
                                                                            </select>
                                rs[field.name]}

```

```

<select
    defaultValue={pro
ductData.kota}
    onChange={handl
eChangeKota}
    className="mt-4
block w-full bg-gray-200 dark:bg-gray-
800 border-gray-600 text-gray-800
dark:text-gray-200 rounded-md shadow-
sm focus:ring focus:ring-indigo-200"
>
    {kotaList.map((kot
a) => (
        <option
key={kota.id} value={kota.id}>
            {kota.name}
        </option>
    )}}
</select>

<select
    defaultValue={pro
ductData.kecamatan}
    onChange={handl
eChangeKecamatan}
    className="mt-4
block w-full bg-gray-200 dark:bg-gray-
800 border-gray-600 text-gray-800
dark:text-gray-200 rounded-md shadow-
sm focus:ring focus:ring-indigo-200"
>
    {kecamatanList.m
ap((kecamatan) => (
        <option
            key={kecam
atan.id}
            value={keca
matan.id}
        >
            {kecamatan.
name}
        </option>
    )}}
</select>

<select
    defaultValue={pro
ductData.kelurahan}
    onChange={handl
eChangeKelurahan}
    className="mt-4
block w-full bg-gray-200 dark:bg-gray-
800 border-gray-600 text-gray-800
dark:text-gray-200 rounded-md shadow-
sm focus:ring focus:ring-indigo-200"
>
    {kelurahanList.ma
p((kelurahan) => (
        <option
            key={kelurah
an.id}
            value={kelur
ahan.id}
        >
            {kelurahan.n
ame}
        </option>
    )}}
</select>
</div>

```



```

        {quillFields.map((field,
index) => (
  <div className="mt-4" key={index}>
    <InputLabel
      htmlFor={field.name}
      value={field.label}
    />

    {/* Render deskripsi jika ada */}
    {field.deskripsi && (
      <div className="text-sm text-
gray-500 mt-2">
        {/* Gunakan parse untuk
merender deskripsi HTML */}
        {parse(field.deskripsi)}
      </div>
    )}

    <ReactQuill
      theme="snow"
      value={productData[field.name]}
      onChange={(content) =>
        handleQuillChange(field.name
, content)
      }
      modules={modules}
      formats={formats}
      autoComplete={field.autoComple
te}

      placeholder={`Tulis ${field.label}
di sini...`}

```

```

      className="mt-1 block h-96 w-
full bg-white dark:bg-gray-900 rounded
border dark:border-gray-600"
    />
    <InputError
      message={errors[field.name]}
      className="mt-2"
    />
  </div>
)}}

    <div className="flex
items-center justify-end mt-4 pt-20">
      <PrimaryButton
        className="ms-
4"
        disabled={proces
sing}
      >
        Simpan
      </PrimaryButton>
    </div>
  </form>
</div>
</div>
</div>
</AuthenticatedLayout>
);
};

export default edit;

```

```

import { useState } from "react";

import AuthenticatedLayout from
"@/Layouts/AuthenticatedLayout";

import Button from
"@/Components/Button";

import { router, Link, usePage } from
"@inertiajs/react";

import Swal from "sweetalert2";

const Index = ({ Biodata = [] }) => {
  const { auth } = usePage().props;

  const [isDeleting, setIsDeleting] =
useState(false);

  const defaultImage =
"/image/user.png";

  const Delete = (slug) => {
    Swal.fire({
      title: "Apakah Anda yakin?",
      text: "Anda tidak akan dapat
mengembalikan ini!",
      icon: "warning",
      showCancelButton: true,
      confirmButtonColor: "#3085d6",
      cancelButtonColor: "#d33",
      confirmButtonText: "Ya, hapus
saja!",
    }).then((result) => {
      if (result.isConfirmed) {
        setIsDeleting(true);

        router.delete(route("biodata.de
lete", slug), {
          onSuccess: () => {
            setIsDeleting(false);

```

```

Swal.fire(
  "Dihapus!",
  "Biodata telah
dihapus.",
  "success"
);
},
  onError: () => {
    setIsDeleting(false);

    Swal.fire(
      "Gagal!",
      "Terjadi kesalahan
saat menghapus biodata.",
      "error"
    );
  },
});
}
});
};

return (
  <AuthenticatedLayout>
    <div className="container mx-
auto mt-10 px-4">
      {/* Hero Section */}
      <section className="h-
[50rem] lg:h-[30rem] bg-gradient-to-tr
from-blue-600 via-indigo-500 to-purple-
500 text-white flex flex-col justify-center
items-center relative overflow-hidden
rounded-md">
        <div className="text-
center z-10 max-w-3xl">

```

```
<h1 className="text-2xl
xl:text-5xl font-extrabold drop-shadow-lg
mb-6">
```

Memahami Jenis-Jenis
CV

```
</h1>
```

```
<p className="text-xl
xl:text-2xl mb-8">
```

dan pelajari format CV
yang sesuai dengan kebutuhan

Anda.

```
</p>
```

```
<div className="grid
grid-cols-1 md:grid-cols-3 gap-6 text-left
p-3 md:p-0">
```

```
<div className="bg-
white text-gray-800 rounded-lg shadow-
lg p-6">
```

```
<h3
className="text-lg xl:text-xl font-bold
mb-2">
```

CV Kronologis

```
</h3>
```

```
<p
className="text-gray-600">
```

Menampilkan
pengalaman kerja dan pendidikan

Anda secara urut
berdasarkan waktu.

```
</p>
```

```
</div>
```

```
<div className="bg-
white text-gray-800 rounded-lg shadow-
lg p-6">
```

```
<h3
className="text-lg xl:text-xl font-bold
mb-2">
```

CV Fungsional

```
</h3>
```

```
<p
className="text-gray-600">
```

Fokus pada
keterampilan dan keahlian Anda,
bukan urutan
pengalaman kerja.

```
</p>
```

```
</div>
```

```
<div className="bg-
white text-gray-800 rounded-lg shadow-
lg p-6">
```

```
<h3
className="text-lg xl:text-xl font-bold
mb-2">
```

CV Kombinasi

```
</h3>
```

```
<p
className="text-gray-600">
```

Gabungan antara
format kronologis dan
fungsional, ideal
untuk berbagai profesi.

```
</p>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<div className="absolute
inset-0 bg-gradient-to-b from-
transparent to-black opacity-20"></div>
```

```
</section>
```

```
{/* Tombol Tambah */}
```

```

        <div className="mb-6 flex
justify-end">

        <Link
href={route("biodata.create")}>

        <Button className="bg-
yellow-400 hover:bg-yellow-500 text-
gray-900 font-semibold px-4 py-2
rounded-lg shadow-lg transition
duration-300">

                Tambah Biodata

        </Button>

        </Link>

</div>

    {/* Grid Biodata */}

    {Biodata.length > 0 ? (

        <div className="grid grid-
cols-1 md:grid-cols-2 lg:grid-cols-3 gap-
6">

            {Biodata.map((item,
index) => (

                <div

                    key={index}

                    className="bg-
white dark:bg-gray-800 rounded-lg
shadow-md overflow-hidden transition
transform hover:scale-105 duration-300"

                    >

                        {/* Foto */}

                        <img

                            src={item.foto ? item.foto :
defaultImage}

                            onError={(e) => (e.target.src =
defaultImage)} // Jika gambar rusak,
pakai default

                            alt={item.nama}

```

```

                    className="w-full h-96 object-cover
rounded-t-lg"

                    />

                                {/* Konten Biodata
*/}

                                <div className="p-
6">

                                    <h3

                                        className="text-lg font-semibold text-
gray-800 dark:text-white mb-2">

                                            Nama:

                                            {item.nama}

                                    </h3>

                                    {auth.roles.includ
es("admin") && (

                                        <p

                                            className="text-gray-600 dark:text-
gray-400 text-sm mb-2">

                                                Pemilik CV:

                                                {item.user.name}

                                        </p>

                                        )}

                                        <p

                                            className="text-gray-600 dark:text-
gray-400 text-sm mb-2">

                                                Email:

                                                {item.email}

                                        </p>

                                        <p

                                            className="text-gray-600 dark:text-
gray-400 text-sm mb-3">

                                                Jenis CV:

                                                {item.jenis_CV}

                                        </p>

```

```

        { /* Tombol Aksi */}

        <div
            className="grid grid-cols-2 md:grid-
            cols-4 gap-3 mt-4 justify-items-center">

            <Link

                href={route(
                    "biodata.e
dit",

                    item.slug
                )}

            >

            <Button
                className="bg-yellow-400 hover:bg-
                yellow-500 text-gray-900 font-semibold
                rounded-lg shadow-md px-6 py-2">

                Edit

            </Button>

            </Link>

            <Button

                className=
                "bg-red-500 hover:bg-red-600 text-white
                font-semibold rounded-lg shadow-md
                px-6 py-2"

                onClick={()
=> Delete(item.slug)}

                disabled={is
Deleting}

            >

            {isDeleting

                ?

                "Menghapus..."

                : "Hapus"}

            </Button>

```

```

        <Link

            href={route(
                "biodata.s
how",

                item.slug
            )}

        >

        <Button
            className="bg-blue-500 hover:bg-blue-
            600 text-white font-semibold rounded-lg
            shadow-md px-6 py-2">

            Lihat

        </Button>

    </Link>

    <a

        href={route("
layout", [item.slug])}

        className=
        "bg-green-500 hover:bg-green-600 text-
        white font-semibold rounded-lg shadow-
        md inline-block px-6 py-2"

    >

        Ekspor PDF

    </a>

</div>

</div>

</div>

    )))

</div>

) : (

    <div className="text-
    center mt-10">

```

```

        <h2 className="text-2xl
font-semibold text-gray-700 dark:text-
gray-300">

```

Tidak ada Biodata

```

        </h2>

```

```

        <p className="text-
gray-500 dark:text-gray-400 mt-2">

```

Silakan tambahkan biodata baru menggunakan tombol di

atas.

```

        </p>

```

```

    </div>

```

```

  })

```

```

</div>

```

```

</AuthenticatedLayout>

```

```

);

```

```

};

```

```

export default Index;

```

```

import { useState } from "react";

```

```

import { Link, usePage } from
"@inertiajs/react";

```

```

import AuthenticatedLayout from
"@/Layouts/AuthenticatedLayout";

```

```

import Button from
"@/Components/Button";

```

```

import parse from "html-react-parser";

```

```

import html2canvas from "html2canvas";

```

```

const show = ({ Biodata }) => {

```

```

  const { auth } = usePage().props;

```

```

    const defaultImage =
"/image/user.png";

```

```

    return (

```

```

      <AuthenticatedLayout>

```

```

        <div className="container mx-
auto px-4 py-8">

```

```

          { /* Hero Section */ }

```

```

          <div className=" bg-
gradient-to-tr from-indigo-700 via-purple-
500 to-pink-500 text-white rounded-lg
shadow-lg overflow-hidden mb-8">

```

```

            <div className="flex flex-
col md:flex-row items-center p-6">

```

```

              <div className="w-48 h-
48 md:w-64 md:h-64 mb-4 md:mb-0">

```

```

                <img

```

```

                  src={

```

```

                    Biodata.foto

```

```

                    ?

```

```

                    `/storage/${Biodata.foto.replace(

```

```

                      "public/",

```

```

                      ""

```

```

                    ))`

```

```

                    : defaultImage

```

```

                }

```

```

                alt="user"

```

```

                className="w-full
h-full object-cover rounded-full shadow-
md"

```

```

              />

```

```

            </div>

```

```

            <div className="text-
center md:text-left md:ml-6 flex-1">

```

```

        <h1 className="text-
2xl font-bold text-white mb-2">
        BIODATA:
        {Biodata.nama}

        </h1>

        <p className="text-
white text-lg md:text-xl mb-4">

        Profil lengkap
        biodata {Biodata.nama}.

        </p>

        {/* Dropdown & Tombol
Export */}

        <div className="flex
flex-col md:flex-row items-center gap-
4">

        <a href={route("layo
ut", [Biodata.slug])}

        className="bg-
yellow-400 hover:bg-yellow-500 text-
gray-900 font-semibold py-2 px-4
rounded-lg transition duration-200"

        >

        Download CV

        </a>

        </div>

        </div>

        </div>

        </div>

        {/* Informasi Biodata */}

        <div

```

```

        className="bg-white
dark:bg-gray-800 rounded-lg shadow-lg
overflow-hidden p-6"

        >

        <div className="grid grid-
cols-1 md:grid-cols-2 gap-6">

        {/* Nama */}

        <div className="border
dark:border-gray-700 rounded-lg p-4">

        <h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">

        Nama:

        </h3>

        <p className="text-
gray-600 dark:text-gray-400">

        {Biodata.nama}

        </p>

        </div>

        {/* Email */}

        <div className="border
dark:border-gray-700 rounded-lg p-4">

        <h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">

        Email:

        </h3>

        <p className="text-
gray-600 dark:text-gray-400">

        {Biodata.email}

        </p>

        </div>

```

```
        {/* Nomor Telepon */}
        <div className="border
dark:border-gray-700 rounded-lg p-4">
            <h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">
```

Nomor Telepon:

```
</h3>
```

```
<p className="text-
gray-600 dark:text-gray-400">
```

```
{Biodata.phone}
```

```
</p>
```

```
</div>
```

```
{/* Domisili Sekarang */}
```

```
<div className="border
dark:border-gray-700 rounded-lg p-4">
```

```
<h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">
```

Alamat:

```
</h3>
```

```
<p className="text-
gray-600 dark:text-gray-400">
```

```
{Biodata.lokasi}
```

```
</p>
```

```
</div>
```

```
{/* Asal Daerah */}
```

```
<div className="border
dark:border-gray-700 rounded-lg p-4">
```

```
<h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">
```

Alamat Detail:

```
</h3>
```

```
<p className="text-
gray-600 dark:text-gray-400">
```

```
{Biodata.asal}
```

```
</p>
```

```
</div>
```

```
{/* Skill */}
```

```
<div className="border
dark:border-gray-700 rounded-lg p-4">
```

```
<h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">
```

Skill (Keahlian):

```
</h3>
```

```
<p className="text-
gray-600 dark:text-gray-400">
```

```
{parse(Biodata?.skill
?? "Data belum diisi")}
```

```
</p>
```

```
</div>
```

```
{/* Profil Singkat */}
```

```
<div className="border
dark:border-gray-700 rounded-lg p-4
md:col-span-2">
```

```
<h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">
```

Profil Singkat:

```
</h3>
```

```
<p className="text-
gray-600 dark:text-gray-400">
```

```
{parse(Biodata?.prof
il_singkat ?? "Data belum diisi")}
```

```
</p>
```



```

</div>

{ /* Pengalaman Kerja */ }

<div className="border
dark:border-gray-700 rounded-lg p-4
md:col-span-2">

  <h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">

    Pengalaman Kerja:

  </h3>

  <p className="text-
gray-600 dark:text-gray-400">

    {parse(Biodata?.pen
galaman ?? "Data belum di isi")}

  </p>

</div>

{ /* Riwayat Pendidikan */ }

<div className="border
dark:border-gray-700 rounded-lg p-4
md:col-span-2">

  <h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">

    Riwayat Pendidikan:

  </h3>

  <p className="text-
gray-600 dark:text-gray-400">

    {parse(Biodata?.pen
didikan ?? "Data belum di isi")}

  </p>

</div>

```

```

<div className="border
dark:border-gray-700 rounded-lg p-4
md:col-span-2">

  <h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">

    Sertifikat:

  </h3>

  <p className="text-
gray-600 dark:text-gray-400">

    {parse(Biodata?.sert
ifikat ?? "Data belum diisi")}

  </p>

</div>

<div className="border
dark:border-gray-700 rounded-lg p-4
md:col-span-2">

  <h3 className="text-
gray-800 dark:text-gray-200 font-
semibold">

    Bahasa yg dikuasai:

  </h3>

  <p className="text-
gray-600 dark:text-gray-400">

    {parse(Biodata?.bah
asa ?? "Data belum diisi")}

  </p>

</div>

</div>

</div>

</AuthenticatedLayout>

);

};

```

```
export default show;
```

```
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 } from "@inertiajs/react";

import { IconBrandWhatsapp } from
"@tabler/icons-react";
```

```
export default function CheckEmail() {

  const [email, setEmail] = useState("");

  const [errors, setErrors] =
useState({});

  const handleSubmit = (e) => {

    e.preventDefault();

    router.post(

      "/check-email",

      { email },

      {

        onError: (err) =>
setErrors(err),

      }

    );

  };

};
```

```
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/628820  
19843866?text=Masukkan%20nama%2  
0dan%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>
);
}

```

```

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={showPasswor
d ? "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="Masuk
kan password baru"

    />

    {/* Toggle Password
Visibility */}

    <button

      type="button"
```

```
className="absolut
e 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
_confirmation"

    value="Konfirmasi
Password Baru"

  />

  <TextInput

    id="password_confir
mation"

    type={showPasswor
d ? "text" : "password"}

    name="password_c
onfirmation"

    value={passwordCo
nfirmation}

    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="absolut
e right-3 top-1/2 transform -translate-y-
1/2 text-gray-500"

        onClick={() =>
setShowPassword(!showPassword)}

        >

            {showPassword ?
"Hide" : "Show"}

        </button>

        <InputError

            message={errors.pa
ssword_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>

);

}

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: "#d333",

            cancelButtonColor: "#3085d6",

            confirmButtonText: "Ya, hapus!",

        }).then((result) => {

            if (result.isConfirmed) {

                destroy(route("admin.destroy",
                id), {

                    onSuccess: () => {

                        Swal.fire({

                            title: "Terhapus!",

                            text: "Data pengguna
                            berhasil dihapus.",

                            icon: "success",

                        });

                    },

                    onError: () => {

                        Swal.fire({

                            title: "Error!",

```

```

                            text: "Terjadi
                            kesalahan saat menghapus pengguna.",

                            icon: "error",

                        });

                    },

                });

            }

        });

    };

    const handleSubmit = (e) => {

        e.preventDefault();

        if (!selectedUser || !selectedUser.id)
        {

            setErrors({ user_id: "Silakan pilih
            pengguna terlebih dahulu" });

            return;

        }

        router.post('/update-password', {

            user_id: selectedUser.id, //
            Pastikan hanya mengirim ID, bukan
            seluruh objek

            password,

            password_confirmation:
            passwordConfirmation,

        }, {

            onError: (err) => setErrors(err),

        });

    };

```

```

return (
  <AuthenticatedLayout>

    <div className="container mx-
auto px-4 py-8 text-gray-900 dark:text-
gray-200">

      <section className="bg-
gradient-to-tr from-indigo-500 to-purple-
500 text-white py-12 rounded-lg
shadow-lg mb-8 text-center">

        <h1 className="text-4xl
font-bold mb-4">Selamat Datang di
Manajemen Pengguna</h1>

        <p className="text-
lg">Kelola semua pengguna dengan
mudah dan efisien.</p>

      </section>

      {success && (
        <div className="mb-6 bg-
green-100 text-green-700 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-lg shadow-lg p-4 flex flex-col
items-center text-center">

            <img src={image}
alt={u.name} className="w-24 h-24
rounded-full mb-4 shadow-md" />

```

```

          <h2 className="text-
lg font-semibold">{u.name}</h2>

          <p className="text-
gray-600 text-sm mb-4">{u.email}</p>

          <button onClick={() =>
setSelectedUser(u)} className="bg-
indigo-500 text-white py-2 px-4 rounded-
lg shadow hover:bg-indigo-600 flex
items-center justify-center">

            <IconKey
className="mr-2" /> Ganti Password

          </button>

          <button onClick={() =>
handleDelete(u.id)} className="bg-red-
500 text-white py-2 px-4 rounded-lg
shadow hover:bg-red-600 flex items-
center justify-center mt-2">

            <IconTrash
className="mr-2" /> Hapus

          </button>

        </div>

      )})

    </div>

    {selectedUser && (
      <div className="fixed
inset-0 flex items-center justify-center
bg-gray-900 bg-opacity-50">

        <div className="bg-
white dark:bg-gray-800 p-6 rounded-lg
shadow-lg w-96">

          <h2 className="text-
xl font-semibold mb-4">Ganti Password
{selectedUser.name}</h2>

          {/* 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="Masuk
                kan password baru"

            />

            {/* Toggle Password
            Visibility */}

            <button

                type="button"

                className="absolut
                e 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
                _confirmation"

                value="Konfirmasi
                Password Baru"

            />

            <TextInput

                id="password_confir
                mation"

                type={showPassword ? "text" : "password"}

                name="password_c
                onfirmation"

                value={passwordCo
                nfirmation}

                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="absolut
                e right-3 top-1/2 transform -translate-y-
                1/2 text-gray-500"

```

```

        onClick={() =>
setShowPassword(!showPassword)}
        >
        {showPassword ?
"Hide" : "Show"}
    </button>
    <InputError
        message={errors.pa
ssword_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>
</div>
</div>
    )}
</div>
</AuthenticatedLayout>
    );
}

```

