INNOVATIVE: Journal Of Social Science Research
Volume 5 Nomor 2 Tahun 2025 Page 1482-1499
E-ISSN 2807-4238 and P-ISSN 2807-4246
Website: https://j-innovative.org/index.php/Innovative

Aplikasi System Informasi Pengolahan Data Penduduk Berbasis Web (Studi Kasus Desa Aek Nabara, Kec. Simangumban, Kab. Tapanuli Utara)

Swandi Tambunan[1✉], Vitri Tundjungsari[2]
Universitas Esa Unggul
Email: swandytambunan@gmail.com[1✉]

Abstrak

Aplikasi sistem informasi pengolahan data penduduk berbasis website studi kasus Desa Aek Nabara kecamatan simangumban kabupaten tapanuli utara. Penelitian ini dilakukan untuk merancang dan membangun aplikasi sistem informasi pengolahan data penduduk pada Desa Aek Nabara, yang dimana hingga saat ini pelayanan pada kantor desa masih menggunakan sistem manual yaitu petugas kantor desa mendatangi rumah warga untuk melakukan pendataan sehingga waktu yang diperlukan menjadi tidak efisien (proses pendataan data-data penduduk seperti data kelahiran, data kematian, data pendatang, dan serta laporan data penduduk yang belum ada. Penelitian ini bertujuan agar dapat memudahkan pegawai kantor kepala desa dalam melakukan proses pendataan penduduk dan laporan bulanan. metode pengembangan perangkat lunak yang digunakan adalah metode waterfall. Teknik pengumpulan data pada penelitian ini menggunakan wawancara, observasi dan studi pustaka. Penelitian ini juga dapat menghasilkan sebuah sistem informasi pendataan penduduk yang dapat menyajikan informasi secara cepat dan efisien serta dapat menyajikan laporan yang diperlukan.

Kata Kunci: *Sistem Informasi, Pengelohan, Data Penduduk, Website, Waterfall*

Abstract

Information system application for processing population data based on the case study website in Aek Nabara Village, Simangumban District, North Tapanuli Regency. This research was conducted to design and build an information system application for processing population data in Aek Nabara Village, where until now services at the village office are still using a manual system, namely village office workers visiting residents' homes to collect data so that the time needed becomes inefficient (process data collection of population data such as birth data, death data, migrant data, and as well as population data reports that do not yet exist.This study aims to make it easier for village head office employees to carry out the population data collection process and monthly reports.The software development method used is waterfall method.Data collection techniques in this study used interviews, observation and literature study.This research can also produce a population data collection information system that can present information quickly and efficiently and can present the necessary report.
Keywords: *Information System, Processing, Population Data, Website, Waterfall*

PENDAHULUAN

Dalam Perkembangan teknologi saat ini semakin pesat. Dengan kemajuan teknologi saat ini, kita dapat membuat tugas menjadi lebih mudah. Kita dapat menemukan informasi yang kita butuhkan dan memperluas jaringan komunikasi kita dengan kemajuan teknologi. Salah satu teknologi yang paling banyak digunakan saat ini adalah komputer. Komputer telah banyak digunakan dalam membantu pekerjaan manusia terutama dalam pengolaghan data (Haswan dkk., 2018).

Kantor pemerintahan tingkat paling bawah adalah Kantor Lurah/ Kepala Desa yang dimana terselenggaranya pusat pelayanan pemerintahan desa yang tidak terlepas dari kepentingan masyarakat. Kantor desa merupakan suatu instansi pemerintahan yang melakukan pengolahan data kependudukan khususnya dalam pembuatan Kartu Tanda Penduduk (KTP), Kartu Keluarga (KK), Akte kelahiran, surat kematian, surat keterangan pendatang, dan surat keterangan pindahan penduduk.(Tarina, 2015).

Kantor Desa Aek Nabara merupakan salah satu cabang Instansi Pemerintah yang terdapat di Kecamatan Simangumban, Kabupaten Tapanuli Utara. Pelayanan di Kantor Desa Aek Nabara dalam pendataan dan kepengurusan masih dilakukan dengan cara manual dimana petugas kantor kepala desa mendatangi rumah - rumah warga untuk melakukan pendataan sehingga waktu yang diperlukan menjadi kurang efisien dan timbul masalah kehilangan data (Arifin dkk., t.t.).

Adapun tujuan pada penelitian ini yaitu untuk menghasilkan program aplikasi sistem informasi pengolahan data penduduk pada Desa Aek Nabara Kecamatan Simangumban.

METODE PENELITIAN

Dalam pengumpulan data yang digunakan untuk mengumpulkan informasi terkait laporan perancangan sistem pengolahan data penduduk di Desa Aek Nabara, peneliti menggunakan beberapa metode antara lain Studi Literatur, Observasi, dan Wawancara. Tujuan dari metode yang digunakan oleh peneliti dalam pengumpulan data yaitu untuk mendapatkan informasi yang akurat dan terpercaya supaya dapat digunakan sebagai dasar analisis dan penarikan kesimpulan. Menggunakan metode pengumpulan data studi literatur sangat relevan digunakan dengan topik penelitian atau studi yang sedang dilakukan saat ini dengan sumber – sumber yang dapat ditemukan dari buku, jurnal ilmiah, artikel, makalah, dan melalui internet.

HASIL DAN PEMBAHASAN

Metode Analisa Masalah (PIECES)

Menganalisa identifikasi permasalahan untuk membangun sistem dilakukan dengan spesifik, ini adalah beberapa aspek sehingga terciptanya *sistem* antara lain (*Perfomance, Information, Economy, Control, Eficiency and service*) Analisis PIECES ini sangat penting untuk membangun *sistem* sehingga tercapainya tujuan yang di inginkan.

Table 1. Metode Analisa Masalah (PIECES)

No	Kategori	Sistem Berjalan	Sistem Usulan
1	*Perfomance* (Kinerja)	Proses berjalan saat ini pada pendataan penduduk di desa aek nabara masih lambat dalam pengelolaan data kependudukan sehingga petugas harus merekap dari buku registrasi yang nantinya akan dijadikan acuan dalam pembuatan data kependudukan. Misalnya pada kecepatan pemrosesan data, seperti waktu yang dibutuhkan untuk mencatat atau memperbaharui informasi penduduk.	Membuat *website* agar penduduk desa bisa melihat pengumuman, membuat laporan, dan membuat kritik dan saran kepada kepala desa setempat.
2	*Information* (Informasi)	Dalam prosedur penginputan data – data warga masih tergolong manual jadi masih membutuhkan waktu lama, dan akurasi data penduduk yang dihasilkan dari system manual juga masih kurang akurat.	Kepala desa dapat menyebarkan pengumuman kepada penduduk sekitar dan mencari data laporan masuk, kritik dan saran di dalam aplikasi *website* secara mudah dan cepat.

3	*Economy* (Ekonomi)	Dalam segi ekonomi ada biaya yang terlibat dalam pengolahan data penduduk yang dilakukan secara manual antara lain termasuk biaya tenaga kerja, kertas, dan peralatan. Biaya tenaga kerja yang dimaksud yaitu Ketika perangkat desa mendatangi rumah – rumah warga untuk mendata ulang warganya. Sehingga dibutuhkan perubahan system yang lebih efisien yang dapat mengurangi pengeluaran, maka dari itu dengan adanya system yang diusulkan diharapkan tingkat efektif dan efesiensi biaya untuk masa yang akan datang lebih baik.	Pemilik dapat melihat laporan harian tanpa harus turun langsung kelapangan.
4	*Control* (Kontrol)	Masih banyak ditemukan kertas – kertas yang ditemukan pada system berjalan saat ini seperti setiap pembuatan surat, warga masih harus membawa fotocopyan KTP, KK, dan lain - lain. Sehinggga menimbulkan penumpukan kertas dikantor desa. Dan dalam proses berjalan saat ini berkas maupun dokumen warga desa masih berbentuk kertas yang dimana keamanan tersebut kurang aman dan gampang tercecer.	Pencatatan data disimpan pada database sehingga dapat di cek langsung melalui aplikasi yang sudah terhubung ke database.
5	*Efficiency* (Efisiensi)	Permasalahan dalam proses berjalan saat ini yaitu dalam pendataan penduduk yang dimana masih dilakukan secara manual jadi proses menginput, mengolah data, dan mendapatkan informasi masih membutuhkan waktu yang lama. Maka dari itu system yang diusulkan dapat mempermudah dalam pencarian informasi data penduduk dan lebih efisien	Pengecekan data laporan, kritik dan saran dapat dilakukan melalui database sehingga untuk pengecekan data lebih mudah dan lebih tersusun rapih untuk melakukan report harian.
6	*Service* (Layanan)	Pada aspek ini permasalahan yang timbul yaitu Ketika pencarian suatu berkas atau dokumen masyarakat yang tercecer dalam bentuk kertas yang dimana dapat menyulitkan petugas dalam mengelola atau menginput data penduduk yang sewaktu – waktu diperlukan. Dengan adanya system yang diusulkan ini pendataan penduduk dilakukan secara komputerisasi dengan suatu system pengolahan data kependudukan yang baik dan dapat mempermudah petugas kantor desa dalam memberikan informasi secara lengkap dan akurat dari data kependudukan.	Memberikan akses secara langsung pada penduduk setempat untuk mengecek pengumuman, membuat laporan masuk, membuat kritik dan saran untuk pemerintah setempat.

Analisa Kebutuhan Sistem (*Fungsional dan Non-Fungsional*)

Analisis kebutuhan *fungsional* merupakan gambaran dari proses-proses mengenai sistem yang berjalan pada sistem ini. Pada dasarnya, ada tiga hal yang dikerjakan sistem ini, menerima data pengiriman, mengolah data pengiriman dan mengeluarkan respon hasil pengiriman (pembukuan).

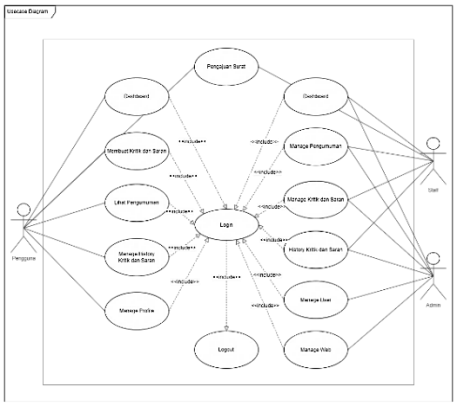
Tabel 2. Analisis Kebutuhan Fungsional

No	Peran	Deskripsi
1	Kepala Desa	• Login • Dashboard • Melihat pemberitahuan atau pengumuman • Melihat laporan masuk • Melihat kritikan dan saran masuk • Logout
2	Admin	• Login • Dashboard • Mengelola pemberitahuan atau pengumuman • Mengelola laporan masuk • Mengelola kritikan dan saran masuk • Logout
3	Pengguna atau Penduduk Desa	• Register • Login • Dashboard • Melihat pemberitahuan atau pengumuman • Membuat laporan • Melihat laporan masuk • Membuat kritikan dan saran masuk • Logout

Analisis kebutuhan *non-fungsional* yang dimaksud disini yaitu analisis kebutuhan perangkat lunak. Analisis perangkat lunak dibutuhkan untuk mengetahui apa yang di butuhkan untuk membangun sistem.

Tahap Perancangan Sistem

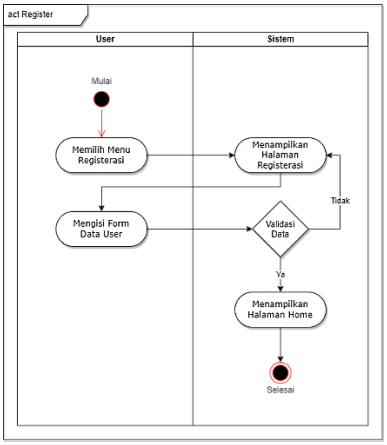
Use Case Diagram



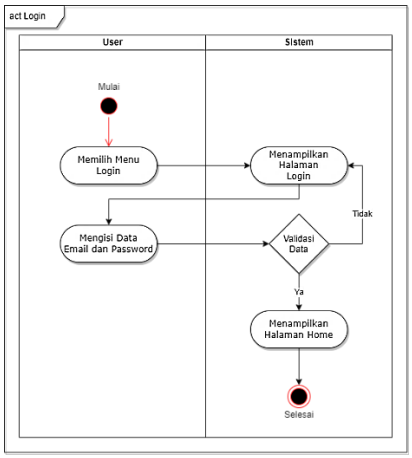
Gambar 1. Use Case Diagram

Activity Diagram

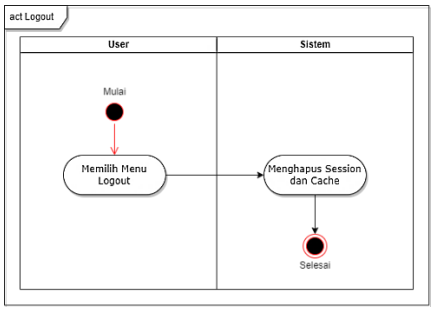
Activity Diagram merupakan rancangan aliran aktivitas atau aliran kerja dalam sebuah sistem yang akan dijalankan. *Activity Diagram* juga digunakan untuk mendefinisikan atau mengelompokan aluran tampilan dari sistem tersebut. *Activity Diagram* memiliki komponen dengan bentuk tertentu yang dihubungkan dengan tanda panah. Panah tersebut mengarah ke-urutan aktivitas yang terjadi dari awal hingga akhir. Adapun *Activity Diagram* usulan antara lain:



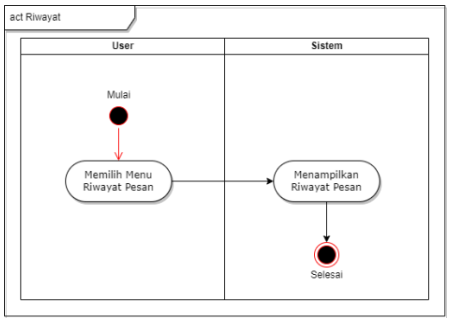
Gambar 2. Activity Diagram – Register



Gambar 3. Activity Diagram – Login

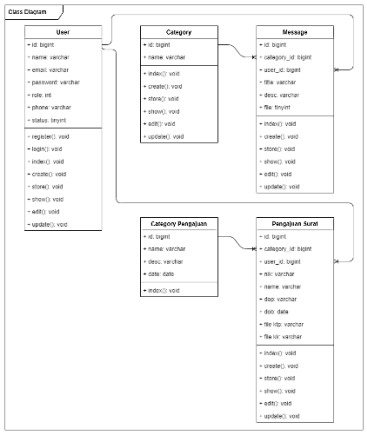


Gambar 4. Activity Diagram – Logout



Gambar 5. Activity Diagram – Riwayat Pesan

Class Diagram



Gambar 6. Class Diagram

Tahap Implementasi

1. *Library Object Oriented (Package OPP)*

Framework ini memiliki banyak pilihan library object oriented yang tidak dimiliki oleh framework lain, hal ini akan sangat berguna untuk membangun aplikasi yang kompleks. Berikut adalah library atau package yang digunakan dalam pengembangan aplikasi:

a. *Laravel UI*

Laravel UI digunakan untuk membuat beberapa fitur yang berkaitan dengan autentifikasi secara otomatis pada aplikasi yang sudah memiliki pengaturan bawaan yang akan dimasukan kedalam Route, Controller, Model, Migration, dan View. Fitur yang biasa digunakan dalam pengembangan aplikasi, yaitu: *Login*, Register, dan Logout.

```
protected $redirectTo = RouteServiceProvider::HOME;

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

/**
 * Get a validator for an incoming registration request.
 *
 * @param  array  $data
 * @return \Illuminate\Contracts\Validation\Validator
 */
protected function validator(array $data)
{
    return Validator::make($data, [
        'name' => ['required', 'string', 'max:255'],
        'username' => ['required', 'string', 'max:255', 'unique:users'],
        'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
        'password' => ['required', 'string', 'min:8', 'confirmed'],
    ]); <- #52-57 return Validator::make
} <- #51-58 protected function validator(array $data)

/**
 * Create a new user instance after a valid registration.
 *
 * @param  array  $data
 * @return \App\Models\User
 */
protected function create(array $data)
{
    return User::create([
        'name' => $data['name'],
        'username' => $data['username'],
        'email' => $data['email'],
        'address' => $data['address'],
        'phone' => $data['phone'],
        'password' => Hash::make($data['password']),
        'email_verified_at' => now(),
    ])->syncRoles([3]); <- #68-76 return User::create
} <- #67-77 protected function create(array $data)
<- #13-78 class RegisterController extends Controller
```

Gambar 7. Script Laravel UI

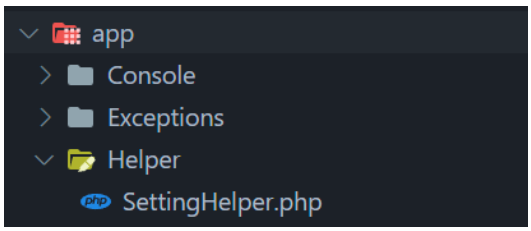
b. *Laravel Permission*

Laravel Permission adalah package atau paket untuk membuat pembatasan hak akses pengguna pada aplikasi. Laravel Permission terdapat 2 jenis pembatasan, yaitu: Role dan Permission. Role dapat diartikan sebagai level atau jabtan yang dimiliki sedangkan Permission dapat diartikan sebagai pembatasan pengguna pada setiap halaman. Pengguna dapat dibatasi hak aksesnya secara individual pada setiap halaman menggunakan permission atau dapat dikelompokan hak aksesnya sesuai dengan Role yang dimilikinya. Pada aplikasi ini pembatasan yang digunakan adalah Role dan Permission, Role yang telah ditetapkan ada 3, yaitu: Kepala Desa, Admin, dan Pengguna, sedangankan untuk Permission setiap Role, yaitu :

1) Kepala Desa : dashboard, melihat pengumuman, melihat laporan masuk dan melihat kritik dan saran
2) Admin : Dashboard, Mengelola Pengumuman, Mengelola Laporan Masuk dan Mengelola Kritik dan Saran
3) Pengguna atau Penduduk Desa: Melihat Pengumuman, Membuat Laporan, Melihat Laporan Pengguna Lain dan Membuat Kritik dan Saran

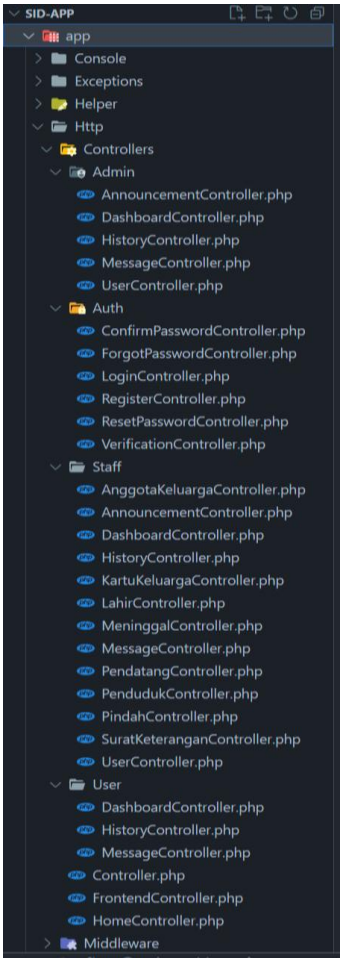
2. Struktur Folder

a. Folder Helper



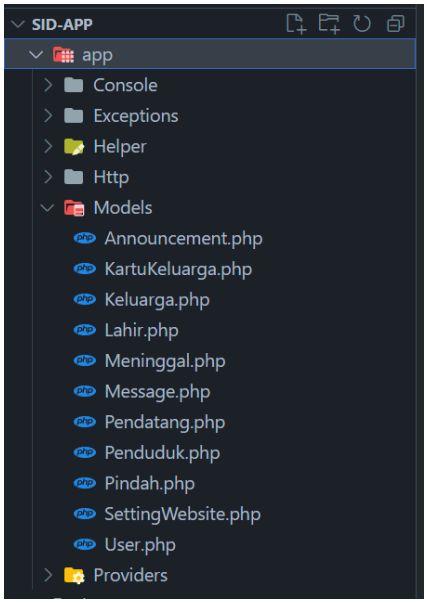
Gambar 8. Folder Helper

b. Folder Controller



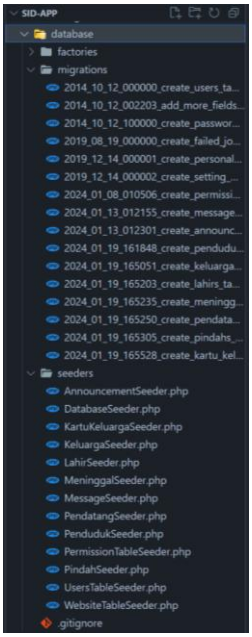
Gambar 9. Folder Controller

c. Folder Model



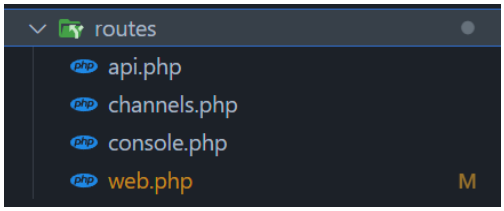
Gambar 10. Folder Model

d. Folder Database



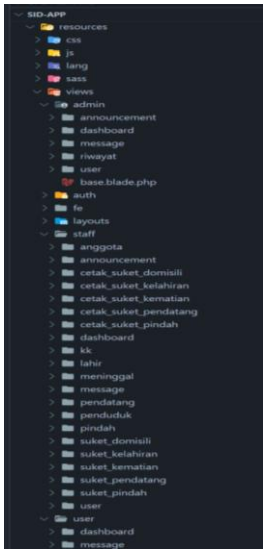
Gambar 11. Folder Database

e. Folder Routes



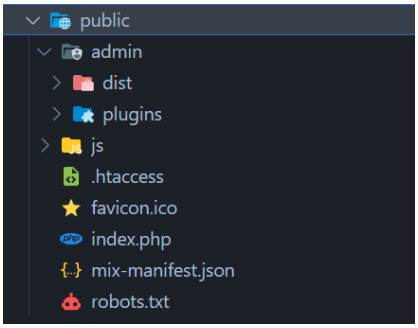
Gambar 12. Folder Routes

f. Folder Resources



Gambar 13. Folder Resources

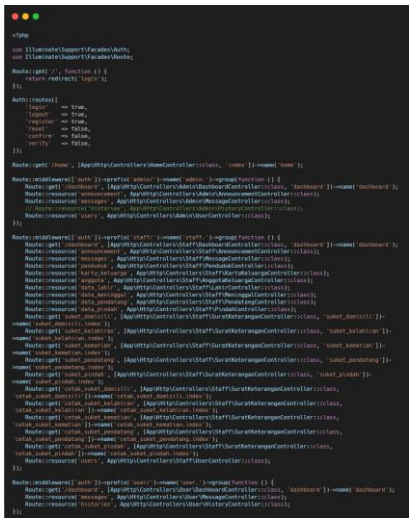
g. Folder Assets



Gambar 14. Folder Assets

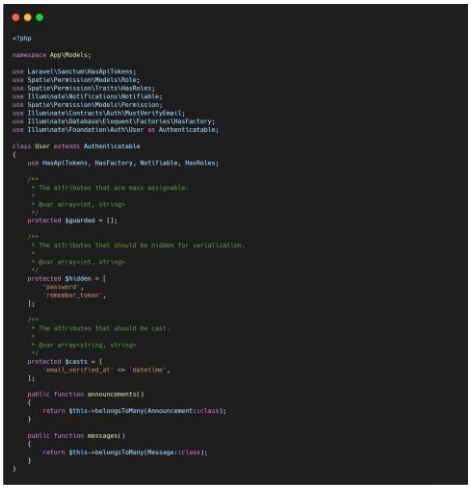
3. Script (source Code)

a. Script Routes



Gambar 15. Script Routes

b. Script Models

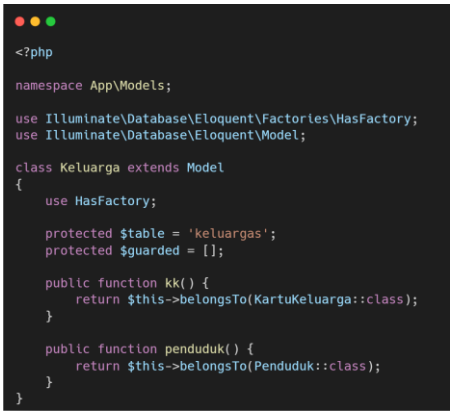


```php
<?php

namespace App\Models;

use Laravel\Sanctum\HasApiTokens;
use Spatie\Permission\Models\Role;
use Spatie\Permission\Traits\HasRoles;
use Illuminate\Notifications\Notifiable;
use Spatie\Permission\Models\Permission;
use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;

class User extends Authenticatable
{
    use HasApiTokens, HasFactory, Notifiable, HasRoles;

    /**
     * The attributes that are mass assignable.
     *
     * @var array<int, string>
     */
    protected $guarded = [];

    /**
     * The attributes that should be hidden for serialization.
     *
     * @var array<int, string>
     */
    protected $hidden = [
        'password',
        'remember_token',
    ];

    /**
     * The attributes that should be cast.
     *
     * @var array<string, string>
     */
    protected $casts = [
        'email_verified_at' => 'datetime',
    ];

    public function announcements()
    {
        return $this->belongsToMany(Announcement::class);
    }

    public function messages()
    {
        return $this->belongsToMany(Message::class);
    }
}
```

Gambar 16. Script Models Users

```php
<?php

namespace App\Models;

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

class Penduduk extends Model
{
    use HasFactory;

    protected $table = 'penduduks';
    protected $guarded = [];

    public function kk() {
        return $this->belongsTo(KartuKeluarga::class);
    }

    public function author() {
        return $this->belongsTo(User::class);
    }
}
```

Gambar 17. Script Models Penduduk



```php
<?php

namespace App\Models;

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

class Keluarga extends Model
{
    use HasFactory;

    protected $table = 'keluargas';
    protected $guarded = [];

    public function kk() {
        return $this->belongsTo(KartuKeluarga::class);
    }

    public function penduduk() {
        return $this->belongsTo(Penduduk::class);
    }
}
```

Gambar 18. Script Models Keluarga

```php
<?php

namespace App\Models;

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

class KartuKeluarga extends Model
{
    use HasFactory;

    protected $table = 'kartu_keluargas';
    protected $guarded = [];

    public function penduduk() {
        return $this->belongsTo(Penduduk::class);
    }

    public function author() {
        return $this->belongsTo(User::class);
    }
}
```

Gambar 19. Script Models Kartu Keluarga

c. Script Controllers

```php
<?php

namespace App\Http\Controllers\Admin;

use App\Models\User;
use Illuminate\Http\Request;
use App\Http\Controllers\Controller;
use App\Models\Announcement;
use App\Models\Message;

class DashboardController extends Controller
{
    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('permission:dashboard-admin-C', ['only' =>
['dashboard','index','list','show']]);
        $this->middleware('permission:dashboard-admin-R', ['only' => ['create','store']]);
        $this->middleware('permission:dashboard-admin-U', ['only' => ['edit','update']]);
        $this->middleware('permission:dashboard-admin-D', ['only' => ['destroy']]);
    }

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function dashboard()
    {
        $data['title']          = 'Dashboard';
        $data['users']          = User::count();
        $data['announcements']  = Announcement::count();
        $data['critics']        = Message::where('category', 'kritik')->count();
        $data['recommendation'] = Message::where('category', 'saran')->count();
        return view('admin.dashboard.index', $data);
    }
}
```

Gambar 20. Script Controllers Dashboard Admin

```php
<?php

namespace App\Http\Controllers\Staff;

use App\Models\User;
use App\Models\Message;
use App\Models\Announcement;
use App\Http\Controllers\Controller;
use App\Models\KartuKeluarga;
use App\Models\Meninggal;
use App\Models\Pendatang;
use App\Models\Penduduk;
use App\Models\Pindah;

class DashboardController extends Controller
{
    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('permission:dashboard-staff-C', ['only' =>
['dashboard','index','list','show']]);
        $this->middleware('permission:dashboard-staff-R', ['only' => ['create','store']]);
        $this->middleware('permission:dashboard-staff-U', ['only' => ['edit','update']]);
        $this->middleware('permission:dashboard-staff-D', ['only' => ['destroy']]);
    }

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function dashboard()
    {
        $data['title']          = 'Dashboard';
        $data['users']          = User::count();
        $data['announcements']  = Announcement::count();
        $data['critics']        = Message::where('category', 'kritik')->count();
        $data['recommendation'] = Message::where('category', 'saran')->count();
        $data['penduduk']       = Penduduk::count();
        $data['kk']             = KartuKeluarga::count();
        $data['lk']             = Penduduk::count();
        $data['pr']             = Penduduk::count();
        $data['lahir']          = Penduduk::count();
        $data['meninggal']      = Meninggal::count();
        $data['pendatang']      = Pendatang::count();
        $data['pindah']         = Pindah::count();
        return view('staff.dashboard.index', $data);
    }
}
```

Gambar 21. Script Controllers Dashboard Staff

```php
<?php

namespace App\Http\Controllers\User;

use App\Models\User;
use App\Models\Message;
use App\Models\Announcement;
use Illuminate\Http\Request;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Auth;

class DashboardController extends Controller
{
    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('permission:dashboard-user-C', ['only' =>
['dashboard','index','list','show']]);
        $this->middleware('permission:dashboard-user-R', ['only' => ['create','store']]);
        $this->middleware('permission:dashboard-user-U', ['only' => ['edit','update']]);
        $this->middleware('permission:dashboard-user-D', ['only' => ['destroy']]);
    }

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function dashboard()
    {
        $data['title']          = 'Dashboard';
        $data['users']          = User::count();
        $data['announcement']   = Announcement::where('id',1)->first();
        $data['announcements']  = Announcement::count();
        $data['critics']        = Message::where('category', 'kritik')->where('author_id',
Auth::user()->id)->count();
        $data['recommendation'] = Message::where('category', 'saran')->where('author_id',
Auth::user()->id)->count();
        return view('user.dashboard.index', $data);
    }
}
```

Gambar 22. Script Controllers Dashboard User

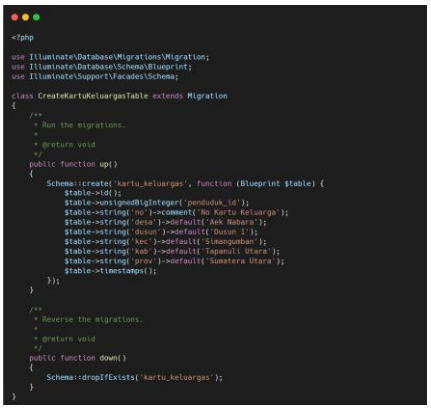
d. Script Database

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateUsersTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }

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

Gambar 23. Script Database Users

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreatePendudukTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('penduduks', function (Blueprint $table) {
            $table->id();
            $table->string('nik');
            $table->string('name');
            $table->string('bop')->comment('Tempat Lahir');
            $table->date('bod')->comment('Tanggal Lahir');
            $table->enum('gender', ['LK','PR']);
            $table->string('village')->comment('Desa');
            $table->string('hamlet')->comment('Dusun');
            $table->string('religion')->comment('Agama');
            $table->string('kawin');
            $table->string('job')->comment('Pekerjaan');
            $table->enum('status',['Ada','Meninggal','Pindah'])->comment('Status Nyawa');
            $table->timestamps();
        });
    }

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

Gambar 24. Script Database Penduduk



```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateKartuKeluargasTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('kartu_keluargas', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('penduduk_id');
            $table->string('no')->comment('No Kartu Keluarga');
            $table->string('desa')->default('Aek Nabara');
            $table->string('dusun')->default('Dusun 1');
            $table->string('kec')->default('Simangumban');
            $table->string('kab')->default('Tapanuli Utara');
            $table->string('prov')->default('Sumatera Utara');
            $table->timestamps();
        });
    }

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

Gambar 25. Script Database Kartu Keluarga

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateKeluargasTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('keluargas', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('kk_id');
            $table->unsignedBigInteger('penduduk_id');
            $table->string('hubungan');
            $table->timestamps();
        });
    }

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

Gambar 26. Script Database Keluarga

Tahap Pengujian

Black-box Testing

Black-Box Testing ini memiliki tujuan untuk membuktikan fungsionalitas yang ada pada aplikasi, sehingga tidak adanya fitur yang error atau tidak berjalan sesuai dengan yang di harapkan. Berikut tabel hasil dari *Black-Box Testing*:

Tabel 3. Hasil Black-Box Testing

No	Menu	Sekenario Pengujian	Hasil yang diharapkan	Hasil Pengujian	
				Sesuai	Tidak
		Admin			
1	*Login*	1. Mengisikan data *customer* 2. Menakan tombol *login*.	Tidak ada masalah saat melakukan *login*.	✓	
2	*Logout*	1. Menakan tombol *logout*.	Tidak ada masalah saat melakukan *logout.*	✓	
3	Dashboard	1. Memilih menu dashboard.	Tidak ada masalah saat mengakses halaman dashboard.	✓	
4	Mengelola Pengumuman	1. Memilih menu pengumuman.	Tidak ada masalah saat melihat detail dan masalah saat	✓	

		2. Melengkapi form untuk membuat pengumuman baru. 3. Menakan tombol simpan untuk menyimpan data pengumuman baru.	melakukan tambah, edit, hapus data pengumuman.	
5	Mengelola Laporan Masuk	1. Memilih menu laporan masuk. 2. Melihat data laporan masuk. 3. Melakukan mengubah status, memberi tanggapan, dan menghapus data laporan masuk	Tidak ada masalah saat melihat detail dan masalah saat melakukan mengubah status, memberikan tanggapan, hapus data laporan masuk.	✓
6	Mengelola Kritik dan Saran	1. Memilih menu kritik dan saran. 2. Melihat data kritik dan saran. 3. Melakukan mengubah status, memberi tanggapan, dan menghapus data kritik dan saran.	Tidak ada masalah saat melihat detail dan masalah saat melakukan mengubah status, memberikan tanggapan, hapus data kritik dan saran.	✓
		Pengguna		
1	*Login*	1. Mengisikan data *customer* 2. Menakan tombol *login*.	Tidak ada masalah saat melakukan *login*.	✓
2	*Register*	1. Mengisikan data *customer* 2. Menakan tombol *register*.	Tidak ada masalah saat melakukan *register.*	✓
3	*Logout*	1. Menakan tombol *logout*.	Tidak ada masalah saat melakukan *logout.*	✓
4	Melihat Pengumuman	1. Memilih menu laporan masuk 2. Melihat data pengumuman.	Tidak ada masalah saat melihat detail data pengumuman.	✓
5	Membuat Laporan Masuk	1. Memilih menu laporan. 2. Melengkapi form untuk membuat laporan baru. 3. Menakan tombol simpan untuk menyimpan data laporan baru.	Tidak ada masalah saat melihat detail dan masalah saat melakukan tambah, edit, hapus data laporan.	✓
6	Membuat Kritik dan Saran	1. Memilih menu kritik dan saran. 2. Melengkapi form untuk membuat kritik dan saran baru.	Tidak ada masalah saat melihat detail dan masalah saat melakukan tambah, edit, hapus data kritik dan saran.	✓

		3. Menekan tombol simpan untuk menyimpan data kritik dan saran baru.		
		Kepala Desa		
1	*Login*	1. Mengisikan data *customer* 2. Menakan tombol *login*.	Tidak ada masalah saat melakukan *login*.	✓
2	*Logout*	1. Menakan tombol *logout*.	Tidak ada masalah saat melakukan *logout.*	✓
3	Dashboard	1. Memilih menu dashboard.	Tidak ada masalah saat mengakses halaman dashboard.	✓
4	Mengelola Pengumuman	1. Memilih menu pengumuman. 2. Melengkapi form untuk membuat pengumuman baru. 3. Menakan tombol simpan untuk menyimpan data pengumuman baru.	Tidak ada masalah saat melihat detail dan masalah saat melakukan tambah, edit, hapus data pengumuman.	✓
5	Melihat Laporan Masuk	1. Memilih menu laporan masuk. 2. Melihat data laporan masuk.	Tidak ada masalah saat melihat data laporan masuk.	✓
6	Melihat Kritik dan Saran	1. Memilih menu kritik dan saran. 2. Melihat data kritik dan saran.	Tidak ada masalah saat melihat data kritik dan saran.	✓

SIMPULAN

Berdasarkan hasil penelitian dan pembahasan dapat ditarik kesimpulan bahwa aplikasi berbasis website tersebut dikembangkan menggunakan Laravel sebagai Backend yaitu *Framework* dari PHP yang menggunakan teknologi *Server Side-Rendering* (SSR), sedangkan *Frontend*-nya menggunakan Bootstrap yang sudah mengkombinasikan teknologi HTML, CSS, dan JS untuk membuat desain tampilan yang dapat dilihat melalui browser, dan Database yang digunakan untuk menyimpan dan mengelola data adalah MySQL. Aplikasi ini memiliki beberapa fitur yang dapat diakses oleh pelanggan, fitur unggulannya adalah fitur kritik dan saran untuk kepala desa setempat, dengan adanya fitur tersebut diharapkan dapat membantu kepala desa dalam mengelola.

DAFTAR PUSTAKA

Arifin, M., Bobbi, M., & Nst, K. (T.T.). Perancangan Aplikasi Pengolahan Data Penduduk Desa Janji Kecamatan Bilah Barat Kabupaten Labuhanbatu Berbasis Web Oleh.

Atmojo, W. T., Dazki, E., & Bima, A. (2019). Sistem Informasi Pengelolaan Data Penduduk Desa Parakanlima Sukabumi Berbasis Web (Vol. 2). Https://Teknojurnal.Com/Definisi-Internet-Of-Things/

Haswan, F., Kuantan Singingi Jl Gatot Subroto, I. K., & Kuantan Riau, T. (2018). Perancangan Sistem Informasi Pendataan Penduduk Kelurahan Sungai Jering Berbasis Web Dengan Object Oriented Programming. 1(2), 92–100.

Josi, A., Akuntansi, K., Prabumulih, S., Patra No, J. L., Sukaraja, K., & Selatan, K. P. (2017). Stmik-Musirawas Lubuklinggau 50 Penerapan Metode Prototiping Dalam Pembangunan Website Desa (Studi Kasus Desa Sugihan Kecamatan Rambang). Dalam JTI (Vol. 9, Nomor 1).

Journal, A., & Rustam, A. (T.T.). Volume 4 Issue 1 27 Aisyah Journal Of Informatics And Electrical Engineering Perancangan Sistem Informasi Inventory Berbasis Web Pada Gudang Di Pt. Spin Warriors. Http://Jti.Aisyahuniversity.Ac.Id/Index.Php/AJIEE

Nurcahyanti Fakultas Ilmu Komputer Universitas Darwan Ali Sampit -Kalimantan Tengah Abstrak, Y. (T.T.). Sistem Informasi Pendataan Penduduk Desa Ganepo Berbasis Dekstop.

Renaldo, A. (T.T.). Perancangan Website Interaktif Mengenai Gedung Joang 45 Jakarta. Https://Doi.Org/10.52969/Semnasikj.V1i1.30

Tarina, V. (2015). Rancang Bangun Sistem Informasi Pendataan Penduduk Berbasis Dekstoppada Kelurahan Simpang Perlang. Rancang Bangun Sistem Informasi Pendataan Penduduk Berbasis Dekstoppada Kelurahan Simpang Perlang. Https://Lppm.Atmaluhur.Ac.Id/Wp-Content/Uploads/2015/12/Jurnal_1122510069_Vonny-Tarina.Pdf

Tumanggor, F., & Silalahi, M. (2023). Rancang Bangun Sistem Informasi Pengelolaan Material Spray Painting Di Pt Wohlrab Indonesia. Jurnal Comasie, 08(1).

Watik, D., Novitasari, F., Trisiana, D. A., Ppkn, M., Slamet, U., Surakarta, R., Universitas, D., & Surakarta, S. R. (T.T.). Analisis Peran Pemerintahan Dalam Pengendalian Pertumbuhan Penduduk.

Zen, C. E., Namira, S., & Rahayu, T. (2022). Rancang Ulang Desain UI (User Interface) Company Profile Berbasis Website Menggunakan Metode UCD (User Centered Design). Dalam Seminar Nasional Mahasiswa Ilmu Komputer Dan Aplikasinya (SENAMIKA) Jakarta-Indonesia.