DAFTAR PUSTAKA

Fathansyah . 2012. Basis Data, Bandung : Informatika Bandung.(Diakses pada tanggal 05 April 2020)

Fathansyah. 2015. *Basis Data.* Bandung: Informatika Bandung.(Diakses pada tanggal 05 April 2020)

Fowler, Martin. 2005. *UML Distilled Edisi 3,* Yogyakarta: Andi.(Diakses pada tanggal 05 April 2020)

Hendini, Ade. 2016. *Pemodelan UML Sistem Informasi Monitoring Penjualan dan Stok Barang (Studi Kasus: Dostro Zhezha Pontianak).* Jurnal Khatulistiwa, Vol. 4 No. 2.(Diakses pada tanggal 11 April 2020)

Hermawan, J, 2004. *"Analisa Desain dan Pemrograman Obyek dengan UML dan Visual Basic.Net".* Yogyakarta : Andi.
(Diakses pada tanggal 11 April 2020)

Hutahaean. 2015. Jeperson. *Konsep Sistem Informasi.* Yogyakarta : Deepublish.(Diakses pada tanggal 28 April 2020)

Iskandaria. 2012. Contoh Pengujian Black Box. http://kafegue.com/contohpengujian-black-box-testing/. (Diakses pada tanggal 28 April 2020)

Jogiyanto. 2005. *Analisis dan Disain Sistem Informasi: Pendekatan Terstruktur*.(Diakses pada tanggal 28 April 2020)

Kendal, Kenneth E dan Kendall, Julie E.2003. *Analisa dan Perancangan Sistem Jilid 1.* Jakarta: PT. INDEKS Kelompok Gramedia. (Diakses pada tanggal 20 Agustus 2020)

Mulyani, Sri. 2016. *Sistem Informasi Manajemen*. Bandung: Abdi Sistematika.(Diakses pada tanggal 20 Agustus 2020)

Nugroho, Ade. 2005. *Analisis dan Perancangan Sistem Informasi Dengan Metodologi Berorientasi Objek.* Bandung: Informatika.(Diakses pada tanggal 25 September 2020)

Nugroho, Adi. 2006. *E-commerce*. Informatika Bandung. Bandung.(Diakses pada tanggal 25 September 2020)

Pressman, R.S.2015. Rekayasa Perangkat Lunak: Pendekatan Praktisi Buku I. Yogyakarta : Andi.(Diakses pada tanggal 25 September 2020)

Pressman, Roger. 1997. *" Rekayasa Perangkat Lunak (Buku Satu)".* Yogyakarta: Andi.(Diakses pada tanggal 04 Januari 2021)

Prof. Dr. Sri Mulyani. (2016). Metode Analisis dan Perancangan Sistem. Bandung: Abdi SisteMatika.(Diakses pada tanggal 04 Januari 2021)

Reitz, Joan M. 2004. *Dictionary for Library for Library and Information Science.*(Diakses pada tanggal 04 Januari 2021)

Rudiyanto, Arief M. 2011. *Pemrograman Web Dinamis Menggunakan PHP dan MySQL*. Yogyakarta. Andi Offset.(Diakses pada tanggal 27 Februari 2021)

Saputra, Alhadi 2012, *Manajemen Basis Data MYSQL Pada Situs FTP Lapan Bandung.*(Diakses pada tanggal 27 Februari 2021)

Shihab. 2011. Metode White Box dan Black Box Testing. http://rijjasihabuddin. blogspot.com/2014/03/metode-white-box-dan-black-box-testing.html.(Diakses pada tanggal 05 Maret 2021)

Sidik Betha., 2012, Pemrograman Web dengan PHP, Informatika, Bandung.(Diakses pada tanggal 06 Maret 2021)

Sidik Betha., 2012. *Framework CodeIgniter*. Bandung: Informatika, Bandung.(Diakses pada tanggal 06 Maret 2021)

Sutabri, Tata. 2012. *Analisis Sistem Informasi*. Yogyakarta: Andi.*Teori dan Praktek Aplikasi Bisnis.* Yogyakarta: Andi.(Diakses pada tanggal 07 Maret 2021)

Utami dan Hartanto. 2012. Konsep Perangkat Lunak ERD. Bandung: Informatika.(Diakses pada tanggal 20 Maret 2021)

Wahana Komputer dan Andi. 2009. *ShortCourse: PHP Programming*. Semarang: Wahana Komputer; Yogyakarta: Andi. Westport: Libraries Unlimited.(Diakses pada tanggal 20 Maret 2021)

Yudi Priyadi, Kolaborasi SQL Dan ERD Dalam Implementasi Database, Penerbit Andi, Yogyakarta, 2014.(Diakses pada tanggal 20 Maret 2021)

LAMPIRAN

Controller

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class AdminController extends Controller
{
      public function index(){
      return view('dashboard',[
          'title' => 'dashboard'
      ]);
   }
}

<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class Controller extends BaseController
{
   use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
}

<?php

namespace App\Http\Controllers;

use Illuminate\Support\Facades\Hash;

use App\Models\Mahasiswa;
use App\Models\User;

class CreateUserController extends Controller
{
   public function index($id)
   {
      $student = Mahasiswa::find($id);
```

```
        if (!$student) {
            return "not found";
        }

        $user = new User;
        $user->name = $student->nama_mahasiswa;
        $user->username = $student->nim;
        $user->email = $student->nim . ".example@student.unhas.ac.id";
        $user->level = "editor";
        $user->password = Hash::make($student->nim);
        $user->save();
        $student->user = true;
        $student->save();

        return redirect()->back();
    }
}

<?php

namespace App\Http\Controllers;

use App\Models\Dospem;
use App\Models\Jurusan;
use App\Models\Skripsi;
use App\Models\Mahasiswa;
use Illuminate\Http\Request;
use Illuminate\Routing\Controller;
use \Cviebrock\EloquentSluggable\Services\SlugService;
use Illuminate\Support\Facades\Auth;

class DashboardPostController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $user = Auth::user();

        if ($user->level == "uploader") {
            return redirect('uploader/create');
        }

        return view('skripsi_post', [
```

```
            'posts'      =>      Skripsi::latest()->filter(request(['search',      'jurusan',
'mahasiswa']))->get(),
        ]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('create', [
            'jurusans' => Jurusan::all(),
            'dospems' => Dospem::all(),
            'mahasiswas' => Mahasiswa::all(),
            'skripsis' => Skripsi::all()
        ]);
    }

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

        $skripsi = new skripsi;
        $skripsi->judul = $request->judul;
        $skripsi->slug = $request->slug;
        $skripsi->jurusan_id = $request->jurusan_id;
        $skripsi->mahasiswa_id = $request->mahasiswa_id;
        $skripsi->file = $request->file('file')->store('files');
        $skripsi->abstrak = $request->abstrak;
        $skripsi->angkatan = $request->angkatan;
        $skripsi->save();

        $dospem = Dospem::find([
            'dospem1' => $request->dospem_id1,
            'dospem2' => $request->dospem_id2,
        ]);
        $skripsi->dospems()->attach($dospem);
        return redirect('/skripsipost')->with('success', 'Skripsi telah ditambahkan');
    }

    /**
```

```
     * Display the specified resource.
     *
     * @param  \App\Models\Skripsi  $post
     * @return \Illuminate\Http\Response
     */
    public function show(Skripsi $post)
    {
        return view('show', ['post' => $post]);
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  \App\Models\Skripsi  $post
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        $post = Skripsi::findOrFail($id);
        return view('edit', [
            'post' => $post,
            'jurusans' => Jurusan::all(),
            'dospems' => Dospem::all(),
            'mahasiswas' => Mahasiswa::all(),
            'skripsis' => Skripsi::all()
        ]);
        // return view('edit',[
        //     'post' => $post,
        //     'jurusans' => Jurusan::all(),
        //     'dospems' => Dospem::all(),
        //     'mahasiswas'=>Mahasiswa::all(),
        //     'skripsis' =>Skripsi::all()
        // ]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \App\Models\Skripsi  $post
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {

        $post = Skripsi::findOrFail($id);
        if ($request->hasFile('file')) {
            $path = $request->file('file')->store('public/files');
```

```
            $post->file = $path;
        }
        $post->update([
            'judul' => $request->judul,
            'mahasiswa_id' => $request->mahasiswa_id,
            'jurusan_id' => $request->jurusan_id,
            'file' => $path,
            'slug' => $request->slug
        ]);

        $dospem = Dospem::find([
            'dospem1' => $request->dospem_id1,
            'dospem2' => $request->dospem_id2,
        ]);

        $post->dospems()->sync($dospem);

        if ($post) {
            return redirect('/skripsipost')->with('success', 'Skripsi telah Update');
        } else {
            return redirect()
                ->back()
                ->withInput()
                ->with([
                    'error' => 'Some problem has occured, please try again'
                ]);
        }

        // $rules = $this->validate($request, [
        //     'judul' => 'required',
        //     'mahasiswa_id' => 'required',
        //     'jurusan_id' => 'required',
        //     'dospem_id' => 'required',
        //     'required' => 'required|string|max:155'
        // ]);
        // if ($request->slug !=$post) {
        //     # code...
        // }
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  \App\Models\Skripsi  $post
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
```

```
        $post = Skripsi::findOrFail($id);
        $post->destroy($id);
        // $post->delete();

        // Skripsi::destroy($post->slug);
        return redirect('/skripsipost')->with('success', 'Skripsi telah terhapus');
    }

    public function checkSlug(Request $request)
    {
        $slug = SlugService::createSlug(Skripsi::class, 'slug', $request->judul);
        return response()->json(['slug' => $slug]);
    }
}

<?php

namespace App\Http\Controllers;

use App\Models\Dospem;
use App\Models\Jurusan;
use App\Models\Skripsi;
use App\Models\Mahasiswa;
use Illuminate\Http\Request;
use Illuminate\Routing\Controller;

class DospemPostController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('dospem.dospem_post', [
            'posts' => Dospem::all()
        ]);
    }

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

```
    return view('dospem.create', [
        'jurusans' => Jurusan::all(),
        'dospems' => Dospem::all(),
        'mahasiswas' => Mahasiswa::all(),
        'skripsis' => Skripsi::all()
    ]);
}

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

    $dospem = new Dospem;
    $dospem->nama_dosen = $request->nama_dosen;
    $dospem->nip = $request->nip;
    $dospem->jurusan_id = $request->jurusan_id;
    $dospem->save();

    return redirect('/dospempost');
}

/**
 * Display the specified resource.
 *
 * @param  \App\Models\Dospem  $post
 * @return \Illuminate\Http\Response
 */
public function show(Dospem $post)
{
    return view('show', ['post' => $post]);
}

/**
 * Show the form for editing the specified resource.
 *
 * @param  \App\Models\Dospem  $post
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    $post = Dospem::findOrFail($id);
    return view('dospem.edit', [
        'post' => $post,
```

```
            'jurusans' => Jurusan::all(),
            'dospems' => Dospem::all(),
            'mahasiswas' => Mahasiswa::all(),
            'skripsis' => Skripsi::all()
        ]);
        // return view('edit',[
        //     'post' => $post,
        //     'jurusans' => Jurusan::all(),
        //     'dospems' => Dospem::all(),
        //     'mahasiswas'=>Mahasiswa::all(),
        //     'skripsis' =>Skripsi::all()
        // ]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \App\Models\Skripsi  $post
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {

        $post = Dospem::findOrFail($id);
        $post->update([
            'nama_dosen' => $request->nama_dosen,
            'nip' => $request->nip,
            'jurusan_id' => $request->jurusan_id,
        ]);

        if ($post) {
            return redirect('/dospempost')->with('success', 'Data Dosen
Pembimbing telah Update');
        } else {
            return redirect()
                ->back()
                ->withInput()
                ->with([
                    'error' => 'Some problem has occured, please try again'
                ]);
        }
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  \App\Models\Mahasiswa  $post
```

```
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {

        $post = Dospem::findOrFail($id);
        $post->destroy($id);
        // $post->delete();

        // Skripsi::destroy($post->slug);
        return redirect('/dospempost')->with('success', 'Data dosen pembimbing
telah terhapus');
    }

    // public function checkSlug(Request $request){
    //     $slug = SlugService::createSlug(Skripsi::class, 'slug', $request->judul);
    //     return response()->json(['slug'=>$slug]);
    // }
}

<?php

namespace App\Http\Controllers;

use App\Models\Skripsi;
use App\Models\Mahasiswa;
use App\Models\Dospem;
use App\Models\Jurusan;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Http\Controllers\Controller;

class HomeController extends Controller
{
    public function index()
    {
        $skripsi = Skripsi::latest()->get();
        if (request("search")) {
            $term = request("search");
            $skripsi = Skripsi::where("angkatan", $term)
                ->orWhere("judul", $term)
                ->orderBy("angkatan", "desc")
                ->get();
        }
        return view('home', [
            'title' => 'home',
            'posts' => $skripsi,
            'jurusan' => Jurusan::latest()->get()
```

```
        ]);
    }

    public function singlePost(Skripsi $skripsi)
    {
        $prodi = Jurusan::latest()->get();
        return view('home_post', [
            'title' => 'home_post',
            'post' => $skripsi,
            'prodi' => $prodi
        ]);
    }

    public function jurusanPost(Jurusan $jurusan)
    {
        return view('jurusan_post', [
            'title' => 'jurusan',
            'jurusan1' => Jurusan::latest()->get(),
            'jurusan' => $jurusan
        ]);
    }
}

<?php

namespace App\Http\Controllers;

use App\Models\Jurusan;
use Illuminate\Http\Request;

class JurusanController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view("jurusan.index", [ 'majors' => Jurusan::all() ]);
    }

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

```
{
    return view("jurusan.create");
}

/**
 * Store a newly created resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $model = new Jurusan();
    $model->prodi = $request->name;
    $model->slug = strtolower($request->name);

    try {
        $model->save();
        return redirect("prodi")->with(["success" => "Data added"]);
    } catch (\Throwable $th) {
        return redirect("prodi")->with(["error" => "Duplicate Entry"]);
    }
}

/**
 * Display the specified resource.
 *
 * @param  \App\Models\Jurusan  $jurusan
 * @return \Illuminate\Http\Response
 */
public function show(Jurusan $jurusan)
{
    //
}

/**
 * Show the form for editing the specified resource.
 *
 * @param  \App\Models\Jurusan  $jurusan
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    $model = Jurusan::findOrFail($id);
    return view("jurusan.edit", [ 'major' => $model ]);
}

/**
```

```
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \App\Models\Jurusan  $jurusan
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {
        $model = Jurusan::findOrFail($id);
        $model->prodi = $request->name;
        $model->slug = strtolower($request->name);

        try {
            $model->save();
            return redirect("prodi");
        } catch (\Throwable $th) {
            return redirect("prodi")->with(["error" => "Fail to update"]);
        }
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  \App\Models\Jurusan  $jurusan
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        try {
            Jurusan::findOrFail($id)->delete();
            return redirect("prodi");
        } catch (\Throwable $th) {
            return redirect("prodi")->with(["error" => "Fail To Delete"]);
        }
    }
}

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Routing\Controller;
use Illuminate\Support\Facades\Auth;
use Hash;
use Session;

class LoginController extends Controller
```

```
{
    public function index(){
        return view('login');
    }

public function proses_login(Request $request)
    {
        request()->validate([
        'username' => 'required',
        'password' => 'required',
        ]);

        $credentials = $request->only('username', 'password');
        if (Auth::attempt($credentials)) {
            $user = Auth::user();
            if ($user->level == 'admin') {
                return redirect()->intended('admin');
            } elseif ($user->level == 'editor') {
                return redirect()->intended('editor');
            } elseif ($user->level == 'uploader') {
                return redirect('uploader/create');
            }
            return redirect('/');
        }
        return redirect('login')->withSuccess('Oppes! Silahkan Cek Inputanmu');
    }
    public function logout(Request $request) {
        $request->session()->flush();
        Auth::logout();
        return Redirect('login');
    }


    // public function authenticate(Request $request){
    //     $credentials = $request->validate([
    //         'email' => 'required|email:dns',
    //         'password' => 'required'
    //     ]);

    //     if (Auth::attempt($credentials)) {
    //         $request->session()->regenerate();

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

    //     return back()->withErrors('loginError','login failed');
    // }
}
```

```php
<?php

namespace App\Http\Controllers;

use App\Models\Dospem;
use App\Models\Jurusan;
use App\Models\Skripsi;
use App\Models\Mahasiswa;
use Illuminate\Http\Request;
use Illuminate\Routing\Controller;
use \Cviebrock\EloquentSluggable\Services\SlugService;

class MahasiswaPostController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('mahasiswa_post', [
            'posts' => Mahasiswa::all()
        ]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('mahasiswa.create', [
            'jurusans' => Jurusan::all(),
            'dospems' => Dospem::all(),
            'mahasiswas' => Mahasiswa::all(),
            'skripsis' => Skripsi::all()
        ]);
    }

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

```
{

    $mahasiswa = new Mahasiswa;
    $mahasiswa->nama_mahasiswa = $request->nama_mahasiswa;
    $mahasiswa->nim = $request->nim;
    $mahasiswa->jurusan_id = $request->jurusan_id;
    $mahasiswa->angkatan = $request->angkatan;
    $mahasiswa->save();

    $dospem = Dospem::find([
        'dospem1' => $request->dospem_id1,
        'dospem2' => $request->dospem_id2,
    ]);
    $mahasiswa->dospems()->attach($dospem);

    return redirect('/mahasiswapost');
}

/**
 * Display the specified resource.
 *
 * @param  \App\Models\Mahasiswa  $post
 * @return \Illuminate\Http\Response
 */
public function show(Mahasiswa $post)
{
    return view('show', ['post' => $post]);
}

/**
 * Show the form for editing the specified resource.
 *
 * @param  \App\Models\Mahasiswa  $post
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    $post = Mahasiswa::findOrFail($id);
    return view('mahasiswa.edit', [
        'post' => $post,
        'jurusans' => Jurusan::all(),
        'dospems' => Dospem::all(),
        'mahasiswas' => Mahasiswa::all(),
        'skripsis' => Skripsi::all()
    ]);
    // return view('edit',[
    //     'post' => $post,
    //     'jurusans' => Jurusan::all(),
```

```
        //     'dospems' => Dospem::all(),
        //     'mahasiswas'=>Mahasiswa::all(),
        //     'skripsis' =>Skripsi::all()
        // ]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \App\Models\Skripsi  $post
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {

        $post = Mahasiswa::findOrFail($id);
        $post->update([
            'nama_mahasiswa' => $request->nama_mahasiswa,
            'nim' => $request->nim,
            'jurusan_id' => $request->jurusan_id,
            'angkatan' => $request->angkatan,
        ]);

        $dospem = Dospem::find([
            'dospem1' => $request->dospem_id1,
            'dospem2' => $request->dospem_id2,
        ]);

        $post->dospems()->sync($dospem);

        if ($post) {
            return redirect('/mahasiswapost')->with('success', 'Data mahasiswa
telah Update');
        } else {
            return redirect()
                ->back()
                ->withInput()
                ->with([
                    'error' => 'Some problem has occured, please try again'
                ]);
        }

        // $rules = $this->validate($request, [
        //     'judul' => 'required',
        //     'mahasiswa_id' => 'required',
        //     'jurusan_id' => 'required',
        //     'dospem_id' => 'required',
```

```
        //     'required' => 'required|string|max:155'
        // ]);
        // if ($request->slug !=$post) {
        //     # code...
        // }
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  \App\Models\Mahasiswa  $post
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {

        $post = Mahasiswa::findOrFail($id);
        $post->destroy($id);
        // $post->delete();

        // Skripsi::destroy($post->slug);
        return redirect('/mahasiswapost')->with('success', 'Data mahasiswa telah terhapus');
    }

    public function checkSlug(Request $request)
    {
        $slug = SlugService::createSlug(Skripsi::class, 'slug', $request->judul);
        return response()->json(['slug' => $slug]);
    }
}

<?php

namespace App\Http\Controllers;

use App\Models\Dospem;
use App\Models\Jurusan;
use App\Models\Skripsi;
use App\Models\Mahasiswa;
use App\Models\User;
use Illuminate\Http\Request;
use Illuminate\Routing\Controller;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Storage;
use \Cviebrock\EloquentSluggable\Services\SlugService;
```

```
class UploaderController extends Controller
{
    /**
    * Show the form for creating a new resource.
    *
    * @return \Illuminate\Http\Response
    */
   public function create()
   {
      return view('uploader.create', [
         'jurusans' => Jurusan::all(),
         'dospems' => Dospem::all(),
         'mahasiswas' => Mahasiswa::all(),
         'skripsis' => Skripsi::all()
      ]);
   }

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

      $skripsi = new skripsi;
      $skripsi->judul = $request->judul;
      $skripsi->slug = $request->slug;
      $skripsi->jurusan_id = $request->jurusan_id;
      $skripsi->mahasiswa_id = $request->mahasiswa_id;
      $skripsi->file = $request->file('file')->store('files');
      $skripsi->abstrak = $request->abstrak;
      $skripsi->angkatan = $request->angkatan;
      $skripsi->save();

      $dospem = Dospem::find([
         'dospem1' => $request->dospem_id1,
         'dospem2' => $request->dospem_id2,
      ]);
      $skripsi->dospems()->attach($dospem);
      return redirect('uploader/create')->with('success', 'Skripsi telah
ditambahkan');
   }
}

<?php
```

```
namespace App\Http\Controllers;

use App\Models\Dospem;
use App\Models\Jurusan;
use App\Models\Skripsi;
use App\Models\Mahasiswa;
use App\Models\User;
use Illuminate\Http\Request;
use Illuminate\Routing\Controller;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Storage;
use \Cviebrock\EloquentSluggable\Services\SlugService;

class UserPostController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('user.user_post', [
            'posts' => User::all()
        ]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('user.create', [
            'jurusans' => Jurusan::all(),
            'dospems' => Dospem::all(),
            'mahasiswas' => Mahasiswa::all(),
            'skripsis' => Skripsi::all(),
            'users' => User::all(),

        ]);
    }

    /**
     * Store a newly created resource in storage.
```

```
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {

        $user = new User;
        $user->name = $request->name;
        $user->username = $request->username;
        $user->email = $request->email;
        $user->level = $request->level;
        $user->password = Hash::make($request->password);
        $user->save();

        return redirect('/userpost');
    }

    /**
     * Display the specified resource.
     *
     * @param  \App\Models\Dospem  $post
     * @return \Illuminate\Http\Response
     */
    public function show(User $post)
    {
        return view('show', ['post' => $post]);
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  \App\Models\User  $post
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        $post = User::findOrFail($id);
        return view('user.edit', [
            'post' => $post,
            'jurusans' => Jurusan::all(),
            'dospems' => Dospem::all(),
            'mahasiswas' => Mahasiswa::all(),
            'skripsis' => Skripsi::all(),
            'users' => User::all(),
        ]);
        // return view('edit',[
        //     'post' => $post,
```

```
    //     'jurusans' => Jurusan::all(),
    //     'dospems' => Dospem::all(),
    //     'mahasiswas'=>Mahasiswa::all(),
    //     'skripsis' =>Skripsi::all()
    // ]);
}

/**
 * Update the specified resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @param  \App\Models\User  $post
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id)
{

    $post = User::findOrFail($id);
    $post->update([
        'name' => $request->name,
        'username' => $request->username,
        'email' => $request->email,
        'level' => $request->level,
        'password' => Hash::make($request->password),
    ]);

    if ($post) {
        return redirect('/userpost')->with('success', 'Data user telah Update');
    } else {
        return redirect()
            ->back()
            ->withInput()
            ->with([
                'error' => 'Some problem has occured, please try again'
            ]);
    }

    // $rules = $this->validate($request, [
    //     'judul' => 'required',
    //     'mahasiswa_id' => 'required',
    //     'jurusan_id' => 'required',
    //     'dospem_id' => 'required',
    //     'required' => 'required|string|max:155'
    // ]);
    // if ($request->slug !=$post) {
    //     # code...
    // }
}
```

```
    /**
     * Remove the specified resource from storage.
     *
     * @param  \App\Models\Mahasiswa  $post
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {

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

        $is_student = Mahasiswa::where('nim', $post->username)->first();
        if ($is_student) {
            $is_student->user = false;
            $is_student->save();
        }

        $post->destroy($id);
        // $post->delete();

        // Skripsi::destroy($post->slug);
        return redirect('/userpost')->with('success', 'Data dosen pembimbing telah
terhapus');
    }

    // public function checkSlug(Request $request){
    //     $slug = SlugService::createSlug(Skripsi::class, 'slug', $request->judul);
    //     return response()->json(['slug'=>$slug]);
    // }
}

<?php

namespace App\Http\Controllers;

use App\Models\Dospem;
use App\Models\Jurusan;
use App\Models\Mahasiswa;
use App\Models\Skripsi;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class UserUploadController extends Controller
{
    public function index()
    {
        $nim = Auth::user()->username;
```

```
        $student = Mahasiswa::where('nim', $nim)->first();

        if (!$student) {
            return redirect()->back();
        }

        return view("mahasiswa.upload.index", ["student" => $student, "lecture"
=> Dospem::all()]);
    }

    public function store(Request $request, $id)
    {
        $student = Mahasiswa::where('nim', $id)->first();

        $skripsi = new Skripsi();
        $skripsi->judul = $request->judul;
        $skripsi->slug = $request->slug;
        $skripsi->jurusan_id = $student->jurusan_id;
        $skripsi->mahasiswa_id = $student->id;
        $skripsi->file = $request->file('file')->store('files');
        $skripsi->abstrak = $request->abstrak;
        $skripsi->angkatan = $student->angkatan;
        $skripsi->save();

        $dospem = Dospem::find([
            'dospem1' => $request->dospem_id1,
            'dospem2' => $request->dospem_id2,
        ]);
        $skripsi->dospems()->attach($dospem);

        return redirect("/editor")->with('success', 'Skripsi telah ditambahkan');
    }
}
```

Middleware

```
<?php

namespace App\Http\Middleware;

use Illuminate\Auth\Middleware\Authenticate as Middleware;

class Authenticate extends Middleware
{
    /**
     * Get the path the user should be redirected to when they are not
authenticated.
```

```
     *
     * @param  \Illuminate\Http\Request  $request
     * @return string|null
     */
    protected function redirectTo($request)
    {
        if (! $request->expectsJson()) {
            return route('login');
        }
    }
}

<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class Cek_login
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle(Request $request, Closure $next, $roles)
    {
        if (!Auth::check()) {
            return redirect('login');
        }
        $user = Auth::user();

        if($user->level == $roles)
            return $next($request);


        return redirect('login')->with('error',"kamu gak punya akses");
    }
}

<?php

namespace App\Http\Middleware;
```

```
use Illuminate\Cookie\Middleware\EncryptCookies as Middleware;

class EncryptCookies extends Middleware
{
    /**
     * The names of the cookies that should not be encrypted.
     *
     * @var array
     */
    protected $except = [
        //
    ];
}

<?php

namespace App\Http\Middleware;

use
Illuminate\Foundation\Http\Middleware\PreventRequestsDuringMaintenance
as Middleware;

class PreventRequestsDuringMaintenance extends Middleware
{
    /**
     * The URIs that should be reachable while maintenance mode is enabled.
     *
     * @var array
     */
    protected $except = [
        //
    ];
}

<?php

namespace App\Http\Middleware;

use App\Providers\RouteServiceProvider;
use Closure;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class RedirectIfAuthenticated
{
    /**
     * Handle an incoming request.
     *
```

```
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @param  string|null  ...$guards
     * @return mixed
     */
    public function handle(Request $request, Closure $next, ...$guards)
    {
        $guards = empty($guards) ? [null] : $guards;

        foreach ($guards as $guard) {
            if (Auth::guard($guard)->check()) {
                return redirect(RouteServiceProvider::HOME);
            }
        }

        return $next($request);
    }
}

<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\TrimStrings as Middleware;

class TrimStrings extends Middleware
{
    /**
     * The names of the attributes that should not be trimmed.
     *
     * @var array
     */
    protected $except = [
        'current_password',
        'password',
        'password_confirmation',
    ];
}

<?php

namespace App\Http\Middleware;

use Illuminate\Http\Middleware\TrustHosts as Middleware;

class TrustHosts extends Middleware
{
    /**
```

```
     * Get the host patterns that should be trusted.
     *
     * @return array
     */
    public function hosts()
    {
        return [
            $this->allSubdomainsOfApplicationUrl(),
        ];
    }
}

<?php

namespace App\Http\Middleware;

use Illuminate\Http\Middleware\TrustProxies as Middleware;
use Illuminate\Http\Request;

class TrustProxies extends Middleware
{
    /**
     * The trusted proxies for this application.
     *
     * @var array|string|null
     */
    protected $proxies;

    /**
     * The headers that should be used to detect proxies.
     *
     * @var int
     */
    protected $headers =
        Request::HEADER_X_FORWARDED_FOR |
        Request::HEADER_X_FORWARDED_HOST |
        Request::HEADER_X_FORWARDED_PORT |
        Request::HEADER_X_FORWARDED_PROTO |
        Request::HEADER_X_FORWARDED_AWS_ELB;
}

<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\VerifyCsrfToken as Middleware;

class VerifyCsrfToken extends Middleware
```

```
{
    /**
     * The URIs that should be excluded from CSRF verification.
     *
     * @var array
     */
    protected $except = [
        //
    ];
}

<?php

namespace App\Http;

use Illuminate\Foundation\Http\Kernel as HttpKernel;

class Kernel extends HttpKernel
{
    /**
     * The application's global HTTP middleware stack.
     *
     * These middleware are run during every request to your application.
     *
     * @var array
     */
    protected $middleware = [
        // \App\Http\Middleware\TrustHosts::class,
        \App\Http\Middleware\TrustProxies::class,
        \Fruitcake\Cors\HandleCors::class,
        \App\Http\Middleware\PreventRequestsDuringMaintenance::class,
        \Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,
        \App\Http\Middleware\TrimStrings::class,

\Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,
    ];

    /**
     * The application's route middleware groups.
     *
     * @var array
     */
    protected $middlewareGroups = [
        'web' => [
            \App\Http\Middleware\EncryptCookies::class,

\Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
            \Illuminate\Session\Middleware\StartSession::class,
```

```
            // \Illuminate\Session\Middleware\AuthenticateSession::class,
            \Illuminate\View\Middleware\ShareErrorsFromSession::class,
            \App\Http\Middleware\VerifyCsrfToken::class,
            \Illuminate\Routing\Middleware\SubstituteBindings::class,
        ],

        'api' => [
            //
\Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful::clas
s,
            'throttle:api',
            \Illuminate\Routing\Middleware\SubstituteBindings::class,
        ],
    ];

    /**
     * The application's route middleware.
     *
     * These middleware may be assigned to groups or used individually.
     *
     * @var array
     */
    protected $routeMiddleware = [
        'auth' => \App\Http\Middleware\Authenticate::class,
        'auth.basic'                                                    =>
\Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
        'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,
        'can' => \Illuminate\Auth\Middleware\Authorize::class,
        'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
        'password.confirm'                                              =>
\Illuminate\Auth\Middleware\RequirePassword::class,
        'signed' => \Illuminate\Routing\Middleware\ValidateSignature::class,
        'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
        'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
        'cek_login' => \App\Http\Middleware\Cek_login::class // baru
    ];
}
```

Model

```
<?php

namespace App\Models;

use App\Models\Skripsi;
use App\Models\Mahasiswa;
```

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

class Dospem extends Model
{
   use HasFactory;
      protected $fillable = [
      'nama_dosen', 'nip', 'jurusan_id'
   ];
   public function mahasiswas(){
      return $this->belongsToMany(Mahasiswa::class);
   }
      public function skripsis(){
      return $this->belongsToMany(Skripsi::class);
   }
      public function jurusan(){
         return $this->belongsTo(Jurusan::class);
   }
}

<?php

namespace App\Models;

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

class DospemMahasiswa extends Model
{
   use HasFactory;
   protected $table = 'dospem_mahasiswa';
}

<?php

namespace App\Models;

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

class Home extends Model
{
   use HasFactory;
   protected                    $fillable                    =
['title','slug','dosen_pembimbing1','dosen_pembimbing2','penulis','nim','jurusa
n','abstrak'];
   protected $guarded = ['id'];
}
```

```php
<?php

namespace App\Models;

use App\Models\Skripsi;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;

class Jurusan extends Model
{
    use HasFactory;
        public function skripsi(){
            return $this->hasMany(Skripsi::class);
    }
}

<?php

namespace App\Models;

use App\Models\Dospem;
use App\Models\Jurusan;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;

class Mahasiswa extends Model
{
    protected $guard = 'mahasiswas';
    protected $fillable = [
        'nama_mahasiswa', 'nim', 'jurusan_id'
    ];
    use HasFactory;
    public function dospems()
    {
        return $this->belongsToMany(Dospem::class);
    }

    public function jurusan()
    {
        return $this->belongsTo(Jurusan::class);
    }
}

<?php

namespace App\Models;
```

```
class ModelHome
{
   private static $home_posts = [
         [
         'title' => 'Remediasi Air Asam Tambang Menggunakan Modified Clay',
         'slug' => 'remediasi-air-asam-tambang-menggunakan-modified-clay',
         'dosen_pembimbing1' => 'Elvi Restiawaty, S.T., P.D.Eng., Ph.D.',
         'dosen_pembimbing2' => 'Wibawa Hendra Saputera, S.Si., M.Si., M.Sc., Ph.D.',
         'penulis' => 'Elicia Kusuma',
         'nim' => '13017082',
         'jurusan' => 'Sistem Informasi',
         'tanggal_input' => '2021-09-25',
         'abstrak' => 'Sumur SDW-5ST1 adalah salah satu cored well dari Lapangan Sadewa, salah satu lapangan yang akan di kembangkan. Untuk mengetahui efek anisotropi pada bataun reservoir dilakukan pengukuran anisotropi inti batuan dengan tomografi ultrasonic dimana pada tomografi ini dapat mencintrakan struktur yang lebih kecil dibandingkan tomografi seismik. Data tingkat anisotropi sampel inti batuan diperoleh melalui pengukuran waktu tempuh gelombang P dan S dengan metode tomografi ultrasonic. Pengukuran dilakukan pada berbagai sudut didasari model Vertically Transverse Isotropic (VTI). Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat fisis dan sifat elastis setiap sampel. Data hasil poengukuran tomografi ultrasonik juga diolah menggunakan algoritma MSIRT menghasilkan data berupa sifat elastis sampel inti batuan. Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat elastis, sifat fisis, dan petrografi hasil deskripsi setiap sampel. Sampel inti batuan SDW-5ST1 merupakan batupasir dari batuan reservoir upper Miocene dengan fasies upper slope fan dan terendapkan pada system tract lowstand. Hasil evaluasi factor upscaling menunjukkan bahwa tidak didapat korelasi yang cukup baik antara data log sumur dan data inti bataun di fasies upper slope fan. Berdasarkan analisis pengaruh sifat batuan terhadap anisotropi, untuk inti batuan SDW-ST1 batupasir, anisotropi dipengaruhi oleh struktur pelapisan, %lempung, porositas intergranular, porositas, permeabilitas, impedansi akustik, Vp/Vs, modulus shear, modulus bulk, modulus young, poisson’s ratio, dan konstanta lame. .'
         ],
         [
         'title' => 'Keanekaragaman Fosil Tumbuhan di Gunung Walat, Jawa Barat',
         'slug' => 'keanekaragaman-fosil-tumbuhan-di-gunung-walat-jawa-barat',
         'dosen_pembimbing1' => 'Elvi Restiawaty, S.T., P.D.Eng., Ph.D.',
         'dosen_pembimbing2' => 'Wibawa Hendra Saputera, S.Si., M.Si., M.Sc., Ph.D.',
         'penulis' => 'Elicia Kusuma',
         'nim' => '13017082',
         'jurusan' => 'Matematika',
```

```
        'tanggal_input' => '2021-09-25',
        'abstrak' => 'Sumur SDW-5ST1 adalah salah satu cored well dari Lapangan Sadewa, salah satu lapangan yang akan di kembangkan. Untuk mengetahui efek anisotropi pada bataun reservoir dilakukan pengukuran anisotropi inti batuan dengan tomografi ultrasonic dimana pada tomografi ini dapat mencintrakan struktur yang lebih kecil dibandingkan tomografi seismik. Data tingkat anisotropi sampel inti batuan diperoleh melalui pengukuran waktu tempuh gelombang P dan S dengan metode tomografi ultrasonic. Pengukuran dilakukan pada berbagai sudut didasari model Vertically Transverse Isotropic (VTI). Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat fisis dan sifat elastis setiap sampel. Data hasil poengukuran tomografi ultrasonik juga diolah menggunakan algoritma MSIRT menghasilkan data berupa sifat elastis sampel inti batuan. Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat elastis, sifat fisis, dan petrografi hasil deskripsi setiap sampel. Sampel inti batuan SDW-5ST1 merupakan batupasir dari batuan reservoir upper Miocene dengan fasies upper slope fan dan terendapkan pada system tract lowstand. Hasil evaluasi factor upscaling menunjukkan bahwa tidak didapat korelasi yang cukup baik antara data log sumur dan data inti bataun di fasies upper slope fan. Berdasarkan analisis pengaruh sifat batuan terhadap anisotropi, untuk inti batuan SDW-ST1 batupasir, anisotropi dipengaruhi oleh struktur pelapisan, %lempung, porositas intergranular, porositas, permeabilitas, impedansi akustik, Vp/Vs, modulus shear, modulus bulk, modulus young, poisson’s ratio, dan konstanta lame. .'
        ]
        ];

        public static function all(){
          return collect(self::$home_posts);
        }

        public static function find($slug){
           $posts = static::all();
           return $posts->firstWhere('slug',$slug);
        }
}

<?php

namespace App\Models;


class ModelHome
{
   private static $home_posts = [
        [
        'title' => 'Remediasi Air Asam Tambang Menggunakan Modified Clay',
        'slug' => 'remediasi-air-asam-tambang-menggunakan-modified-clay',
```

'dosen_pembimbing1' => 'Elvi Restiawaty, S.T., P.D.Eng., Ph.D.',
'dosen_pembimbing2' => 'Wibawa Hendra Saputera, S.Si., M.Si., M.Sc., Ph.D.',
'penulis' => 'Elicia Kusuma',
'nim' => '13017082',
'jurusan' => 'Sistem Informasi',
'tanggal_input' => '2021-09-25',
'abstrak' => 'Sumur SDW-5ST1 adalah salah satu cored well dari Lapangan Sadewa, salah satu lapangan yang akan di kembangkan. Untuk mengetahui efek anisotropi pada bataun reservoir dilakukan pengukuran anisotropi inti batuan dengan tomografi ultrasonic dimana pada tomografi ini dapat mencintrakan struktur yang lebih kecil dibandingkan tomografi seismik. Data tingkat anisotropi sampel inti batuan diperoleh melalui pengukuran waktu tempuh gelombang P dan S dengan metode tomografi ultrasonic. Pengukuran dilakukan pada berbagai sudut didasari model Vertically Transverse Isotropic (VTI). Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat fisis dan sifat elastis setiap sampel. Data hasil poengukuran tomografi ultrasonik juga diolah menggunakan algoritma MSIRT menghasilkan data berupa sifat elastis sampel inti batuan. Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat elastis, sifat fisis, dan petrografi hasil deskripsi setiap sampel. Sampel inti batuan SDW-5ST1 merupakan batupasir dari batuan reservoir upper Miocene dengan fasies upper slope fan dan terendapkan pada system tract lowstand. Hasil evaluasi factor upscaling menunjukkan bahwa tidak didapat korelasi yang cukup baik antara data log sumur dan data inti bataun di fasies upper slope fan. Berdasarkan analisis pengaruh sifat batuan terhadap anisotropi, untuk inti batuan SDW-ST1 batupasir, anisotropi dipengaruhi oleh struktur pelapisan, %lempung, porositas intergranular, porositas, permeabilitas, impedansi akustik, Vp/Vs, modulus shear, modulus bulk, modulus young, poisson’s ratio, dan konstanta lame. .'
],
[
'title' => 'Keanekaragaman Fosil Tumbuhan di Gunung Walat, Jawa Barat',
'slug' => 'keanekaragaman-fosil-tumbuhan-di-gunung-walat-jawa-barat',
'dosen_pembimbing1' => 'Elvi Restiawaty, S.T., P.D.Eng., Ph.D.',
'dosen_pembimbing2' => 'Wibawa Hendra Saputera, S.Si., M.Si., M.Sc., Ph.D.',
'penulis' => 'Elicia Kusuma',
'nim' => '13017082',
'jurusan' => 'Matematika',
'tanggal_input' => '2021-09-25',
'abstrak' => 'Sumur SDW-5ST1 adalah salah satu cored well dari Lapangan Sadewa, salah satu lapangan yang akan di kembangkan. Untuk mengetahui efek anisotropi pada bataun reservoir dilakukan pengukuran anisotropi inti batuan dengan tomografi ultrasonic dimana pada tomografi ini dapat mencintrakan struktur yang lebih kecil dibandingkan tomografi seismik.

Data tingkat anisotropi sampel inti batuan diperoleh melalui pengukuran waktu tempuh gelombang P dan S dengan metode tomografi ultrasonic. Pengukuran dilakukan pada berbagai sudut didasari model Vertically Transverse Isotropic (VTI). Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat fisis dan sifat elastis setiap sampel. Data hasil poengukuran tomografi ultrasonik juga diolah menggunakan algoritma MSIRT menghasilkan data berupa sifat elastis sampel inti batuan. Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat elastis, sifat fisis, dan petrografi hasil deskripsi setiap sampel. Sampel inti batuan SDW-5ST1 merupakan batupasir dari batuan reservoir upper Miocene dengan fasies upper slope fan dan terendapkan pada system tract lowstand. Hasil evaluasi factor upscaling menunjukkan bahwa tidak didapat korelasi yang cukup baik antara data log sumur dan data inti bataun di fasies upper slope fan. Berdasarkan analisis pengaruh sifat batuan terhadap anisotropi, untuk inti batuan SDW-ST1 batupasir, anisotropi dipengaruhi oleh struktur pelapisan, %lempung, porositas intergranular, porositas, permeabilitas, impedansi akustik, Vp/Vs, modulus shear, modulus bulk, modulus young, poisson’s ratio, dan konstanta lame. .'

```
        ]
        ];

        public static function all(){
          return collect(self::$home_posts);
        }

        public static function find($slug){
           $posts = static::all();
           return $posts->firstWhere('slug',$slug);
        }
}


Home::create([
   'title' => 'Keanekaragaman Fosil Tumbuhan di Gunung Walat, Jawa Barat',
        'slug' => 'keanekaragaman-fosil-tumbuhan-di-gunung-walat-jawa-barat',
        'dosen_pembimbing1' => 'Elvi Restiawaty, S.T., P.D.Eng., Ph.D.',
        'dosen_pembimbing2' => 'Wibawa Hendra Saputera, S.Si., M.Si., M.Sc., Ph.D.',
        'penulis' => 'Elicia Kusuma',
        'nim' => '13017082',
        'jurusan' => 'Matematika',
        'abstrak' => 'Sumur SDW-5ST1 adalah salah satu cored well dari Lapangan Sadewa, salah satu lapangan yang akan di kembangkan. Untuk mengetahui efek anisotropi pada bataun reservoir dilakukan pengukuran anisotropi inti batuan dengan tomografi ultrasonic dimana pada tomografi ini dapat mencintrakan struktur yang lebih kecil dibandingkan tomografi seismik. Data tingkat anisotropi sampel inti batuan diperoleh melalui pengukuran waktu
```

tempuh gelombang P dan S dengan metode tomografi ultrasonic. Pengukuran dilakukan pada berbagai sudut didasari model Vertically Transverse Isotropic (VTI). Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat fisis dan sifat elastis setiap sampel. Data hasil poengukuran tomografi ultrasonik juga diolah menggunakan algoritma MSIRT menghasilkan data berupa sifat elastis sampel inti batuan. Setelah data parameter anisotropi sampel inti batuan diperoleh, dilakukan analisa hubungan parameter anisotropi dengan sifat elastis, sifat fisis, dan petrografi hasil deskripsi setiap sampel. Sampel inti batuan SDW-5ST1 merupakan batupasir dari batuan reservoir upper Miocene dengan fasies upper slope fan dan terendapkan pada system tract lowstand. Hasil evaluasi factor upscaling menunjukkan bahwa tidak didapat korelasi yang cukup baik antara data log sumur dan data inti bataun di fasies upper slope fan. Berdasarkan analisis pengaruh sifat batuan terhadap anisotropi, untuk inti batuan SDW-ST1 batupasir, anisotropi dipengaruhi oleh struktur pelapisan, %lempung, porositas intergranular, porositas, permeabilitas, impedansi akustik, Vp/Vs, modulus shear, modulus bulk, modulus young, poisson’s ratio, dan konstanta lame. .'
]);

```
<?php

namespace App\Models;

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

class Post extends Model
{
   use HasFactory;
}

<?php

namespace App\Models;
use Cviebrock\EloquentSluggable\Sluggable;
use App\Models\Dospem;
use App\Models\Jurusan;
use App\Models\Mahasiswa;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;

class Skripsi extends Model
{
      use Sluggable,HasFactory;
      protected $fillable = [
      'judul', 'abstrak', 'slug', 'mahasiswa_id','jurusan_id','dospem_id'
   ];
   protected $with =['jurusan','dospems','mahasiswa'];
```

```
    public function jurusan(){
        return $this->belongsTo(Jurusan::class);
    }
    public function dospems(){
        return $this->belongsToMany(Dospem::class);
    }
    public function mahasiswa(){
        return $this->belongsTo(Mahasiswa::class);
    }
    public function getRouteKeyName(){
        return 'slug';
    }
    public function sluggable(): array
        {
            return [
            'slug' => [
                'source' => 'judul'
            ]
            ];
        }
        public function scopeFilter($query, array $filters){
            $query->when($filters['search'] ?? false, function($query, $search){
                return $query->where('judul','like','%'.request('search').'%')
                ->orWhere(function($query){
                    $query-
>whereYear('created_at','like','%'.request('search').'%');
                })
                ->orWhereHas('jurusan', function($query){
                    $query->where('prodi','like','%'.request('search').'%');
                })
                ->orWhereHas('mahasiswa', function($query){
                    $query-
>where('nama_mahasiswa','like','%'.request('search').'%');
                });
            });

        }
}

<?php

namespace App\Models;

use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;
```

```
use Laravel\Sanctum\HasApiTokens;

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

    /**
     * The attributes that are mass assignable.
     *
     * @var string[]
     */
    protected $fillable = [
        'name',
        'username',
        'email',
        'level',
        'password',
    ];

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

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

Providers

```
<?php

namespace App\Providers;

use Illuminate\Support\ServiceProvider;

class AppServiceProvider extends ServiceProvider
```

```
{
    /**
     * Register any application services.
     *
     * @return void
     */
    public function register()
    {
        //
    }

    /**
     * Bootstrap any application services.
     *
     * @return void
     */
    public function boot()
    {
        //
    }
}

<?php

namespace App\Providers;

use Illuminate\Foundation\Support\Providers\AuthServiceProvider as ServiceProvider;
use Illuminate\Support\Facades\Gate;

class AuthServiceProvider extends ServiceProvider
{
    /**
     * The policy mappings for the application.
     *
     * @var array
     */
    protected $policies = [
        // 'App\Models\Model' => 'App\Policies\ModelPolicy',
    ];

    /**
     * Register any authentication / authorization services.
     *
     * @return void
     */
    public function boot()
    {
```

```
        $this->registerPolicies();

        //
    }
}
```

```
<?php

namespace App\Providers;

use Illuminate\Support\Facades\Broadcast;
use Illuminate\Support\ServiceProvider;

class BroadcastServiceProvider extends ServiceProvider
{
    /**
     * Bootstrap any application services.
     *
     * @return void
     */
    public function boot()
    {
        Broadcast::routes();

        require base_path('routes/channels.php');
    }
}
```

Bootstrap

```
<?php

/*
|--------------------------------------------------------------------------
| Create The Application
|--------------------------------------------------------------------------
|
| The first thing we will do is create a new Laravel application instance
| which serves as the "glue" for all the components of Laravel, and is
| the IoC container for the system binding all of the various parts.
|
*/

$app = new Illuminate\Foundation\Application(
    $_ENV['APP_BASE_PATH'] ?? dirname(__DIR__)
);
```

```
/*
|--------------------------------------------------------------------------
| Bind Important Interfaces
|--------------------------------------------------------------------------
|
| Next, we need to bind some important interfaces into the container so
| we will be able to resolve them when needed. The kernels serve the
| incoming requests to this application from both the web and CLI.
|
*/

$app->singleton(
    Illuminate\Contracts\Http\Kernel::class,
    App\Http\Kernel::class
);

$app->singleton(
    Illuminate\Contracts\Console\Kernel::class,
    App\Console\Kernel::class
);

$app->singleton(
    Illuminate\Contracts\Debug\ExceptionHandler::class,
    App\Exceptions\Handler::class
);

/*
|--------------------------------------------------------------------------
| Return The Application
|--------------------------------------------------------------------------
|
| This script returns the application instance. The instance is given to
| the calling script so we can separate the building of the instances
| from the actual running of the application and sending responses.
|
*/

return $app;
```

View

```
{{-- @dd($posts) --}}
@extends('layouts.maindash')
@section('container')
<main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
    <h1 class="h2">Create Data Dosen Pembimbing </h1>
    <div class="col-lg-8">
        <form method="post" action="/dospempost" enctype="multipart/form-
data">
```

```
        @csrf
        <div class="mb-3">
          <label for="nama_dosen" class="form-label">Nama Dosen</label>
          <input type="text" class="form-control @error('nama_dosen') is-
invalid @enderror" id="nama_dosen"
            name="nama_dosen" required autofocus value="{{
old('nama_dosen') }}">
          @error('nama_dosen')
          <div class="invalid-feedback">
            {{ $message }}
          </div>
          @enderror
        </div>

        <div class="mb-3">
          <label for="nip" class="form-label">NIP</label>
          <input type="text" class="form-control @error('nip') is-invalid
@enderror" id="nip" name="nip" required
            autofocus value="{{ old('nip') }}">
          @error('nip')
          <div class="invalid-feedback">
            {{ $message }}
          </div>
          @enderror
        </div>

        <div class="mb-3">
          <label for="jurusan" class="form-label">Prodi</label>
          <select class="form-select" name="jurusan_id">
            @foreach ($jurusans as $j)
            @if (old('jurusan_id'==$j->id))
            <option value="{{ $j->id }}" selected>{{ $j->prodi }}</option>
            @else
            <option value="{{ $j->id }}">{{ $j->prodi }}</option>
            @endif
            @endforeach
          </select>
        </div>

        <button type="submit" class="btn btn-primary">Submit</button>
      </form>
    </div>
</main>
@endsection

{{-- @dd($posts) --}}
@extends('layouts.maindash')
@section('container')
```

```
   <main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
      <div>
         <h1 class="h2">Dosen Pembimbing Post</h1>
         <a href="/dospempost/create" class="btn btn-primary mb-3 mt-
3">Create Data Dosen Pembimbing</a>

         @if (session()->has('success'))
            <div class="alert alert-success" role="alert">
               {{ session('success') }}
            </div>
         @endif

         <div class="table-responsive">
            <table class="table table-striped table-sm">
               <thead>
                  <tr>
                     <th scope="col">#</th>
                     <th scope="col">Nama Dosen</th>
                     <th scope="col">NIP</th>
                     <th scope="col">Prodi</th>
                     <th scope="col">Action</th>
                  </tr>
               </thead>
               <tbody>
                  @foreach ($posts as $p)
                     <tr>
                        <td>{{ $loop->iteration }}</td>
                        <td>{{ $p->nama_dosen }}</td>
                        <td>{{ $p->nip }}</td>
                        <td>{{ $p->jurusan->prodi }}</td>
                        <td>
                           <a href="{{ route('dospempost.edit', $p->id) }}"
class="badge bg-warning"><span
                                 data-feather="edit"></span></a>
                           <form action="{{ route('dospempost.destroy', $p->id)
}}" method="post"
                              class="d-inline">
                              @method('DELETE')
                              @csrf
                              <button class="badge bg-danger border-0"
                                 onclick="return confirm('Apa kamu
yakin?')"><span
                                    data-feather="x-circle"></span></button>
                           </form>
                        </td>
                     </tr>
                  @endforeach
               </tbody>
```

```
            </table>
          </div>
        </div>
    </main>
@endsection

{{-- @dd($posts) --}}
@extends('layouts.maindash')
@section('container')
<main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
    <h1 class="h2">Edit Data Dosen Pembimbing </h1>
    <div class="col-lg-8">
        <form method="post" action="{{ route('dospempost.update', $post->id)
}}" enctype="multipart/form-data">
            @csrf
            @method('put')
            <div class="mb-3">
                <label for="nama_dosen" class="form-label">Nama dosen</label>
                <input type="text" class="form-control @error('nama_dosen') is-
invalid @enderror" id="nama_dosen"
                    name="nama_dosen" required autofocus value="{{
old('nama_dosen', $post->nama_dosen) }}">
                @error('nama_dosen')
                <div class="invalid-feedback">
                    {{ $message }}
                </div>
                @enderror
            </div>

            <div class="mb-3">
                <label for="nip" class="form-label">NIP</label>
                <input type="text" class="form-control @error('nip') is-invalid
@enderror" id="nip" name="nip" required
                    autofocus value="{{ old('nip',$post->nip) }}">
                @error('nip')
                <div class="invalid-feedback">
                    {{ $message }}
                </div>
                @enderror
            </div>

            <div class="mb-3">
                <label for="jurusan" class="form-label">Prodi</label>
                <select class="form-select" name="jurusan_id">
                    @foreach ($jurusans as $j)
                    @if (old('jurusan_id',$post->jurusan_id)==$j->id)
                    <option value="{{ $j->id }}" selected>{{ $j->prodi }}</option>
                    @else
```

```
                <option value="{{ $j->id }}">{{ $j->prodi }}</option>
                @endif
                @endforeach
            </select>
        </div>

        <button type="submit" class="btn btn-primary">Submit</button>
    </form>
  </div>
</main>
@endsection

{{-- @dd($posts) --}}
@extends('layouts.maindash')
@section('container')
<main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
  <h1 class="h2">Create Data Program Studi</h1>
  <div class="col-lg-8">
    <form method="post" action="{{ route('prodi.store') }}"
enctype="multipart/form-data">
        @csrf
        <div class="mb-3">
          <label for="name" class="form-label">Nama Prodi</label>
          <input type="text" class="form-control @error('name') is-invalid
@enderror" id="name"
            name="name" required autofocus value="{{ old('name') }}">
          @error('name')
          <div class="invalid-feedback">
            {{ $message }}
          </div>
          @enderror
        </div>

        <button type="submit" class="btn btn-primary">Submit</button>
    </form>
  </div>
</main>
@endsection

{{-- @dd($posts) --}}
@extends('layouts.maindash')
@section('container')
<main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
  <h1 class="h2">Edit Data Program Studi</h1>
  <div class="col-lg-8">
    <form method="post" action="{{ route('prodi.update', $major->id) }}"
enctype="multipart/form-data">
        @csrf
```

```
        @method('put')
        <div class="mb-3">
          <label for="name" class="form-label">Nama</label>
          <input type="text" class="form-control @error('name') is-invalid
@enderror"
            id="name" name="name" required autofocus
            value="{{ old('name', $major->prodi) }}">
          @error('name')
          <div class="invalid-feedback">
            {{ $message }}
          </div>
          @enderror
        </div>

        <button type="submit" class="btn btn-primary">Submit</button>
      </form>
    </div>
</main>
@endsection

{{-- @dd($posts) --}}
@extends('layouts.maindash')
@section('container')
<main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
    <div>
      <h1 class="h2">Dosen Program Studi</h1>
      <a href="{{ route('prodi.create') }}" class="btn btn-primary mb-3 mt-
3">Create Data Program Studi</a>

      @if (session()->has('success'))
      <div class="alert alert-success" role="alert">
        {{ session('success') }}
      </div>
      @endif

      @if (session()->has('error'))
      <div class="alert alert-danger" role="alert">
        {{ session('error') }}
      </div>
      @endif

      <div class="table-responsive">
        <table class="table table-striped table-sm">
          <thead>
            <tr>
              <th scope="col">#</th>
              <th scope="col">Nama</th>
              <th scope="col">Slug</th>
```

```
                <th scope="col">Action</th>
              </tr>
            </thead>
            <tbody>
              @foreach ($majors as $major)
              <tr>
                <td>{{ $loop->iteration }}</td>
                <td>{{ $major->prodi }}</td>
                <td>{{ $major->slug }}</td>
                <td>
                  <a href="{{ route('prodi.edit', $major->id) }}" class="badge
bg-warning"><span
                      data-feather="edit"></span></a>
                  <form action="{{ route('prodi.destroy', $major->id) }}"
method="post" class="d-inline">
                    @method('DELETE')
                    @csrf
                    <button class="badge bg-danger border-0"
                      onclick="return confirm('Apa kamu yakin?')"><span
                        data-feather="x-circle"></span></button>
                  </form>
                </td>
              </tr>
              @endforeach
            </tbody>
          </table>
        </div>
    </div>
</main>
@endsection

<!doctype html>
<html lang="en">

<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Bootstrap CSS -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2Q
vZ6jIW3" crossorigin="anonymous">
    <link        rel="stylesheet"        href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.7.1/font/bootstrap-icons.css">
    <script src="https://unpkg.com/feather-icons"></script>
```

```
    <link rel="stylesheet" href="/css/style.css">
    <title>JurnalWeb | {{ $title }}</title>
</head>

<body>
    @include('partials.navbar')
    <div class="container mt-3">
        @yield('container')
    </div>


    <!-- Javascript bootstrap-->
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.mi
n.js"
        integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nb
Tov4+1p" crossorigin="anonymous">
    </script>

    <script>
        feather.replace()
    </script>
</body>

</html>

<!doctype html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="Mark Otto, Jacob Thornton, and Bootstrap
contributors">
    <meta name="generator" content="Hugo 0.88.1">
    <title>Dashboard Template · Bootstrap v5.1</title>

    <link                                                    rel="canonical"
href="https://getbootstrap.com/docs/5.1/examples/dashboard/">

    <!-- Bootstrap core CSS -->
    <link href="/css/bootstrap.min.css" rel="stylesheet">

    <link rel="stylesheet" type="text/css" href="/css/trix.css">
    <script type="text/javascript" src="/js/trix.js"></script>
    <style>
```

```
        .bd-placeholder-img {
            font-size: 1.125rem;
            text-anchor: middle;
            -webkit-user-select: none;
            -moz-user-select: none;
            user-select: none;
        }

        @media (min-width: 768px) {
            .bd-placeholder-img-lg {
                font-size: 3.5rem;
            }
        }

        trix-toolbar [data-trix-button-group="file-tools"] {
            display: none;
        }

    </style>

    <!-- Custom styles for this template -->
    <link href="/css/dashboard.css" rel="stylesheet">
</head>

<body>
    @include('partials.navdash')
    <div class="container mt-3">
        @yield('container')
    </div>

    <script src="/js/bootstrap.bundle.min.js"></script>

    <script src="https://cdn.jsdelivr.net/npm/feather-
icons@4.28.0/dist/feather.min.js"
        integrity="sha384-
uO3SXW5IuS1ZpFPKugNNWqTZRRglnUJK6UAZ/gxOX80nxEkN9NcGZ
Tftn6RzhGWE" crossorigin="anonymous">
    </script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js@2.9.4/dist/Chart.min.js"
        integrity="sha384-
zNy6FEbO50N+Cg5wap8IKA4M/ZnLJgzc6w2NqACZaK0u0FXfOWRRJOn
QtpZun8ha" crossorigin="anonymous">
    </script>
    <script src="/js/dashboard.js"></script>
</body>

<!doctype html>
<html lang="en">
```

```
<head>
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <meta name="description" content="">
   <meta name="author" content="Mark Otto, Jacob Thornton, and Bootstrap
contributors">
   <meta name="generator" content="Hugo 0.88.1">
   <title>Dashboard Template · Bootstrap v5.1</title>

   <link                                                    rel="canonical"
href="https://getbootstrap.com/docs/5.1/examples/dashboard/">

   <!-- Bootstrap core CSS -->
   <link href="/css/bootstrap.min.css" rel="stylesheet">

   <link rel="stylesheet" type="text/css" href="/css/trix.css">
   <script type="text/javascript" src="/js/trix.js"></script>
   <style>
      .bd-placeholder-img {
         font-size: 1.125rem;
         text-anchor: middle;
         -webkit-user-select: none;
         -moz-user-select: none;
         user-select: none;
      }

      @media (min-width: 768px) {
         .bd-placeholder-img-lg {
            font-size: 3.5rem;
         }
      }

      trix-toolbar [data-trix-button-group="file-tools"] {
         display: none;
      }

   </style>

   <!-- Custom styles for this template -->
   <link href="/css/dashboard.css" rel="stylesheet">
</head>

<body>
   @include('partials.nav_uploader')
   <div class="container mt-3">
      @yield('container')
   </div>
```

```
    <script src="/js/bootstrap.bundle.min.js"></script>

    <script src="https://cdn.jsdelivr.net/npm/feather-
icons@4.28.0/dist/feather.min.js"
        integrity="sha384-
uO3SXW5IuS1ZpFPKugNNWqTZRRglnUJK6UAZ/gxOX80nxEkN9NcGZ
Tftn6RzhGWE" crossorigin="anonymous">
    </script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js@2.9.4/dist/Chart.min.js"
        integrity="sha384-
zNy6FEbO50N+Cg5wap8IKA4M/ZnLJgzc6w2NqACZaK0u0FXfOWRRJOn
QtpZun8ha" crossorigin="anonymous">
    </script>
    <script src="/js/dashboard.js"></script>
</body>

@extends('layouts.uploader')
@section('container')
    <main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
        <h1 class="h2">Create Skripsi </h1>
        <div class="col-lg-8">
            <form method="post" action={{ route('user.upload.store', Auth::user()-
>username) }}
                enctype="multipart/form-data">
                @csrf
                <div class="mb-3">
                    <label for="judul" class="form-label">Judul</label>
                    <input type="text" class="form-control @error('judul') is-invalid
@enderror" id="judul"
                        name="judul" required autofocus value="{{ old('judul') }}">
                    @error('judul')
                        <div class="invalid-feedback">
                            {{ $message }}
                        </div>
                    @enderror
                </div>

                <div class="mb-3">
                    <label for="slug" class="form-label">Slug</label>
                    <input type="text" class="form-control @error('slug') is-invalid
@enderror" id="slug"
                        name="slug" disabled readonly value="{{ old('slug') }}">
                    @error('judul')
                        <div class="invalid-feedback">
                            {{ $message }}
                        </div>
                    @enderror
```

```
            </div>

            <div class="mb-3">
               <label for="mahasiswa" class="form-label">Mahasiswa</label>
               <h6>{{ $student->nama_mahasiswa }}</h6>
            </div>

            <div class="mb-3">
               <label for="angkatan" class="form-label">Angkatan</label>
               <h6>{{ $student->angkatan }}</h6>
            </div>

            <div class="mb-3">
               <label for="jurusan" class="form-label">Prodi</label>
               <h6>{{ $student->jurusan->prodi }}</h6>
            </div>

            <div class="mb-3">
               <label for="dospem1" class="form-label">Dosen Pembimbing
Utama</label>
               <select class="form-select" name="dospem_id1">
                  @foreach ($lecture as $d)
                     <option value="{{ $d->id }}">{{ $d->nama_dosen
}}</option>
                  @endforeach
               </select>
            </div>

            <div class="mb-3">
               <label for="dospem2" class="form-label">Dosen Pembimbing
Kedua</label>
               <select class="form-select" name="dospem_id2">
                  @foreach ($lecture as $d)
                     <option value="{{ $d->id }}">{{ $d->nama_dosen
}}</option>
                  @endforeach
               </select>
            </div>

            <div class="mb-3">
               <label for="file" class="form-label">Input file Skripsi</label>
               <input class="form-control @error('file') is-invalid @enderror"
type="file" id="file"
                  name="file">
               @error('file')
                  <div class="invalid-feedback">
                     {{ $message }}
                  </div>
```

```
            @enderror
        </div>

        <div class="mb-3">
            <label for="cover" class="form-label">Input cover Skripsi</label>
            <input class="form-control @error('cover') is-invalid @enderror" type="file" id="cover"
                name="cover">
            @error('cover')
                <div class="invalid-feedback">
                    {{ $message }}
                </div>
            @enderror
        </div>

        <div class="mb-3">
            @error('abstrak')
                <p class="text-danger">{{ $message }}</p>
            @enderror
            <label for="abstrak" class="form-label">Abstrak</label>
            <input id="abstrak" type="hidden" name="abstrak" value="{{ old('abstrak') }}" required>
            <trix-editor input="abstrak"></trix-editor>
        </div>


        <button type="submit" class="btn btn-primary">Submit</button>
    </form>
</div>
</main>
<script>
    const judul = document.querySelector('#judul');
    const slug = document.querySelector('#slug');

    judul.addEventListener('change', function() {
        fetch('/skripsipost/checkSlug?judul=' + judul.value)
            .then(response => response.json())
            .then(data => slug.value = data.slug)
    });

    document.addEventListener('trix-file-accept', function(e) {
        e.preventDefault();
    })
</script>
@endsection

{{-- @dd($posts) --}}
```

```
@extends('layouts.maindash')
@section('container')
   <main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
      <h1 class="h2">Create Data Mahasiswa </h1>
      <div class="col-lg-8">
         <form method="post" action="/mahasiswapost"
enctype="multipart/form-data">
            @csrf
            <div class="mb-3">
               <label for="nama_mahasiswa" class="form-label">Nama
Mahasiswa</label>
               <input type="text" class="form-control
@error('nama_mahasiswa') is-invalid @enderror"
                  id="nama_mahasiswa" name="nama_mahasiswa" required
autofocus value="{{ old('nama_mahasiswa') }}">
               @error('nama_mahasiswa')
                  <div class="invalid-feedback">
                     {{ $message }}
                  </div>
               @enderror
            </div>

            <div class="mb-3">
               <label for="angkatan" class="form-label">Angkatan</label>
               <input type="text" class="form-control @error('angkatan') is-
invalid @enderror" id="angkatan"
                  name="angkatan" required autofocus value="{{ old('angkatan')
}}">
               @error('angkatan')
                  <div class="invalid-feedback">
                     {{ $message }}
                  </div>
               @enderror
            </div>

            <div class="mb-3">
               <label for="nim" class="form-label">NIM</label>
               <input type="text" class="form-control @error('nim') is-invalid
@enderror" id="nim"
                  name="nim" required autofocus value="{{ old('nim') }}">
               @error('nim')
                  <div class="invalid-feedback">
                     {{ $message }}
                  </div>
               @enderror
            </div>

            <div class="mb-3">
```

```
            <label for="jurusan" class="form-label">Prodi</label>
            <select class="form-select" name="jurusan_id">
              @foreach ($jurusans as $j)
                @if (old('jurusan_id' == $j->id))
                  <option value="{{ $j->id }}" selected>{{ $j->prodi }}</option>
                @else
                  <option value="{{ $j->id }}">{{ $j->prodi }}</option>
                @endif
              @endforeach
            </select>
          </div>

          <div class="mb-3">
            <label for="dospem1" class="form-label">Dosen Pembimbing Utama</label>
            <select class="form-select" name="dospem_id1">
              @foreach ($dospems as $d)
                <option value="{{ $d->id }}">{{ $d->nama_dosen }}</option>
              @endforeach
            </select>
          </div>

          <div class="mb-3">
            <label for="dospem2" class="form-label">Dosen Pembimbing Kedua</label>
            <select class="form-select" name="dospem_id2">
              @foreach ($dospems as $d)
                <option value="{{ $d->id }}">{{ $d->nama_dosen }}</option>
              @endforeach
            </select>
          </div>

          <button type="submit" class="btn btn-primary">Submit</button>
        </form>
      </div>
    </main>
@endsection

{{-- @dd($posts) --}}
@extends('layouts.maindash')
@section('container')
    <main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
      <h1 class="h2">Edit Data Mahasiswa </h1>
      <div class="col-lg-8">
```

```
        <form method="post" action="{{ route('mahasiswapost.update', $post->id) }}" enctype="multipart/form-data">
            @csrf
            @method('put')
            <div class="mb-3">
              <label for="nama_mahasiswa" class="form-label">Nama Mahasiswa</label>
              <input type="text" class="form-control @error('nama_mahasiswa') is-invalid @enderror" id="nama_mahasiswa"
                name="nama_mahasiswa" required autofocus
                value="{{ old('nama_mahasiswa', $post->nama_mahasiswa) }}">
              @error('nama_mahasiswa')
                <div class="invalid-feedback">
                  {{ $message }}
                </div>
              @enderror
            </div>

            <div class="mb-3">
              <label for="nim" class="form-label">NIM</label>
              <input type="text" class="form-control @error('nim') is-invalid @enderror" id="nim" name="nim" required
                autofocus value="{{ old('nim', $post->nim) }}">
              @error('nim')
                <div class="invalid-feedback">
                  {{ $message }}
                </div>
              @enderror
            </div>

            <div class="mb-3">
              <label for="jurusan" class="form-label">Prodi</label>
              <select class="form-select" name="jurusan_id">
                @foreach ($jurusans as $j)
                  @if (old('jurusan_id', $post->jurusan_id) == $j->id)
                    <option value="{{ $j->id }}" selected>{{ $j->prodi }}</option>
                  @else
                    <option value="{{ $j->id }}">{{ $j->prodi }}</option>
                  @endif
                @endforeach
              </select>
            </div>

            <div class="mb-3">
              <label for="dospem1" class="form-label">Dosen Pembimbing Utama</label>
```

```
            <select class="form-select" name="dospem_id1">
              @foreach ($dospems as $d)
                @if (old('dospem_id', $post->dospem_id) == $d->id)
                  <option value="{{ $d->id }}" selected>{{ $d->nama_dosen }}</option>
                @else
                  <option value="{{ $d->id }}">{{ $d->nama_dosen }}</option>
                @endif
              @endforeach
            </select>
          </div>

          <div class="mb-3">
            <label for="dospem2" class="form-label">Dosen Pembimbing Kedua</label>
            <select class="form-select" name="dospem_id2">
              @foreach ($dospems as $d)
                @if (old('dospem_id', $post->dospem_id) == $d->id)
                  <option value="{{ $d->id }}" selected>{{ $d->nama_dosen }}</option>
                @else
                  <option value="{{ $d->id }}">{{ $d->nama_dosen }}</option>
                @endif
              @endforeach
            </select>
          </div>


          <button type="submit" class="btn btn-primary">Submit</button>
        </form>
      </div>
    </main>
@endsection
```

Web

```
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\HomeController;
use App\Http\Controllers\AdminController;
use App\Http\Controllers\CreateUserController;
use App\Http\Controllers\LoginController;
use App\Http\Controllers\UserPostController;
use App\Http\Controllers\DospemPostController;
```

```
use App\Http\Controllers\DashboardPostController;
use App\Http\Controllers\MahasiswaPostController;
use App\Http\Controllers\JurusanController;
use App\Http\Controllers\UploaderController;
use App\Http\Controllers\UserUploadController;

Route::get('/', [LoginController::class, 'index']);
Route::get('/home/{skripsi:slug}', [HomeController::class, 'singlePost']);

Route::get('/skripsipost/checkSlug', [DashboardPostController::class,
'checkSlug'])->middleware('auth');
Route::resource('/skripsipost', DashboardPostController::class)-
>middleware('auth');
Route::resource('/mahasiswapost', MahasiswaPostController::class)-
>middleware('auth');
Route::resource('/dospempost', DospemPostController::class)-
>middleware('auth');
Route::resource('/prodi', JurusanController::class)->middleware('auth');
Route::get('/skripsi/prodi/{jurusan:slug}', [HomeController::class,
'jurusanPost'])->name('skripsi_per_prodi');
Route::resource('/userpost', UserPostController::class)->middleware('auth');
Route::resource('/uploader', UploaderController::class);

Route::get('login', [LoginController::class, 'index'])->name('login');
Route::post('proses_login', [LoginController::class, 'proses_login'])-
>name('proses_login');
Route::get('logout', [LoginController::class, 'logout'])->name('logout');


Route::group(['middleware' => ['auth']], function () {
   Route::group(['middleware' => ['cek_login:admin']], function () {
      /*
                  Route Khusus untuk role admin
         */
      Route::resource('admin', AdminController::class);
      Route::post("user/create/{id}", [CreateUserController::class, 'index'])-
>name('user.create');
   });
   Route::group(['middleware' => ['cek_login:editor']], function () {
      /*
                  Route Khusus untuk role editor
         */
      Route::get('editor', [HomeController::class, 'index'])->name('editor');
      Route::get("user/upload", [UserUploadController::class, 'index'])-
>name('user.upload.index');
      Route::post("user/upload/{id}", [UserUploadController::class, 'store'])-
>name('user.upload.store');
   });
```

```
});
```