LAMPIRAN

Migrasi:

```
<?php

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

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password');
            $table->string('nim')->nullable();
            $table->string('fakultas')->nullable();
            $table->string('jurusan')->nullable();
            $table->enum('status', ['level0', 'level1','level2','level3','level4','level5','level6','level7','level8','level9','level10'])-
>default('level0');
            // $table->string('rewards')->nullable();
            $table->rememberToken();
            $table->timestamps();
        });

        Schema::create('password_reset_tokens', function (Blueprint $table) {
            $table->string('email')->primary();
            $table->string('token');
            $table->timestamp('created_at')->nullable();
        });

        Schema::create('sessions', function (Blueprint $table) {
            $table->string('id')->primary();
            $table->foreignId('user_id')->nullable()->index();
            $table->string('ip_address', 45)->nullable();
            $table->text('user_agent')->nullable();
            $table->longText('payload');
            $table->integer('last_activity')->index();
        });
    }

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

<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
```

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

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('messages', function (Blueprint $table) {
            $table->id();
            $table->foreignId('user_id')->constrained();
            $table->foreignId('recipient_id')->constrained('users');
            $table->text('text')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('messages');
    }
};<?php

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

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('f_a_q_s', function (Blueprint $table) {
            $table->id();
            $table->string('judul')->nullable();
            $table->string('pertanyaan')->nullable();
            $table->text('jawaban')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('f_a_q_s');
    }
};
<?php

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

return new class extends Migration
{
```

```
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('jadwals', function (Blueprint $table) {
            $table->id();
            $table->foreignId('user_id');
            $table->string('nama')->nullable();
            $table->string('nim')->nullable();
            $table->string('fakultas')->nullable();
            $table->string('jurusan')->nullable();
            $table->date('waktu')->nullable();
            $table->text('masalah')->nullable();
            $table->string('antrian')->nullable();
            $table->string('status');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('jadwals');
    }
};
<?php

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

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('testis', function (Blueprint $table) {
            $table->id();
            $table->foreignId('user_id');
            $table->string('nama')->nullable();
            $table->string('nim')->nullable();
            $table->string('fakultas')->nullable();
            $table->string('jurusan')->nullable();
            $table->string('pendapat')->nullable();
            $table->integer('rating')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('testis');
    }
};
<?php

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

```
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        $teams = config('permission.teams');
        $tableNames = config('permission.table_names');
        $columnNames = config('permission.column_names');
        $pivotRole = $columnNames['role_pivot_key'] ?? 'role_id';
        $pivotPermission = $columnNames['permission_pivot_key'] ?? 'permission_id';

        if (empty($tableNames)) {
            throw new \Exception('Error: config/permission.php not loaded. Run [php artisan config:clear] and try again.');
        }
        if ($teams && empty($columnNames['team_foreign_key'] ?? null)) {
            throw new \Exception('Error: team_foreign_key on config/permission.php not loaded. Run [php artisan
config:clear] and try again.');
        }

        Schema::create($tableNames['permissions'], function (Blueprint $table) {
            //$table->engine('InnoDB');
            $table->bigIncrements('id'); // permission id
            $table->string('name');       // For MyISAM use string('name', 225); // (or 166 for InnoDB with
Redundant/Compact row format)
            $table->string('guard_name'); // For MyISAM use string('guard_name', 25);
            $table->timestamps();

            $table->unique(['name', 'guard_name']);
        });

        Schema::create($tableNames['roles'], function (Blueprint $table) use ($teams, $columnNames) {
            //$table->engine('InnoDB');
            $table->bigIncrements('id'); // role id
            if ($teams || config('permission.testing')) { // permission.testing is a fix for sqlite testing
                $table->unsignedBigInteger($columnNames['team_foreign_key'])->nullable();
                $table->index($columnNames['team_foreign_key'], 'roles_team_foreign_key_index');
            }
            $table->string('name');       // For MyISAM use string('name', 225); // (or 166 for InnoDB with
Redundant/Compact row format)
            $table->string('guard_name'); // For MyISAM use string('guard_name', 25);
            $table->timestamps();
            if ($teams || config('permission.testing')) {
                $table->unique([$columnNames['team_foreign_key'], 'name', 'guard_name']);
            } else {
                $table->unique(['name', 'guard_name']);
            }
        });

        Schema::create($tableNames['model_has_permissions'], function (Blueprint $table) use ($tableNames,
$columnNames, $pivotPermission, $teams) {
            $table->unsignedBigInteger($pivotPermission);

            $table->string('model_type');
            $table->unsignedBigInteger($columnNames['model_morph_key']);
            $table->index([$columnNames['model_morph_key'], 'model_type'],
'model_has_permissions_model_id_model_type_index');
```

```
            $table->foreign($pivotPermission)
                ->references('id') // permission id
                ->on($tableNames['permissions'])
                ->onDelete('cascade');
            if ($teams) {
                $table->unsignedBigInteger($columnNames['team_foreign_key']);
                $table->index($columnNames['team_foreign_key'], 'model_has_permissions_team_foreign_key_index');

                $table->primary([$columnNames['team_foreign_key'], $pivotPermission, $columnNames['model_morph_key'],
'model_type'],
                    'model_has_permissions_permission_model_type_primary');
            } else {
                $table->primary([$pivotPermission, $columnNames['model_morph_key'], 'model_type'],
                    'model_has_permissions_permission_model_type_primary');
            }

        });

        Schema::create($tableNames['model_has_roles'], function (Blueprint $table) use ($tableNames, $columnNames,
$pivotRole, $teams) {
            $table->unsignedBigInteger($pivotRole);

            $table->string('model_type');
            $table->unsignedBigInteger($columnNames['model_morph_key']);
            $table->index([$columnNames['model_morph_key'], 'model_type'],
'model_has_roles_model_id_model_type_index');

            $table->foreign($pivotRole)
                ->references('id') // role id
                ->on($tableNames['roles'])
                ->onDelete('cascade');
            if ($teams) {
                $table->unsignedBigInteger($columnNames['team_foreign_key']);
                $table->index($columnNames['team_foreign_key'], 'model_has_roles_team_foreign_key_index');

                $table->primary([$columnNames['team_foreign_key'], $pivotRole, $columnNames['model_morph_key'],
'model_type'],
                    'model_has_roles_role_model_type_primary');
            } else {
                $table->primary([$pivotRole, $columnNames['model_morph_key'], 'model_type'],
                    'model_has_roles_role_model_type_primary');
            }
        });

        Schema::create($tableNames['role_has_permissions'], function (Blueprint $table) use ($tableNames, $pivotRole,
$pivotPermission) {
            $table->unsignedBigInteger($pivotPermission);
            $table->unsignedBigInteger($pivotRole);

            $table->foreign($pivotPermission)
                ->references('id') // permission id
                ->on($tableNames['permissions'])
                ->onDelete('cascade');

            $table->foreign($pivotRole)
                ->references('id') // role id
                ->on($tableNames['roles'])
                ->onDelete('cascade');

            $table->primary([$pivotPermission, $pivotRole], 'role_has_permissions_permission_id_role_id_primary');
        });

        app('cache')
            ->store(config('permission.cache.store') != 'default' ? config('permission.cache.store') : null)
            ->forget(config('permission.cache.key'));
    }

    /**
```

```
     * Reverse the migrations.
     */
    public function down(): void
    {
        $tableNames = config('permission.table_names');

        if (empty($tableNames)) {
            throw new \Exception('Error: config/permission.php not found and defaults could not be merged. Please publish the package configuration before proceeding, or drop the tables manually.');
        }

        Schema::drop($tableNames['role_has_permissions']);
        Schema::drop($tableNames['model_has_roles']);
        Schema::drop($tableNames['model_has_permissions']);
        Schema::drop($tableNames['roles']);
        Schema::drop($tableNames['permissions']);
    }
};
```

Models:

```
. <?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 App\Enums\RewardStatus;
use Spatie\Permission\Traits\HasRoles;
class User extends Authenticatable
{
    use HasFactory, Notifiable,HasRoles;

    /**
     * The attributes that are mass assignable.
     *
     * @var array<int, string>
     */
    protected $fillable = [
        'name',
        'email',
        'password',
        'status',
        'nim',
        'fakultas',
        'jurusan',
        'rewards'
    ];

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

    /**
     * Get the attributes that should be cast.
     *
```

```
     * @return array<string, string>
     */
    protected function casts(): array
    {
        return [
            'email_verified_at' => 'datetime',
            'password' => 'hashed',
            'status' => RewardStatus::class,
        ];
    }


}
<?php

namespace App\Models;

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

class testi extends Model
{
    use HasFactory;

    protected $fillable = [
        'user_id', 'nama', 'nim', 'fakultas', 'jurusan', 'pendapat', 'rating',
    ];
}
<?php

namespace App\Models;

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

class Message extends Model
{
    use HasFactory;

    public $table = 'messages';
    protected $fillable = ['user_id', 'text', 'recipient_id'];


    public function user(): BelongsTo {
        return $this->belongsTo(User::class, 'user_id');
    }
    public function recipient(): BelongsTo {
        return $this->belongsTo(User::class, 'recipient_id');
    }


    public function getTimeAttribute(): string {
        return $this->created_at ? $this->created_at->format("d M Y, H:i:s") : 'N/A';
    }


}
<?php
```

```
namespace App\Models;

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

class Jadwal extends Model
{
    use HasFactory;
    protected $fillable = ['user_id','nama','nim','fakultas','jurusan',
    'masalah','waktu','antrian','status'];
}
<?php

namespace App\Models;

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

class FAQ extends Model
{
    use HasFactory;
    protected $fillable = ['judul', 'pertanyaan', 'jawaban'];
}
```

CONTROLLERS:

```
. <?php

namespace App\Http\Controllers\Auth;

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

class RegisteredUserController extends Controller
{
    /**
     * Display the registration view.
     */
    public function create(): Response
    {
        return Inertia::render('Auth/Register');
    }

    /**
     * Handle an incoming registration request.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function store(Request $request): RedirectResponse
    {
```

```
        $request->validate([
            'name' => 'required|string|max:255',
            'email' => 'required|string|lowercase|email|max:255|unique:'.User::class,
            'password' => ['required', 'confirmed', Rules\Password::defaults()],
            'nim' => 'required|string|max:255',
            'fakultas' => 'required|string|max:255',
            'jurusan' => 'required|string|max:255',
        ]);

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

        $Role = Role::findByName('mahasiswa');
        $user->assignRole($Role);
    event(new Registered($user));


        Auth::login($user);

        return redirect(route('dashboard', absolute: false));
    }
}
<?php

namespace App\Http\Controllers;

use Inertia\Inertia;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Facades\Redirect;
use Illuminate\Support\Str;
use Illuminate\Http\Request;
use Illuminate\Validation\Rule;
use Illuminate\Support\Facades\Auth;
use App\Models\User;
use Spatie\Permission\Models\Role;
use Illuminate\Http\RedirectResponse;
use Illuminate\Support\Facades\Hash;


class AdminController extends Controller
{
    public function index()
    {
        $user = User::with('roles')->get();
        return Inertia::render('Admin/index', [
            'user' => $user
        ]);
    }

    public function destroy(User $user)
    {
```

```
        $user->delete();

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

}
<?php

namespace App\Http\Controllers;

use App\Jobs\SendMessage;
use App\Models\Message;
use App\Models\User;
use Illuminate\Http\JsonResponse;
use Illuminate\Http\Request;
use Inertia\Inertia;
use App\Enums\RewardStatus;

class ChatController extends Controller
{

    // public function index()
    // {
    //     $user = auth()->user();

    //     return Inertia::render('ChatBox', [
    //         'user' => [
    //             'id' => $user->id,
    //             'name' => $user->name,
    //             'email' => $user->email,
    //         ],
    //         'rootUrl' => url('/'),
    //     ]);

    // }
    public function index2(): JsonResponse
    {
        // Mengembalikan semua pengguna dengan ID dan nama
        $users = User::select('id', 'name')->get();

        return response()->json($users);
    }
    public function chatUsers()
    {
        return Inertia::render('ChatUsers', [
            'users' => User::with('roles')->get()->map(function ($user) {
                return [
                    'id' => $user->id,
                    'name' => $user->name,
                    'roles' => $user->roles->pluck('name')->toArray(), // Ambil nama role sebagai array
                ];
            }),
        ]);
    }


    public function chatBox($userId)
```

```
    {
        $recipient = User::findOrFail($userId);
        return Inertia::render('ChatBox', [
            'recipient' => [
                'id' => $recipient->id,
                'name' => $recipient->name,
            ],
            'rootUrl' => url('/'),
        ]);
    }


    public function messages(Request $request): JsonResponse
    {
        $validated = $request->validate([
            'recipient_id' => 'required|exists:users,id',
        ]);

        $messages = Message::with('user:id,name')
            ->where(function ($query) use ($validated) {
                $query->where('user_id', auth()->id())
                    ->where('recipient_id', $validated['recipient_id']);
            })
            ->orWhere(function ($query) use ($validated) {
                $query->where('user_id', $validated['recipient_id'])
                    ->where('recipient_id', auth()->id());
            })
            ->latest()
            ->limit(50)
            ->get()
            ->append('time');

        return response()->json($messages);
    }


    public function message(Request $request): JsonResponse
    {
        $validated = $request->validate([
            'text' => 'required|string|max:255',
            'recipient_id' => 'required|exists:users,id', // Tambahkan validasi untuk recipient_id
        ]);

        $message = Message::create([
            'user_id' => auth()->id(),
            'text' => $validated['text'],
            'recipient_id' => $validated['recipient_id'], // Pastikan field ini ditambahkan
        ]);

        SendMessage::dispatch($message);

return response()->json([
    'success' => true,
    'refresh' => true,
    'message' => "Message created and job dispatched.",
]);

    }

public function deleteChat(Request $request): JsonResponse
{
    $validated = $request->validate([
        'recipient_id' => 'required|exists:users,id',
```

```
        'keep_history' => 'required|boolean',
    ]);

    // Kalau tidak ingin menyimpan riwayat chat → hapus
    if (!$validated['keep_history']) {
        Message::where(function ($query) use ($validated) {
            $query->where('user_id', auth()->id())
                ->where('recipient_id', $validated['recipient_id']);
        })->orWhere(function ($query) use ($validated) {
            $query->where('user_id', $validated['recipient_id'])
                ->where('recipient_id', auth()->id());
        })->delete();
    }

    // Proses update status user seperti sebelumnya
    $user = auth()->user();
    $currentStatus = $user->status;

    if ($currentStatus instanceof RewardStatus && $currentStatus !== RewardStatus::LEVEL_10) {
        $levels = RewardStatus::cases();
        $nextLevelIndex = array_search($currentStatus, $levels) + 1;
        $nextLevel = $levels[$nextLevelIndex] ?? null;

        if ($nextLevel) {
            $user->update(['status' => $nextLevel]);
        }
    }

    return response()->json([
        'success' => true,
        'message' => $validated['keep_history']
            ? 'Chat ended, riwayat disimpan.'
            : 'Chat ended, riwayat dihapus.',
    ]);
}

    public function showCheckEmailForm()
    {
        return Inertia::render('CheckEmail');
    }

    // Memvalidasi email dan menampilkan halaman ganti password
    public function checkEmail(Request $request)
    {
        $request->validate([
            'email' => 'required|email',
        ]);

        $user = User::where('email', $request->input('email'))->first();

        if ($user) {
            return Inertia::render('GantiPassword', [
                'userId' => $user->id,
            ]);
        }

        return redirect()->back()->withErrors(['email' => 'Email tidak ditemukan']);
    }

    public function updatePassword(Request $request)
```

```
    {
        $request->validate([
            'user_id' => 'required|exists:users,id',
            'password' => 'required|confirmed|min:8',
        ]);

        $user = User::find($request->input('user_id'));

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

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

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


}
<?php

namespace App\Http\Controllers;

use App\Models\FAQ;
use Inertia\Inertia;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Facades\Redirect;
use Illuminate\Support\Str;
use Illuminate\Http\Request;
use Illuminate\Validation\Rule;

class FAQController extends Controller
{
    public function index()
    {
        $FAQ = FAQ::all();

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

    public function create()
    {
        return Inertia::render('FAQ/create');
    }
    public function store(Request $request)
    {
        // Validasi input
        $request->validate([
            'judul' => 'required',
            'pertanyaan' => 'required',
            'jawaban' => 'required',
        ]);

        // Menyimpan FAQ ke dalam database
        FAQ::create([
            'judul' => $request->input('judul'),
            'pertanyaan' => $request->input('pertanyaan'),
```

```
            'jawaban' => $request->input('jawaban'),
        ]);

        return Redirect::route('FAQ.index')->with('success', 'FAQ berhasil ditambahkan');
    }


    public function edit(FAQ $FAQ)
    {
        return Inertia::render('FAQ/update', [
            'FAQ' => $FAQ,
        ]);
    }
    public function update(Request $request, FAQ $FAQ)
    {
        $request->validate([
            'judul' => 'required',
            'pertanyaan' => 'required',
            'jawaban' => 'required',
        ]);

        $FAQ->update([
            'judul' => $request->input('judul'),
            'pertanyaan' => $request->input('pertanyaan'),
            'jawaban' => $request->input('jawaban'),
        ]);
        return Redirect::route('FAQ.index')->with('success', 'FAQ berhasil di update');
    }

    public function destroy(FAQ $FAQ)
    {
        $FAQ->delete();

        return Redirect::route('FAQ.index')->with('success', 'FAQ berhasil di hapus');
    }
}
<?php

namespace App\Http\Controllers;

use App\Models\Jadwal;
use App\Models\User;
use Inertia\Inertia;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Facades\Redirect;
use Illuminate\Support\Str;
use Illuminate\Http\Request;
use Illuminate\Validation\Rule;
use App\Enums\RewardStatus;

class JadwalController extends Controller
{
    public function index()
    {
        $user = auth()->user(); // Ambil user yang sedang login

        // Jika user memiliki role "mahasiswa", tampilkan hanya jadwal miliknya
        if ($user->hasRole('mahasiswa')) {
            $Jadwal = Jadwal::where('user_id', $user->id)->get();
        } else {
            // Jika bukan mahasiswa, tampilkan semua jadwal
            $Jadwal = Jadwal::all();
```

```
    }

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


public function create()
{
    return Inertia::render('Jadwal/create');
}
public function store(Request $request)
{
    // Validasi input
    $request->validate([

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

    // Ambil nomor antrian terakhir untuk hari ini
    $nomorAntrianTerakhir = Jadwal::whereDate('created_at', today())->max('antrian');
    $nomorAntrianBaru = $nomorAntrianTerakhir ? $nomorAntrianTerakhir + 1 : 1;

    // Menyimpan Jadwal ke dalam database
    Jadwal::create([
        'user_id' => auth()->id(),
        'nama' => auth()->user()->name,
        'nim' => auth()->user()->nim,
        'waktu' => now(),
        'status' => 'antrian',
        'antrian' => $nomorAntrianBaru,
        'fakultas' => auth()->user()->fakultas,
        'jurusan' => auth()->user()->jurusan,
        'masalah' => $request->input('masalah'),
    ]);

    // Ambil user yang sedang login
    $user = auth()->user();

    // Pastikan status user sudah dalam bentuk Enum
    $currentStatus = $user->status; // Tanpa tryFrom

    if ($currentStatus instanceof RewardStatus && $currentStatus !== RewardStatus::LEVEL_10) {
        // Ambil level berikutnya dalam enum
        $levels = RewardStatus::cases();
        $nextLevelIndex = array_search($currentStatus, $levels) + 1;
        $nextLevel = $levels[$nextLevelIndex] ?? null;

        if ($nextLevel) {
            // Update status user ke level berikutnya
            $user->update(['status' => $nextLevel]);
        }
    }

    return Redirect::route('Jadwal.index')->with('success', 'Jadwal berhasil ditambahkan dan status user diperbarui.');
}

public function edit(Jadwal $Jadwal)
{
    return Inertia::render('Jadwal/update', [
        'Jadwal' => $Jadwal,
```

```
        ]);
    }
    public function update(Request $request, Jadwal $Jadwal)
    {
        $request->validate([
            'nim' => 'required',
            'fakultas' => 'required',
            'jurusan' => 'required',
            'masalah' => 'required',
        ]);

        $Jadwal->update([
            'nim' => $request->input('nim'),
            'fakultas' => $request->input('fakultas'),
            'jurusan' => $request->input('jurusan'),
            'masalah' => $request->input('masalah'),
        ]);
        return Redirect::route('Jadwal.index')->with('success', 'Jadwal berhasil di update');
    }
    public function selesai(Jadwal $Jadwal)
    {
        $Jadwal->update(['status' => 'selesai']);

        return Redirect::route('Jadwal.index')->with('success', 'Jadwal berhasil di selesaikan');
    }

    public function destroy(Jadwal $Jadwal)
    {
        $Jadwal->delete();

        return Redirect::route('Jadwal.index')->with('success', 'Jadwal berhasil di hapus');
    }
}
<?php

namespace App\Http\Controllers;

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

class TestiController extends Controller
{

    public function index()
{
    $user = auth()->user(); // Ambil user yang sedang login

    // Jika user memiliki role "mahasiswa", tampilkan hanya jadwal miliknya
    if ($user->hasRole('mahasiswa')) {
        $Testi = Testi::where('user_id', $user->id)->get();
    } else {
        // Jika bukan mahasiswa, tampilkan semua jadwal
        $Testi = Testi::all();
    }

    return Inertia::render('Testimoni/index', ['Testi' => $Testi]);
}
public function index2()
{
    $Testi = Testi::where('rating', 5)
    ->orderBy('created_at', 'desc')
    ->limit(5)
    ->get();
```

```
    return Inertia::render('Welcome', ['Testi' => $Testi]);
}


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

    public function store(Request $request)
    {
        $request->validate([
            'rating' => 'required|integer|min:1|max:5',
            'pendapat' => 'nullable|string|max:255',
        ]);

        Testi::create([
            'user_id' => auth()->id(),
            'nama' => auth()->user()->name,
            'nim' => auth()->user()->nim,
            'fakultas' => auth()->user()->fakultas,
            'jurusan' => auth()->user()->jurusan,
            'rating' => $request->input('rating'),
            'pendapat' => $request->input('pendapat'),
        ]);

        return Redirect::route('Testi.index')->with('success', 'Testi berhasil di tambah');
    }

    public function edit(Testi $Testi)
    {
        return Inertia::render('Testimoni/update', [
            'Testi' => $Testi,
        ]);
    }
    public function update(Request $request, Testi $Testi)
    {
        $request->validate([
            'nim' => 'required',
            'fakultas' => 'required',
            'jurusan' => 'required',
            'rating' => 'required|integer|min:1|max:5',
            'pendapat' => 'nullable|string|max:255',
        ]);

        $Testi->update([
            'nim' => $request->input('nim'),
            'fakultas' => $request->input('fakultas'),
            'jurusan' => $request->input('jurusan'),
            'rating' => $request->input('rating'),
            'pendapat' => $request->input('pendapat'),
        ]);
        return Redirect::route('Testi.index')->with('success', 'Testi berhasil di update');
    }

    public function destroy(Testi $Testi)
    {
        $Testi->delete();

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

ROUTE:

```
. <?php

use App\Http\Controllers\ProfileController;
use App\Http\Controllers\ChatController;
use App\Http\Controllers\TestiController;
use App\Http\Controllers\FAQController;
use App\Http\Controllers\JadwalController;
use App\Http\Controllers\AdminController;
use App\Models\User;
use App\Models\ChatMessage;
use App\Events\MessageSent;
use Illuminate\Foundation\Application;
use Illuminate\Support\Facades\Route;
use Inertia\Inertia;

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

Route::get('/', [TestiController::class, 'index2'])->name('testimoni.index');

Route::get('/lupa-password', [ChatController::class, 'showCheckEmailForm'])->name('show.check.email');
Route::post('/check-email', [ChatController::class, 'checkEmail'])->name('check.email');
Route::post('/update-password', [ChatController::class, 'updatePassword'])->name('update.password');

Route::get('/dashboard', function () {
    return Inertia::render('Dashboard');
})->middleware(['auth', 'verified'])->name('dashboard');

Route::middleware('auth')->group(function () {
    Route::get('/profile', [ProfileController::class, 'edit'])->name('profile.edit');
    Route::patch('/profile', [ProfileController::class, 'update'])->name('profile.update');
    Route::delete('/profile', [ProfileController::class, 'destroy'])->name('profile.destroy');

    Route::get('/home', [ChatController::class, 'chatUsers'])->name('chat.users');
    Route::get('/chat/{userId}', [ChatController::class, 'chatBox'])->name('chat.box');
    Route::get('/users', [ChatController::class, 'index2'])->name('users');
    Route::get('/messages', [ChatController::class, 'messages'])
        ->name('messages');
    Route::post('/message', [ChatController::class, 'message'])
        ->name('message');
    Route::post('/chat/end', [ChatController::class, 'deleteChat'])->name('chat.end');

    Route::get('/FAQ', [FAQController::class, 'index'])->name('FAQ.index');
    Route::get('/FAQ/create', [FAQController::class, 'create'])->name('FAQ.create');
    Route::post('/FAQ/store', [FAQController::class, 'store'])->name('FAQ.store');
    Route::get('/FAQ/edit/{FAQ}', [FAQController::class, 'edit'])->name('FAQ.edit');
```

```
    Route::post('/FAQ/update/{FAQ}', [FAQController::class, 'update'])->name('FAQ.update');
    Route::delete('/FAQ/delete/{FAQ}', [FAQController::class, 'destroy'])->name('FAQ.delete');

    Route::get('/Jadwal', [JadwalController::class, 'index'])->name('Jadwal.index');
    Route::get('/Jadwal/create', [JadwalController::class, 'create'])->name('Jadwal.create');
    Route::post('/Jadwal/store', [JadwalController::class, 'store'])->name('Jadwal.store');
    Route::get('/Jadwal/edit/{Jadwal}', [JadwalController::class, 'edit'])->name('Jadwal.edit');
    Route::post('/Jadwal/update/{Jadwal}', [JadwalController::class, 'update'])->name('Jadwal.update');
    Route::post('Jadwal/selesai{jadwal}', [JadwalController::class, 'selesai'])->name('Jadwal.selesai');
    Route::delete('/Jadwal/delete/{Jadwal}', [JadwalController::class, 'destroy'])->name('Jadwal.delete');

    Route::get('/Testi', [TestiController::class, 'index'])->name('Testi.index');
    Route::get('/Testi/create', [TestiController::class, 'create'])->name('Testi.create');
    Route::post('/Testi/store', [TestiController::class, 'store'])->name('Testi.store');
    Route::get('/Testi/edit/{Testi}', [TestiController::class, 'edit'])->name('Testi.edit');
    Route::post('/Testi/update/{Testi}', [TestiController::class, 'update'])->name('Testi.update');
    Route::delete('/Testi/delete/{Testi}', [TestiController::class, 'destroy'])->name('Testi.delete');
});


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

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

});
require __DIR__ . '/auth.php';
```

VIEWS:

```
import './bootstrap';
import '../css/app.css';

import { createRoot } from 'react-dom/client';
import { createInertiaApp } from '@inertiajs/react';
import { resolvePageComponent } from 'laravel-vite-plugin/inertia-helpers';

const appName = import.meta.env.VITE_APP_NAME || 'Laravel';

createInertiaApp({
    title: (title) => `${title} - ${appName}`,
    resolve: (name) => resolvePageComponent(`./Pages/${name}.jsx`, import.meta.glob('./Pages/**/*.jsx')),
    setup({ el, App, props }) {
        const root = createRoot(el);

        root.render(<App {...props} />);
    },
    progress: {
        color: '#4B5563',
    },
});
// Import statements
import React from "react";
import PublicLayout from "@/Layouts/PublicLayout";
import { IconStarFilled,IconBrandWhatsapp, IconBrandInstagram, IconBrandFacebook, IconBrandX } from
"@tabler/icons-react";

function Welcome({Testi}) {
    return (
        <PublicLayout>
```

```
<div className="bg-gradient-to-b from-green-100 via-teal-100 to-blue-100 min-h-screen flex flex-col pt-16">
  {/* Hero Section */}
  <header className="bg-gradient-to-r from-teal-400 to-blue-500 text-white py-20 text-center">
    <h2 className="text-4xl font-bold">
      Selamat Datang di Helpdesk Kampus UMPAR
    </h2>
    <p className="mt-4 text-lg py-3">
      Solusi cepat untuk kebutuhan dan pertanyaan mahasiswa.
    </p>
    <a
      href="/login"
      className="mt-6 px-6 py-2 bg-white text-teal-500 font-bold rounded-lg shadow-md hover:bg-teal-100"
    >
      Mulai Sekarang
    </a>
  </header>

  <div className="bg-white dark:bg-gray-900 py-6 px-6 sm:px-6 lg:px-8">
    <div className="flex flex-wrap justify-center items-center gap-6 md:gap-10">
      <img
        src="/img/logoumparr.png"
        alt="Universitas Logo"
        width={120}
        height={120}
      />
      <img
        src="/img/kampus-merdeka.png"
        alt="Kampus Merdeka"
        width={150}
        height={80}
      />
      <img
        src="/img/akreditasi-b.png"
        alt="Akreditasi"
        width={100}
        height={100}
      />
      <img
        src="/img/khazanah-bugis.png"
        alt="Khazanah Bugis"
        width={120}
        height={120}
      />
    </div>
    <div className="mt-6 bg-blue-100 dark:bg-blue-900 text-center py-3 px-4 rounded-lg">
      <p className="text-gray-700 dark:text-gray-200 text-sm md:text-base font-semibold">
        Visi: Menjadi Perguruan Tinggi yang Unggul dalam
        Ilmu Pengetahuan, Teknologi dan Seni berdasarkan
        Nilai-nilai Islam Tahun 2023.
      </p>
    </div>
  </div>

  {/* About Section */}
  <section id="about" className="py-16 bg-green-50">
    <div className="container mx-auto px-4 text-center">
      <h3 className="text-2xl font-bold text-teal-800">
        Apa itu Helpdesk Kampus?
      </h3>
      <p className="mt-4 text-teal-600">
        Helpdesk Kampus adalah platform yang dirancang untuk
```

```
            membantu mahasiswa dalam mengakses informasi,
            memecahkan masalah akademik, dan memberikan solusi
            seputar kebutuhan kampus dengan cepat dan
            efjawabanen.
          </p>
        </div>
      </section>

      {/* Features Section */}
      <section id="features" className="py-16 bg-white">
        <div className="container mx-auto px-4 text-center">
          <h3 className="text-2xl font-bold text-teal-800">
            Fitur Utama
          </h3>
          <div className="grid grid-cols-1 md:grid-cols-3 gap-8 mt-8">
            <div className="p-6 bg-green-50 shadow-md rounded-lg">
              <h4 className="text-xl font-bold text-teal-600">
                FAQ Otomatis
              </h4>
              <p className="mt-2 text-teal-600">
                Dapatkan jawaban instan untuk pertanyaan
                umum tentang kampus.
              </p>
            </div>
            <div className="p-6 bg-green-50 shadow-md rounded-lg">
              <h4 className="text-xl font-bold text-teal-600">
                Layanan Tiket
              </h4>
              <p className="mt-2 text-teal-600">
                Ajukan masalah atau keluhan dan pantau
                statusnya secara real-time.
              </p>
            </div>
            <div className="p-6 bg-green-50 shadow-md rounded-lg">
              <h4 className="text-xl font-bold text-teal-600">
                Live Chat
              </h4>
              <p className="mt-2 text-teal-600">
                Berinteraksi langsung dengan petugas kampus
                untuk mendapatkan bantuan.
              </p>
            </div>
          </div>
        </div>
      </section>

    {/* Testimonials Section */}
    <section id="testimonials" className="py-16 bg-green-100">
      <div className="container mx-auto px-4 text-center">
        <h3 className="text-2xl font-bold text-teal-800">
          Apa Kata Mereka?
        </h3>
        {Testi.length > 0 ? (
          <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-8 mt-8">
            {Testi.map((testi) => (
              <div
                key={testi.id}
                className="p-6 bg-white shadow-lg rounded-lg"
              >
```

```
                <p className="italic text-teal-600">
                  "{testi.pendapat}"
                </p>
                <div className="flex justify-center gap-1 text-yellow-500">
                    {[...Array(testi.rating)].map((_, i) => (
                      <IconStarFilled key={i} className="w-5 h-5" />
                    ))}
                  </div>
                <h4 className="mt-4 text-teal-600 font-bold">
                  - {testi.nama}, {testi.fakultas}, {testi.jurusan}
                </h4>
              </div>
            ))}
          </div>
        ) : (
          <p className="text-gray-500 mt-6">
            Belum ada testimoni dengan rating 5.
          </p>
        )}
      </div>
    </section>

      {/* Contact Section */}
      <section id="contact" className="py-16 bg-teal-50">
    <div className="container mx-auto px-4 text-center">
      <h3 className="text-2xl font-bold text-teal-800">Kontak Kami</h3>
      <p className="mt-4 text-teal-600">
        Ikuti kami di sosial media untuk update terbaru dan bantuan cepat.
      </p>

      {/* Sosial Media Links */}
      <div className="mt-8 flex flex-wrap justify-center gap-6">
        {/* WhatsApp */}
        <a
          href="https://wa.me/6281234567890"
          target="_blank"
          rel="noopener noreferrer"
          className="flex items-center space-x-2 text-teal-600 hover:text-teal-800 transition duration-300"
        >
          <IconBrandWhatsapp className="h-14 w-14" />
          <span className="hidden sm:inline">WhatsApp</span>
        </a>

        {/* Instagram */}
        <a
          href="https://instagram.com/yourusername"
          target="_blank"
          rel="noopener noreferrer"
          className="flex items-center space-x-2 text-teal-600 hover:text-teal-800 transition duration-300"
        >
          <IconBrandInstagram className="h-14 w-14" />
          <span className="hidden sm:inline">Instagram</span>
        </a>

        {/* Facebook */}
        <a
          href="https://facebook.com/yourusername"
          target="_blank"
          rel="noopener noreferrer"
          className="flex items-center space-x-2 text-teal-600 hover:text-teal-800 transition duration-300"
```

```
                >
                  <IconBrandFacebook className="h-14 w-14" />
                  <span className="hidden sm:inline">Facebook</span>
                </a>

                {/* Twitter */}
                <a
                  href="https://twitter.com/yourusername"
                  target="_blank"
                  rel="noopener noreferrer"
                  className="flex items-center space-x-2 text-teal-600 hover:text-teal-800 transition duration-300"
                >
                  <IconBrandX className="h-14 w-14" />
                  <span className="hidden sm:inline">Twitter</span>
                </a>
              </div>
            </div>
          </section>

            {/* Footer */}
            <footer className="bg-teal-500 py-4 text-center text-white">
              <p>
                &copy; {new Date().getFullYear()} Helpdesk UMPAR. Semua
                Hak Dilindungi.
              </p>
            </footer>
          </div>
        </PublicLayout>
    );
}

export default Welcome;
import { useEffect, useState } from 'react';
import Checkbox from '@/Components/Checkbox';
import GuestLayout from '@/Layouts/GuestLayout';
import InputError from '@/Components/InputError';
import InputLabel from '@/Components/InputLabel';
import PrimaryButton from '@/Components/PrimaryButton';
import TextInput from '@/Components/TextInput';
import { Head, Link, useForm } from '@inertiajs/react';

export default function Login({ status, canResetPassword }) {
    const { data, setData, post, processing, errors, reset } = useForm({
        email: '',
        password: '',
        remember: false,
    });

    useEffect(() => {
        return () => {
            reset('password');
        };
    }, []);

    const submit = (e) => {
        e.preventDefault();
        post(route('login'));
    };

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

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

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

        <form onSubmit={submit} className="bg-white p-6 rounded-lg shadow-md">
            <div>
                <InputLabel htmlFor="email" value="Email" />
                <TextInput
                    id="email"
                    type="email"
                    name="email"
                    value={data.email}
                    className="mt-1 block w-full border border-teal-300 rounded-md shadow-sm focus:ring-teal-500 focus:border-teal-500"
                    autoComplete="username"
                    isFocused={true}
                    onChange={(e) => setData('email', e.target.value)}
                />
                <InputError message={errors.email} className="mt-2" />
            </div>

            <div className="relative mt-4">
                <InputLabel htmlFor="password" value="Password" />
                <TextInput
                    id="password"
                    type={showPassword ? "text" : "password"}
                    name="password"
                    value={data.password}
                    className="mt-1 block w-full border border-teal-300 rounded-md shadow-sm focus:ring-teal-500 focus:border-teal-500"
                    autoComplete="current-password"
                    onChange={(e) => setData("password", e.target.value)}
                />
                <button
                    type="button"
                    className="absolute right-3 top-1/2 transform -translate-y-1/2 text-teal-600"
                    onClick={() => setShowPassword(!showPassword)}
                >
                    {showPassword ? "Hide" : "Show"}
                </button>
                <InputError message={errors.password} className="mt-2" />
            </div>

            <div className="block mt-4">
                <label className="flex items-center justify-between">
                    <div>
                    <Checkbox
                        name="remember"
                        checked={data.remember}
                        onChange={(e) => setData('remember', e.target.checked)}
                    />
                    <span className="ms-2 text-sm text-teal-600">Remember me</span>
                    </div>
                    <Link
                        href={route('show.check.email')}
                        className="text-sm text-teal-600 hover:text-teal-800 font-semibold"
```

```
                >
                    Lupa Password?
                </Link>
            </label>
        </div>

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

            {canResetPassword && (
                <Link
                    href={route('register')}
                    className="text-sm text-teal-600 hover:text-teal-800 font-semibold"
                >
                    Don't have an account?
                </Link>
            )}
            <PrimaryButton className="bg-teal-600 hover:bg-teal-700 text-white px-4 py-2 rounded-lg shadow-md" disabled={processing}>
                Log in
            </PrimaryButton>
        </div>
    </form>
</GuestLayout>
);
}
import { useEffect } from 'react';
import GuestLayout from '@/Layouts/GuestLayout';
import InputError from '@/Components/InputError';
import InputLabel from '@/Components/InputLabel';
import PrimaryButton from '@/Components/PrimaryButton';
import TextInput from '@/Components/TextInput';
import { Head, Link, useForm } from '@inertiajs/react';

export default function Register() {
    const { data, setData, post, processing, errors, reset } = useForm({
        name: '',
        email: '',
        nim: '',
        fakultas: '',
        jurusan: '',
        password: '',
        password_confirmation: '',
    });

    useEffect(() => {
        return () => {
            reset('password', 'password_confirmation');
        };
    }, []);

    const submit = (e) => {
        e.preventDefault();
        post(route('register'));
    };

    return (
        <GuestLayout>
            <Head title="Register" />
            <div className="bg-white dark:bg-gray-800 p-6 rounded-lg shadow-lg w-full max-w-md">
                <h2 className="text-center text-2xl font-bold text-gray-900 dark:text-gray-100">Register</h2>
```

```
<form onSubmit={submit} className="mt-6">
  <div>
    <InputLabel htmlFor="name" value="Name" />
    <TextInput
      id="name"
      name="name"
      value={data.name}
      className="mt-1 block w-full border border-teal-300 rounded-md shadow-sm focus:ring-teal-500 focus:border-teal-500"
      autoComplete="name"
      isFocused={true}
      onChange={(e) => setData('name', e.target.value)}
      required
    />
    <InputError message={errors.name} className="mt-2" />
  </div>
  {/* Input NIM */}
  <div className="mt-4">
    <InputLabel htmlFor="nim" value="NIM" />
    <TextInput
      id="nim"
      type="text"
      name="nim"
      value={data.nim}
      className="mt-1 block w-full border border-teal-300 rounded-md shadow-sm focus:ring-teal-500 focus:border-teal-500"
      onChange={(e) => setData('nim', e.target.value)}
      required
    />
    <InputError message={errors.nim} className="mt-2" />
  </div>

  {/* Input Fakultas */}
  <div className="mt-4">
    <InputLabel htmlFor="fakultas" value="Fakultas" />
    <TextInput
      id="fakultas"
      type="text"
      name="fakultas"
      value={data.fakultas}
      className="mt-1 block w-full border border-teal-300 rounded-md shadow-sm focus:ring-teal-500 focus:border-teal-500"
      onChange={(e) => setData('fakultas', e.target.value)}
      required
    />
    <InputError message={errors.fakultas} className="mt-2" />
  </div>

  {/* Input Jurusan */}
  <div className="mt-4">
    <InputLabel htmlFor="jurusan" value="Jurusan" />
    <TextInput
      id="jurusan"
      type="text"
      name="jurusan"
      value={data.jurusan}
      className="mt-1 block w-full border border-teal-300 rounded-md shadow-sm focus:ring-teal-500 focus:border-teal-500"
      onChange={(e) => setData('jurusan', e.target.value)}
      required
    />
    <InputError message={errors.jurusan} className="mt-2" />
  </div>
```

```
            <div className="mt-4">
              <InputLabel htmlFor="email" value="Email" />
              <TextInput
                id="email"
                type="email"
                name="email"
                value={data.email}
                className="mt-1 block w-full border border-teal-300 rounded-md shadow-sm focus:ring-teal-500 focus:border-teal-500"
                autoComplete="username"
                onChange={(e) => setData('email', e.target.value)}
                required
              />
              <InputError message={errors.email} className="mt-2" />
            </div>


            <div className="mt-4">
              <InputLabel htmlFor="password" value="Password" />
              <TextInput
                id="password"
                type="password"
                name="password"
                value={data.password}
                className="mt-1 block w-full border border-teal-300 rounded-md shadow-sm focus:ring-teal-500 focus:border-teal-500"
                autoComplete="new-password"
                onChange={(e) => setData('password', e.target.value)}
                required
              />
              <InputError message={errors.password} className="mt-2" />
            </div>

            <div className="mt-4">
              <InputLabel htmlFor="password_confirmation" value="Confirm Password" />
              <TextInput
                id="password_confirmation"
                type="password"
                name="password_confirmation"
                value={data.password_confirmation}
                className="mt-1 block w-full border border-teal-300 rounded-md shadow-sm focus:ring-teal-500 focus:border-teal-500"
                autoComplete="new-password"
                onChange={(e) => setData('password_confirmation', e.target.value)}
                required
              />
              <InputError message={errors.password_confirmation} className="mt-2" />
            </div>

            <div className="flex items-center justify-between mt-6">
              <Link
                href={route('login')}
                className="text-sm text-teal-600 hover:text-teal-800 font-semibold"
              >
                Already registered?
```

```
                </Link>
                <PrimaryButton class="bg-teal-600 hover:bg-teal-700 text-white px-4 py-2 rounded-lg shadow-md" disabled={processing}>
                    Register
                </PrimaryButton>
            </div>
        </form>
    </div>
</GuestLayout>
);
}

import AuthenticatedLayout from '@/Layouts/AuthenticatedLayout';
import Status from '@/Components/Status';
import { Link } from '@inertiajs/react';
import { IconMessageCircle, IconHelpCircle, IconTicket, IconUsers } from '@tabler/icons-react';

export default function Dashboard() {
    return (
        <AuthenticatedLayout>
            <div class="h-auto w-full flex flex-col items-center justify-center p-6">


                {/* Menu Navigasi */}
                <div class="mt-8 grid grid-cols-2 md:grid-cols-4 gap-6 w-full max-w-4xl">
                    <Link
                        href="/FAQ"
                        class="flex flex-col items-center justify-center bg-blue-500 hover:bg-blue-600 text-white font-semibold p-4 rounded-lg shadow-lg transition-all duration-300"
                    >
                        <IconHelpCircle class="w-12 h-12" />
                        <span class="mt-2 text-sm">FAQ</span>
                    </Link>

                    <Link
                        href="/Jadwal"
                        class="flex flex-col items-center justify-center bg-green-500 hover:bg-green-600 text-white font-semibold p-4 rounded-lg shadow-lg transition-all duration-300"
                    >
                        <IconTicket class="w-12 h-12" />
                        <span class="mt-2 text-sm">Layanan Tiket</span>
                    </Link>

                    <Link
                        href="/home"
                        class="flex flex-col items-center justify-center bg-yellow-500 hover:bg-yellow-600 text-white font-semibold p-4 rounded-lg shadow-lg transition-all duration-300"
                    >
                        <IconMessageCircle class="w-12 h-12" />
                        <span class="mt-2 text-sm">Live Chat</span>
                    </Link>

                    <Link
                        href="/Testi"
                        class="flex flex-col items-center justify-center bg-purple-500 hover:bg-purple-600 text-white font-semibold p-4 rounded-lg shadow-lg transition-all duration-300"
                    >
                        <IconUsers class="w-12 h-12" />
                        <span class="mt-2 text-sm">Testimoni</span>
                    </Link>
```

```
            </div>
                        {/* Status Akun */}
                        <div className="w-full max-w-4xl">
                <Status />
            </div>
          </div>
      </AuthenticatedLayout>
    );
}
import React, { useState } from "react";
import { useForm, router } from "@inertiajs/react";
import AuthenticatedLayout from "@/Layouts/AuthenticatedLayout";
import Swal from "sweetalert2";
import { IconKey, IconTrash } from "@tabler/icons-react";
import image from "/public/img/user.png";
import InputError from "@/Components/InputError";
import InputLabel from "@/Components/InputLabel";
import PrimaryButton from "@/Components/PrimaryButton";
import TextInput from "@/Components/TextInput";

export default function Index({ user, success }) {
    const [password, setPassword] = useState('');
    const [errors, setErrors] = useState({});
    const [passwordConfirmation, setPasswordConfirmation] = useState('');
    const [selectedUser, setSelectedUser] = useState(null);
    const [showPassword, setShowPassword] = useState(false);

    const handleDelete = (id) => {
        Swal.fire({
            title: "Apakah Anda yakin?",
            text: "Data pengguna akan dihapus secara permanen!",
            icon: "warning",
            showCancelButton: true,
            confirmButtonColor: "#d33",
            cancelButtonColor: "#3085d6",
            confirmButtonText: "Ya, hapus!",
        }).then((result) => {
            if (result.isConfirmed) {
                router.delete(route("admin.destroy", id), {
                    onSuccess: () => {
                        Swal.fire({
                            title: "Terhapus!",
                            text: "Data pengguna berhasil dihapus.",
                            icon: "success",
                        });
                    },
                    onError: () => {
                        Swal.fire({
                            title: "Error!",
                            text: "Terjadi kesalahan saat menghapus pengguna.",
                            icon: "error",
                        });
                    },
                });
            }
        });
    };

    const handleSubmit = (e) => {
        e.preventDefault();

        if (!selectedUser || !selectedUser.id) {
            setErrors({ user_id: "Silakan pilih pengguna terlebih dahulu" });
```

```
        return;
    }

    router.post('/update-password', {
        user_id: selectedUser.id,
        password,
        password_confirmation: passwordConfirmation,
    }, {
        onError: (err) => setErrors(err),
        onSuccess: () => {
            setSelectedUser(null);
            setPassword('');
            setPasswordConfirmation('');
            Swal.fire({
                title: "Berhasil!",
                text: "Password berhasil diperbarui.",
                icon: "success",
            });
        }
    });
};

return (
    <AuthenticatedLayout>
        <div className="container mx-auto px-4 py-8 text-gray-900 dark:text-gray-200">
            {/* Header */}
            <section className="bg-gradient-to-r from-teal-400 to-blue-500 text-white py-12 rounded-lg shadow-md mb-8 text-center">
                <h1 className="text-4xl font-bold mb-2">Manajemen Pengguna</h1>
                <p className="text-lg">Kelola akun pengguna dengan mudah dan cepat.</p>
            </section>

            {/* Success Message */}
            {success && (
                <div className="mb-6 bg-green-100 border border-green-300 text-green-700 p-4 rounded-lg text-center shadow-sm">
                    {success}
                </div>
            )}

            {/* User Cards */}
            <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-6">
                {user.map((u) => (
                    <div key={u.id} className="bg-white dark:bg-gray-800 rounded-xl shadow-md p-4 flex flex-col items-center text-center border border-teal-100">
                        <img src={image} alt={u.name} className="w-24 h-24 rounded-full mb-4 shadow-md" />
                        <h2 className="text-lg font-bold text-teal-700">{u.name}</h2>
                        <p className="text-gray-600 text-sm mb-4">{u.email}</p>
                        <button
                            onClick={() => setSelectedUser(u)}
                            className="bg-teal-500 text-white py-2 px-4 rounded-lg shadow hover:bg-teal-600 flex items-center justify-center w-full mb-2"
                        >
                            <IconKey className="mr-2" /> Ganti Password
                        </button>
                        <button
                            onClick={() => handleDelete(u.id)}
```

```
                    className="bg-red-500 text-white py-2 px-4 rounded-lg shadow hover:bg-red-600 flex items-center
justify-center w-full"
                  >
                    <IconTrash className="mr-2" /> Hapus
                  </button>
                </div>
              ))}
            </div>

            {/* Modal Ganti Password */}
            {selectedUser && (
              <div className="fixed inset-0 z-50 flex items-center justify-center bg-black bg-opacity-50">
                <div className="bg-white dark:bg-gray-800 p-6 rounded-xl shadow-xl w-full max-w-md">
                  <h2 className="text-xl font-bold mb-4 text-teal-700">
                    Ganti Password {selectedUser.name}
                  </h2>
                  <form onSubmit={handleSubmit} className="space-y-6">
                    {/* Password */}
                    <div className="relative">
                      <InputLabel htmlFor="password" value="Password Baru" />
                      <TextInput
                        id="password"
                        type={showPassword ? "text" : "password"}
                        name="password"
                        value={password}
                        className="mt-1 block w-full border border-gray-300 rounded-lg px-4 py-2"
                        onChange={(e) => setPassword(e.target.value)}
                        placeholder="Masukkan password baru"
                      />
                      <button
                        type="button"
                        className="absolute right-3 top-1/2 transform -translate-y-1/2 text-sm text-gray-500"
                        onClick={() => setShowPassword(!showPassword)}
                      >
                        {showPassword ? "Sembunyikan" : "Lihat"}
                      </button>
                      <InputError message={errors.password} className="mt-2" />
                    </div>

                    {/* Konfirmasi Password */}
                    <div>
                      <InputLabel htmlFor="password_confirmation" value="Konfirmasi Password" />
                      <TextInput
                        id="password_confirmation"
                        type={showPassword ? "text" : "password"}
                        name="password_confirmation"
                        value={passwordConfirmation}
                        className="mt-1 block w-full border border-gray-300 rounded-lg px-4 py-2"
                        onChange={(e) => setPasswordConfirmation(e.target.value)}
                        placeholder="Ulangi password baru"
                      />
                      <InputError message={errors.password_confirmation} className="mt-2" />
                    </div>
```

```
                        <div className="flex justify-end space-x-2">
                            <button
                                type="button"
                                onClick={() => setSelectedUser(null)}
                                className="px-4 py-2 bg-gray-300 hover:bg-gray-400 text-gray-800 rounded-lg"
                            >
                                Batal
                            </button>
                            <PrimaryButton type="submit" className="bg-teal-600 hover:bg-teal-700 text-white px-4 py-2
rounded-lg">
                                Simpan
                            </PrimaryButton>
                        </div>
                    </form>
                </div>
            </div>
        )}
    </div>
</AuthenticatedLayout>
);
}
import { useState } from "react";
import Swal from "sweetalert2";
import { Link, router,usePage } from "@inertiajs/react";
import AuthenticatedLayout from "@/Layouts/AuthenticatedLayout";
import Button from "@/Components/Button";
import { IconPlus, IconTrash, IconEdit } from "@tabler/icons-react";

const Index = ({ FAQ }) => {
    const { auth } = usePage().props;
    const [isDeleting, setIsDeleting] = useState(false);
    const [openIndex, setOpenIndex] = useState(null);

    const toggleFAQ = (index) => {
        setOpenIndex(openIndex === index ? null : index);
    };

    const Delete = (id) => {
        Swal.fire({
            title: "Apakah Anda yakin?",
            text: "Anda tidak akan dapat mengembalikan ini!",
            icon: "warning",
            showCancelButton: true,
            confirmButtonColor: "#3085d6",
            cancelButtonColor: "#d33",
            confirmButtonText: "Ya, hapus saja!",
        }).then((result) => {
            if (result.isConfirmed) {
                setIsDeleting(true);
                router.delete(route("FAQ.delete", id), {
                    onSuccess: () => {
                        setIsDeleting(false);
                        Swal.fire("Dihapus!", "FAQ telah dihapus.", "success");
                    },
                    onError: () => {
                        setIsDeleting(false);
                        Swal.fire(
                            "Gagal!",
                            "Terjadi kesalahan saat menghapus FAQ.",
                            "error"
                        );
                    },
                });
```

```
        }
    });
};
const Daf = auth.roles.includes("operator") || auth.roles.includes("admin");
return (
    <AuthenticatedLayout>
        <div className="container mx-auto px-4 py-8">
            <div className="text-center text-gray-900 dark:text-white">
                <h1 className="text-3xl xl:text-5xl font-bold mb-6">
                    Daftar FAQ
                </h1>
                {Daf && (
                    <Link href={route("FAQ.create")}>
                        <Button className="flex items-center justify-center gap-2 bg-blue-600 hover:bg-blue-700 text-white font-semibold py-2 px-4 rounded-lg shadow-lg transition-all duration-300">
                            <IconPlus className="w-5 h-5" />
                            Tambah Masukan
                        </Button>
                    </Link>
                )}
            </div>

            {/* Daftar FAQ */}
            <div className="mt-8 grid grid-cols-1 md:grid-cols-2 xl:grid-cols-3 gap-6">
                {FAQ && FAQ.length > 0 ? (
                    FAQ.map((item, index) => (
                        <div
                            key={index}
                            className="bg-white dark:bg-gray-800 shadow-lg rounded-lg overflow-hidden transition-transform transform hover:scale-105 duration-300"
                        >
                            <div className="p-6">
                                <button
                                    onClick={() => toggleFAQ(index)}
                                    className="w-full text-left text-xl font-semibold text-gray-800 dark:text-gray-200 flex justify-between items-center"
                                >
                                    {item.judul}
                                    <span className="text-gray-500">
                                        {openIndex === index ? "−" : "+"}
                                    </span>
                                </button>

                                {openIndex === index && (
                                    <div className="mt-4 text-gray-700 dark:text-gray-300">
                                        <p className="font-medium">
                                            {item.pertanyaan}
                                        </p>
                                        <p
                                            className="mt-2 text-sm"
                                            dangerouslySetInnerHTML={{
                                                __html: item.jawaban.replace(
                                                    /\n/g,
                                                    "<br />"
                                                ),
                                            }}
                                        ></p>
                                    </div>
                                )}
                            </div>
                            {Daf && (
                                <div className="flex justify-between items-center bg-gray-100 dark:bg-gray-900 p-4">
```

```
                        <Link
                            href={route("FAQ.edit", item.id)}
                            className="text-blue-500 hover:text-blue-700 font-medium flex items-center"
                        >
                            <IconEdit className="w-5 h-5 mr-1" />{" "}
                            Edit
                        </Link>
                        <button
                            onClick={() => Delete(item.id)}
                            disabled={isDeleting}
                            className="text-red-500 hover:text-red-700 font-medium flex items-center"
                        >
                            <IconTrash className="w-5 h-5 mr-1" />{" "}
                            Hapus
                        </button>
                    </div>
                ))}
            </div>
        ))
    ) : (
        <p className="text-gray-500 text-center col-span-full">
            Tidak ada FAQ
        </p>
    )}
</div>
</div>
</AuthenticatedLayout>
);
};

export default Index;
```

```
import { useState } from "react";
import Swal from "sweetalert2";
import { Link, router } from "@inertiajs/react";
import AuthenticatedLayout from "@/Layouts/AuthenticatedLayout";
import Button from "@/Components/Button";
import { IconClock, IconTrash, IconEdit, IconUser, IconList, IconCheck } from "@tabler/icons-react";

const Index = ({ Jadwal }) => {
    const [isDeleting, setIsDeleting] = useState(false);

    const Delete = (id) => {
        Swal.fire({
            title: "Apakah Anda yakin?",
            text: "Anda tidak akan dapat mengembalikan ini!",
            icon: "warning",
            showCancelButton: true,
            confirmButtonColor: "#3085d6",
            cancelButtonColor: "#d33",
            confirmButtonText: "Ya, hapus saja!",
        }).then((result) => {
            if (result.isConfirmed) {
                setIsDeleting(true);
                router.delete(route("Jadwal.delete", id), {
                    onSuccess: () => {
                        setIsDeleting(false);
                        Swal.fire("Dihapus!", "Jadwal telah dihapus.", "success");
                    },
                    onError: () => {
                        setIsDeleting(false);
                        Swal.fire("Gagal!", "Terjadi kesalahan saat menghapus jadwal.", "error");
                    },
                });
            }
```

```
        });
    };
    const handleSelesai = (id) => {
        Swal.fire({
            title: "Tandai sebagai selesai?",
            text: "Data ini akan diupdate menjadi 'Selesai'.",
            icon: "question",
            showCancelButton: true,
            confirmButtonColor: "#16a34a",
            cancelButtonColor: "#d33",
            confirmButtonText: "Ya, selesai",
        }).then((result) => {
            if (result.isConfirmed) {
                router.post(route("Jadwal.selesai", id), {
                    onSuccess: () => {
                        Swal.fire("Berhasil!", "Status telah diubah menjadi selesai.", "success");
                    },
                    onError: () => {
                        Swal.fire("Gagal!", "Gagal menyelesaikan jadwal.", "error");
                    },
                });
            }
        });
    };

    return (
        <AuthenticatedLayout>
            <div className="container mx-auto px-4 py-8">
                <div className="text-center text-gray-900 dark:text-white">
                    <h1 className="text-3xl xl:text-5xl font-bold mb-6">Layanan Tiket</h1>
                    <p className="text-gray-600 dark:text-gray-400 mb-4">
                        Kelola jadwal helpdesk dengan mudah dan cepat.
                    </p>
                    <Link href={route("Jadwal.create")}>
                        <Button className="flex items-center justify-center gap-2 bg-blue-600 hover:bg-blue-700 text-white font-
semibold py-2 px-4 rounded-lg shadow-lg transition-all duration-300">
                            <IconList className="w-5 h-5" />
                            Tambah Jadwal
                        </Button>
                    </Link>
                </div>

                {/* Daftar Jadwal */}
                <div className="mt-8 grid grid-cols-1 md:grid-cols-2 xl:grid-cols-3 gap-6">
                    {Jadwal && Jadwal.length > 0 ? (
                        Jadwal.map((item, index) => (
                            <div
                                key={index}
                                className="bg-white dark:bg-gray-800 shadow-lg rounded-lg overflow-hidden transition-
transform transform hover:scale-105 duration-300"
                            >
                                <div className="p-6">
                                    {/* Nama */}
                                    <h2 className="text-xl font-semibold text-gray-800 dark:text-gray-200 flex items-center gap-2">
                                        <IconUser className="w-6 h-6 text-blue-500" />
                                        {item.nama}
                                    </h2>

                                    {/* NIM & Fakultas/Jurusan */}
                                    <p className="text-gray-600 dark:text-gray-400 mt-2">NIM: {item.nim}</p>
                                    <p className="text-gray-600 dark:text-gray-400">Fakultas: {item.fakultas} - {item.jurusan}</p>
```

```
                {/* Waktu */}
                <div className="mt-3 flex items-center gap-2 text-sm">
                  <IconClock className="w-5 h-5 text-yellow-500" />
                  <span className="font-semibold text-gray-700 dark:text-gray-300">{item.waktu}</span>
                </div>

                {/* Antrian & Status */}
                <div className="mt-4 space-y-2">
                  {/* Antrian */}
                  <span
                    className={`px-3 py-1 text-xs font-semibold rounded-full w-fit ${item.antrian <= 5
                      ? "bg-green-500 text-white"
                      : item.antrian <= 9
                        ? "bg-yellow-500 text-white"
                        : "bg-red-500 text-white"
                      }`}
                  >
                    Antrian: {item.antrian}
                  </span>

                  {/* Status */}
                  <span
                    className={`px-3 py-1 text-xs font-semibold rounded-full w-fit flex items-center gap-1 ${item.status.toLowerCase() === "selesai"
                      ? "bg-green-600 text-white"
                      : "bg-yellow-500 text-white"
                      }`}
                  >
                    Status:
                    {item.status.toLowerCase() === "selesai" ? (
                      <>
                        <IconCheck className="w-4 h-4 text-white" />
                        Selesai
                      </>
                    ) : (
                      <>
                        <IconClock className="w-4 h-4 text-white" />
                        {item.status}
                      </>
                    )}
                  </span>
                </div>

                {/* Masalah */}
                <p className="text-gray-700 dark:text-gray-300 mt-4 text-sm">
                  {item.masalah.length > 100 ? `${item.masalah.substring(0, 100)}...` : item.masalah}
                </p>
              </div>

              {/* Footer: Edit & Hapus */}
              <div className="flex justify-between items-center bg-gray-100 dark:bg-gray-900 p-4">
                <Link
                  href={route("Jadwal.edit", item.id)}
                  className="text-blue-500 hover:text-blue-700 font-medium flex items-center"
                >
                  <IconEdit className="w-5 h-5 mr-1" /> Edit
                </Link>
                <button
                  onClick={() => Delete(item.id)}
                  disabled={isDeleting}
                  className="text-red-500 hover:text-red-700 font-medium flex items-center"
```

```
                        >
                            <IconTrash className="w-5 h-5 mr-1" /> Hapus
                        </button>
                    </div>

                    {/* Tambahkan tombol selesaikan jika belum selesai */}
                    {item.status.toLowerCase() !== "selesai" && (
                        <div className="flex justify-center bg-gray-50 dark:bg-gray-800 p-3 border-t border-gray-200 dark:border-gray-700">
                            <button
                                onClick={() => handleSelesai(item.id)}
                                className="text-green-600 hover:text-green-800 font-medium flex items-center gap-1"
                            >
                                <IconCheck className="w-5 h-5" />
                                Tandai Selesai
                            </button>
                        </div>
                    )}

                </div>

            ))
        ) : (
            <p className="text-gray-500 text-center col-span-full">Tidak ada jadwal yang tersedia</p>
        )}
    </div>
</div>
</AuthenticatedLayout>
);
};

export default Index;
import { useState } from "react";
import Swal from "sweetalert2";
import { Link, router } from "@inertiajs/react";
import AuthenticatedLayout from "@/Layouts/AuthenticatedLayout";
import Button from "@/Components/Button";
import StarDisplay from "@/Components/StarDisplay";
import { IconTrash, IconEdit, IconStarFilled, IconUser } from "@tabler/icons-react";

const Index = ({ Testi }) => {
    const [isDeleting, setIsDeleting] = useState(false);

    const Delete = (id) => {
        Swal.fire({
            title: "Apakah Anda yakin?",
            text: "Anda tidak akan dapat mengembalikan ini!",
            icon: "warning",
            showCancelButton: true,
            confirmButtonColor: "#3085d6",
            cancelButtonColor: "#d33",
            confirmButtonText: "Ya, hapus saja!",
        }).then((result) => {
            if (result.isConfirmed) {
                setIsDeleting(true);
                router.delete(route("Testi.delete", id), {
                    onSuccess: () => {
                        setIsDeleting(false);
                        Swal.fire("Dihapus!", "Testimoni telah dihapus.", "success");
                    },
                    onError: () => {
                        setIsDeleting(false);
                        Swal.fire("Gagal!", "Terjadi kesalahan saat menghapus testimoni.", "error");
                    },
                });
```

```
        }
      });
  };

  return (
    <AuthenticatedLayout>
      <div className="container mx-auto px-4 py-8">
        <div className="text-center text-gray-900 dark:text-white">
          <h1 className="text-3xl xl:text-5xl font-bold mb-6">Testimoni Pengguna</h1>
          <p className="text-gray-600 dark:text-gray-400 mb-4">
            Lihat pengalaman dan ulasan dari pengguna kami.
          </p>
          <Link href={route("Testi.create")}>
            <Button className="flex items-center justify-center gap-2 bg-blue-600 hover:bg-blue-700 text-white font-semibold py-2 px-4 rounded-lg shadow-lg transition-all duration-300">
              Tambah Testimoni
            </Button>
          </Link>
        </div>

        {/* Daftar Testimoni */}
        <div className="mt-8 grid grid-cols-1 md:grid-cols-2 xl:grid-cols-3 gap-6">
          {Testi && Testi.length > 0 ? (
            Testi.map((item, index) => (
              <div
                key={index}
                className="bg-white dark:bg-gray-800 shadow-lg rounded-lg overflow-hidden transition-transform transform hover:scale-105 duration-300 p-6"
              >
                <div className="flex items-center mb-4">
                  <div className="w-12 h-12 bg-blue-500 text-white flex items-center justify-center rounded-full">
                    <IconUser className="w-6 h-6" />
                  </div>
                  <div className="ml-3">
                    <h2 className="text-lg font-semibold text-gray-800 dark:text-gray-200">{item.nama}</h2>
                    <div className="flex items-center gap-1 text-yellow-500">
                      {[...Array(item.rating)].map((_, i) => (
                        <IconStarFilled key={i} className="w-5 h-5" />
                      ))}
                    </div>
                  </div>
                </div>

                <p className="text-gray-700 dark:text-gray-300 mt-2 text-sm">
                  {item.pendapat.length > 150 ? `${item.pendapat.substring(0, 150)}...` : item.pendapat}
                </p>

                {/* Tombol Edit & Hapus */}
                <div className="flex justify-between items-center bg-gray-100 dark:bg-gray-900 p-4 mt-4">
                  <Link href={route("Testi.edit", item.id)} className="text-blue-500 hover:text-blue-700 font-medium flex items-center">
                    <IconEdit className="w-5 h-5 mr-1" /> Edit
                  </Link>
                  <button
                    onClick={() => Delete(item.id)}
                    disabled={isDeleting}
                    className="text-red-500 hover:text-red-700 font-medium flex items-center"
                  >
```

```
                                <IconTrash className="w-5 h-5 mr-1" /> Hapus
                            </button>
                        </div>
                    </div>
                ))
            ) : (
                <p className="text-gray-500 text-center col-span-full">Belum ada testimoni.</p>
            )}
        </div>
      </div>
    </AuthenticatedLayout>
  );
};

export default Index;
```

```
import React, { useEffect, useRef, useState } from "react";
import Message from "./Message.jsx";
import MessageInput from "./MessageInput.jsx";
import axios from "axios";
import { Link, usePage } from "@inertiajs/react";

const ChatBox = ({ rootUrl, recipient }) => {
  const { csrf_token } = usePage().props; // Untuk Laravel CSRF token
  const [messages, setMessages] = useState([]);
  const scroll = useRef();

  // Fungsi untuk mengambil ulang pesan
  const fetchMessages = async () => {
    try {
      const { data } = await axios.get(`${rootUrl}/messages`, { params: { recipient_id: recipient.id } });
      setMessages(data);
      scroll.current?.scrollIntoView({ behavior: "smooth" }); // Scroll ke bawah setelah refresh
    } catch (err) {
      console.error("Error fetching messages:", err.message);
    }
  };

  useEffect(() => {
    fetchMessages();

    return () => setMessages([]); // Bersihkan state saat unmount
  }, [rootUrl, recipient.id]);

const endChat = async () => {
  const keepHistory = window.confirm("Apakah Anda ingin menyimpan riwayat chat? Klik OK untuk Simpan, Cancel untuk Hapus.");

  try {
    await axios.post(`${rootUrl}/chat/end`, {
      recipient_id: recipient.id,
      keep_history: keepHistory // true jika OK, false jika Cancel
    }, {
      headers: { 'X-CSRF-TOKEN': csrf_token }
    });

    alert(keepHistory ? "Chat ended. Riwayat disimpan." : "Chat ended. Riwayat dihapus.");
    window.location.href = "/home";
  } catch (err) {
    console.error("Error ending chat:", err.message);
  }
};
```

```
    return (
        <div className="max-w-2xl mx-auto bg-white shadow rounded-lg">
            {/* Header */}
            <div className="p-4 border-b flex justify-between items-center">
                <Link href="/home" className="text-blue-500">← Kembali</Link>
                <h2 className="font-semibold">{recipient.name}</h2>
                <div className="flex space-x-2">
                    {/* <button onClick={fetchMessages} className="bg-gray-500 text-white px-3 py-1 rounded">
                        Refresh 🔄
                    </button> */}
                    <button onClick={endChat} className="bg-red-500 text-white px-3 py-1 rounded">
                        End Chat
                    </button>
                </div>
            </div>

            {/* Chat Messages */}
            <div className="h-96 overflow-y-auto p-4 space-y-3">
                {messages.map((message, index) => (
                    <Message key={index} message={message} />
                ))}
                <span ref={scroll}></span>
            </div>

            {/* Input Chat */}
            <div className="p-4 border-t">
                <MessageInput rootUrl={rootUrl} recipientId={recipient.id} />
            </div>
        </div>
    );
};

export default ChatBox; import React from "react";
import { Link, usePage } from "@inertiajs/react";
import AuthenticatedLayout from '@/Layouts/AuthenticatedLayout';

const ChatUsers = ({ users }) => {
    const { auth } = usePage().props; // Mengambil data user yang sedang login

    // Cek apakah user memiliki role "mahasiswa"
    const isMahasiswa = auth.roles.includes("mahasiswa");

    // Jika user adalah mahasiswa, hanya bisa melihat operator
    const filteredUsers = isMahasiswa
        ? users.filter(user => user.roles.includes("operator"))
        : users;

    return (
        <AuthenticatedLayout>
            <div className="max-w-lg mx-auto p-6 bg-white rounded shadow">
                <h2 className="text-lg font-semibold mb-4">Pilih pengguna untuk chat</h2>

                {filteredUsers.length > 0 ? (
                    <ul className="space-y-3">
                        {filteredUsers.map((user) => (
```

```
                    <li key={user.id} className="p-3 border rounded-lg hover:bg-gray-100">
                      <Link href={`/chat/${user.id}`} className="font-medium">
                        {user.name} ({user.roles.join(", ")})
                      </Link>
                    </li>
                  ))}
                </ul>
              ) : (
                <p className="text-gray-500">Tidak ada operator yang tersedia.</p>
              )}
            </div>
        </AuthenticatedLayout>
    );
};

export default ChatUsers;
import React from "react";

const Message = ({ userId, message }) => {
    const isCurrentUser = userId === message.user_id;

    return (
        <div
            className={`flex ${isCurrentUser ? "justify-end" : "justify-start"} mb-2`}
        >
            <div
                className={`${
                    isCurrentUser ? "bg-blue-500 text-white" : "bg-gray-200 text-black"
                } px-4 py-2 rounded max-w-xs break-words`}
            >
                {!isCurrentUser && message.user && (
                    <p className="font-bold text-sm mb-1">{message.user.name}</p>
                )}
                <p className="text-sm">{message.text}</p>

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

export default Message;
import React, { useState } from "react";
import axios from "axios";

const MessageInput = ({ rootUrl, recipientId }) => {
    const [message, setMessage] = useState("");

const sendMessage = async (e) => {
    e.preventDefault();
    const trimmedMessage = message.trim();
    if (trimmedMessage === "") {
        alert("Please enter a message!");
        return;
    }

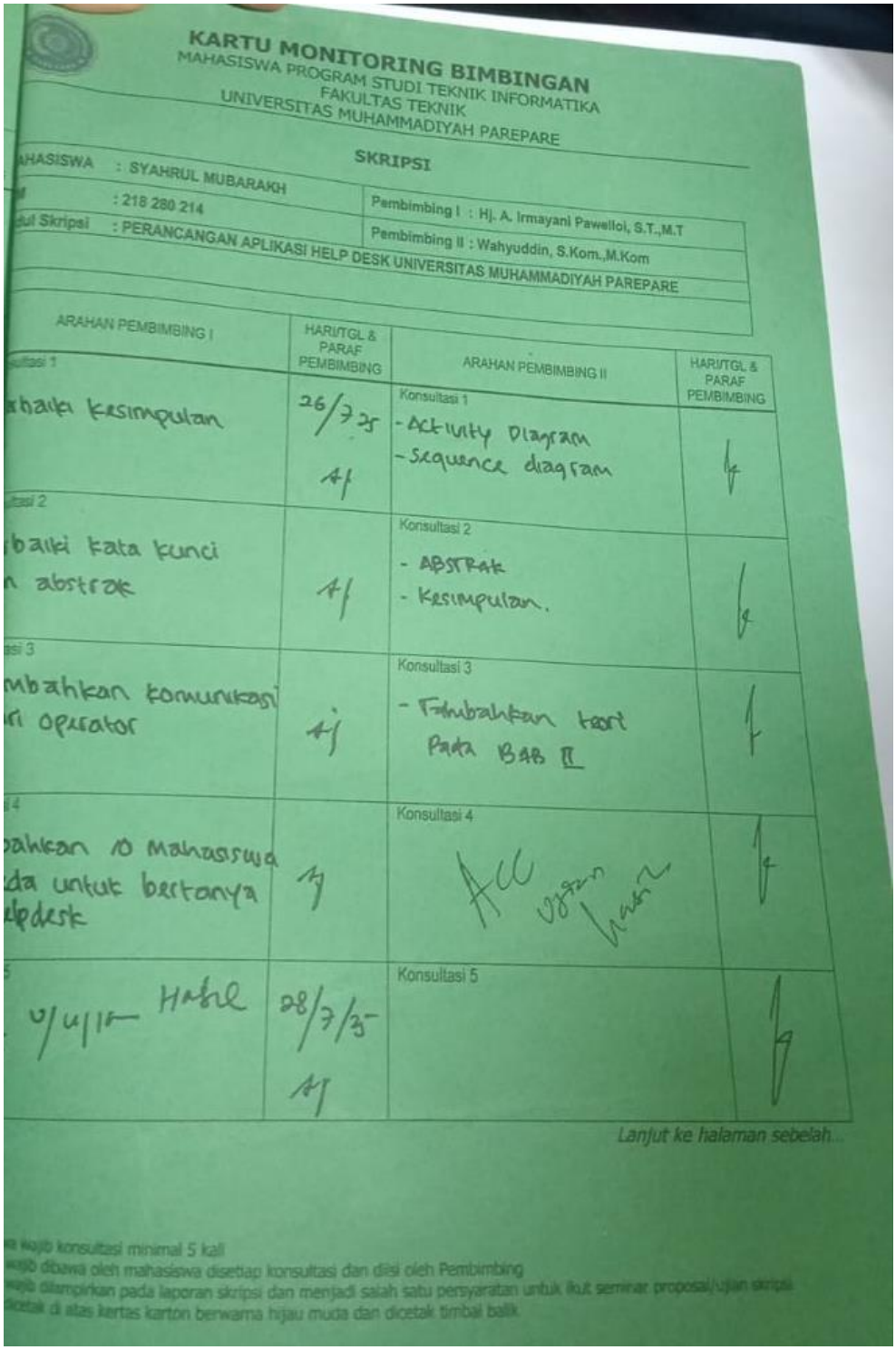
    try {
        const response = await axios.post(`${rootUrl}/message`, {
            text: trimmedMessage,
            recipient_id: recipientId,
        });

        // Cek respons dari backend dan reload jika berhasil
```

```
      if (response.data.success) {
        setMessage(""); // Kosongkan input
        location.reload(); // Refresh halaman
      }
    } catch (err) {
      console.error("Error sending message:", err.message);
    }
};


  return (
    <form onSubmit={sendMessage} className="flex items-center">
      <input
        type="text"
        className="flex-1 p-2 border border-gray-300 rounded-l focus:outline-none focus:ring-2 focus:ring-blue-500"
        placeholder="Type your message..."
        value={message}
        onChange={(e) => setMessage(e.target.value)}
        autoComplete="off"
      />
      <button
        type="submit"
        className="px-4 bg-blue-500 text-white rounded-r hover:bg-blue-600"
      >
        Send
      </button>
    </form>
  );
};

export default MessageInput;
```



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

SKRIPSI

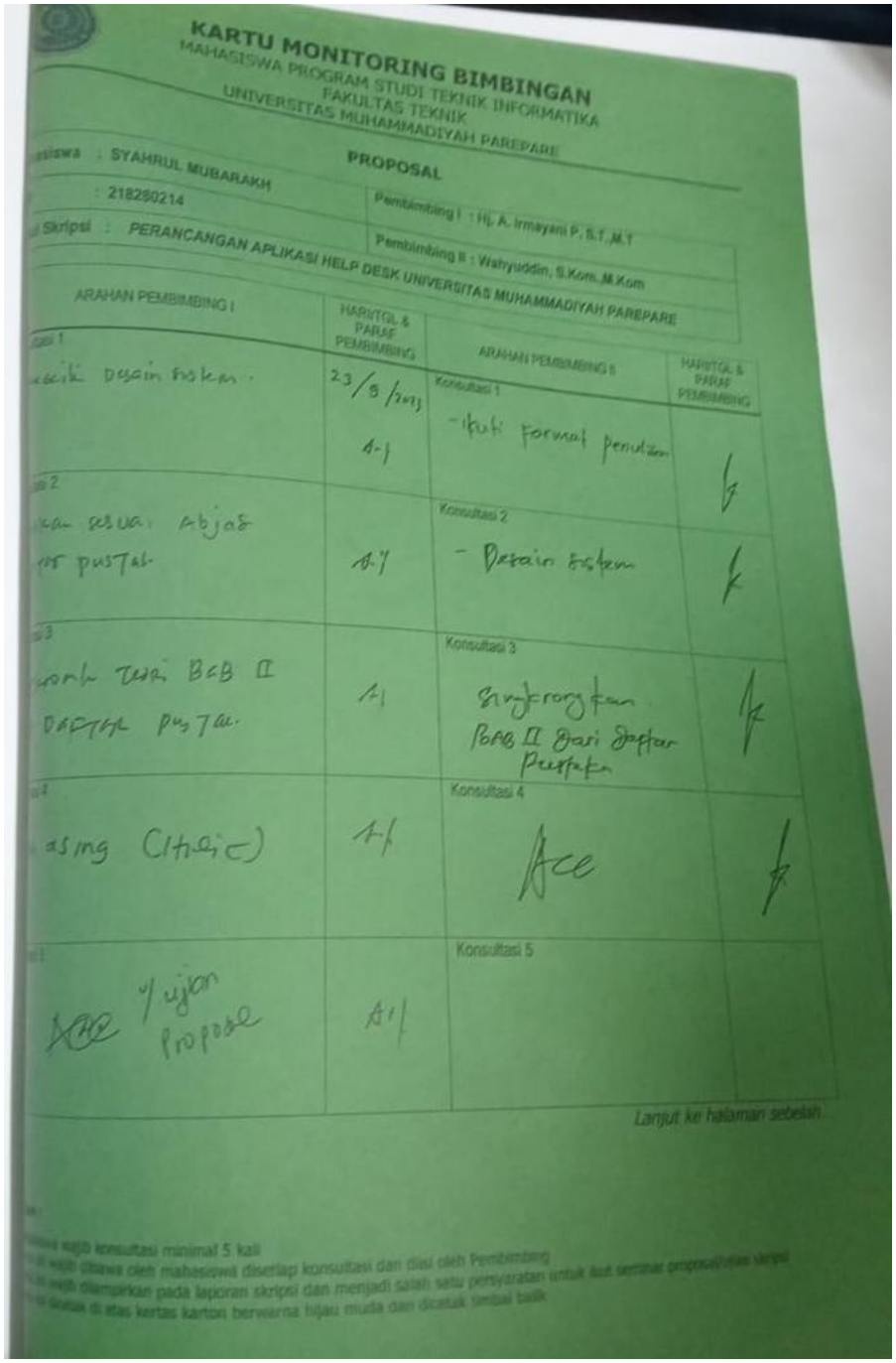
AHASISWA : SYAHRUL MUBARAKH
: 218 280 214
ul Skripsi : PERANCANGAN APLIKASI HELP DESK UNIVERSITAS MUHAMMADIYAH PAREPARE

Pembimbing I : Hj. A. Irmayani Pawelloi, S.T.,M.T
Pembimbing II : Wahyuddin, S.Kom.,M.Kom

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
tasi 1 baiki kesimpulan	26/7 25 Af	Konsultasi 1 - Activity Diagram - Sequence diagram	
tasi 2 baiki kata kunci n abstrak	Af	Konsultasi 2 - ABSTRAK - Kesimpulan.	
si 3 mbahkan komunikasi ri operator	Af	Konsultasi 3 - Tambahkan teori Pada Bab II	
i 4 ahkan 10 mahasiswa da untuk bertanya lpdesk		Konsultasi 4 ACC ujian hasil	
5 5/11/25 Hasil	28/7/25 Af	Konsultasi 5	

Lanjut ke halaman sebelah...

a wajib konsultasi minimal 5 kali
wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik



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

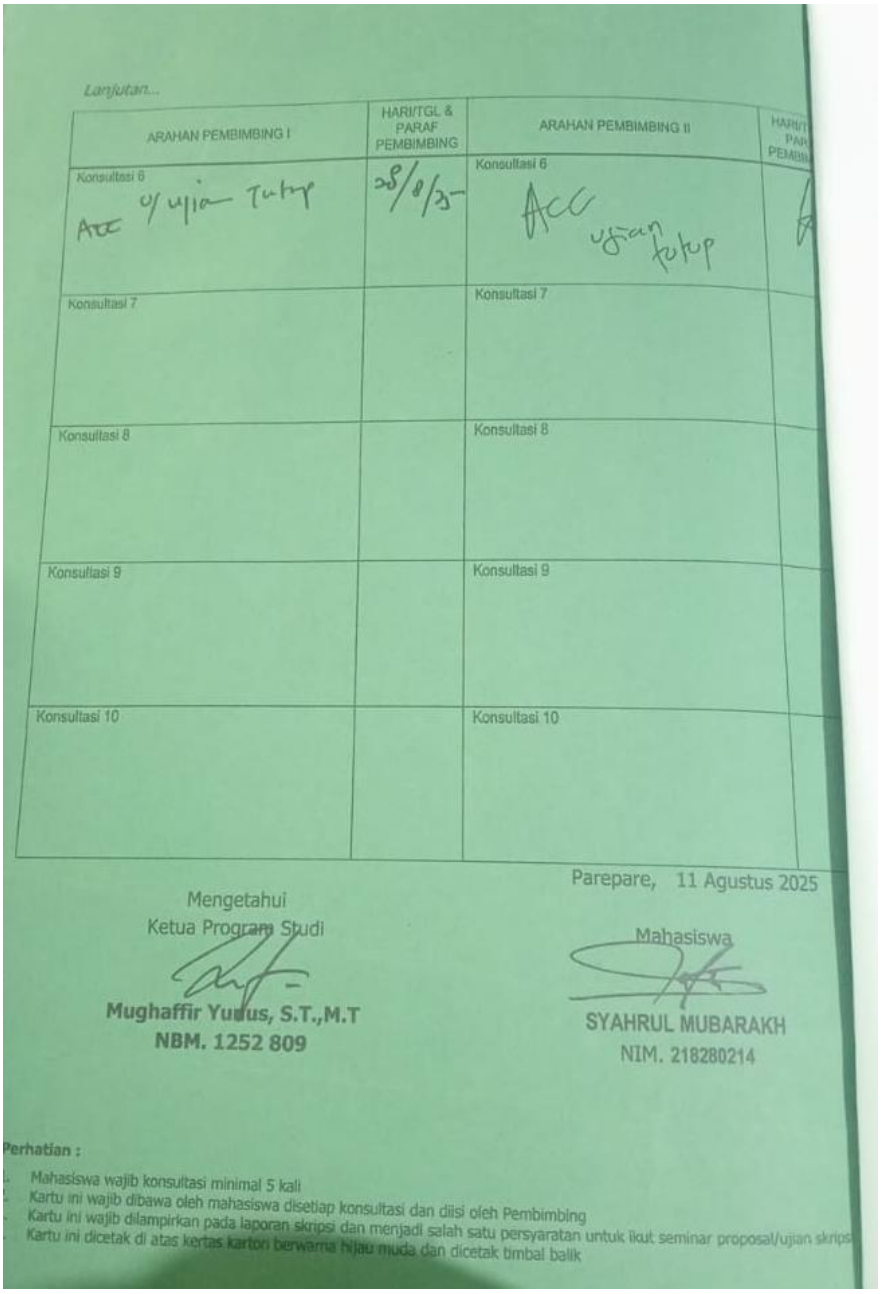
PROPOSAL

siswa : SYAHRUL MUBARAKH | Pembimbing I : Hj. A. Irmayani P, S.T, M.T
: 218280214 | Pembimbing II : Wahyuddin, S.Kom, M.Kom
Skripsi : PERANCANGAN APLIKASI HELP DESK UNIVERSITAS MUHAMMADIYAH PAREPARE

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Bab 1 ...kasih Desain Sistem.	23/8/2023 A-1	Konsultasi 1 - ikuti format penulisan	
Bab 2 ...kan sesuai Abjad ...pustak	A-1	Konsultasi 2 - Desain Sistem	
Bab 3 ...mbah Teori BAB II DAFTAR pustaka.	A-1	Konsultasi 3 Singkronkan BAB II dari Daftar Pustaka	
Bab 4 ...asing (Utuh)c)	A-1	Konsultasi 4 Ace	
Bab 5 Ace ...ujian Propose	A-1	Konsultasi 5	

Lanjut ke halaman sebelah

...wajib konsultasi minimal 5 kali
...wajib diisi oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
...wajib diserahkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/hasil skripsi
...dicetak di atas kertas karton berwarna hijau muda dan dicetak bolak balik



Lanjutan...

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/T PAR PEMBI
Konsultasi 6 Acc u/ ujian Tutup	28/8/25	Konsultasi 6 Acc ujian tutup	
Konsultasi 7		Konsultasi 7	
Konsultasi 8		Konsultasi 8	
Konsultasi 9		Konsultasi 9	
Konsultasi 10		Konsultasi 10	

Parepare, 11 Agustus 2025

Mengetahui
Ketua Program Studi

Mughaffir Yunus, S.T.,M.T
NBM. 1252 809

Mahasiswa

SYAHRUL MUBARAKH
NIM. 218280214

Perhatian :

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