

**LAPORAN AKHIR PROJECT PEMROGRAMAN WEB FRAMEWORK
SISTEM INFORMASI MANAJEMEN PERPUSTAKAAN DIGITAL
'KOLEKSIKU'**



**Disusun oleh:
DELIMA MASYA'UL LAILLIYAH
L200230061**

A

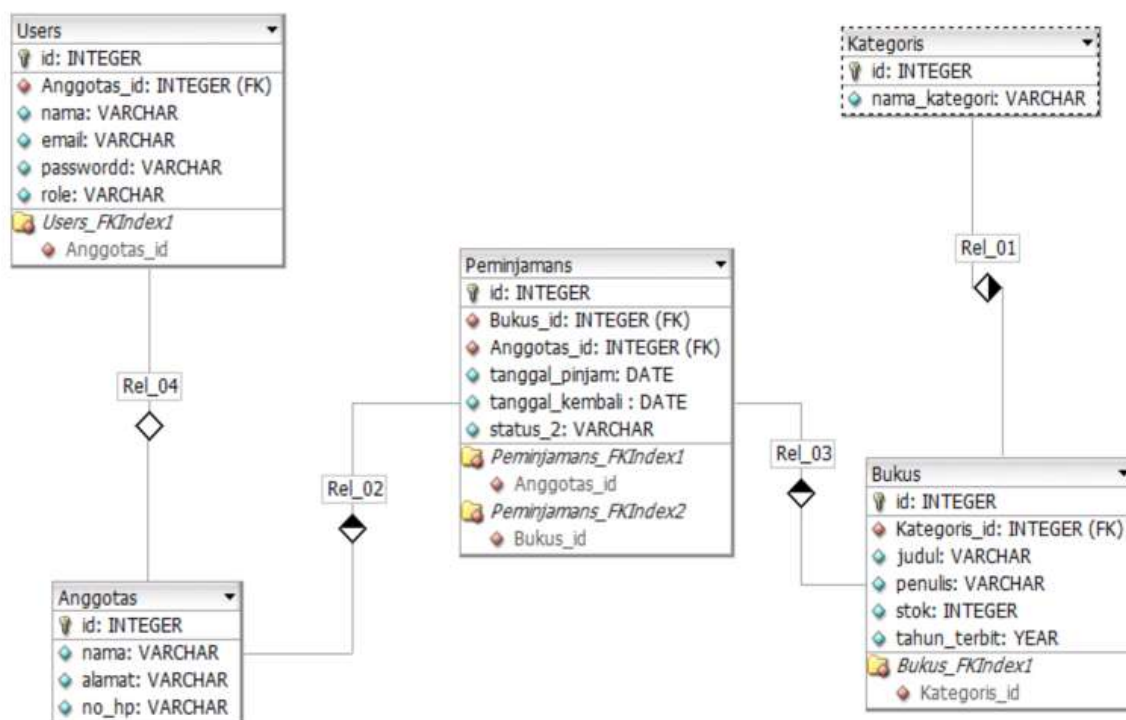
**TEKNIK INFORMATIKA
FAKULTAS KOMUNIKASI DAN INFORMATIKA
UNIVERSITAS MUHAMMADIYAH SURAKARTA
2025/2026**

Deskripsi Project

Project ini merupakan aplikasi peminjaman buku berbasis website yang dibuat untuk membantu pengelolaan data perpustakaan agar lebih rapi dan mudah digunakan. Melalui aplikasi ini, data buku, kategori buku, penerbit, anggota, serta peminjaman buku dapat dicatat dan dikelola dalam satu system, sehingga lebih tertata dan mengurangi kesalahan. Aplikasi ini memiliki dua jenis pengguna, yaitu admin dan anggota. Admin bertugas mengelola seluruh data yang ada di sistem, seperti menambah, mengubah, dan menghapus data buku, kategori, penerbit, anggota, serta data peminjaman. Sementara itu, anggota hanya dapat melihat daftar buku yang tersedia dan melihat data peminjaman miliknya sendiri tanpa bisa mengubah data tersebut. Selain itu, sistem ini dilengkapi dengan fitur login agar setiap pengguna hanya dapat mengakses halaman sesuai dengan perannya. Tampilan dashboard juga disesuaikan dengan jenis pengguna yang login sehingga informasi yang ditampilkan menjadi lebih jelas dan mudah dipahami.

Perancangan Database

Project yang dibangun menggunakan perancangan database yang terdiri dari tabel-tabel berikut:



Framework MVC

a. Route

```
<?php
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\{DashboardController,BukuController,KategoriController,AnggotaController,PeminjamanController};
use App\Http\Controllers\Auth\AuthenticatedSessionController;

Route::get('/', function () {
    return view('welcome');
});

Route::get('/login', [AuthenticatedSessionController::class, 'create'])->name('login');
Route::post('/login', [AuthenticatedSessionController::class, 'store']);
Route::post('/logout', [AuthenticatedSessionController::class, 'destroy'])->name('logout');
Route::get('/register', [App\Http\Controllers\Auth\RegisteredUserController::class, 'create'])->name('register');
Route::post('/register', [App\Http\Controllers\Auth\RegisteredUserController::class, 'store']);

Route::middleware('auth')->group(function () {
    Route::get('/dashboard', [DashboardController::class, 'index'])->name('dashboard');
    Route::get('/peminjaman', [PeminjamanController::class, 'index'])->name('peminjaman.index');
    Route::middleware('role:admin')->group(function () {
        Route::get('/buku/create', [BukuController::class, 'create'])->name('buku.create');
        Route::post('/buku', [BukuController::class, 'store'])->name('buku.store');
        Route::get('/buku/{buku}/edit', [BukuController::class, 'edit'])->name('buku.edit');
        Route::put('/buku/{buku}', [BukuController::class, 'update'])->name('buku.update');
        Route::delete('/buku/{buku}', [BukuController::class, 'destroy'])->name('buku.destroy');
        Route::resource('kategori', KategoriController::class);
        Route::resource('anggota', AnggotaController::class);
        Route::resource('peminjaman', PeminjamanController::class)->except(['index']);
    });
    Route::get('/buku', [BukuController::class, 'index'])->name('buku.index');
    Route::get('/buku/{buku}', [BukuController::class, 'show'])->name('buku.show');
});
```

Route Autentikasi :

- login: Digunakan untuk menampilkan halaman masuk dan memproses data login pengguna.
- register: Digunakan untuk pendaftaran user baru agar bisa masuk ke dalam sistem sebagai anggota.
- logout: Digunakan untuk mengakhiri sesi pengguna dan kembali ke halaman awal.

Route Dashboard:

- dashboard: Route utama setelah login yang berfungsi menampilkan ringkasan data statistik perpustakaan dalam bentuk kartu (cards).

Route Manajemen Buku (CRUD & View):

- buku.index & buku.show: Route akses bersama untuk melihat daftar koleksi buku dan detail informasi tiap buku.
- buku (create, store, edit, update, destroy): Kumpulan rute yang diproteksi khusus Admin untuk menambah, mengubah, dan menghapus data buku dari database.

Route Manajemen Data Master (Admin Only):

- kategori: Route resource untuk mengelola pengelompokan jenis buku (CRUD).
- anggota: Route resource untuk mengelola data identitas pengguna yang terdaftar di perpustakaan.

Route Peminjaman (Transactions):

- peminjaman.index: Route untuk menampilkan data transaksi peminjaman (Admin melihat seluruh data, Anggota melihat data pribadi).
- peminjaman (create, edit, update, destroy): Route operasional transaksi

peminjaman yang dikelola sepenuhnya oleh Admin.

b. Controller dan Function in Controller

- **AuthenticatedSessionController**

```
<?php
namespace App\Http\Controllers\Auth;
use App\Http\Controllers\Controller;
use App\Http\Requests\Auth\LoginRequest;
use App\Providers\RouteServiceProvider;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\View\View;

class AuthenticatedSessionController extends Controller
{
    public function create(): View
    {
        return view('auth.login');
    }
    public function store(LoginRequest $request): RedirectResponse
    {
        $request->authenticate();
        $request->session()->regenerate();
        return redirect()->intended(RouteServiceProvider::HOME);
    }
    public function destroy(Request $request): RedirectResponse
    {
        Auth::guard('web')->logout();
        $request->session()->invalidate();
        $request->session()->regenerateToken();
        return redirect('/');
    }
}
```

- create: Menampilkan halaman formulir login bagi pengguna.
- store: Melakukan validasi data login dan membuat sesi (session) jika data sesuai.
- destroy: Menghapus sesi pengguna (Logout) dan mengarahkan kembali ke halaman utama.

- **DashboardController**

```

<?php
namespace App\Http\Controllers;
use App\Models\Buku;
use App\Models\Kategori;
use App\Models\User;
use App\Models\Peminjaman;
use App\Models\Anggota;
use Illuminate\Support\Facades\Auth;

class DashboardController extends Controller{
    public function index(){
        $user = auth()->user();
        $data = [
            'jumlahBuku'      => Buku::count(),
            'jumlahKategori' => Kategori::count(),
            'jumlahAnggota'   => Anggota::count(),
            'jumlahPeminjaman' => ($user->role === 'admin')
                                ? Peminjaman::count()
                                : Peminjaman::where('anggota_id', $user->id)->count(),
        ];
        if ($user->role == 'anggota') {
            $data['pinjamanSaya'] = Peminjaman::where('anggota_id', $user->id)->count();
        } else {
            $data['pinjamanSaya'] = 0;
        }
        if ($user->role === 'admin') {
            return view('admin.dashboard', $data);
        } else {
            return view('anggota.dashboard', $data);
        }
    }
}

```

- DashboardController ini bertugas mengumpulkan data statistik dari database untuk disajikan kepada pengguna berdasarkan role mereka.
- Fungsi index():
 - Mengambil jumlah total buku, kategori, dan anggota menggunakan metode count() dari masing-masing Model.
 - Logika Multi-Role: Jika Admin, sistem menampilkan total seluruh peminjaman di perpustakaan. Jika Anggota, sistem memfilter data sehingga hanya menampilkan jumlah "Pinjaman Saya".
 - Mengarahkan pengguna secara otomatis ke halaman admin.dashboard atau anggota.dashboard sesuai dengan hak aksesnya
- **BukuController**

```

<?php
namespace App\Http\Controllers;
use App\Models\Buku;
use App\Models\Kategori;
use Illuminate\Http\Request;
class BukuController extends Controller{
    public function index()
    {
        $buku = Buku::all();
        return view('buku.index', compact('buku'));
    }
    public function create()
    {
        $kategori = Kategori::all();
        return view('buku.create', compact('kategori'));
    }
    public function show(Buku $buku)
    {
        return view('buku.show', compact('buku'));
    }
    public function store(Request $request)
    {
        $request->validate([
            'judul' => 'required',
            'penulis' => 'required',
            'tahun_terbit' => 'required',
            'stok' => 'required',
            'kategori_id' => 'required',
        ]);

```

```

        Buku::create($request->all());
        return redirect()->route('buku.index');
    }
    public function edit($id)
    {
        $buku = Buku::findOrFail($id);
        $kategori = Kategori::all();
        return view('buku.edit', compact('buku', 'kategori'));
    }
    public function update(Request $request, Buku $buku)
    {
        $buku->update($request->all());
        return redirect()->route('buku.index');
    }
    public function destroy(Buku $buku)
    {
        $buku->delete();
        return redirect()->route('buku.index');
    }
}

```

- index: Menampilkan tabel daftar buku secara keseluruhan.
- create & store: Digunakan oleh Admin untuk menampilkan form tambah buku dan menyimpan data buku baru ke database.
- edit & update: Digunakan oleh Admin untuk memproses perubahan data pada buku yang sudah ada.
- destroy: Menghapus data buku tertentu dari sistem.
- show: Menampilkan informasi detail mengenai satu judul buku tertentu.

- **PeminjamanController**

```
<?php
namespace App\Http\Controllers;
use App\Models\Peminjaman;
use App\Models\Anggota;
use App\Models\Buku;
use Illuminate\Http\Request;
class PeminjamanController extends Controller
{
    public function index()
    {
        $user = auth()->user();
        if ($user->role === 'admin') {
            $allPeminjaman = Peminjaman::with(['anggota', 'buku'])->get();
        } else {
            $allPeminjaman = Peminjaman::with(['anggota', 'buku'])
                ->where('anggota_id', $user->id)
                ->get();
        }
        return view('peminjaman.index', compact('allPeminjaman'));
    }

    public function create()
    {
        $anggota = \App\Models\Anggota::all();
        $buku = \App\Models\Buku::all();
        return view('peminjaman.create', compact('anggota', 'buku'));
    }

    public function store(Request $request)
    {
        $validatedData = $request->validate([
            'anggota_id' => 'required|exists:anggotas,id',
            'buku_id' => 'required|exists:bukus,id',
            'tanggal_pinjam' => 'required|date',
            'tanggal_kembali' => 'nullable|date',
            'status' => 'required',
        ]);
        Peminjaman::create($validatedData);
        return redirect()->route('peminjaman.index');
    }

    public function destroy(Peminjaman $peminjaman)
    {
        $peminjaman->delete();
        return redirect()->route('peminjaman.index');
    }

    public function edit($id)
    {
        $peminjaman = Peminjaman::findOrFail($id);
        $anggota = \App\Models\Anggota::all();
        $buku = \App\Models\Buku::all();
        return view('peminjaman.edit', compact('peminjaman', 'anggota', 'buku'));
    }
}
```

```

public function show(Peminjaman $peminjaman)
{
    return view('peminjaman.show', compact('peminjaman'));
}

public function update(Request $request, $id)
{
    $validatedData = $request->validate([
        'anggota_id' => 'required|exists:anggotas,id',
        'buku_id' => 'required|exists:bukus,id',
        'tanggal_pinjam' => 'required|date',
        'tanggal_kembali' => 'nullable|date',
        'status' => 'required',
    ]);
    $peminjaman = Peminjaman::findOrFail($id);
    $peminjaman->update($validatedData);
    return redirect()->route('peminjaman.index')->with('success', 'Data berhasil diperbarui');
}
}

```

- index(): Menampilkan daftar transaksi peminjaman. Bagi Admin, fungsi ini menampilkan semua data, sedangkan bagi Anggota, hanya menampilkan data "Pinjaman Saya".
 - create() menyediakan form untuk memilih anggota dan buku yang tersedia.
 - store() melakukan validasi input (ID anggota, ID buku, tanggal pinjam) dan menyimpan transaksi baru ke database.
 - edit() & update() : Digunakan untuk mengubah data transaksi, seperti memperbarui tanggal pengembalian atau merubah status peminjaman (misal: dari "dipinjam" menjadi "kembali").
 - destroy() : Berfungsi untuk menghapus record transaksi peminjaman tertentu dari sistem.
 - show(): Menampilkan detail lengkap satu transaksi peminjaman tertentu.
- **KategoriController**

```

<?php
namespace App\Http\Controllers;
use App\Models\Kategori;
use Illuminate\Http\Request;
class KategoriController extends Controller
{
    public function index()
    {
        $allKategori = Kategori::all();
        return view('kategori.index', compact('allKategori'));
    }
    public function create()
    {
        return view('kategori.create');
    }
    public function store(Request $request)
    {
        $validatedData = $request->validate([
            'nama_kategori' => 'required|max:100',
        ]);
        Kategori::create($validatedData);
        return redirect()->route('kategori.index');
    }
    public function show(Kategori $kategori)
    {
        return view('kategori.show', compact('kategori'));
    }
    public function edit(Kategori $kategori)
    {
        return view('kategori.edit', compact('kategori'));
    }
    public function update(Request $request, Kategori $kategori)
    {
        $validatedData = $request->validate([
            'nama_kategori' => 'required|max:100',
        ]);
        $kategori->update($validatedData);

        return redirect()->route('kategori.index');
    }
    public function destroy(Kategori $kategori)
    {
        $kategori->delete();
        return redirect()->route('kategori.index');
    }
}

```

- index(): Mengambil seluruh data dari tabel kategori untuk ditampilkan dalam bentuk daftar pada halaman manajemen kategori.
- create() & store(): create() menampilkan formulir untuk menambah

kategori baru, sedangkan store() berfungsi melakukan validasi input nama_kategori dan menyimpannya ke database.

- edit() & update(): Digunakan untuk mengubah nama kategori yang sudah ada. Fungsi update() memastikan data yang diubah tetap divalidasi sebelum diperbarui di database.
- destroy(): Berfungsi untuk menghapus kategori tertentu yang sudah tidak diperlukan lagi dari sistem.
- show(): Menampilkan detail informasi dari satu kategori spesifik.

- **AnggotaController**

```
<?php
namespace App\Http\Controllers;
use App\Models\Anggota;
use Illuminate\Http\Request;
use App\Models\Buku;
use App\Models\Kategori;
class AnggotaController extends Controller
{
    public function index()
    {
        $allAnggota = Anggota::all();
        return view('anggota.index', compact('allAnggota'));
    }
    public function create()
    {
        return view('anggota.create');
    }
    public function store(Request $request)
    {
        $validatedData = $request->validate([
            'nama' => 'required|max:100',
            'alamat' => 'required',
            'no_hp' => 'required',
        ]);
        Anggota::create($validatedData);
        return redirect()->route('anggota.index');
    }
}
```

```

public function show(Anggota $anggota)
{
    return view('anggota.show', compact('anggota'));
}
public function edit($id)
{
    $anggota = Anggota::findOrFail($id);
    return view('anggota.edit', compact('anggota'));
}
public function update(Request $request, $id)
{
    $anggota = Anggota::findOrFail($id);
    $anggota->update([
        'nama' => $request->nama,
    ]);
    $anggota->update([
        'alamat' => $request->alamat,
    ]);
    $anggota->update([
        'no_hp' => $request->no_hp,
    ]);
    return redirect()->route('anggota.index')
        ->with('success', 'Data anggota berhasil diupdate');
}

```

```

public function destroy($id)
{
    $anggota = Anggota::findOrFail($id);
    $anggota->delete();
    return redirect()->route('anggota.index')
        ->with('success', 'Data anggota berhasil dihapus');
}
public function dashboard()
{
    $jumlahBuku = Buku::count();
    $jumlahKategori = Kategori::count();
    $jumlahAnggota = Anggota::count();
    $jumlahPeminjaman = Peminjaman::count();
    return view('anggota.dashboard', compact(
        'jumlahBuku',
        'jumlahKategori',
        'jumlahAnggota',
        'jumlahPeminjaman'
    ));
}

```

- index(): Menampilkan daftar seluruh anggota yang terdaftar dalam sistem untuk kebutuhan manajemen admin.
- create() & store(): create() menyediakan formulir pendaftaran anggota baru, sementara store() melakukan validasi input (Nama, Alamat, dan Nomor HP) serta menyimpannya ke database.
- edit() & update(): Memungkinkan admin untuk mencari data anggota

berdasarkan ID (findOrFail) dan memperbarui informasi identitas mereka jika terjadi perubahan.

- destroy(): Memberikan otoritas kepada admin untuk menghapus record anggota dari database sistem.
- dashboard(): Fungsi khusus yang bertindak sebagai penyaji data ringkasan. Fungsi ini menghitung jumlah total buku, kategori, anggota, dan peminjaman untuk ditampilkan sebagai informasi utama pada dashboard anggota.

c. Model

- **Model Anggota**

```
<?php

namespace App\Models;

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

class Anggota extends Model
{
    use HasFactory;
    protected $guarded = [];
    public function peminjamans(): HasMany
    {
        return $this->hasMany(Peminjaman::class);
    }
}
```

Berfungsi mengelola data identitas pengguna. Model ini memiliki relasi HasMany terhadap model Peminjaman, yang berarti satu anggota dapat melakukan banyak transaksi peminjaman buku.

- **Model Kategori**

```

<?php

namespace App\Models;

use App\Models\Buku;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Relations\HasMany;
class Kategori extends Model
{
    protected $table = 'kategoris';
    protected $guarded = [];
    public function bukus(): HasMany
    {
        return $this->hasMany(Buku::class);
    }
}

```

Digunakan untuk mengelompokkan buku. Model ini memiliki relasi HasMany ke model Buku, yang berarti satu kategori (misal: "Novel") bisa dimiliki oleh banyak judul buku.

- **Model Peminjaman**

```

<?php

namespace App\Models;
use Illuminate\Database\Eloquent\Model;
use App\Models\Anggota;
use App\Models\Buku;

class Peminjaman extends Model
{
    protected $table = 'peminjamans'; // PENTING (lihat poin 2)
    protected $fillable = [
        'anggota_id',
        'buku_id',
        'tanggal_pinjam',
        'tanggal_kembali',
        'status',
    ];
    public function anggota()
    {
        return $this->belongsTo(\App\Models\Anggota::class, 'anggota_id');
    }
    public function buku()
    {
        return $this->belongsTo(Buku::class);
    }
}

```

Model transaksi utama yang menghubungkan tabel Anggota dan Buku. Model

ini menggunakan relasi BelongsTo untuk menarik data detail anggota dan detail buku yang sedang dipinjam.

- **Model Buku**

```
<?php

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

class Buku extends Model
{
    use HasFactory;
    protected $table = 'buku'; // pastikan sesuai
    protected $fillable = [
        'judul',
        'penulis',
        'tahun_terbit',
        'stok',
        'kategori_id',
    ];
    public function kategori()
    {
        return $this->belongsTo(Kategori::class);
    }
    public function peminjamans()
    {
        return $this->hasMany(Peminjaman::class);
    }
}
```

Berfungsi mengelola informasi koleksi buku (judul, penulis, stok). Model ini memiliki relasi BelongsTo ke model Kategori dan HasMany ke model Peminjaman.

- **Model User**

```

<?php

namespace App\Models;
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
{
    protected $fillable = [
        'name',
        'email',
        'password',
        'role',
    ];
    public function isAdmin()
    {
        return $this->role === 'admin';
    }
    protected $hidden = [
        'password',
        'remember_token'
    ];
    public function anggota()
    {
        return $this->belongsTo(Anggota::class, 'user_id');
    }
}

```

Model bawaan Laravel yang telah dimodifikasi untuk mendukung fitur Multi-Role, sehingga sistem dapat mengenali apakah pengguna yang login bertindak sebagai Admin atau Anggota.

d. File Migration

- Create kategori

```

<?php

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

return new class extends Migration
{
    public function up(): void
    {
        Schema::create('kategoris', function (Blueprint $table) {
            $table->id();
            $table->string('nama_kategori');
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('kategoris');
    }
};

```

- Create anggota

```

<?php

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

return new class extends Migration
{
    public function up(): void
    {
        Schema::create('anggotas', function (Blueprint $table) {
            $table->id();
            $table->string('nama');
            $table->string('alamat');
            $table->string('no_hp', 20);
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('anggotas');
    }
};

```

- Create peminjaman

```

<?php

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

return new class extends Migration
{
    public function up(): void
    {
        Schema::create('peminjamans', function (Blueprint $table) {
            $table->id();
            $table->foreignId('anggota_id')->constrained('anggotas')->cascadeOnUpdate()->restrictOnDelete();
            $table->foreignId('buku_id')->constrained('bukus')->cascadeOnUpdate()->restrictOnDelete();
            $table->date('tanggal_pinjam');
            $table->date('tanggal_kembali')->nullable();
            $table->string('status')->default('dipinjam');
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('peminjamans');
    }
};

```

- Create buku

```

<?php

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

return new class extends Migration
{
    public function up(): void
    {
        Schema::create('bukus', function (Blueprint $table) {
            $table->id();
            $table->string('judul', 255);
            $table->string('pengarang', 100);
            $table->year('tahun_terbit');
            $table->integer('stok');
            $table->foreignId('penerbit_id')->constrained()->cascadeOnUpdate()->restrictOnDelete();
            $table->foreignId('kategori_id')->constrained()->cascadeOnUpdate()->restrictOnDelete();
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('bukus');
    }
};

```

e. View Page

- layout.dashboard

```

<!DOCTYPE html>
<html lang="id">
<head>
  <meta charset="UTF-8">
  <title>koleksiku</title>
  <link rel="stylesheet" href="{{ asset('css/dashboard.css') }}">
</head>
<body>

<div class="wrapper">
  <div class="sidebar">
    <h2><span>k</span>oleksiku</h2>
    <a href="{{ route('dashboard') }}">Home</a>
    {{-- MENU ADMIN --}}
    @if(auth()->user()->role === 'admin')
      <a href="{{ route('buku.index') }}">Buku</a>
      <a href="{{ route('kategori.index') }}">Kategori</a>
      <a href="{{ route('anggota.index') }}">Anggota</a>
      <a href="{{ route('peminjaman.index') }}">Peminjaman</a>
    @endif
    {{-- MENU ANGGOTA --}}
    @if(auth()->user()->role === 'anggota')
      <a href="{{ route('buku.index') }}">Buku</a>
      <a href="{{ route('peminjaman.index') }}">Peminjaman Saya</a>
    @endif
  </div>

  <div class="main">
    <nav class="navbar">
      <div class="nav-left">
        <a href="{{ route('dashboard') }}" class="btn-home" style="text-decoration:none;">🏠 Home</a>
      </div>
      <div class="nav-right">
        <span>
          Login sebagai: <b>{{ auth()->user()->name }}</b>
          ({{ auth()->user()->role }})
        </span>
        <form action="{{ route('logout') }}" method="POST" style="display:inline; margin-left: 10px;">
          @csrf
          <button type="submit" class="btn-logout">Logout</button>
        </form>
      </div>
    </nav>
    <hr style="opacity: 0.1;">
    <main class="content">
      @yield('content')
    </main>
  </div>
</div>
</body>
</html>

```

- layout.header

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Manajemen Data Buku</title>
  <link rel="stylesheet" href="{{ asset('css/style.css') }}">
</head>
<body>
<div class="container">
  <h1>Manajemen Peminjaman Buku</h1>
  <div class="nav">
  </div>

```

- admin.dashboard

```

@extends('layout.dashboard')
@section('content')
<h2>Selamat Datang, {{ auth()->user()->name }}</h2>
<div class="cards">
  <div class="card blue">
    <h3>Jumlah Buku</h3>
    <p>{{ $jumlahBuku }}</p>
  </div>
  <div class="card blue">
    <h3>Kategori</h3>
    <p>{{ $jumlahKategori }}</p>
  </div>
  <div class="card blue">
    <h3>Anggota</h3>
    <p>{{ $jumlahAnggota }}</p>
  </div>
  <div class="card blue">
    <h3>Pinjaman</h3>
    <p>{{ $jumlahPeminjaman }}</p>
  </div>
</div>
<div class="info-box">
  <h3>Informasi Aturan Peminjaman</h3>
  <ol>
    <li>Maksimal peminjaman 1 minggu</li>
    <li>Maksimal 3 buku</li>
    <li>Denda Rp 15.000 / minggu</li>
    <li>Konfirmasi ke petugas</li>
  </ol>

```



```

@include('layout.header')
<h3>Edit Anggota</h3>
<form action="{{ route('anggota.update', $anggota) }}" method="POST">
    @csrf
    @method('PUT')
    <div class="form-group">
        <label>Nama Anggota:</label>
        <input type="text" name="nama" value="{{ $anggota->nama }}">
    </div>
    <div class="form-group">
        <label>Alamat:</label>
        <input type="text" name="alamat" value="{{ $anggota->alamat }}">
    </div>

    <div class="form-group">
        <label>No HP:</label>
        <input type="text" name="no_hp" value="{{ $anggota->no_hp }}">
    </div>
    <button type="submit" class="tombol">Update</button>
</form>
<br>
<a href="{{ route('anggota.index') }}" class="tombol">Kembali</a>
@include('layout.footer')

```

- anggota.index

```

<h3>Anggota</h3>
<a href="{{ route('anggota.create') }}" class="tombol">Tambah</a>
<table>
    <thead>
        <tr>
            <th>No</th>
            <th>Nama Anggota</th>
            <th>Alamat Anggota</th>
            <th>Nomor Anggota</th>
            <th>Aksi</th>
        </tr>
    </thead>
    <tbody>
        @foreach ($allAnggota as $key => $r)
            <tr>
                <td>{{ $key + 1 }}</td>
                <td>{{ $r->nama }}</td>
                <td>{{ $r->alamat }}</td>
                <td>{{ $r->no_hp }}</td>
                <td>
                    <a href="{{ route('anggota.edit', $r->id) }}" class="tombol">Edit</a>
                    <form action="{{ route('anggota.destroy', $r->id) }}"
                        method="POST"
                        style="display:inline;"
                        @csrf
                        @method('DELETE')
                    <button type="submit" class="tombol" onclick="return confirm('Yakin hapus data?')">
                        Hapus
                    </button>
                </td>
            </tr>
        </foreach>
    </tbody>
</table>

```

- buku.create

```

@include('layout.header')
<h3>Tambah Buku</h3>
<form action="{{ route('buku.store') }}" method="POST">
    @csrf
    <div class="form-group">
        <label>Judul Buku</label>
        <input type="text" name="judul" required>
    </div>
    <div class="form-group">
        <label>Penulis</label>
        <input type="text" name="penulis" class="form-control" required>
    </div>
    <div class="form-group">
        <label>Kategori</label>
        <select name="kategori_id">
            @foreach($kategori as $k)
                <option value="{{ $k->id }}">{{ $k->nama_kategori }}</option>
            @endforeach
        </select>
    </div>
    <div class="form-group">
        <label>Tahun Terbit</label>
        <input type="number" name="tahun_terbit" required>
    </div>
    <div class="form-group">
        <label>Stok</label>
        <input type="number" name="stok" required>
    </div>
    <button type="submit" class="tombol">Simpan</button>
</form>

```

- buku.edit

```

@include('layout.header')
<h3>Edit Buku</h3>
<form action="{{ route('buku.update', $buku->id) }}" method="POST">
    @csrf
    @method('PUT')
    <div class="form-group">
        <label>Judul Buku</label>
        <input type="text" name="judul" value="{{ $buku->judul }}" required>
    </div>
    <div class="form-group">
        <label>Kategori</label>
        <select name="kategori_id" required>
            @foreach($kategori as $k)
                <option value="{{ $k->id }}"
                    {{ $buku->kategori_id == $k->id ? 'selected' : '' }}>
                    {{ $k->nama }}
                </option>
            @endforeach
        </select>
    </div>
    <div class="form-group">
        <label>Penulis</label>
        <input type="text" name="penulis"
            value="{{ $buku->penulis }}" required>
    </div>

```

```

        <div class="form-group">
            <label>Penulis</label>
            <input type="text" name="penulis"
                value="{{ $buku->penulis }}" required>
        </div>
        <div class="form-group">
            <label>Tahun Terbit</label>
            <input type="number" name="tahun_terbit"
                value="{{ $buku->tahun_terbit }}" required>
        </div>
        <div class="form-group">
            <label>Stok</label>
            <input type="number" name="stok" value="{{ $buku->stok }}" required>
        </div>
        <button type="submit" class="tombol">Update</button>
    </form>
    <br>
    <a href="{{ route('buku.index') }}" class="tombol">Kembali</a>

    @include('layout.footer')

```

- buku.index

```

    @include('layout.header')
    <h3>Data Buku</h3>
    @auth
        @if(auth()->user()->role === 'admin')
            <a href="{{ route('buku.create') }}" class="tombol">Tambah</a>
        @endif
    @endauth
    <table border="1" cellpadding="8">
        <tr>
            <th>No</th>
            <th>Judul</th>
            <th>Penulis</th>
            <th>Tahun</th>
            <th>Stok</th>
            {{-- Sembunyikan header Aksi jika bukan admin --}}
            @if(auth()->user()->role === 'admin')
                <th>Aksi</th>
            @endif
        </tr>
        @foreach($buku as $b)
            <tr>
                <td>{{ $loop->iteration }}</td>
                <td>{{ $b->judul }}</td>
                <td>{{ $b->penulis }}</td>
                <td>{{ $b->tahun_terbit }}</td>
                <td>{{ $b->stok }}</td>
                <td>

```

```

@if(auth()->user()->role === 'admin')
    <a href="{{ route('buku.show', $b->id) }}" class="tombol">Detail</a>
    <a href="{{ route('buku.edit', $b->id) }}" class="tombol">Edit</a>
    <form action="{{ route('buku.destroy', $b->id) }}" method="POST" style="display:inline">
        @csrf
        @method('DELETE')
        <button type="submit" class="tombol" onclick="return confirm('Yakin hapus data?')">
            Hapus
        </button>
    </form>
@endif
</td>
</tr>
@endforeach
</table>
<br>
<a href="{{ route('dashboard') }}" class="tombol">Home</a>
@include('layout.footer')

```

- buku.show

```

@include('layout.header')
<h3>Detail Buku</h3>
<table border="1" cellpadding="8" cellspacing="0">
    <tr>
        <td width="150">Judul Buku</td>
        <td width="10">:</td>
        <td>{{ $buku->judul }}</td>
    </tr>
    <tr>
        <td>Tahun Terbit</td>
        <td>:</td>
        <td>{{ $buku->tahun_terbit }}</td>
    </tr>
    <tr>
        <td>Penulis</td>
        <td>:</td>
        <td>{{ $buku->penulis }}</td>
    </tr>
    <tr>
        <td>Stok</td>
        <td>:</td>
        <td>{{ $buku->stok }}</td>
    </tr>
</table>
<br>
<a href="{{ route('buku.index') }}" class="tombol">Kembali</a>
@include('layout.footer')

```

- peminjaman.create

```

@include('layout.header')
<h3>Tambah Peminjaman</h3>
<form action="{{ route('peminjaman.store') }}" method="POST">
    @csrf <div class="form-group">
        <label>Anggota</label>
        <select name="anggota_id" class="form-control">
            @foreach($anggota as $a)
                <option value="{{ $a->id }}">{{ $a->nama }}</option>
            @endforeach
        </select>
    </div>
    <div class="form-group">
        <label>Buku</label>
        <select name="buku_id">
            @foreach($buku as $b)
                <option value="{{ $b->id }}">{{ $b->judul }}</option>
            @endforeach
        </select>
    </div>
    <div class="form-group">
        <label>Tanggal Pinjam</label>
        <input type="date" name="tanggal_pinjam">
    </div>
    <div class="form-group">
        <label>Tanggal Kembali</label>
        <input type="date" name="tanggal_kembali">
    </div>
    <div class="form-group">
        <label>Status</label>
        <select name="status">
            <option value="dipinjam">Dipinjam</option>
            <option value="dikembalikan">Dikembalikan</option>
        </select>
    </div>
    <button type="submit" class="tombol">Simpan</button>
</form>
@include('layout.footer')

```

- peminjaman.edit

```

@include('layout.header')
<h3>Edit Peminjaman</h3>
<form action="{{ route('peminjaman.update', $peminjaman->id) }}" method="POST">
    @csrf
    @method('PUT')
    <div class="form-group">
        <label>Anggota</label>
        <select name="anggota_id" class="form-control">
            @foreach($anggota as $a)
                <option value="{{ $a->id }}" {{ isset($peminjaman) && $peminjaman->anggota_id == $a->id ? 'selected' : '' }}>
                    {{ $a->nama }} </option>
            @endforeach
        </select>
    </div>
    <div class="form-group">
        <label>Buku</label>
        <select name="buku_id">
            @foreach($buku as $b)
                <option value="{{ $b->id }}"
                    {{ $peminjaman->buku_id == $b->id ? 'selected' : '' }}>
                    {{ $b->judul }}
                </option>
            @endforeach
        </select>
    </div>
    <div class="form-group">
        <label>Tanggal Kembali</label>
        <input type="date" name="tanggal_kembali"
            value="{{ $peminjaman->tanggal_kembali }}">
    </div>
    <div class="form-group">
        <label>Status</label>
        <select name="status">
            <option value="dipinjam" {{ $peminjaman->status == 'dipinjam' ? 'selected' : '' }}>
                Dipinjam
            </option>
            <option value="dikenbalikan" {{ $peminjaman->status == 'dikenbalikan' ? 'selected' : '' }}>
                Dikenbalikan
            </option>
        </select>
    </div>
    <button type="submit" class="tombol">Update</button>
</form>
<br>
<a href="{{ route('peminjaman.index') }}" class="tombol">Kembali</a>
@include('layout.footer')

```

- **peminjaman.index**

```

@include('layout.header')
<h3>Data Peminjaman</h3>
{{-- Tombol Tambah hanya untuk Admin --}}
@if(auth()->user()->role === 'admin')
    <a href="{{ route('peminjaman.create') }}" class="tombol">Tambah</a>
@endif
<table>
<thead>
    <tr>
        <th>No</th>
        <th>Anggota</th>
        <th>Buku</th>
        <th>Tanggal Pinjam</th>
        <th>Tanggal Kembali</th>
        <th>Status</th>
        {{-- Header Aksi hanya untuk Admin --}}
        @if(auth()->user()->role === 'admin')
            <th>Aksi</th>
        @endif
    </tr>
</thead>
<tbody>

```

```

@foreach ($allPeminjaman as $key => $p)
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| {{ $key + 1 }} | {{ $p->anggota->nama }} | {{ $p->buku->judul }} | {{ $p->tanggal_pinjam }} | {{ $p->tanggal_kembali ?? '-' }} | {{ $p->status ?? '-' }} | @if(auth()->user()->role === 'admin')  Detail Edit {{ csrf_field() }} {{ method_field('DELETE') }} | |

```

- peminjaman.show

```

@include('layout.header')
<h3>Detail Peminjaman</h3>
<table border="1" cellpadding="8" cellspacing="0">
|  |
| --- |
| Nama Anggota  {{ $peminjaman->anggota->nama }} |
| Judul Buku  {{ $peminjaman->buku->judul }} |
| Tanggal Pinjam  {{ $peminjaman->tanggal_pinjam }} |
| Tanggal Kembali  {{ $peminjaman->tanggal_kembali ?? '-' }} |
| Status  {{ $peminjaman->status }} |

```

- kategori.create

```
@include('layout.header')
<h3>Buat Kategori</h3>
<form action="{{ route('kategori.store') }}" method="post">
    @csrf
    <div class="form-group">
        <label for="">Nama Kategori:</label>
        <input type="text" name="nama_kategori" id="" placeholder="Masukkan nama kategori">
    </div>
    <button type="submit" class="tombol">submit</button>
</form>
@include('layout.footer')
```

- kategori.edit

```
@include('layout.header')
<h3>Edit Kategori</h3>
<form action="{{ route('kategori.update', $kategori->id) }}" method="POST">
    @csrf
    @method('PUT')
    <div class="form-group">
        <label for="">Nama Kategori:</label>
        <input type="text" name="nama_kategori" id="" value="{{ $kategori->nama_kategori }}">
    </div>
    <button type="submit" class="tombol">Update</button>
</form>
<br>
<a href="{{ route('kategori.index') }}" class="tombol">Kembali</a>
@include('layout.footer')
```

- kategori.index

```
<h3>Kategori</h3>
<a href="{{ route('kategori.create') }}" class="tombol">Tambah</a>
<table>
    <thead>
        <tr>
            <th>No</th>
            <th>Nama Kategori</th>
            <th>Aksi</th>
        </tr>
    </thead>
    <tbody>
        @foreach ($allKategori as $key => $r)
            <tr>
                <td>{{ $key + 1 }}</td>
                <td>{{ $r->nama_kategori }}</td>
                <td>
                    <form action="{{ route('kategori.destroy', $r->id) }}" method="POST">
                        <a href="{{ route('kategori.edit', $r->id) }}" class="tombol">Edit</a>
                        @csrf
                        @method('DELETE')
                        <button type="submit" class="tombol" onclick="return confirm('Yakin hapus data?')">
                            Hapus
                        </button>
                    </form>
                </td>
            </tr>
        @endforeach
    </tbody>
</table>
<br>
```

Ide Pengembangan (Improvement)

1. Sistem denda : Menghitung denda secara otomatis jika tanggal pengembalian melewati batas 1 minggu (Rp 15.000/minggu). Data denda ini dapat ditampilkan langsung pada dashboard anggota sebagai peringatan.
2. Grafik : Mengubah atau menambahkan tampilan angka statistik yang statis di dashboard admin menjadi grafik interaktif
3. Pencarian cepat: Menambahkan fitur Real-time Search dan menambahkan filter berdasarkan kategori atau penulis untuk memudahkan navigasi saat jumlah data sudah mencapai ribuan.
4. Laporan (PDF/Excel): Menambahkan fitur cetak laporan dalam format PDF atau Excel untuk membantu admin membuat laporan bulanan mengenai sirkulasi buku dan daftar anggota aktif.
5. Sistem notifikasi untuk anggota: Mengintegrasikan layanan email untuk mengirimkan pengingat otomatis kepada anggota 1 hari sebelum batas waktu pengembalian buku habis.