

**SISTEM INFORMASI ADMINISTRASI BERBASIS WEB  
DENGAN MENGGUNAKAN FRAMEWORK LARAVEL  
PADA TPQ AL-HIKMAH**

**LAPORAN PEMROGRAMAN WEB BERBASIS FRAMEWORK**

Oleh:

- |   |                     |
|---|---------------------|
| <b>1. Agiftsany Azhar</b>               | <b>152011513020</b> |
| <b>2. Muhammad Faisal Maulana Putra</b> | <b>152011513009</b> |



**PROGRAM STUDI SISTEM INFORMASI**

**FAKULTAS VOKASI**

**UNIVERSITAS AIRLANGGA**

**SURABAYA**

**2021**

## DAFTAR ISI

<b>DAFTAR ISI.....</b>	<b>ii</b>
<b>BAB I PENDAHULUAN.....</b>	<b>1</b>
<b>1.1 Latar Belakang .....</b>	<b>1</b>
<b>1.2 Tujuan .....</b>	<b>2</b>
<b>1.3 Lingkup Masalah .....</b>	<b>2</b>
<b>1.4 Deskripsi Global Perangkat Lunak .....</b>	<b>2</b>
<b>1.5 Fungsi Produk .....</b>	<b>3</b>
<b>1.6 Karakteristik Pengguna.....</b>	<b>3</b>
<b>BAB II DESAIN .....</b>	<b>4</b>
<b>2.1 Physical Data Model.....</b>	<b>4</b>
<b>2.2 Perancangan Database.....</b>	<b>4</b>
<b>2.2.1 Tabel santri.....</b>	<b>4</b>
<b>2.2.2 Tabel pengurus.....</b>	<b>5</b>
<b>2.2.3 Tabel peran.....</b>	<b>5</b>
<b>2.2.4 Tabel buku .....</b>	<b>5</b>
<b>2.2.5 Tabel bab .....</b>	<b>5</b>
<b>2.2.6 Tabel kemajuan.....</b>	<b>6</b>
<b>2.2.7 Tabel detail_kemajuan .....</b>	<b>6</b>
<b>2.2.8 Tabel detail_peran .....</b>	<b>6</b>
<b>BAB III IMPLEMENTASI.....</b>	<b>7</b>
<b>3.1 Halaman Utama (Home).....</b>	<b>7</b>
<b>3.2 Register.....</b>	<b>7</b>
<b>3.3 Login.....</b>	<b>9</b>
<b>3.4 Dashboard .....</b>	<b>10</b>
<b>3.5 Buku .....</b>	<b>11</b>
<b>3.6 Bab .....</b>	<b>15</b>
<b>3.7 Pengurus.....</b>	<b>20</b>
<b>3.8 Peran.....</b>	<b>27</b>
<b>3.9 Logout.....</b>	<b>31</b>
<b>LAMPIRAN.....</b>	<b>33</b>

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Sistem informasi merupakan kumpulan-kumpulan komponen dalam satu organisasi yang berfungsi untuk mengolah data menjadi informasi. Peranan sistem informasi tidak perlu diragukan lagi karena dengan adanya dukungan sistem informasi yang baik maka suatu sistem informasi yang baik maka organisasi memiliki suatu keunggulan kompetitif dibandingkan dengan yang lain (Kusrini & Koniyo, 2007). Untuk mendapatkan dan menghasilkan informasi, komputer dan teknologinya adalah salah satu alat bantu yang paling tepat. Pemanfaatan komputer sebagai alat bantu mengolah data juga semakin berkembang seiring dengan kemajuan zaman. Hal ini terjadi karena tuntutan kebutuhan akan informasi yang sangat meningkat dan keinginan untuk dapat menyelesaikan pekerjaan dengan cepat.

Proses penerimaan santri baru merupakan salah satu hal penting pada bidang pendidikan. Sistem informasi digunakan untuk mempermudah proses bisnis, mulai dari registrasi santri dan pengurus, input buku beserta babnya, pencapaian dari santri selama belajar, dan peran dari masing-masing pengurus. Oleh karena itu proses bisnis tersebut harus berjalan dengan cepat dan dapat selalu dipantau oleh pengurus maupun admin.

Begitu halnya di TPQ Al-Hikmah yang mana sistem informasi yang lebih baik merupakan kebutuhan yang tidak dapat ditunda lagi. Akan tetapi, pada kenyataannya proses pendataan pada TPQ Al-Hikmah masih menggunakan kertas dan diolah menggunakan Microsoft Excel. Hal ini menjadikan pengolahan data tidak berjalan cepat dan lama untuk diperbarui. karena hasil dari pencapaian santri tidak bisa ditampilkan secara langsung sehingga orang tua santri harus menunggu proses kegiatan belajar pada akhir semester berakhir untuk mengetahui hasilnya. Masalah lain yang muncul adalah saat pengurus yang memasukkan data buku lebih dari satu., maka data dari komputer lain harus digabungkan untuk mendapatkan hasil akhir. Para santri pun tidak bisa meminjam buku hingga proses pendataan buku berakhir sehingga mengakibatkan santri tidak bisa membaca buku sebagai bahan belajarnya.

Untuk mengatasi permasalahan tersebut maka diperlukan suatu software yang dirancang khusus untuk menangani kasus seperti pada TPQ Al-Hikmah. Dengan software ini diharapkan proses bisnis yang terjadi pada TPQ Al-Hikmah dapat berjalan lebih cepat dan hasil pencapaian santri dapat dipantau terus-menerus oleh orang tua santri.

Berdasarkan latar belakang tersebut, maka penulis bermaksud untuk mengembangkan suatu sistem informasi yang digunakan untuk mencatat semua proses bisnis yang ada di TPQ Al-Hikmah berbasis web menggunakan Laravel framework dan MySQL sebagai pengolah database serta Laragon sebagai web server. Tidak lupa software pendukung, seperti Visual Studio Code, Google Chrome, dan SQLyog untuk mempermudah dalam proses pembuatan sistem informasi berbasis web nantinya.

Dengan uraian tersebut, penulis berkeinginan membantu TPQ Al-Hikmah untuk memperbarui sistem informasi yang telah ada sebelumnya sehingga mempermudah semua pihak yang membutuhkan informasi di TPQ Al-hikmah. Atas dasar inilah, penulis tertarik memilih judul **“SISTEM INFORMASI ADMINISTRASI BERBASIS WEB DENGAN MENGGUNAKAN FRAMEWORK LARAVEL PADA TPQ AL-HIKMAH”**.

## **1.2 Tujuan**

Sistem informasi administrasi TPQ Al-Hikmah bertujuan untuk menyediakan sebuah sistem informasi berbasis web yang mempermudah segala pihak yang membutuhkan informasi di TPQ Al-Hikmah.

## **1.3 Lingkup Masalah**

Sistem informasi berbasis web pada sistem informasi administrasi TPQ Al-Hikmah dikembangkan dengan tujuan untuk:

1. Menangani proses register santri dan pengurus
2. Menangani proses login santri, pengurus, dan admin
3. Menangani proses pembelajaran santri
4. Menangani input data buku oleh pengurus
5. Menangani peran pada masing-masing pengurus

## **1.4 Deskripsi Global Perangkat Lunak**

Sistem informasi administrasi TPQ Al-Hikmah adalah sistem informasi berbasis web untuk mengolah proses bisnis yang ada di TPQ Al-Hikmah. Web ini nantinya dapat mempermudah santri dan pengurus untuk melakukan proses pembelajaran di TPQ Al-Hikmah.

## **1.5 Fungsi Produk**

Fungsi dari sistem informasi administrasi TPQ Al-Hikmah adalah untuk memudahkan semua orang khususnya santri dan pengurus dalam mengakses data, menginput data, dan mengelola data agar lebih terstruktur.

## **1.6 Karakteristik Pengguna**

### **1. Santri**

- a. Dapat mengoperasikan laptop/PC
- b. Dapat mengoperasikan sistem informasi yang berbasis web

### **2. Pengurus**

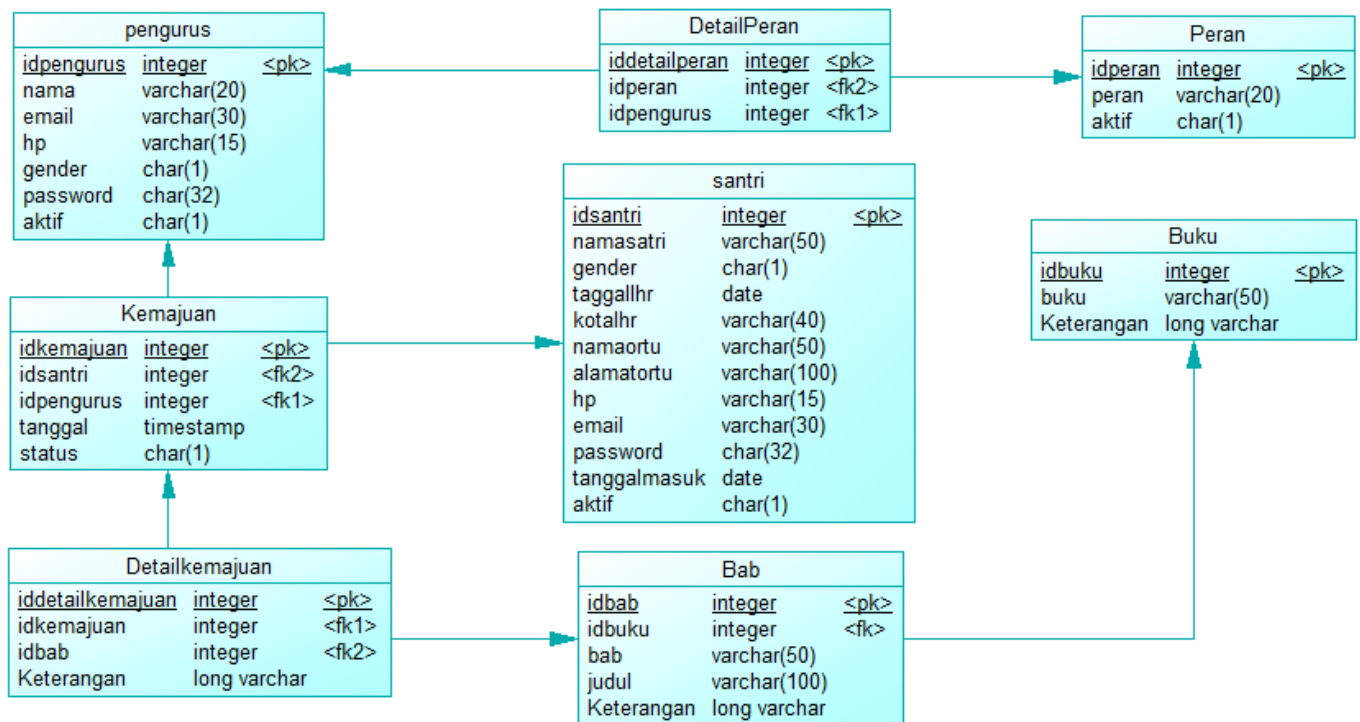
- a. Dapat mengoperasikan laptop/PC
- b. Dapat mengoperasikan sistem informasi yang berbasis web
- c. Memahami proses pengolahan data master
- d. Memahami konsep basis data

### **3. Admin**

- a. Dapat mengoperasikan laptop/PC
- b. Dapat mengoperasikan sistem informasi yang berbasis web
- c. Memahami proses pengolahan data master
- a. Memahami konsep basis data

## BAB II DESAIN

### 2.1 Physical Data Model



Tabel yang digunakan dalam sistem informasi administrasi TPQ Al-Hikmah:

1. Tabel santri
2. Tabel pengurus
3. Tabel peran
4. Tabel buku
5. Tabel bab
6. Tabel kemajuan
7. Tabel detail\_kemajuan
8. Tabel detail\_peran

### 2.2 Perancangan Database

#### 2.2.1 Tabel santri

Nama Kolom	Tipe Data	Keterangan
id	int	primary key
nama	varchar(50)	
gender	char(1)	

tgl_lhr	date	
kota_lhr	varchar(40)	
image	varchar(255)	
nama_ortu	varchar(50)	
alamat_ortu	varchar(100)	
hp	varchar(15)	
email	varchar(50)	
password	varchar(255)	
tgl_masuk	date	
aktif	int	

### 2.2.2 Tabel pengurus

Nama Kolom	Tipe Data	Keterangan
id	int	primary key
nama	varchar(50)	
email	varchar(50)	
hp	varchar(15)	
gender	char(1)	
image	varchar(255)	
password	varchar(255)	
aktif	int	

### 2.2.3 Tabel peran

Nama Kolom	Tipe Data	Keterangan
id	int	primary key
peran	varchar(50)	
aktif	int	

### 2.2.4 Tabel buku

Nama Kolom	Tipe Data	Keterangan
id	int	primary key
buku	varchar(50)	
keterangan	text	

### 2.2.5 Tabel bab

Nama Kolom	Tipe Data	Keterangan
id	int	primary key
id_buku	int	foreign key
bab	varchar(50)	
judul	varchar(100)	

keterangan	text	
------------	------	--

### 2.2.6 Tabel kemajuan

Nama Kolom	Tipe Data	Keterangan
id	int	primary key
id_santri	int	foreign key
id_pengurus	int	foreign key
tanggal	date	
status	char(1)	

### 2.2.7 Tabel detail\_kemajuan

Nama Kolom	Tipe Data	Keterangan
id	int	primary key
id_kemajuan	int	foreign key
id_bab	int	foreign key
keterangan	text	

### 2.2.8 Tabel detail\_peran

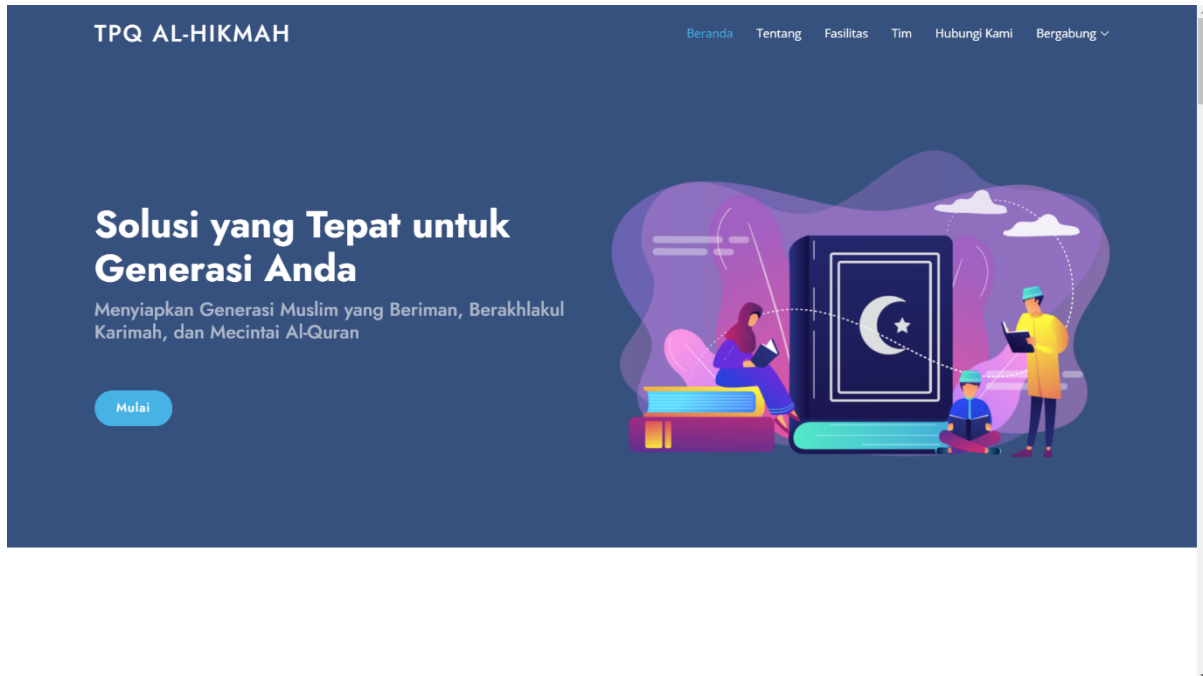
Nama Kolom	Tipe Data	Keterangan
id	int	primary key
id_peran	int	foreign key
id_pengurus	int	foreign key



## BAB III

### IMPLEMENTASI

#### 3.1 Halaman Utama (Home)



Halaman utama (home) merupakan halaman awal ketika user mengakses <https://kel09.tpq-alhikmah.my.id/>. Home juga menjadi tempat sumber informasi yang ada di TPQ Al-Hikmah, mulai dari program unggulan, fasilitas, hingga lokasi dari TPQ Al-Hikmah. Terdapat juga menu bergabung yang mana user dapat melakukan daftar dan/atau login.

#### 3.2 Register

The screenshot displays a registration form titled 'Daftar' on a dark blue background. The form is a light gray box with the following fields: 'Agiftsany' (text), 'Sidoarjo' (text), 'Azhar' (text), 'santri@gmail.com' (text), 'Laki-Laki' (dropdown menu), '29/12/2021' (date picker), 'Mojosantren RT 011 RW 003 Kemas Krian' (text), and '082330725030' (text). There is a password field with masked characters '.....'. A blue 'Daftar' button is at the bottom of the form. Below the button is a link that says 'Punya Akun? Login!'. At the very bottom of the page, there is a small copyright notice: '© Copyright TPQ Al-Hikmah. All rights Reserved'.

Untuk bisa mengakses halaman Dashboard, calon santri diwajibkan untuk daftar terlebih dahulu pada menu Daftar. Ketika semua formulir sudah diisi, maka calon santri menekan tombol daftar. Saat menekan tombol daftar, terjadi proses penyimpanan data ke database sesuai dengan formulir yang diisi tadi dan proses tersebut berlangsung pada file RegisterController.php dengan function yang digunakan untuk menyimpan data tersebut adalah store(). Penulis menggunakan enkripsi password bcrypt agar password yang disimpan lebih aman dan terhindar dari pembajakan. Selain dimasukkan dalam tabel santri, biodata santri juga dimasukkan ke dalam tabel user yang bertujuan untuk otentikasi saat login. Setelah proses pendaftaran berhasil calon santri akan diarahkan ke halaman Login.

### RegisterController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Santri;
use App\Models\User;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\DB;

class RegisterController extends Controller
{
    /**
     * Store a newly created resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        $validatedData = $request->validate([
            'nama' => 'required|min:3|max:50',
            'gender' => 'required',
            'tgl_lhr' => 'required',
            'kota_lhr' => 'required|max:40',
            'nama_ortu' => 'required|min:3|max:50',
            'alamat_ortu' => 'required|max:100',
            'hp' =>
                'required|unique:santris|unique:penguruses',
            'email' =>
                'required|email:dns|unique:santris|unique:penguruses',
            'password' => 'required|min:8|max:32',
        ]);

        $validatedData['password'] = bcrypt($validatedData['password']);

        Santri::create($validatedData);

        $santriid = Santri::latest()->first()->id;
```

```

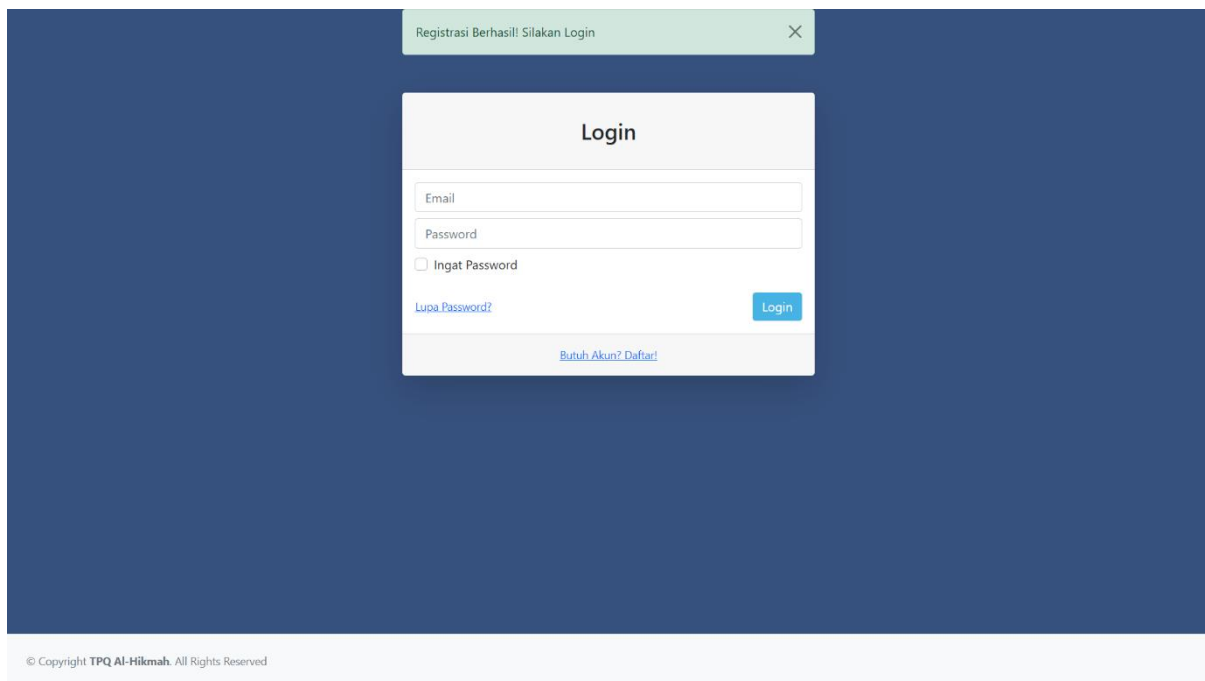
DB::table('users')->insert([
    'id_santri'      => $santriid,
    'nama'           => $validatedData['nama'],
    'email'          => $validatedData['email'],
    'password'       => $validatedData['password'],
    'role'           => 'Santri',
]);

$request->session()->flash('success','Registrasi Berhasil!
Silakan Login');

return redirect('/login');
}
}

```

### 3.3 Login



The screenshot shows a web application interface with a dark blue background. At the top, a green notification box displays the message "Registrasi Berhasil! Silakan Login" with a close button. Below this, a white login form is centered. The form has a title "Login" and contains two input fields for "Email" and "Password". Below the password field is a checkbox labeled "Ingat Password" and a link "Lupa Password?". A blue "Login" button is positioned to the right of the form. At the bottom of the form, there is a link "Butuh Akun? Daftar!". The footer of the page contains the text "© Copyright TPQ Al-Hikmah. All Rights Reserved".

Setelah proses pendaftaran berhasil, maka santri bisa melakukan login pada halaman Login. Pada saat login, terjadi proses validasi yang mana proses tersebut berlangsung pada file LoginController.php dengan function yang digunakan untuk validasi adalah authenticate(). Pada proses ini yang divalidasi adalah email dan passwordnya. Validasi ini tidak menggunakan tabel santri karena proses validasi sendiri hanya bisa divalidasi menggunakan tabel user. Apabila email dan password benar maka session akan di-*regenerate* secara otomatis dan nantinya santri akan diarahkan ke Dashboard.

#### LoginController.php

```
<?php
```

```

namespace App\Http\Controllers;

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

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

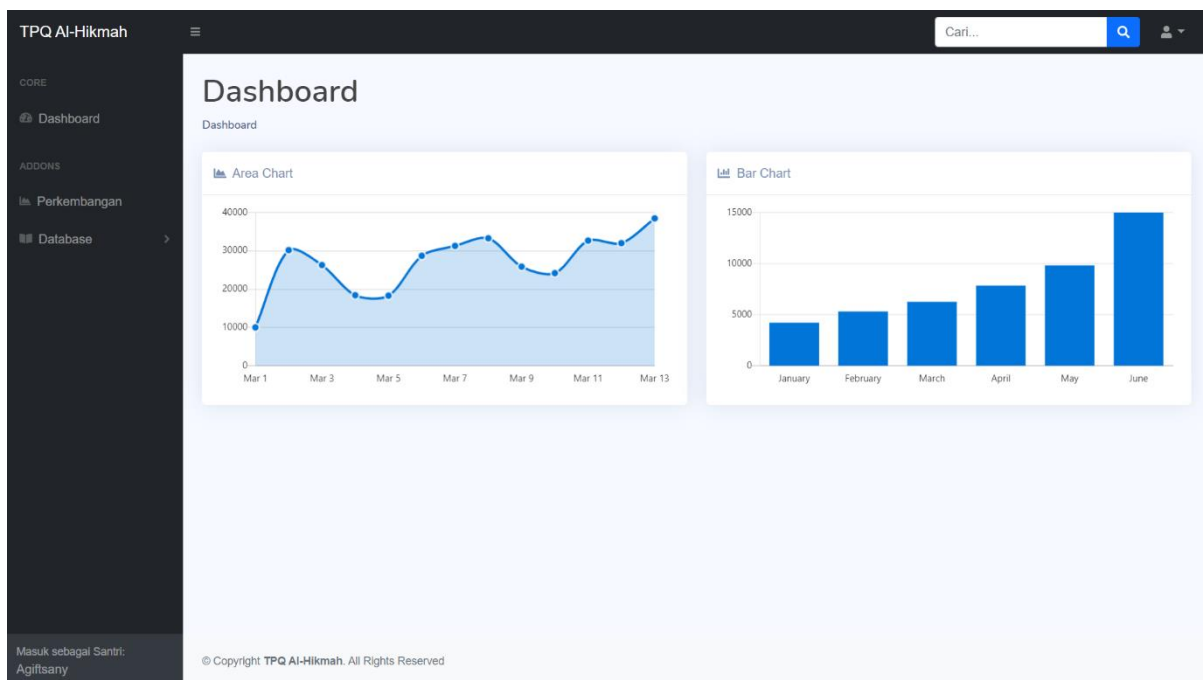
        if (Auth::attempt($credentials)) {
            $request->session()->regenerate();

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

        return back()->with('loginError', 'Login Gagal!');
    }
}

```

### 3.4 Dashboard

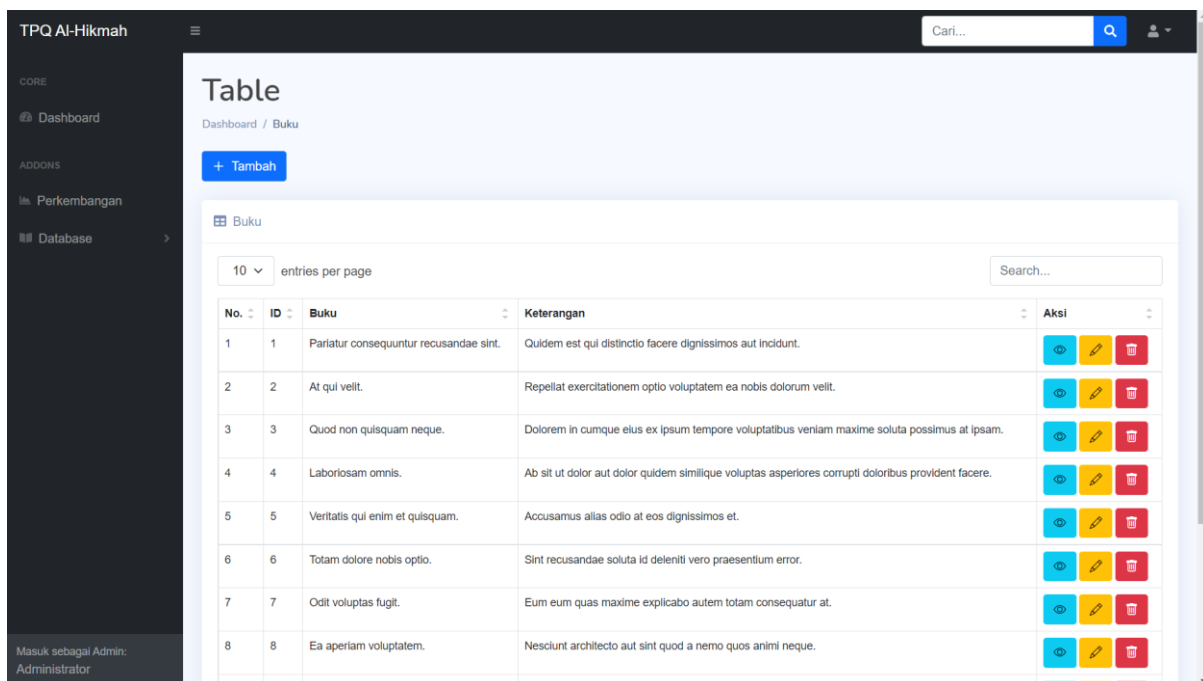


Dalam tampilan Dashboard santri, terdapat menu seperti buku, kemajuan, dan daftar nama pengurus pada TPQ Al-Hikmah. Dashboardnya sendiri juga menggunakan satu page yang mana masing-masing role memiliki akses yang berbeda. Selain itu, terdapat role yang mana role tersebut berganti nama sesuai dengan user yang login.

## dashboard.blade.php

```
<div class="collapse" id="pagesCollapseError" aria-  
labelledby="headingOne" data-bs-parent="#sidenavAccordionPages">  
    <nav class="sb-sidenav-menu-nested nav">  
        <a class="nav-link" href="{{ url('/buku-table') }}>Buku</a>  
        <a class="nav-link" href="{{ url('/kemajuan-  
table') }}">Kemajuan Santri</a>  
        <a class="nav-link" href="{{ url('/pengurus-  
table') }}">Pengurus</a>  
        @can('admin')  
        <a class="nav-link" href="{{ url('/peran-  
table') }}">Peran</a>  
        <a class="nav-link" href="{{ url('/santri-  
table') }}">Santri</a>  
        @endcan  
    </nav>  
</div>  
</div>  
</div>  
</div>  
<div class="sb-sidenav-footer">  
    <div class="small">Masuk sebagai {{ auth()->user()->role }}:</div>  
    {{ auth()->user()->nama }}  
</div>
```

## 3.5 Buku



TPQ Al-Hikmah

CORE

- Dashboard

ADDONS

- Perkembangan
- Database

Masuk sebagai Admin: Administrator

### Table

























Dashboard / Buku

+ Tambah

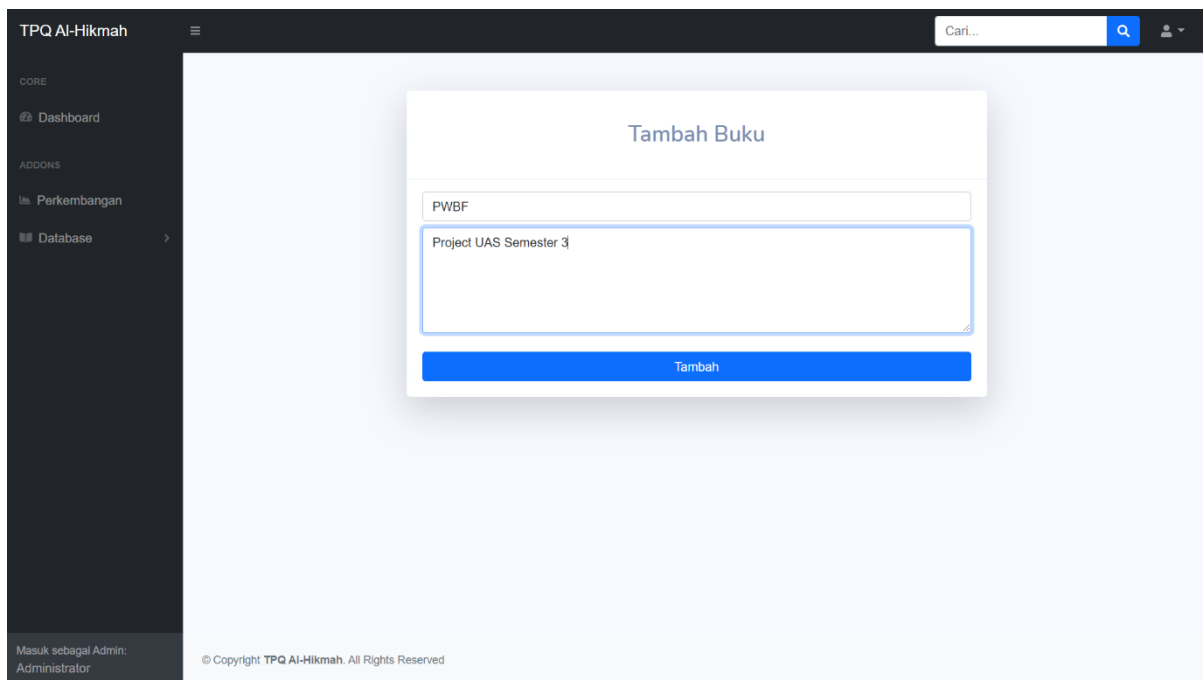
Buku

10 entries per page

Search...

No.	ID	Buku	Keterangan	Aksi
1	1	Parlatur consequuntur recusandae sint.	Quidem est qui distinctio facere dignissimos aut incidunt.	  
2	2	At qui velit.	Repellat exercitationem optio voluptatem ea nobis dolorum velit.	  
3	3	Quod non quisquam neque.	Dolorem in cumque elus ex ipsum tempore voluptatibus veniam maxime soluta possimus at ipsam.	  
4	4	Laboriosam omnis.	Ab sit ut dolor aut dolor quidem similique voluptas asperiores corrupti doloribus provident facere.	  
5	5	Veritatis qui enim et quisquam.	Accusamus alias odio at eos dignissimos et.	  
6	6	Totam dolore nobis optio.	Sint recusandae soluta id deleniti vero praesentium error.	  
7	7	Odit voluptas fugit.	Eum eum quas maxime explicabo autem totam consequatur at.	  
8	8	Ea aperiam voluptatem.	Nesciunt architecto aut sint quod a nemo quos animi neque.	  

Pada halaman Buku, terdapat tombol untuk menambah, mengedit, dan menghapus buku yang mana proses tersebut hanya bisa dilakukan oleh admin dan pengurus. Terdapat juga tombol show yang berfungsi untuk melihat daftar bab yang ada pada buku tersebut.



Pada halaman Tambah Buku, terdapat formulir yang harus diisi, yaitu judul buku dan keterangan. Proses menambah buku berlangsung pada file BukuController.php dengan function yang digunakan store(). Pada function ini terdapat sebuah variabel yang berisikan array dari nilai yang diinputkan pada formulir Tambah Buku. Apabila tombol Tambah ditekan, maka data akan dimasukkan ke tabel buku.

### BukuController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Buku;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class BukuController extends Controller
{
    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        $validatedData = $request->validate([
            'buku'          => 'required|max:50',
            'keterangan'    => 'required',
        ]);
    }
}
```

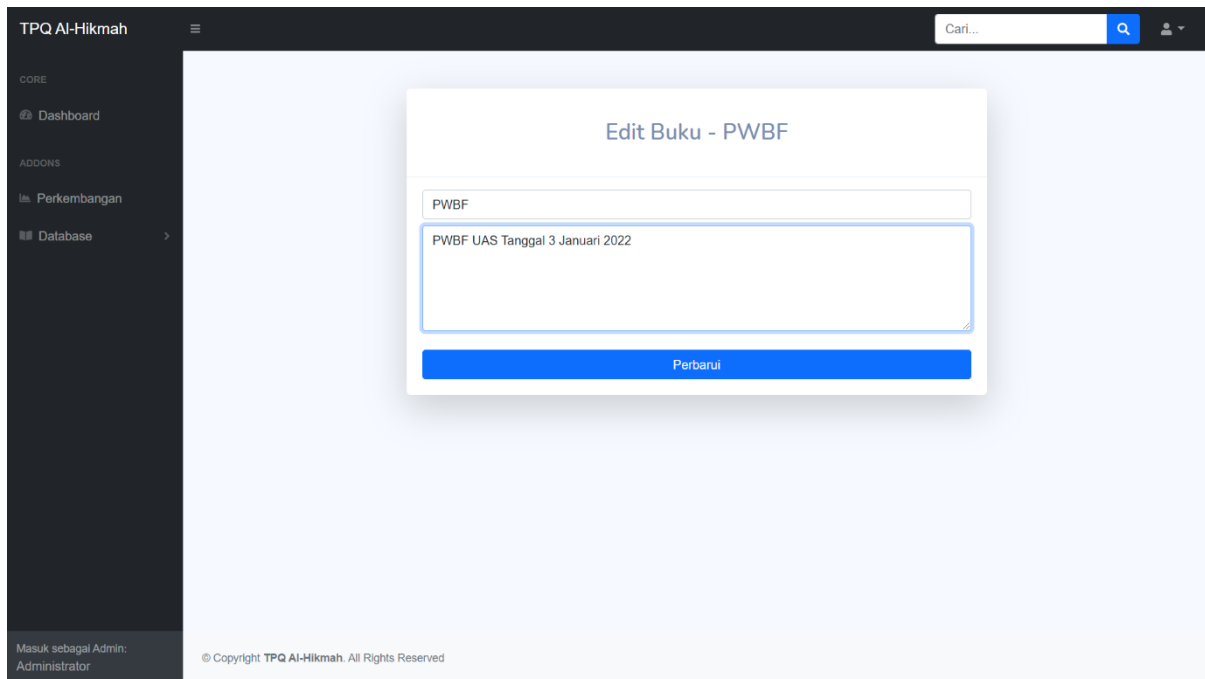
```

        Buku::create($validatedData);

        $request->session()->flash('successBuku','Buku Berhasil
Ditambahkan!');

        return redirect('/buku-table');
    }
}

```



Apabila pada proses penambahan yang dilakukan pada proses sebelumnya ada kesalahan, maka buku tersebut bisa diedit pada halaman Edit Buku. Pada saat menekan tombol edit, function edit() akan menerima parameter yang berisi id dari buku yang dipilih agar nantinya pada halaman Edit Buku hanya menampilkan buku yang dipilih saja. Proses edit buku ini berlangsung pada function udpate(). Proses ini melibatkan parameter request yang dikirimkan oleh aktor apabila terjadi perubahan pada buku. Request tersebut nantinya juga akan digunakan untuk mengambil id dari buku yang dipilih.

### BukuController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Buku;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class BukuController extends Controller
{

```

```

/**
 * Show the form for editing the specified resource.
 *
 * @param \App\Models\Buku $buku
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    return view('dashboard.edit.buku', [
        'buku' => Buku::find($id),
        "title" => Buku::find($id)->buku
    ]);
}

/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @param \App\Models\Buku $buku
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, Buku $buku)
{
    DB::table('bukus')->where('id',$request->id)->update([
        'buku' => $request->buku,
        'keterangan' => $request->keterangan
    ]);

    return redirect('/buku-table')->with('updateBuku','Data Buku
Berhasil Di-Update!');
}
}

```

The screenshot displays a web application interface for managing book data. At the top, a yellow notification banner indicates a successful update: "Data Buku Berhasil Di-Update!". Below this, a table titled "Table" (Dashboard / Buku) shows a list of book records. The table has columns: No., ID, Buku, Keterangan, and Aksi. A single record is visible: No. 11, ID 11, Buku PWBF, Keterangan PWBF UAS Tanggal 3 Januari 2022. The Aksi column contains icons for view, edit, and delete. The interface includes a sidebar on the left with navigation links for Dashboard, Perkembangan, and Database. A top navigation bar features a search bar and a user profile dropdown. The footer shows the user is logged in as "Masuk sebagai Admin: Administrator" and the copyright notice "© Copyright TPQ AI-Hikmah. All Rights Reserved".

Apabila buku sudah tidak digunakan lagi, aktor dapat menghapus buku tersebut dengan menekan tombol hapus. Proses ini melibatkan function `destroy()` yang berisi parameter dari id



buku yang dipilih. Selain itu, terdapat peringatan konfirmasi sebelum menghapus buku tersebut.

### BukuController.php

```
<?php

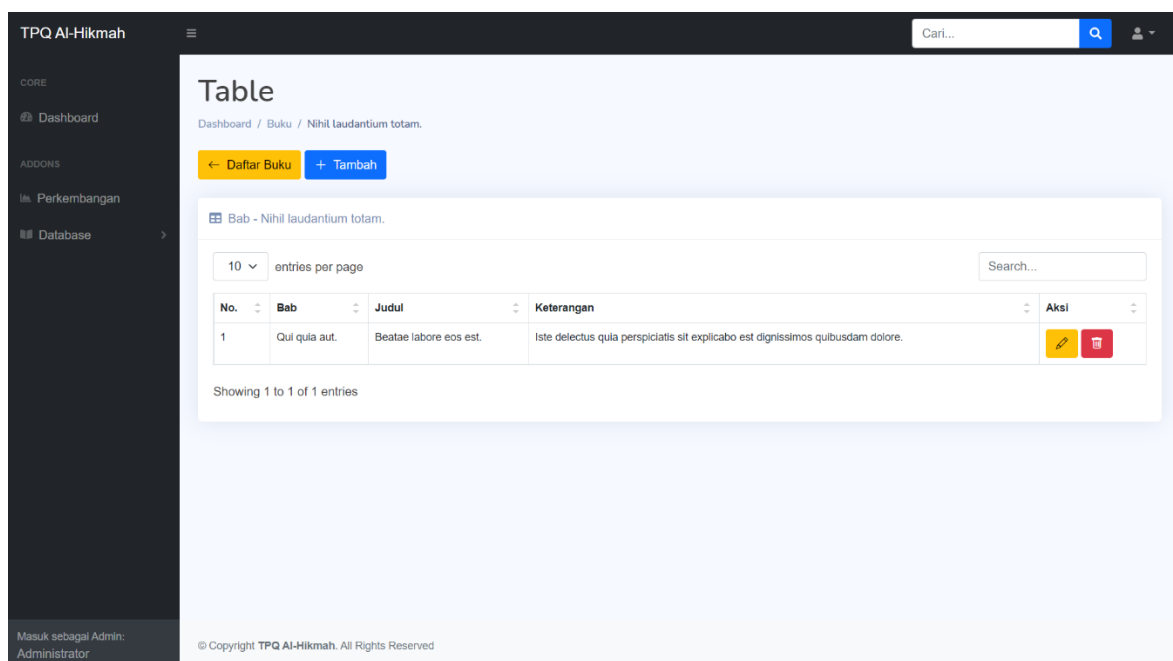
namespace App\Http\Controllers;

use App\Models\Buku;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;



class BukuController extends Controller
{
    /**
     * Remove the specified resource from storage.
     *
     * @param \App\Models\Buku $buku
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        Buku::find($id)->delete();

        return redirect('/buku-table')->with('deleteBuku', 'Buku Berhasil
Dihapus!');
    }
}
```

## 3.6 Bab



The screenshot displays a web application interface for managing book chapters. The sidebar on the left contains navigation links for 'CORE' (Dashboard), 'ADDONS' (Perkembangan, Database), and a login section. The main content area is titled 'Table' and shows a list of chapters. The table has columns for 'No.', 'Bab', 'Judul', 'Keterangan', and 'Aksi'. A single entry is visible with the chapter number 1, title 'Qui quia aut.', and a detailed description. The 'Aksi' column contains edit and delete icons. The interface also includes a search bar, a 'Daftar Buku' button, and a 'Tambah' button.

No.	Bab	Judul	Keterangan	Aksi
1	Qui quia aut.	Beatae labore eos est.	Iste delectus quia perspicatis sit explicabo est dignissimos quibusdam dolore.	 

Pada halaman Bab terdapat perbedaan proses untuk cara menampilkannya. Pada bab, mengandung foreign key yang berasal dari id dari buku sehingga pada function index() mengandung parameter yang berisi id dari buku. Nantinya id tersebut digunakan untuk mencocokkan bab dengan buku yang dipilih sehingga saat tombol show ditekan hanya akan menampilkan bab dari buku yang dipilih saja. Pada bab juga terdapat opsi untuk menambah, mengedit, dan menghapus bab.

#### **BabController.php**

```
<?php

namespace App\Http\Controllers;

use App\Models\Bab;
use App\Models\Buku;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class BabController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index(Buku $id)
    {
        return view('dashboard.show.bab', [
            'bab' => Bab::where('id_buku', $id->id)->get(),
            'buku' => $id,
            'title' => "Buku",
            'counter' => 1,
        ]);
    }
}
```

Pada saat menekan tombol tambah, function create() juga mengandung parameter yang berisi id buku sehingga saat formulir ditampilkan judul buku akan terisi secara otomatis sesuai dengan buku yang dipilih. Pada bagian halaman tambah Bab, terdapat kolom-kolom yang harus diisi, seperti bab, judul bab, dan keterangan. Data-data ini nantinya akan dikirim ke database dengan function store(). Function ini juga mengandung parameter yang berisi request yang didapat setelah aktor menekan tombol tambah.

### BabController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Bab;
use App\Models\Buku;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class BabController extends Controller
{
    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create($id)
    {
        return view('dashboard.create.bab', [
            'buku' => Buku::find($id),
            'title' => "Bab",
        ]);
    }
}
```

```

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validatedData = $request->validate([
        'bab' => 'required|max:50',
        'judul' => 'required|max:100',
        'id_buku' => 'required',
        'keterangan' => 'required',
    ]);

    Bab::create($validatedData);

    $request->session()->flash('successBab', 'Bab Berhasil
Ditambahkan!');

    return redirect('/buku-table');
}
}

```

The screenshot shows a web application interface for TPQ AI-Hikmah. On the left is a dark sidebar with navigation links: CORE (Dashboard), ADDONS (Perkembangan, Database), and a login section (Masuk sebagai Admin: Administrator). The main content area has a light blue background. A modal window titled 'Edit Bab - 2' is centered, containing a form with four input fields: 'bab' with the value '2', 'Kajian Pustaka', 'Nihil laudantium totam.', and 'Pemrograman Web Berbasis Framework Semester 3'. A blue button labeled 'Perbarui' is at the bottom of the form. The footer of the application shows the copyright notice: '© Copyright TPQ AI-Hikmah. All Rights Reserved'.

Bab ini juga bisa diedit apabila ada kesalahan atau pembaruan. Pada saat menekan tombol edit, function edit() akan menerima parameter yang berisi id dari bab yang dipilih agar nantinya pada halaman Edit Bab hanya menampilkan bab yang dipilih saja. Proses edit bab ini berlangsung pada function update(). Proses ini melibatkan parameter request yang diterima apabila aktor telah menekan tombol perbarui. Request tersebut nantinya juga akan digunakan untuk mengambil id dari buku yang dipilih.

## BabController.php

```
<?php

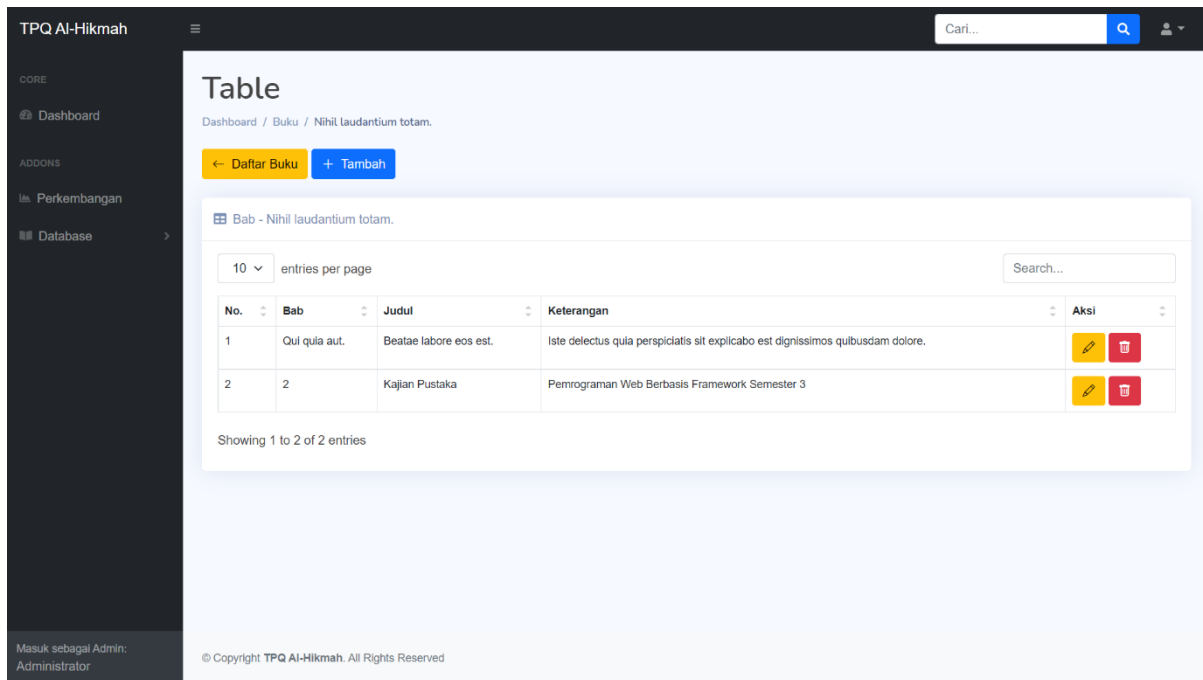
namespace App\Http\Controllers;

use App\Models\Bab;
use App\Models\Buku;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class BabController extends Controller
{
    /**
     * Show the form for editing the specified resource.
     *
     * @param \App\Models\Bab $bab
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        return view('dashboard.edit.bab', [
            'bab' => Bab::find($id),
            'bukus' => Buku::all(),
            'title' => Bab::find($id)->bab
        ]);
    }

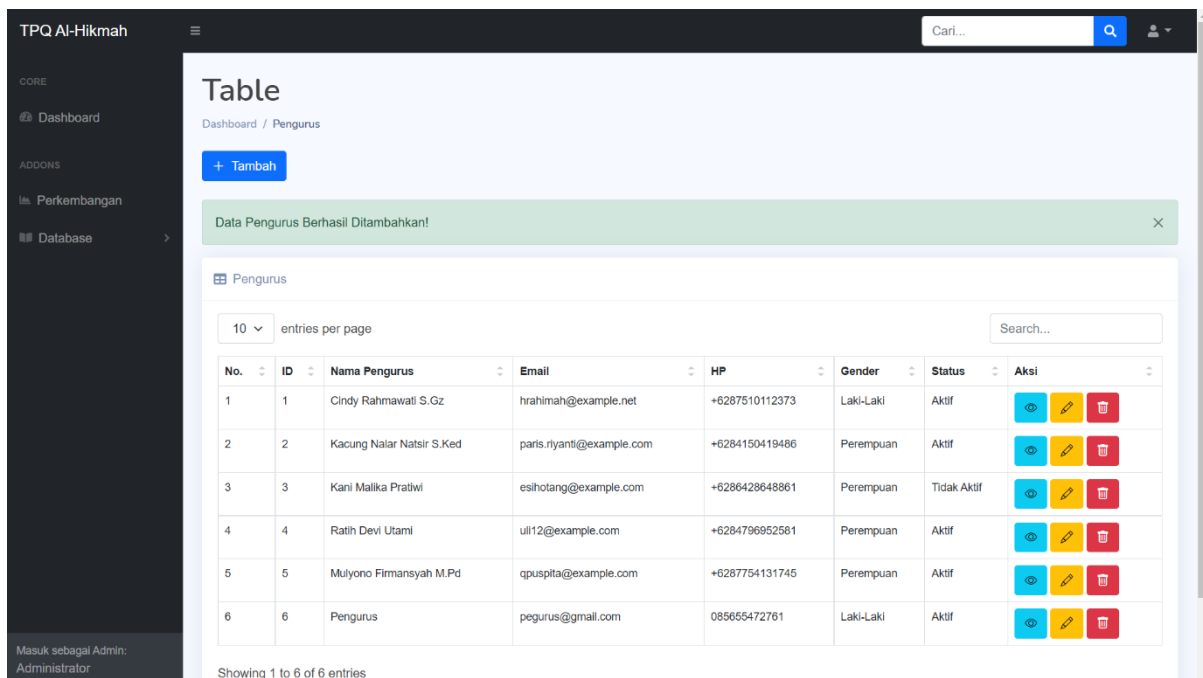
    /**
     * Update the specified resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @param \App\Models\Bab $bab
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request)
    {
        DB::table('babs')->where('id',$request['id'])->update([
            'bab' => $request['bab'],
            'judul' => $request['judul'],
            'id_buku' => $request['id_buku'],
            'keterangan' => $request['keterangan'],
        ]);

        return redirect('/buku-table')->with('updateBab','Data Bab
        Berhasil Di-Update!');
    }
}
```



Apabila bab sudah tidak digunakan lagi, aktor dapat menghapus bab tersebut dengan menekan tombol hapus. Proses ini melibatkan function `destroy()` yang berisi parameter dari id bab yang dipilih. Selain itu, terdapat peringatan konfirmasi sebelum menghapus bab tersebut.

### 3.7 Pengurus



Pada halaman Pengurus, terdapat tombol untuk menambah, mengedit, menghapus, dan show pengurus. Namun, tombol tersebut hanya bisa diakses oleh admin. Sedangkan selain admin hanya bisa READ.

Apabila pada TPQ Al-Hikmah kekurangan pengurus, admin bisa menambah pengurus baru yang nantinya bisa mengakses beberapa fitur yang ada pada TPQ Al-Hikmah. Formulir berisikan nama, jenis kelamin, email, telepon, dan password. Data-data ini kemudian dikirim ke database melalui perintah yang ada pada function store(). Selain disimpan dalam tabel pengurus, data ini juga nantinya juga disimpan pada tabel user karena untuk validasi password diperlukan tabel user sama seperti pada registrasi santri yang telah dibahas sebelumnya. Untuk password juga menggunakan enkripsi bcrypt.

### PengurusController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\User;
use Illuminate\Foundation\Bus\PendingChain;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;

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

```

{
    $validatedData = $request->validate([
        'nama'          => 'required|min:3|max:50',
        'gender'        => 'required',
        'hp'            =>
'required|unique:penguruses|unique:santris',
        'email'         =>
'required|email:dns|unique:penguruses|unique:santris',
        'password'      => 'required|min:8|max:32',
    ]);

    $validatedData['password'] = bcrypt($validatedData['password']);

    Pengurus::create($validatedData);

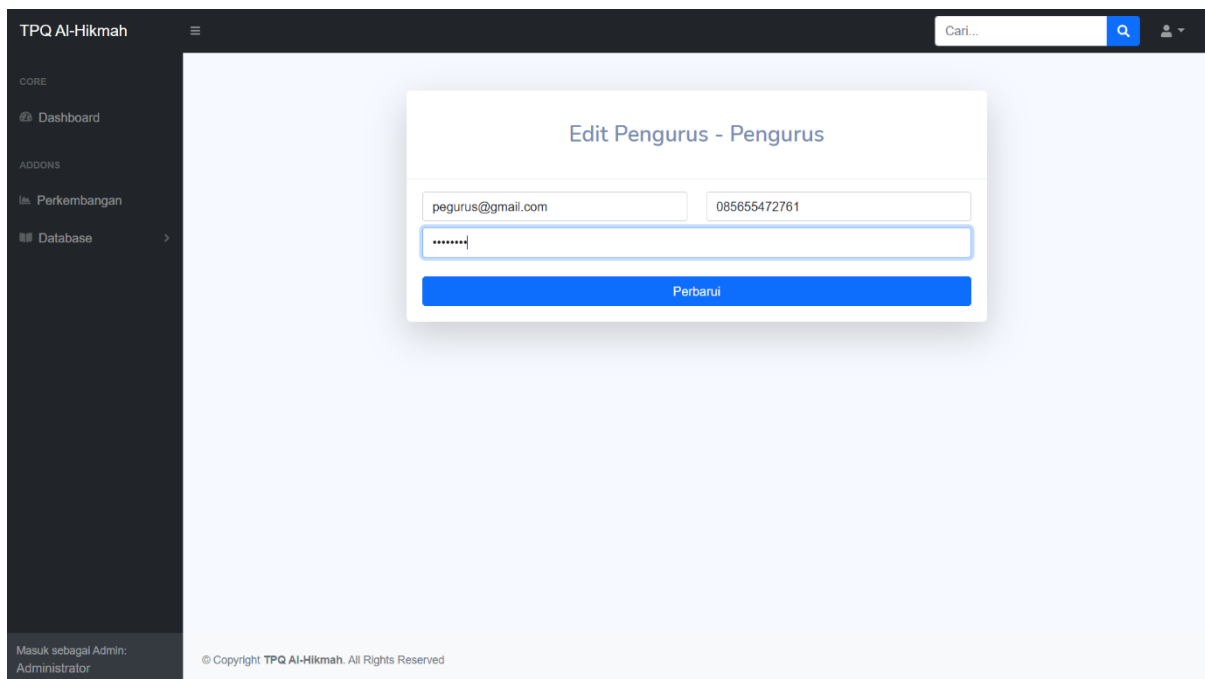
    $pengurusid = Pengurus::latest()->first()->id;

    DB::table('users')->insert([
        'id_pengurus'    => $pengurusid,
        'nama'          => $validatedData['nama'],
        'email'          => $validatedData['email'],
        'password'       => $validatedData['password'],
        'role'           => 'Pengurus',
    ]);

    $request->session()->flash('success','Data Pengurus Berhasil
Ditambahkan!');

    return redirect('/pengurus-table');
}
}

```



Pengurus bisa melakukan update pada data yang telah diregistrasikan sebelumnya. Proses untuk menampilkan formulir edit ini terjadi pada function edit(). Function ini memiliki



parameter yang berisi id dari pengurus yang dipilih. Sedangkan untuk proses update datanya terjadi pada function update() yang memiliki parameter request yang dikirimkan oleh aktor setelah menekan tombol perbarui. Request ini nantinya juga digunakan untuk mencocokkan antara id dengan pengurus yang ingin diedit datanya.

### PengurusController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\User;
use Illuminate\Foundation\Bus\PendingChain;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;

class PengurusController extends Controller
{
    /**
     * Show the form for editing the specified resource.
     *
     * @param \App\Models\Pengurus $pengurus
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        return view('dashboard.edit.pengurus', [
            'pengurus' => Pengurus::find($id),
            "title"     => Pengurus::find($id)->nama
        ]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @param \App\Models\Pengurus $pengurus
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, Pengurus $pengurus, User $user)
    {
        DB::table('penguruses')->where('id',$request->id)->update([
            'hp'          => $request->hp,
            'email'       => $request->email,
            'password'    => Hash::make($request->newPassword)
        ]);

        return redirect('/pengurus-table')->with('update','Data Pengurus Berhasil Di-Update!');
    }
}
```

TPQ Al-Hikmah

Dashboard / Pengurus



















+ Tambah

Data Pengurus Berhasil Di-Update!

Pengurus

10 entries per page

Search...

No.	ID	Nama Pengurus	Email	HP	Gender	Status	Aksi
1	1	Cindy Rahmawati S.Gz	hrahimah@example.net	+6287510112373	Laki-Laki	Aktif	  
2	2	Kacung Nalar Natsir S.Ked	paris.riyanti@example.com	+6284150419486	Perempuan	Aktif	  
3	3	Kani Malika Pratiwi	esihotang@example.com	+6286428648861	Perempuan	Tidak Aktif	  
4	4	Ratih Devi Utami	uul12@example.com	+6284796952581	Perempuan	Aktif	  
5	5	Mulyono Firmansyah M.Pd	qpuspita@example.com	+6287754131745	Perempuan	Aktif	  
6	6	Pengurus	pegurus@gmail.com	085655472761	Laki-Laki	Aktif	  

Masuk sebagai Admin: Administrator

Showing 1 to 6 of 6 entries

Apabila pengurus sudah purnatugas, admin dapat menghapus pengurus tersebut dengan menekan tombol hapus. Proses ini melibatkan function `destroy()` yang berisi parameter dari id pengurus yang dipilih. Pada function ini juga akan menghapus data pada tabel user sehingga pengurus yang telah dihapus tidak bisa login kembali. Selain itu, terdapat peringatan konfirmasi sebelum menghapus pengurus tersebut.

### PengurusController.php

```
<?php

namespace App\Http\Controllers;

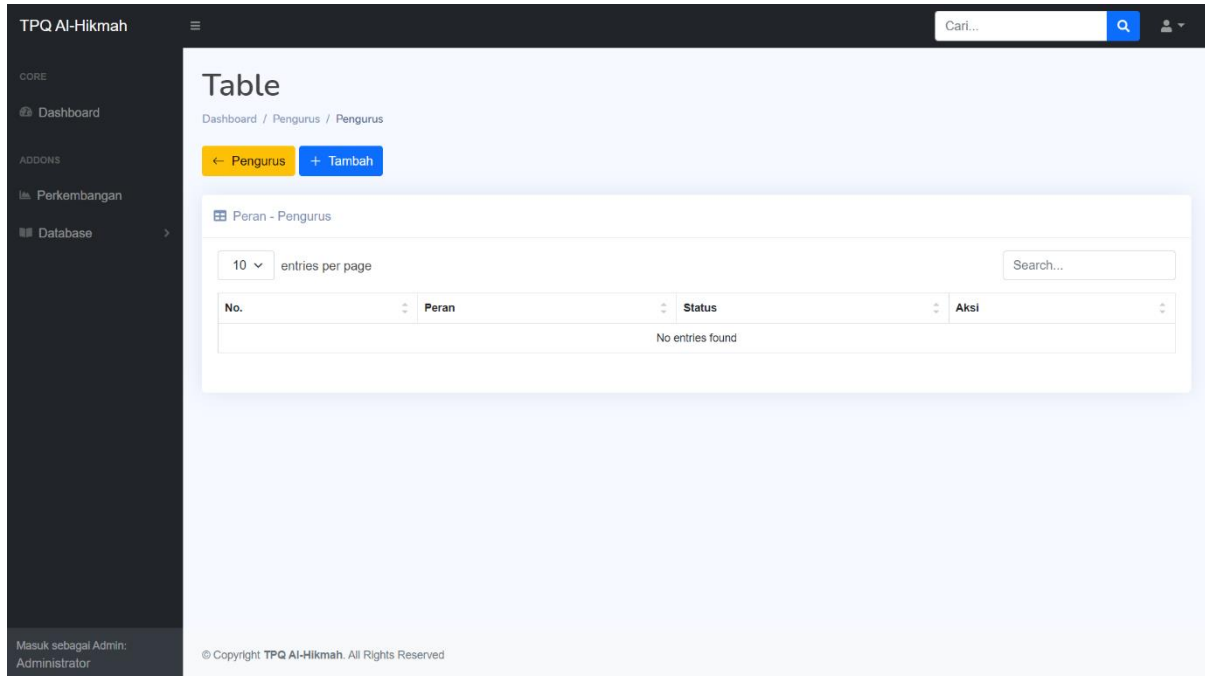
use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\User;
use Illuminate\Foundation\Bus\PendingChain;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;

class PengurusController extends Controller
{
    /**
     * Remove the specified resource from storage.
     *
     * @param \App\Models\Pengurus $pengurus
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        Pengurus::find($id)->delete();
        User::where('id_pengurus', $id)->delete();
    }
}
```

```

        return redirect('/pengurus-table')->with('delete','Pengurus
Berhasil Dihapus!');
    }
}

```



Pada halaman detail peran, terdapat daftar peran yang diikuti oleh pengurus. Detail peran mengandung foreign key yang berasal dari id pengurus sehingga untuk menampilkan tabel detail peran juga diperlukan id dari pengurus. Proses ini berlangsung pada function index().

### Detail\_PeranController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\Peran;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

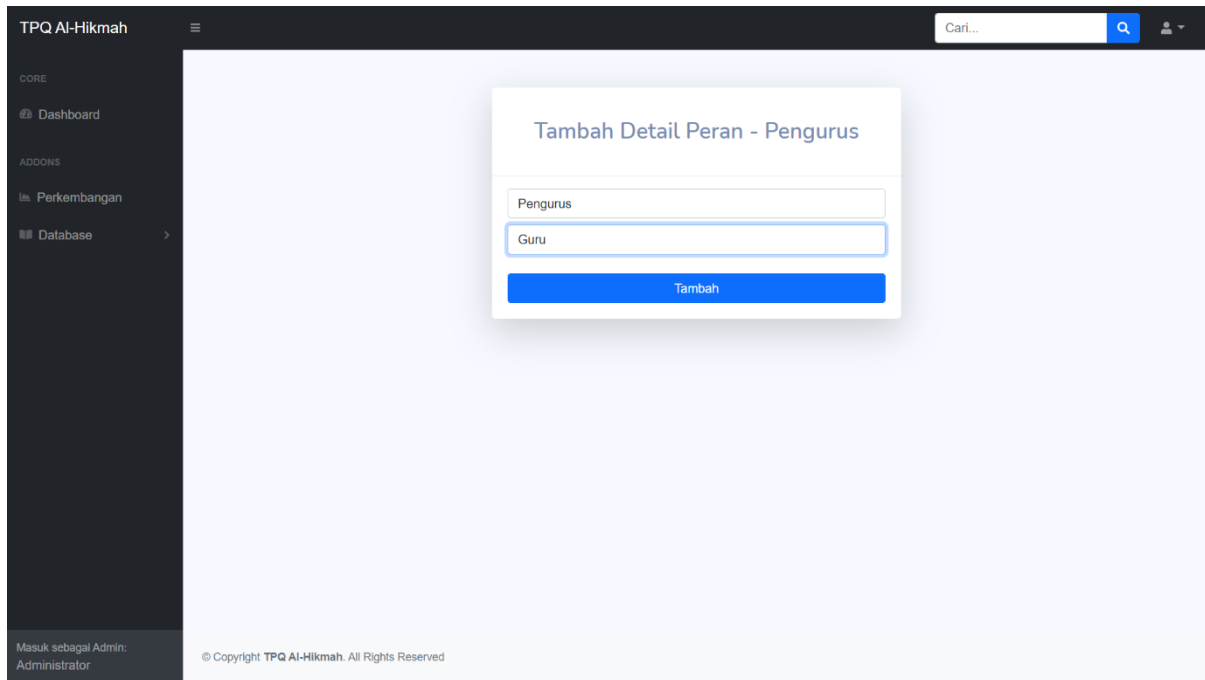
class Detail_PeranController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index($id)
    {
        return view('dashboard.show.detail-peran', [
            'peran' => Detail_Peran::where('id_pengurus', $id)-
>with('peran')->get(),

```

```

        "title"          => Pengurus::find($id)->nama,
        'idpengurus'     => Pengurus::find($id)->id,
        'counter'        => 1
    ]);
}
}

```



Pada saat menekan tombol tambah, function create() juga mengandung parameter yang berisi id pengurus. Pada bagian halaman tambah Peran, terdapat kolom nama peran yang harus diisi. Data ini nantinya akan dikirim ke database dengan function store() yang memiliki parameter request. Request ini didapat setelah aktor menekan tombol tambah.

### Detail\_PeranController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\Peran;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

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

```

{
    return view('dashboard.create.detail-peran', [
        'pengurus' => Pengurus::find($id),
        'title'      => Pengurus::find($id)->nama,
        'peran'      => Peran::all()
    ]);
}

public function getDetailPeran($id)
{
    $detailperan = DB::table('detail__perans')->
>where("id_peran",$id)->pluck('id','peran');
    return response()->json($detailperan);
}

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validatedData = $request->validate([
        'id_pengurus' => 'required',
        'id_peran'    => 'required',
    ]);

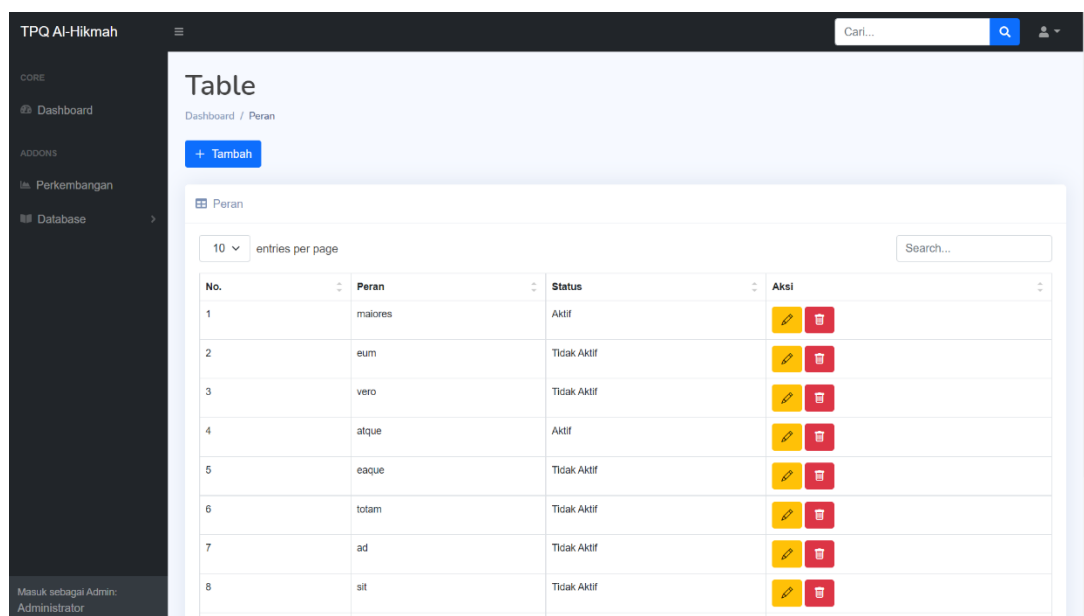
    Detail_Peran::create($validatedData);

    $request->session()->flash('successDetailPeran','Data Peran
Berhasil Ditambahkan!');

    return redirect('/pengurus-table');
}
}

```

### 3.8 Peran



TPQ AI-Hikmah

Cari...

CORE

Dashboard

ADDITIONS

Perkembangan

Database

Masuk sebagai Admin: Administrator

### Table

















Dashboard / Peran

+ Tambah

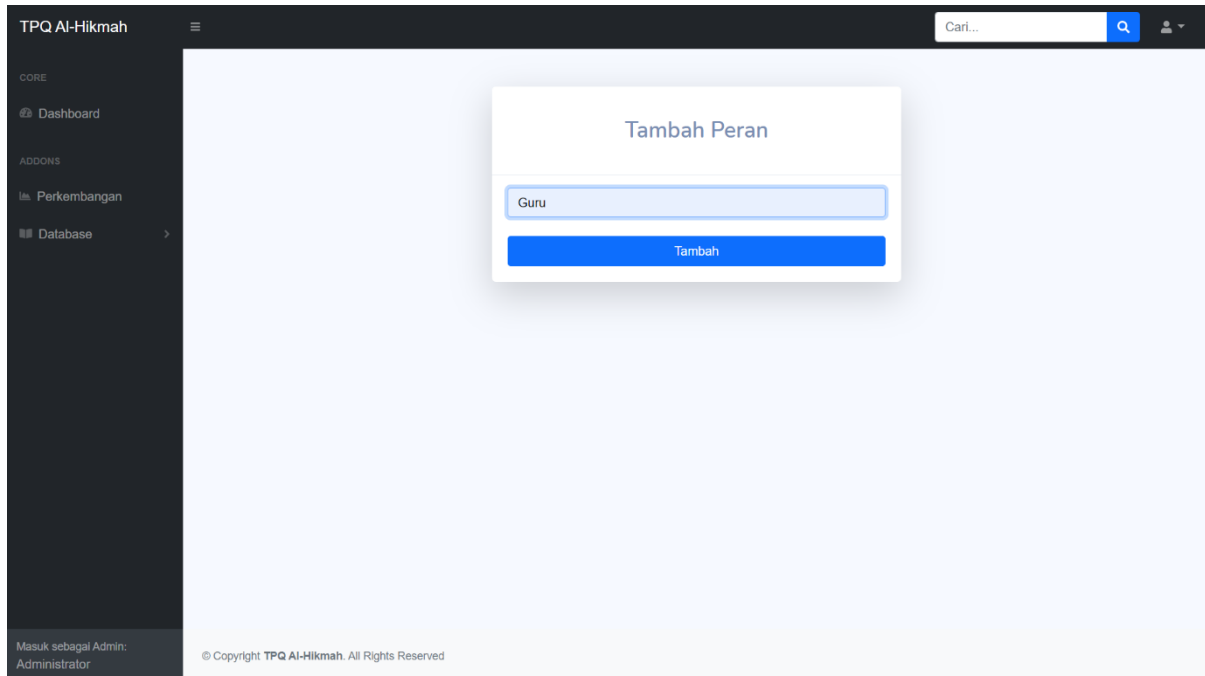
Peran

10 entries per page

Search...

No.	Peran	Status	Aksi
1	maiores	Aktif	 
2	eum	Tidak Aktif	 
3	vero	Tidak Aktif	 
4	atque	Aktif	 
5	eaque	Tidak Aktif	 
6	totam	Tidak Aktif	 
7	ad	Tidak Aktif	 
8	sit	Tidak Aktif	 

Pada halaman Peran, terdapat tombol untuk menambah, mengedit, dan menghapus peran. Namun, tabel peran hanya bisa diakses oleh admin.



Pada halaman Tambah Peran, terdapat kolom yang harus diisi, yaitu nama peran yang ingin ditambahkan. Proses penyimpanan dari nama peran yang baru ini terjadi pada function store(). Apabila tombol Tambah ditekan, maka data akan dimasukkan ke tabel peran.

### PeranController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\Peran;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class PeranController extends Controller
{
    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        $validatedData = $request->validate([
            'peran' => 'required|max:20|unique:perans',
        ]);
    }
}
```

```

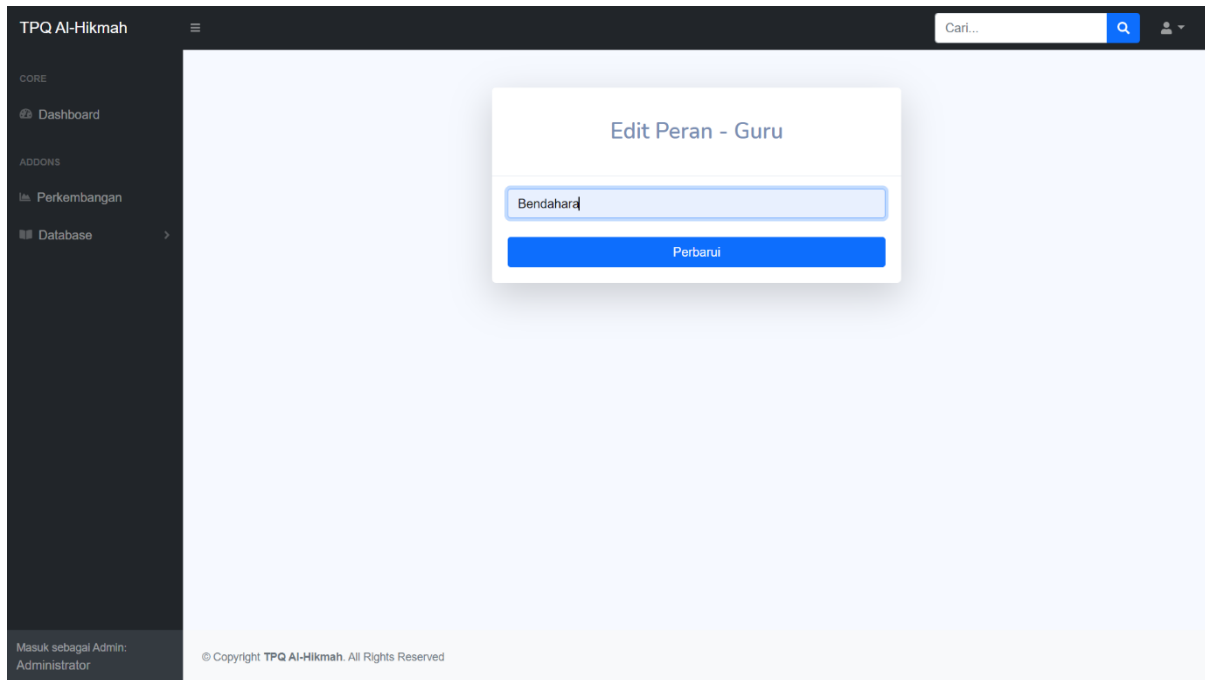
    });

    Peran::create($validatedData);

    $request->session()->flash('success', 'Peran Berhasil
Ditambahkan!');

    return redirect('/peran-table');
}
}

```



Apabila terjadi kesalahan penulisan pada peran, admin bisa mengedit dengan menekan tombol edit. Pada saat tombol ditekan, function edit() parameteranya akan mendapatkan id dari peran yang dipilih. Sedangkan untuk proses updatenya sendiri terjadi pada function store() yang memiliki parameter request. Request ini nantinya juga akan digunakan untuk mencocokkan dengan id peran yang dipilih sehingga tidak terjadi tertukarnya data.

### PeranController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\Peran;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class PeranController extends Controller
{

```

```

/**
 * Show the form for editing the specified resource.
 *
 * @param \App\Models\Peran $peran
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    return view('dashboard.edit.peran', [
        'peran'      => Peran::find($id),
        'title'       => Peran::find($id)->peran
    ]);
}

/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @param \App\Models\Peran $peran
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, Peran $peran)
{
    DB::table('perans')->where('id',$request->id)->update([
        'peran'      => $request->peran,
    ]);

    return redirect('/peran-table')->with('update','Data Peran
    Berhasil Di-Update!');
}
}

```

TPQ AI-Hikmah

CORE

Dashboard

ADDONS

Perkembangan

Database

pwbf\_praktikum.test menyatakan

Apakah Anda yakin ingin menghapus data ini?

Ok

Batal

Cari...

Table

Dashboard / Peran

+ Tambah

Data Peran Berhasil Di-Update!

Peran

10 entries per page

Search...

No.	Peran	Status	Aksi
21	Bendahara	Aktif	<div></div> <div></div>

Showing 21 to 21 of 21 entries

1

2

3

Masuk sebagai Admin: Administrator

© Copyright TPQ AI-Hikmah. All Rights Reserved



Apabila peran sudah tidak digunakan lagi, admin dapat menghapus peran tersebut dengan menekan tombol hapus. Function yang digunakan adalah `destroy()`. Function ini memiliki parameter yang berisi id dari peran yang dipilih.

### PeranController.php

```
<?php

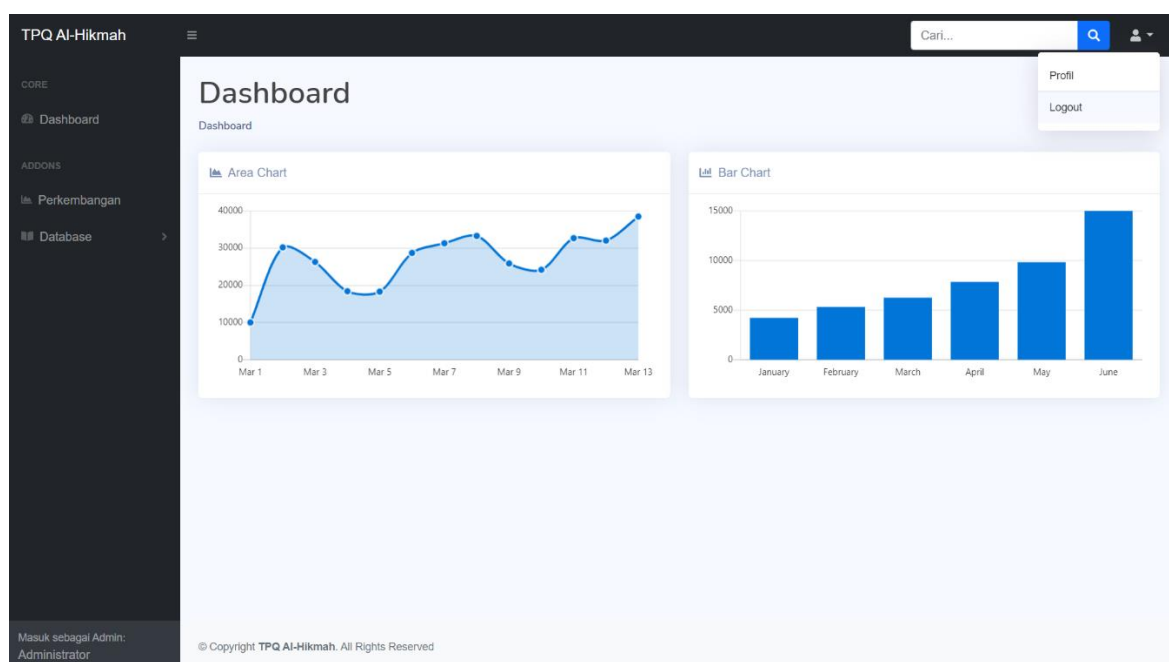
namespace App\Http\Controllers;

use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\Peran;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class PeranController extends Controller
{
    /**
     * Remove the specified resource from storage.
     *
     * @param \App\Models\Peran $peran
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        Peran::find($id)->delete();

        return redirect('/peran-table')->with('delete','Peran Berhasil
Dihapus!');
    }
}
```

## 3.9 Logout



Pada saat menekan tombol logout, function logout() akan mengakhiri *session*-nya dan diarahkan ke halaman utama (Home).

#### LoginController.php

```
<?php

namespace App\Http\Controllers;

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

class LoginController extends Controller
{
    public function logout(Request $request)
    {
        Auth::logout();

        $request->session()->invalidate();

        $request->session()->regenerateToken();

        return redirect('/');
    }
}
```

# LAMPIRAN

## 1. GitHub

GitHub: <https://github.com/agiftsanyazhar/Kel9-PWBF-Praktikum>

## 2. Register

### RegisterController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Santri;
use App\Models\User;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\DB;

class RegisterController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('register', [
            "title" => "Daftar"
        ]);
    }

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

    /**
     * Store a newly created resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        $validatedData = $request->validate([
            'nama' => 'required|min:3|max:50',
            'gender' => 'required',
            'tgl_lhr' => 'required',
            'kota_lhr' => 'required|max:40',
            'nama_ortu' => 'required|min:3|max:50',
            'alamat_ortu' => 'required|max:100',
        ]);
    }
}
```

```

        'hp' =>
        'required|unique:santris|unique:penguruses',
        'email' =>
        'required|email:dns|unique:santris|unique:penguruses',
        'password' => 'required|min:8|max:32',
    ]);

    $validatedData['password'] =
bcrypt($validatedData['password']);

    Santri::create($validatedData);

    $santriid = Santri::latest()->first()->id;

    DB::table('users')->insert([
        'id_santri' => $santriid,
        'nama' => $validatedData['nama'],
        'email' => $validatedData['email'],
        'password' => $validatedData['password'],
        'role' => 'Santri',
    ]);

    $request->session()->flash('success','Registrasi Berhasil!
Silakan Login');

    return redirect('/login');
}

/**
 * Display the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    //
}

/**
 * Show the form for editing the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    //
}

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

```

```

/**
 * Remove the specified resource from storage.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function destroy($id)
{
    Santri::find($id)->delete();
    User::where('id_santri', $id)->delete();

    return redirect('/santri-table')->with('delete','Santri
Berhasil Dihapus!');
}
}

```

### 3. Login

```

Logincontroller.php

<?php

namespace App\Http\Controllers;

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

class LoginController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('login', [
            "title" => "Login"
        ]);
    }

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

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

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

        return back()->with('loginError', 'Login Gagal!');
    }

    public function logout(Request $request)

```

```

{
    Auth::logout();

    $request->session()->invalidate();

    $request->session()->regenerateToken();

    return redirect('/');
}

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

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

/**
 * Display the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    //
}

/**
 * Show the form for editing the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    //
}

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

```

```

{
    //

}

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

```

## 4. Dashboard

**dashboard.blade.php**

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no" />
    <meta name="description" content="" />
    <meta name="author" content="" />
    <title>Dashboard</title>
    <link href="https://cdn.jsdelivrivr.net/npm/simple-
datatables@latest/dist/style.css" rel="stylesheet" />
    <link href="css/dashboard.css" rel="stylesheet" />
    <link href="assets/css/style.css" rel="stylesheet">
    <script src="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.3/js/all.min.js" crossorigin="anonymous"></script>

    <script src="peran.js"></script>

    <!-- Favicons -->
    <link href="img/favicon.png" rel="icon">
    <link href="img/apple-touch-icon.png" rel="apple-touch-icon">

    <link href="assets/vendor/bootstrap-icons/bootstrap-icons.css"
rel="stylesheet">
</head>

<body class="sb-nav-fixed">
    <nav class="sb-topnav navbar navbar-expand navbar-dark bg-
dark">
        <!-- Navbar Brand -->
        <a class="navbar-brand ps-3" href="/">TPQ Al-Hikmah</a>
        <!-- Sidebar Toggle -->
        <button class="btn btn-link btn-sm order-1 order-lg-0 me-4
me-lg-0" id="sidebarToggle" href="#"><i class="fas fa-
bars"></i></button>
        <!-- Navbar Search -->
        <form class="d-none d-md-inline-block form-inline ms-auto
me-0 me-md-3 my-2 my-md-0">
            <div class="input-group">

```

```

        <input class="form-control" type="text"
placeholder="Cari..." aria-label="Search" aria-
describedby="btnNavbarSearch" />
        <button class="btn btn-primary"
id="btnNavbarSearch" type="button"><i class="fas fa-
search"></i></button>
    </div>
</form>
<!-- Navbar-->
<ul class="navbar-nav ms-auto ms-md-0 me-3 me-lg-4">
    <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle"
id="navbarDropdown" href="#" role="button" data-bs-toggle="dropdown"
aria-expanded="false"><i class="fas fa-user fa-fw"></i></a>
        <ul class="dropdown-menu dropdown-menu-end" aria-
labelledby="navbarDropdown">
            <li><a class="dropdown-item"
href="profile">Profil</a></li>
            <li><hr class="dropdown-divider" /></li>
            <form action="/logout" method="post">
                @csrf
                <li><button class="dropdown-item"
type="submit">Logout</button></li>
            </form>
        </ul>
    </li>
</ul>
</nav>
<div id="layoutSidenav">
    <div id="layoutSidenav_nav">
        <nav class="sb-sidenav accordion sb-sidenav-dark"
id="sidenavAccordion">
            <div class="sb-sidenav-menu">
                <div class="nav">
                    <div class="sb-sidenav-menu-
heading">Core</div>
                    <a class="nav-link" href="dashboard-
index">
                        <div class="sb-nav-link-icon"><i
class="fas fa-tachometer-alt"></i></div>
                        Dashboard
                    </a>
                    <div class="sb-sidenav-menu-
heading">Addons</div>
                    <a class="nav-link" href="charts">
                        <div class="sb-nav-link-icon"><i
class="fas fa-chart-area"></i></div>
                        Perkembangan
                    </a>
                    <a class="nav-link collapsed" href="#"
data-bs-toggle="collapse" data-bs-target="#collapsePages" aria-
expanded="false" aria-controls="collapsePages">
                        <div class="sb-nav-link-icon"><i
class="fas fa-book-open"></i></div>
                        Database
                        <div class="sb-sidenav-collapse-
arrow"><i class="fas fa-angle-down"></i></div>
                    </a>
                    <div class="collapse" id="collapsePages"
aria-labelledby="headingTwo" data-bs-parent="#sidenavAccordion">

```



```

<nav class="sb-sidenav-menu-nested nav
accordion" id="sidenavAccordionPages">
    <a class="nav-link collapsed"
href="#" data-bs-toggle="collapse" data-bs-
target="#pagesCollapseError" aria-expanded="false" aria-
controls="pagesCollapseError">
        Table
        <div class="sb-sidenav-
collapse-arrow"><i class="fas fa-angle-down"></i></div>
        </a>
        <div class="collapse"
id="pagesCollapseError" aria-labelledby="headingOne" data-bs-
parent="#sidenavAccordionPages">
            <nav class="sb-sidenav-menu-
nested nav">
                <a class="nav-link"
href="{{ url('/buku-table') }}">Buku</a>
                <a class="nav-link"
href="{{ url('/kemajuan-table') }}">Kemajuan Santri</a>
                <a class="nav-link"
href="{{ url('/pengurus-table') }}">Pengurus</a>
                @can('admin')
                <a class="nav-link"
href="{{ url('/peran-table') }}">Peran</a>
                <a class="nav-link"
href="{{ url('/santri-table') }}">Santri</a>
                @endcan
            </nav>
        </div>
    </nav>
</div>
</div>
<div class="sb-sidenav-footer">
    <div class="small">Masuk sebagai {{ auth()-
>user()->role }}:</div>
    {{ auth()->user()->nama }}
</div>
</nav>
</div>
<div id="layoutSidenav_content">
    @yield('container')
    <footer class="py-4 bg-light mt-auto">
        <div class="container-fluid px-4">
            <div class="d-flex align-items-center justify-
content-between small">
                <div class="text-muted">&copy; Copyright
<strong><span>TPQ Al-Hikmah</span></strong>. All Rights Reserved</div>
            </div>
        </div>
    </footer>
</div>
</div>
<script src="_____ " crossorigin="anonymous"></script>
<script src="js/scripts.js"></script>
<script src="_____ " crossorigin="anonymous"></script>
<script src="js/chart-area-demo.js"></script>
<script src="js/chart-bar-demo.js"></script>
<script src="_____ " crossorigin="anonymous"></script>
<script src="js/datatables-simple-demo.js"></script>
<script src="js/chart-pie-demo.js"></script>

```

```

<!-- =====
* Template Name: NiceAdmin - v2.2.0
* Template URL: _____
* Author: BootstrapMade.com
* License: _____
===== -->
<!-- Vendor JS Files -->
<script
src="assets/vendor/apexcharts/apexcharts.min.js"></script>
    {{-- <script
src="assets/vendor/bootstrap/js/bootstrap.bundle.min.js"></script> --
}}
    <script src="assets/vendor/chart.js/chart.min.js"></script>
    <script src="assets/vendor/echarts/echarts.min.js"></script>
    <script src="assets/vendor/quill/quill.min.js"></script>
    <script src="assets/vendor/simple-datatables/simple-
datatables.js"></script>
    <script src="assets/vendor/tinymce/tinymce.min.js"></script>
    <script src="assets/vendor/php-email-
form/validate.js"></script>

    <!-- Template Main JS File -->
    <script src="assets/js/main.js"></script>
</body>
</html>

```

## 5. Buku

### BukuController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Buku;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class BukuController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index(Buku $buku)
    {
        return view('dashboard.buku-table', [
            'bukus' => Buku::all(),
            'title' => "Buku",
            'counter' => 1,
        ]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
}

```

```

public function create()
{
    return view('dashboard.create.buku', [
        "title" => "Buku"
    ]);
}

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validatedData = $request->validate([
        'buku' => 'required|max:50',
        'keterangan' => 'required',
    ]);

    Buku::create($validatedData);

    $request->session()->flash('successBuku', 'Buku Berhasil
Ditambahkan!');

    return redirect('/buku-table');
}

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

/**
 * Show the form for editing the specified resource.
 *
 * @param \App\Models\Buku $buku
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    return view('dashboard.edit.buku', [
        'buku' => Buku::find($id),
        "title" => Buku::find($id)->buku
    ]);
}

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

```

```

{
    DB::table('bukus')->where('id',$request->id)->update([
        'buku'          => $request->buku,
        'keterangan'    => $request->keterangan
    ]);

    return redirect('/buku-table')->with('updateBuku','Data Buku
Berhasil Di-Update!');
}

/**
 * Remove the specified resource from storage.
 *
 * @param  \App\Models\Buku  $buku
 * @return \Illuminate\Http\Response
 */
public function destroy($id)
{
    Buku::find($id)->delete();

    return redirect('/buku-table')->with('deleteBuku','Buku
Berhasil Dihapus!');
}
}

```

## 6. Bab

### BabController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Bab;
use App\Models\Buku;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class BabController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index(Buku $id)
    {
        return view('dashboard.show.bab', [
            'bab'    => Bab::where('id_buku', $id->id)->get(),
            'buku'   => $id,
            'title'  => "Buku",
            'counter' => 1,
        ]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
}

```

```

public function create($id)
{
    return view('dashboard.create.bab', [
        'buku' => Buku::find($id),
        'title' => "Bab",
    ]);
}

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validatedData = $request->validate([
        'bab' => 'required|max:50',
        'judul' => 'required|max:100',
        'id_buku' => 'required',
        'keterangan' => 'required',
    ]);

    Bab::create($validatedData);

    $request->session()->flash('successBab', 'Bab Berhasil
Ditambahkan!');

    return redirect('/buku-table');
}

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

/**
 * Show the form for editing the specified resource.
 *
 * @param \App\Models\Bab $bab
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    return view('dashboard.edit.bab', [
        'bab' => Bab::find($id),
        'bukus' => Buku::all(),
        'title' => Bab::find($id)->bab
    ]);
}

/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request

```

```

    * @param \App\Models\Bab $bab
    * @return \Illuminate\Http\Response
    */
    public function update(Request $request)
    {
        DB::table('babs')->where('id',$request['id'])->update([
            'bab'          => $request['bab'],
            'judul'        => $request['judul'],
            'id_buku'      => $request['id_buku'],
            'keterangan'   => $request['keterangan'],
        ]);

        return redirect('/buku-table')->with('updateBab','Data Bab
Berhasil Di-Update!');
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param \App\Models\Bab $bab
     * @return \Illuminate\Http\Response
     */
    public function destroy(Request $request, $id)
    {
        Bab::find($id)->delete();

        $request->session()->flash('deleteBab','Bab Berhasil
Dihapus!');

        return redirect('/buku-table');
    }
}

```

## 7. Pengurus

```

PengurusController.php

<?php

namespace App\Http\Controllers;

use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\User;
use Illuminate\Foundation\Bus\PendingChain;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;

class PengurusController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index(Pengurus $pengurus)
    {
        return view('dashboard.pengurus-table', [
            'pengurus' => Pengurus::all(),

```

```

        "title"      => "Pengurus",
        'counter'    => 1
    });
}

/**
 * Show the form for creating a new resource.
 *
 * @return \Illuminate\Http\Response
 */
public function create()
{
    return view('dashboard.create.pengurus', [
        "title" => "Pengurus"
    ]);
}

/**
 * Store a newly created resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validatedData = $request->validate([
        'nama'          => 'required|min:3|max:50',
        'gender'         => 'required',
        'hp'             =>
'required|unique:penguruses|unique:santris',
        'email'          =>
'required|email:dns|unique:penguruses|unique:santris',
        'password'       => 'required|min:8|max:32',
    ]);

    $validatedData['password'] =
bcrypt($validatedData['password']);

    Pengurus::create($validatedData);

    $pengurusid = Pengurus::latest()->first()->id;

    DB::table('users')->insert([
        'id_pengurus'    => $pengurusid,
        'nama'           => $validatedData['nama'],
        'email'           => $validatedData['email'],
        'password'        => $validatedData['password'],
        'role'            => 'Pengurus',
    ]);

    $request->session()->flash('success', 'Data Pengurus Berhasil
Ditambahkan!');

    return redirect('/pengurus-table');
}

/**
 * Display the specified resource.
 *
 * @param  \App\Models\Pengurus  $pengurus
 * @return \Illuminate\Http\Response

```

```

    */
    public function show(Pengurus $pengurus)
    {
        //
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param \App\Models\Pengurus $pengurus
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        return view('dashboard.edit.pengurus', [
            'pengurus' => Pengurus::find($id),
            'title'      => Pengurus::find($id)->nama
        ]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @param \App\Models\Pengurus $pengurus
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, Pengurus $pengurus, User
$user)
    {
        DB::table('penguruses')->where('id',$request->id)->update([
            'hp'            => $request->hp,
            'email'         => $request->email,
            'password'      => Hash::make($request->newPassword)
        ]);

        return redirect('/pengurus-table')->with('update','Data
Pengurus Berhasil Di-Update!');
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param \App\Models\Pengurus $pengurus
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        Pengurus::find($id)->delete();
        User::where('id_pengurus', $id)->delete();

        return redirect('/pengurus-table')->with('delete','Pengurus
Berhasil Dihapus!');
    }
}

```

## 8. Detail Peran

**Detail\_PeranController.php**



```

<?php

namespace App\Http\Controllers;

use App\Models\Detail_Peran;
use App\Models\Pengurus;
use App\Models\Peran;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class Detail_PeranController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index($id)
    {
        return view('dashboard.show.detail-peran', [
            'peran' => Detail_Peran::where('id_pengurus',
$id)->with('peran')->get(),
            'title' => Pengurus::find($id)->nama,
            'idpengurus' => Pengurus::find($id)->id,
            'counter' => 1
        ]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create($id)
    {
        return view('dashboard.create.detail-peran', [
            'pengurus' => Pengurus::find($id),
            'title' => Pengurus::find($id)->nama,
            'peran' => Peran::all()
        ]);
    }

    public function getDetailPeran($id)
    {
        $detailperan = DB::table('detail__perans')-
>where("id_peran", $id)->pluck('id', 'peran');
        return response()->json($detailperan);
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        $validatedData = $request->validate([
            'id_pengurus' => 'required',
            'id_peran' => 'required',
        ]);
    }
}

```

```

        Detail_Peran::create($validatedData);

        $request->session()->flash('successDetailPeran','Data Peran
Berhasil Ditambahkan!');

        return redirect('/pengurus-table');
    }

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

    /**
     * Show the form for editing the specified resource.
     *
     * @param \App\Models\Detail_Peran $detail_Peran
     * @return \Illuminate\Http\Response
     */
    public function edit(Detail_Peran $detail_Peran, $id)
    {
        return view('dashboard.edit.detail-peran', [
            // 'bab' => Bab::find($id),
            // 'bukus' => Buku::all(),
            // "title" => Bab::find($id)->bab

            'pengurus' => Pengurus::find($id),
            "title" => Pengurus::find($id),
            'peran' => Peran::find($id)
        ]);
    }

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

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

```

```
}  
}
```

## 9. Peran

### PeranController.php

```
<?php  
  
namespace App\Http\Controllers;  
  
use App\Models\Detail_Peran;  
use App\Models\Pengurus;  
use App\Models\Peran;  
use Illuminate\Http\Request;  
use Illuminate\Support\Facades\DB;  
  
class PeranController extends Controller  
{  
    /**  
     * Display a listing of the resource.  
     *  
     * @return \Illuminate\Http\Response  
     */  
    public function index(Peran $peran)  
    {  
        return view('dashboard.peran-table', [  
            'peran' => Peran::all(),  
            "title" => "Peran",  
            'counter' => 1  
        ]);  
    }  
  
    /**  
     * Show the form for creating a new resource.  
     *  
     * @return \Illuminate\Http\Response  
     */  
    public function create()  
    {  
        return view('dashboard.create.peran', [  
            "title" => "Peran"  
        ]);  
    }  
  
    /**  
     * Store a newly created resource in storage.  
     *  
     * @param \Illuminate\Http\Request $request  
     * @return \Illuminate\Http\Response  
     */  
    public function store(Request $request)  
    {  
        $validatedData = $request->validate([  
            'peran' => 'required|max:20|unique:perans',  
        ]);  
  
        Peran::create($validatedData);  
    }  
}
```

```

        $request->session()->flash('success','Peran Berhasil
Ditambahkan!');

        return redirect('/peran-table');
    }

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

    /**
     * Show the form for editing the specified resource.
     *
     * @param \App\Models\Peran $peran
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        return view('dashboard.edit.peran', [
            'peran'      => Peran::find($id),
            "title"      => Peran::find($id)->peran
        ]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @param \App\Models\Peran $peran
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, Peran $peran)
    {
        DB::table('perans')->where('id',$request->id)->update([
            'peran'      => $request->peran,
        ]);

        return redirect('/peran-table')->with('update','Data Peran
Berhasil Di-Update!');
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param \App\Models\Peran $peran
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        Peran::find($id)->delete();

        return redirect('/peran-table')->with('delete','Peran Berhasil
Dihapus!');
    }

```

```
}
```

## 10. Logout

### LoginController.php

```
<?php

namespace App\Http\Controllers;

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

class LoginController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('login', [
            "title" => "Login"
        ]);
    }

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

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

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

        return back()->with('loginError', 'Login Gagah!');
    }

    public function logout(Request $request)
    {
        Auth::logout();

        $request->session()->invalidate();

        $request->session()->regenerateToken();

        return redirect('/');
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
}
```

```

public function create()
{
    //

}

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

}

/**
 * Display the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    //

}

/**
 * Show the form for editing the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    //

}

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

}

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

}
}

```

