

CRUD Model dan Resource Controller

Membuat Flash Data

Tutorial selanjutnya adalah membuat CRUD dari model Buku. Namun sebelumnya kita perlu menambahkan fitur flash data. Flash data digunakan untuk menampilkan pesan ketika pengguna melakukan suatu aksi. Contohnya hapus data, tambah data, dan lain-lain. Flash data dapat diimplementasikan menggunakan alert pada bootstrap 4.

Berhasil! Data buku telah disimpan.



Buatlah file baru bernama flash.data.php. Kemudian salin kode yang ada di bawah ini ke dalam file tersebut. File ini akan di injeksi ke dalam template utama. Pada kode tampak terdapat percabangan if. Hal ini berfungsi apabila terdapat flashdata maka layout ini akan menampilkan alert box.

Nama file	resources\views\layouts\flash.data.php
Deskripsi	Layout flash data
<pre>@if (session('status')) <div class="alert alert-primary alert-dismissible fade show" role="alert"> <i class="fa fa-info-circle fa-fw"></i> {{ __(session('status')) }} <button type="button" class="close" data-dismiss="alert" aria- label="Close"> &times; </button> </div> @endif @if (session('error')) <div class="alert alert-danger alert-dismissible fade show" role="alert"> <i class="fa fa-exclamation-triangle fa-fw"></i> {{ __(session('error')) }} <button type="button" class="close" data-dismiss="alert" aria- label="Close"> &times; </button> </div> @endif @if (session('success')) <div class="alert alert-success alert-dismissible fade show" role="alert"> <i class="fa fa-check-circle fa-fw"></i> {{ __(session('success')) }}</pre>	

```

<button type="button" class="close" data-dismiss="alert" aria-label="Close">
  <span aria-hidden="true">&times;</span>
</button>
</div>
@endif

```

 Contoh Flash Data Status



 Contoh Flash Data Success



 Contoh Flash Data Error



Kemudian buka file app.blade.php dan sisipkan kode `@include('layouts.flash')` pada file tersebut sebelum `yield content`. Contoh penggunaan dari flash data akan dicontohkan pada implementasi CRUD nantinya.

Nama file	resources\views\layouts\app.blade.php
Deskripsi	Layout scaffold
...	<pre> <h1 class="h3 mb-4 text-gray-800">@yield('title', 'Halaman Kosong')</h1> @include('layouts.flash') @yield('content') ... </pre>

Mengatur Pagination

Untuk menggunakan pagination versi bootstrap css, buka file `app\Providers\AppServiceProvider.php` kemudian pada kelas `AppServiceProvider` tambahkan kode `Paginator::useBootstrap()`; di dalam method **boot** dan tambahkan sintaks `use Illuminate\Pagination\Paginator;` pada file tersebut.

Selanjutnya kita perlu untuk menambahkan action button pada template. Buka file `app.blade.php`. Kemudian cari kode yang ada di baris pertama dan ganti dengan kode yang ada pada baris kedua.

Nama file	resources\views\layouts\app.blade.php
Deskripsi	Layout scaffold
Cari	<pre> <h1 class="h3 mb-4 text-gray-800">@yield('title', 'Halaman Kosong')</h1> </pre>
Ganti	<pre> <div class="d-sm-flex align-items-center justify-content-between mb-4"> <h1 class="h3 mb-0 text-gray-800">@yield('title', 'Halaman Kosong')</h1> <div> @yield('actions') </div> </div> </pre>

	<pre></div> </div></pre>
--	--------------------------------------

Membuat CRUD Buku

Buatlah resource controller dari model Buku dengan menjalankan perintah Artisan berikut. Opsi `--resource` menandakan bahwa controller yang dibuat adalah resource controller. Opsi `--buku` digunakan untuk mendefinisikan model yang dituju.

```
php artisan make:controller BukuController --resource --model=Buku
```

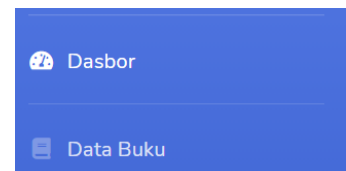
Resource controller adalah salah satu jenis controller yang disediakan oleh Laravel. Pada resource controller masing-masing method dari fungsi CRUD (Create, Read, Update, Delete) sudah disediakan oleh controller. Pemetaan route dari resource controller juga lebih mudah. Sehingga kita dapat fokus ke pengembangan dari CRUD ini sendiri. Namun kamu juga bisa membuat controller secara manual.

Buka `routes/web.php` dan tambahkan resource route. Untuk mendefinisikan resource route sangatlah mudah dengan menggunakan sintaks `Route::resource()`.

Nama file	<code>routes/web.php</code>
Deskripsi	Routes laravel
<pre>... Route::middleware('auth')->group(function() { ... Route::resource('buku', 'BukuController'); }); ...</pre>	

Buka file `sidebar.blade.php` dan tambahkan kode html untuk sidebar data buku.

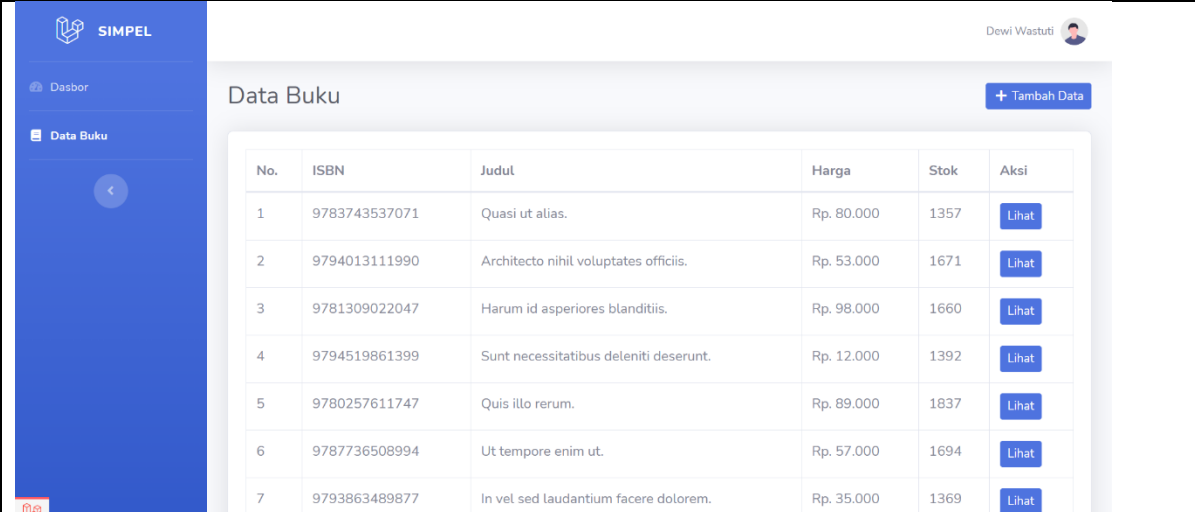
Nama file	<code>resources/views/layouts/sidebar.blade.php</code>
Deskripsi	Sidebar
<pre>... <hr class="sidebar-divider my-0"> <li class="nav-item @if(Route::is('buku.*')) active @endif"> <i class="fas fa-fw fa-book"></i> Data Buku ...</pre>	



1. Daftar Buku

Buka file `BukuController` yang ada di dalam folder `app\Http\Controllers`. Kemudian edit method `index` yang sudah ada menjadi kode seperti di bawah ini.

Nama file	<code>app\Http\Controllers/BukuController.php</code>
-----------	--

Deskripsi	Controller untuk buku
<pre> ... public function index() { return view('buku.index', ['data' => Buku::query() ->latest() ->paginate(10)]); } ... </pre>	
	

Fungsi di atas akan mengambil data dari model buku. Method latest digunakan untuk mengurutkan data dari terbaru ke terlama. Paginate digunakan untuk melakukan pagination atau pembatasan jumlah data per halaman untuk menghemat waktu loading. Buatlah file view index.blade.php ke dalam folder buku.

Nama file	resources\views\buku\index.blade.php
Deskripsi	View untuk indeks buku
<pre> @extends('layouts.app') @section('title', 'Data Buku') @section('actions') <i class="fa fa-plus fa-fw"></i> Tambah Data @endsection @section('content') <div class="card shadow mb-4"> <div class="card-body"> <div class="table-responsive"> <table class="table table-bordered"> <thead> <tr> <th>No.</th> <th>ISBN</th> <th>Judul</th> <th>Harga</th> <th>Stok</th> </pre>	

```

        <th>Aksi</th>
    </tr>
</thead>
<tbody>
    @forelse($data as $buku)
        <tr>
            <td>{{ $data->firstItem() + $loop->index }}</td>
            <td>{{ $buku->isbn }}</td>
            <td>{{ $buku->judul }}</td>
            <td>{{ $buku->harga_rupiah }}</td>
            <td>{{ $buku->stok }}</td>
            <td>
                <a href="{{ route('buku.show', $buku) }}" class="btn btn-
primary btn-sm">Lihat</a>
            </td>
        </tr>
    @empty
        <tr>
            <td colspan="6" class="text-center">Data Kosong</td>
        </tr>
    @endforelse
</tbody>
</table>
</div>

    {{ $data->links() }}
</div>
</div>
@endsection

```

No.	ISBN	Judul	Harga
1	9793727632012	Dignissimos sint possimus ad et.	53000

Pada halaman data buku di kolom harga belum memiliki format rupiah. Sehingga harus dikonversi dari 53000 ke Rp. 53.000. Untuk melakukan ini kita perlu mendefinisikan accessor pada model Buku. Accessor digunakan untuk mentransformasi atribut model ketika diakses. Semisal kita ingin membuat accessor harga_rupiah, maka kita membutuhkan accessor getHargaRupiahAttribute. Buka file model Buku dan tambahkan method di bawah ini.

Nama file	app/Models/Buku.php								
Deskripsi	Model untuk buku								
<pre>... public function getHargaRupiahAttribute(){ return 'Rp. ' . str_replace(',', '.', number_format(\$this->harga)); } ...</pre>									
<table><tr><th>No.</th><th>ISBN</th><th>Judul</th><th>Harga</th></tr><tr><td>1</td><td>9793727632012</td><td>Dignissimos sint possimus ad et.</td><td>Rp. 53.000</td></tr></table>		No.	ISBN	Judul	Harga	1	9793727632012	Dignissimos sint possimus ad et.	Rp. 53.000
No.	ISBN	Judul	Harga						
1	9793727632012	Dignissimos sint possimus ad et.	Rp. 53.000						

Kemudian ganti `<td>{{ $buku->harga }}</td>` dengan kode `<td>{{ $buku->harga_rupiah }}</td>`.

2. Lihat Buku

Buka file BukuController yang terdapat di dalam folder app\Http\Controllers. Edit method show menjadi kode seperti di bawah ini. Method compact digunakan untuk mendapatkan array dari argumen yang dikirimkan. Pada dasarnya sama dengan kode ['buku' => \$buku].

Nama file	app/Http/Controllers/BukuController.php
Deskripsi	Controller untuk buku
<pre>... public function show(Buku \$buku){ return view('buku.view', compact('buku')); } ...</pre>	

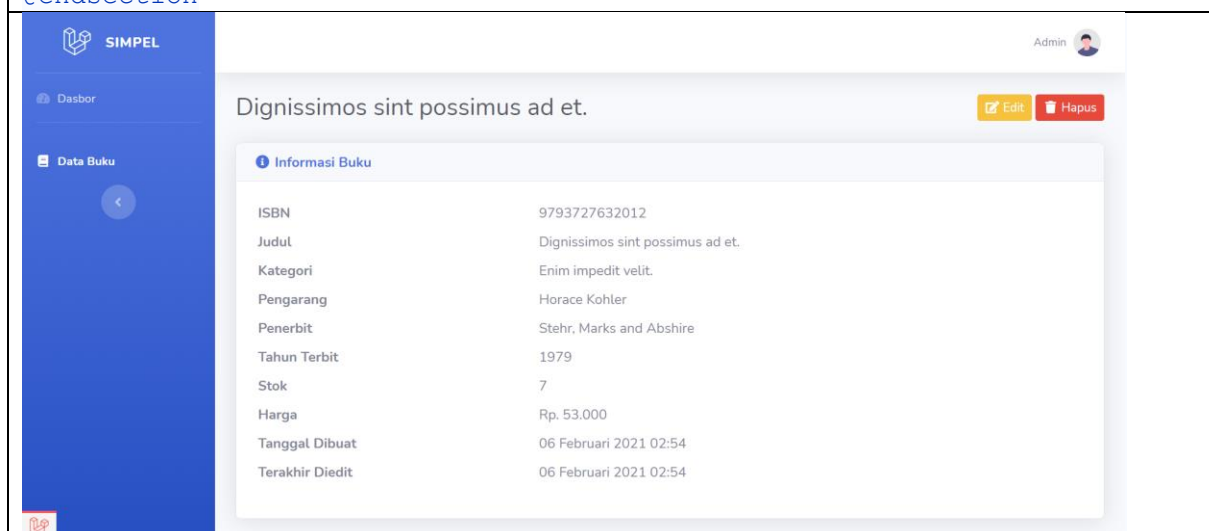
Buatlah file view view.blade.php ke dalam folder buku.

Nama file	resources/views/buku/view.blade.php
Deskripsi	View untuk lihat info buku
<pre>@extends('layouts.app') @section('title', \$buku->judul) @section('actions') <i class="fa fa-edit fa-fw"></i> Edit <form action="{{ route('buku.destroy', \$buku) }}" onsubmit="return confirm('Apakah anda yakin?')" method="POST" class="d-inline-block"> @csrf @method('DELETE') <button type="submit" class="btn btn-danger btn-sm"> <i class="fa fa-trash fa-fw"></i> Hapus </button> </form> @endsection @section('content') <div class="card shadow mb-4"> <div class="card-header"> <h6 class="m-0 font-weight-bold text-primary"> <i class="fa fa-info-circle fa-fw"></i> Informasi Buku </h6> </div> <div class="card-body"> <table class="table table-borderless table-sm"> <tr> <th>ISBN</th> <td>{{ \$buku->isbn }}</td> </tr> <tr> <th>Judul</th> <td>{{ \$buku->judul }}</td> </tr> </table> </div> </div> </pre>	

```

<tr>
    <th>Kategori</th>
    <td>{{ optional($buku->kategori)->nama }}</td>
</tr>
<tr>
    <th>Pengarang</th>
    <td>{{ $buku->pengarang }}</td>
</tr>
<tr>
    <th>Penerbit</th>
    <td>{{ $buku->penerbit }}</td>
</tr>
<tr>
    <th>Tahun Terbit</th>
    <td>{{ $buku->tahun }}</td>
</tr>
<tr>
    <th>Stok</th>
    <td>{{ $buku->stok }}</td>
</tr>
<tr>
    <th>Harga</th>
    <td>{{ $buku->harga_rupiah }}</td>
</tr>
<tr>
    <th>Tanggal Dibuat</th>
    <td>{{ $buku->created_at->translatedFormat('d F Y h:i') }}</td>
</tr>
<tr>
    <th>Terakhir Diedit</th>
    <td>{{ $buku->updated_at->translatedFormat('d F Y h:i') }}</td>
</tr>
</table>
</div>
</div>
@endsection

```



3. Tambah Buku

Buka file BukuController yang terdapat di dalam folder app\Http\Controllers. Edit method create menjadi kode seperti di bawah ini.

Nama file	app/Http/Controllers/BukuController.php
-----------	---

Deskripsi	Controller untuk buku
<pre> ... public function create(){ return view('buku.create', ['data_kategori' => Kategori::all()]); } ... </pre>	

Jangan lupa untuk menambahkan sintaks `use App\Models\Kategori;` di bagian atas kode. Kemudian buatlah file `create.blade.php` ke dalam folder `buku`.

Nama file	resources/views/buku/create.blade.php
Deskripsi	File untuk halaman tambah buku
<pre> @extends('layouts.app') @section('title', 'Tambah Buku') @section('content') <div class="card shadow mb-4"> <div class="card-header"> <h6 class="m-0 font-weight-bold text-primary"> <i class="fa fa-clipboard-list fa-fw"></i> Formulir </h6> </div> <div class="card-body"> <form action="{{ route('buku.store') }}" method="POST"> @csrf <div class="form-group"> <label for="kategori_id">Kategori <x-required/></label> <select class="form-control @error('kategori_id') is-invalid @enderror" name="kategori_id" id="kategori_id"> <option selected disabled hidden>-- Pilih Kategori -- </option> @foreach(\$data_kategori as \$kategori) <option value="{{ \$kategori->id }}" @if(old('kategori_id') == \$kategori->id) selected @endif>{{ \$kategori->nama }}</option> @endforeach </select> @error('kategori_id') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <div class="form-group"> <label for="isbn">ISBN <x-required/></label> <input type="text" class="form-control @error('isbn') is- invalid @enderror" name="isbn" id="isbn" value="{{ old('isbn') }}" placeholder="Kode ISBN" required> @error('isbn') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <div class="form-group"> <label for="judul">Judul <x-required/></label> <input type="text" class="form-control @error('judul') is- invalid @enderror" name="judul" id="judul" value="{{ old('judul') }}" placeholder="Judul buku" required> @error('judul') </pre>	


```

        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="pengarang">Pengarang <x-required/></label>
        <input type="text" class="form-control @error('pengarang') is-
invalid @enderror" name="pengarang" id="pengarang" value="{{
old('pengarang') }}" placeholder="Nama pengarang" required>
        @error('pengarang')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="penerbit">Penerbit <x-required/></label>
        <input type="text" class="form-control @error('penerbit') is-
invalid @enderror" name="penerbit" id="penerbit" value="{{
old('penerbit') }}" placeholder="Nama penerbit" required>
        @error('penerbit')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="tahun">Tahun Terbit <x-required/></label>
        <input type="number" class="form-control @error('tahun') is-
invalid @enderror" name="tahun" id="tahun" value="{{ old('tahun') }}"
placeholder="Tahun terbit" required>
        @error('tahun')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="stok">Stok <x-required/></label>
        <input type="number" class="form-control @error('stok') is-
invalid @enderror" name="stok" id="stok" value="{{ old('stok') }}"
placeholder="Stok saat ini" required>
        @error('stok')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="harga">Harga <x-required/></label>
        <input type="number" class="form-control @error('harga') is-
invalid @enderror" name="harga" id="harga" value="{{ old('harga') }}"
placeholder="Harga jual" required>
        @error('harga')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <button type="submit" class="btn btn-primary">
        <i class="fa fa-save fa-fw"></i>
        <span>Simpan</span>
    </button>
</form>
</div>
</div>
@endsection

```

Kemudian edit method store pada BukuController. Method store akan digunakan ketika formulir tambah buku disubmit. Pada method ini pula validasi data dilakukan. Validasi data dapat dilakukan dengan menggunakan method validate pada obyek request. Apabila gagal, validasi akan secara otomatis kembali ke formulir dan mencetak error melalui fungsi @error pada blade. Fungsi old() pada blade digunakan untuk mencetak data yang gagal divalidasi sehingga pengguna bisa memperbaiki isi formulir. Salin kode berikut pada method store.

Nama file	app/Http/Controllers/BukuController.php
Deskripsi	Controller untuk buku
<pre> ... public function store(Request \$request) { \$request->validate(['kategori_id' => ['required', Rule::exists(Kategori::class, 'id')], 'isbn' => ['required', 'string', 'max:255', Rule::unique(Buku::class)], 'judul' => 'required string max:255', 'pengarang' => 'required string max:255', 'penerbit' => 'required string max:255', 'tahun' => 'required date_format:Y', 'stok' => 'required numeric min:0', 'harga' => 'required numeric min:0',]); \$buku = new Buku; \$buku->kategori_id = \$request->kategori_id; \$buku->isbn = \$request->isbn; \$buku->judul = \$request->judul; \$buku->pengarang = \$request->pengarang; \$buku->penerbit = \$request->penerbit; \$buku->tahun = \$request->tahun; \$buku->stok = \$request->stok; \$buku->harga = \$request->harga; \$buku->save(); return redirect() ->route('buku.show', \$buku) ->with('success', 'Data berhasil ditambahkan.');</pre>	
...	

Tambahkan sintaks `use Illuminate\Validation\Rule;` pada bagian atas. Pada method store dapat dibagi menjadi 3 bagian, yaitu: validasi, penyimpanan, dan redirect.

Rule::exists digunakan untuk memvalidasi bahwa kategori yang dipilih benar-benar ada di dalam tabel kategori. Rule::unique digunakan untuk memvalidasi bahwa data yang diinputkan unik dan belum pernah ada pada tabel buku. Proses penyimpanan dengan model adalah dengan menganalogikan tuple data sebagai obyek dan memanggil method save() untuk menyimpan obyek ke dalam basis data. Fungsi redirect setelah menyimpan adalah menuju informasi buku. Kemudian juga menyertakan flash data success yang akan muncul.

Data berhasil ditambahkan.



4. Edit Buku

Buka file BukuController yang terdapat di dalam folder app\Http\Controllers. Edit method edit menjadi kode seperti di bawah ini.

Nama file	app/Http/Controllers/BukuController.php
Deskripsi	Controller untuk buku
<pre>... public function edit(Buku \$buku) { return view('buku.edit', ['buku' => \$buku, 'data_kategori' => Kategori::all(),]); } ...</pre>	

Kemudian buatlah edit.blade.php ke dalam folder buku dan salin kode berikut.

Nama file	resources/views/buku/edit.blade.php
Deskripsi	View untuk halaman edit buku
<pre>@extends('layouts.app') @section('title', 'Edit Buku') @section('content') <div class="card shadow mb-4"> <div class="card-header"> <h6 class="m-0 font-weight-bold text-primary"> <i class="fa fa-clipboard-list fa-fw"></i> Formulir </h6> </div> <div class="card-body"> <form action="{{ route('buku.update', \$buku) }}" method="POST"> @csrf @method('PATCH') <div class="form-group"> <label for="kategori_id">Kategori <x-required/></label> <select class="form-control @error('kategori_id') is-invalid" @enderror" name="kategori_id" id="kategori_id"> <option selected disabled hidden>-- Pilih Kategori -- </option> @foreach(\$data_kategori as \$kategori) <option value="{{ \$kategori->id }}" @if(old('kategori_id', \$buku->kategori->id) == \$kategori->id) selected @endif>{{ \$kategori->nama</pre>	

```

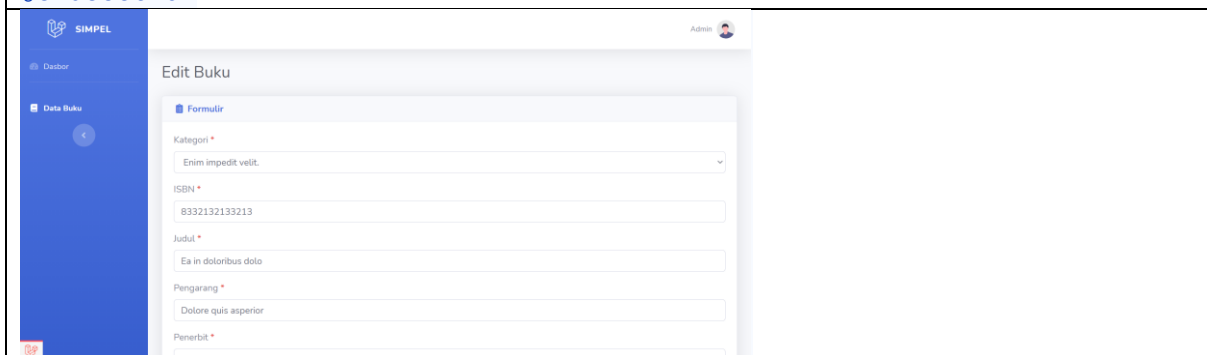
}}</option>
    @endforeach
</select>
    @error('kategori_id')
    <div class="invalid-feedback">{{ $message }}</div>
    @enderror
</div>
    <div class="form-group">
        <label for="isbn">ISBN <x-required/></label>
        <input type="number" class="form-control @error('isbn') is-
invalid @enderror" name="isbn" id="isbn" value="{{ old('isbn', $buku-
>isbn) }}" placeholder="Kode ISBN" required>
        @error('isbn')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="judul">Judul <x-required/></label>
        <input type="text" class="form-control @error('judul') is-
invalid @enderror" name="judul" id="judul" value="{{ old('judul', $buku-
>judul) }}" placeholder="Judul buku" required>
        @error('judul')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="pengarang">Pengarang <x-required/></label>
        <input type="text" class="form-control @error('pengarang') is-
invalid @enderror" name="pengarang" id="pengarang" value="{{
old('pengarang', $buku->pengarang) }}" placeholder="Nama pengarang"
required>
        @error('pengarang')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="penerbit">Penerbit <x-required/></label>
        <input type="text" class="form-control @error('penerbit') is-
invalid @enderror" name="penerbit" id="penerbit" value="{{
old('penerbit', $buku->penerbit) }}" placeholder="Nama penerbit"
required>
        @error('penerbit')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="tahun">Tahun Terbit <x-required/></label>
        <input type="number" class="form-control @error('tahun') is-
invalid @enderror" name="tahun" id="tahun" value="{{ old('tahun', $buku-
>tahun) }}" placeholder="Tahun terbit" required>
        @error('tahun')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <div class="form-group">
        <label for="stok">Stok <x-required/></label>
        <input type="number" class="form-control @error('stok') is-
invalid @enderror" name="stok" id="stok" value="{{ old('stok', $buku-
>stok) }}" placeholder="Stok saat ini" required>
        @error('stok')
        <div class="invalid-feedback">{{ $message }}</div>
    </div>

```

```

        @enderror
    </div>
    <div class="form-group">
        <label for="harga">Harga <x-required/></label>
        <input type="number" class="form-control @error('harga') is-
invalid @enderror" name="harga" id="harga" value="{{ old('harga', $buku-
>harga) }}" placeholder="Harga jual" required>
        @error('harga')
        <div class="invalid-feedback">{{ $message }}</div>
        @enderror
    </div>
    <button type="submit" class="btn btn-primary">
        <i class="fa fa-save fa-fw"></i>
        <span>Simpan</span>
    </button>
</form>
</div>
</div>
@endsection

```



Kode create.blade.php dan edit.blade.php hampir mirip namun berbeda. Pada tampilan edit, data yang sudah ada diisikan ke dalam formulir terlebih dahulu. Kemudian memiliki route action yang berbeda. Edit menggunakan method update untuk action dan menggunakan header PATCH.

Edit method update pada BukuController seperti dibawah ini.

Nama file	app/Http/Controllers/BukuController.php
Deskripsi	Controller untuk buku
<pre> ... public function update(Request \$request, Buku \$buku){ \$request->validate(['kategori_id' => ['required', Rule::exists(Kategori::class, 'id')], 'isbn' => ['required', 'string', 'max:255', Rule::unique(Buku::class)->ignore(\$buku)], 'judul' => 'required string max:255', 'pengarang' => 'required string max:255', 'penerbit' => 'required string max:255', 'tahun' => 'required date_format:Y', 'stok' => 'required numeric min:0', 'harga' => 'required numeric min:0',]); \$buku->kategori_id = \$request->kategori_id; \$buku->isbn = \$request->isbn; \$buku->judul = \$request->judul; </pre>	

```

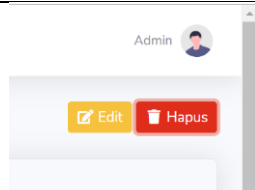
    $buku->pengarang = $request->pengarang;
    $buku->penerbit = $request->penerbit;
    $buku->tahun = $request->tahun;
    $buku->stok = $request->stok;
    $buku->harga = $request->harga;
    $buku->update();

    return redirect()
        ->route('buku.show', $buku)
        ->with('success', 'Data berhasil diperbarui.');
```

Pada Rule::unique terdapat ignore untuk mengabaikan apabila nilai sama dengan data awalnya. Kemudian tidak terdapat inisiasi model baru melainkan mengambil dari model yang sudah ada, memperbarui semua atributnya, dan memanggil update() untuk menyimpan perubahan ke dalam basis data.

5. Hapus Buku

Tombol hapus pada dasarnya sudah ada namun aksi penghapusan belum dijelaskan. Pada method destroy hanya akan ada proses hapus dan redirect. Namun pada penghapusan data buku, data terkait buku seperti penjualan dan penghapusan juga ingin dihapus. Proses ini akan dijelaskan nanti. Sekarang edit method destroy pada BukuController.

Nama file	app/Http/Controllers/BukuController.php
Deskripsi	Controller untuk buku
<pre> ... public function destroy(Buku \$buku) { \$buku->delete(); return redirect() ->route('buku.index') ->with('success', 'Data berhasil dihapus.');</pre>	
	

6. Observer Buku

Observer merupakan fitur dari laravel yang cara kerjanya mirip dengan fungsi trigger pada basis data. Hanya saja fitur observer berjalan pada model eloquent. Fungsi observer adalah mendengarkan event yang terjadi pada sebuah model dan melakukan aksi sesuai yang didefinisikan. BukuObserver bisa digenerate oleh perintah Artisan di bawah ini.

```
php artisan make:observer BukuObserver --model=Buku
```

Observer akan disimpan ke dalam folder app\Observers. Buka file BukuObserver dan edit method deleted. Method ini akan dipanggil setelah model buku dihapus.

Nama file	app/Observers/BukuObserver.php
Deskripsi	Observer untuk model buku
<pre>... public function deleted(Buku \$buku) { foreach(\$buku->pembelian as \$pembelian) { \$pembelian->delete(); } foreach(\$buku->penjualan as \$penjualan) { \$penjualan->delete(); } } ...</pre>	

Untuk menggunakan observer, kamu harus mendaftarkan observer ke dalam app\Providers\EventServiceProvider.php dengan memanggil observer pada model dan mengirimkan kelas observer ke dalam argumen. Contoh kode berikut di bawah ini.

Nama file	app\Providers\EventServiceProvider.php
Deskripsi	Service provider laravel
<pre>use App\Models\Buku; use App\Observers\BukuObserver; ... public function boot() { Buku::observe(BukuObserver::class); } ...</pre>	

Membuat CRUD Kategori

Buatlah resource controller bernama KategoriController menggunakan perintah artisan di bawah ini.

```
php artisan make:controller KategoriController --resource --model=Kategori
```

Buka routes\web.app dan tambahkan route resource di dalam group. Pada routes ini tidak diperlukan method show karena data kategori sudah ditampilkan lengkap di index.

Nama file	routes\web.app
Deskripsi	Routes laravel
<pre>... Route::middleware('auth')->group(function() { Route::resource('kategori', 'KategoriController')->except('show'); }); ...</pre>	

Buka file sidebar.blade.php dan tambahkan kode html untuk menambahkan shortcut sidebar data kategori.

Nama file	resource\views\layouts\sidebar.blade.php
Deskripsi	Sidebar
<pre>... <li class="nav-item @if(Route::is('kategori.*')) active @endif"> <i class="fas fa-fw fa-tags"></i> Data Kategori ...</pre>	

1. Data Kategori

Buka KategoriController dan edit method index seperti di bawah ini. Pada contoh ini withCount digunakan untuk menyertakan jumlah buku dengan kategori tersebut. Metode ini dikenal dengan eager-loading yang sangat bermanfaat dalam mempercepat pemuatan model dari basis data.

Nama file	app\Http\Controllers/KategoriController.php
Deskripsi	Controller untuk kategori
<pre>... public function index() { return view('kategori.index', ['data' => Kategori::query() ->withCount('buku') ->orderBy('nama') ->get()]); } ...</pre>	


Pada data kategori kita tidak menggunakan pagination karena data diperkirakan tidak akan terlalu banyak nantinya. Untuk membuat view, silahkan file view index.blade.php ke dalam folder kategori. Kemudian salin kode berikut ini.


Nama file	resources\views\kategori\index.blade.php
Deskripsi	View untuk indeks kategori
<pre>@extends('layouts.app') @section('title', 'Data Kategori') @section('actions') <i class="fa fa-plus fa-fw"></i> Tambah Data @endsection @section('content') <div class="card shadow mb-4"> <div class="card-body"> <div class="table-responsive"></pre>	


```

<table class="table table-bordered">
  <thead>
    <tr>
      <th>No.</th>
      <th>Nama</th>
      <th>Jumlah Buku</th>
      <th>Aksi</th>
    </tr>
  </thead>
  <tbody>
    @forelse($data as $kategori)
      <tr>
        <td>{{ $loop->iteration }}</td>
        <td>{{ $kategori->nama }}</td>
        <td>{{ $kategori->buku_count }} Buku</td>
        <td>
          <a href="{{ route('kategori.edit', $kategori) }}"
class="btn btn-warning btn-sm">
            <i class="fa fa-edit fa-fw"></i>
            <span>Edit</span>
          </a>
          <form action="{{ route('kategori.destroy', $kategori) }}"
onsubmit="return confirm('Apakah anda yakin?')" method="POST" class="d-
inline-block">
            @csrf
            @method('DELETE')
            <button type="submit" class="btn btn-danger btn-sm">
              <i class="fa fa-trash fa-fw"></i>
              <span>Hapus</span>
            </button>
          </form>
        </td>
      </tr>
    @empty
      <tr>
        <td colspan="4" class="text-center">Tidak ada data</td>
      </tr>
    @endforelse
  </tbody>
</table>
</div>
</div>
</div>
@endsection







```


SIMPEL

Admin 

Dashboard
 Data Buku
 Data Kategori

Data Kategori
 + Tambah Data

No.	Nama	Jumlah Buku	Aksi
1	Aut maiores deserunt.	0 Buku	 Edit  Hapus
2	Enim impedit velit.	8 Buku	 Edit  Hapus
3	Odit esse.	0 Buku	 Edit  Hapus

Hak Cipta © SIMPEL 2021

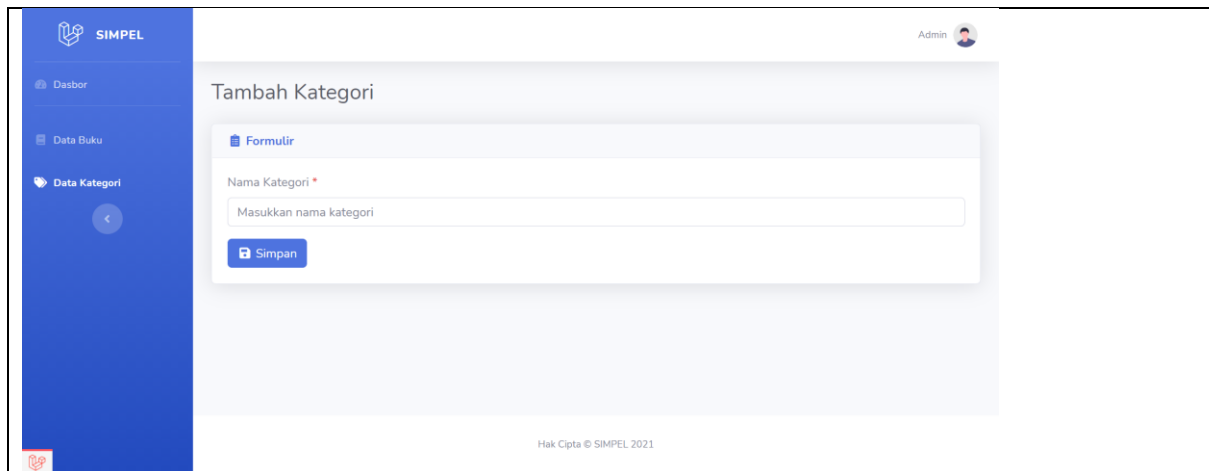
2. Tambah Kategori

Edit method create pada KategoriController sesuai kode di bawah ini.

Nama file	app/Http/Controllers/KategoriController.php
Deskripsi	Controller untuk kategori
<pre>... public function create() { return view('kategori.create'); } ...</pre>	

Kemudian buat file view create.blade.php ke dalam folder kategori dan salin kode di bawah ini. Formulir menu tambah kategori hanya memiliki 1 kolom saja.

Nama file	resources/views/kategori/create.blade.php
Deskripsi	View untuk halaman tambah kategori
<pre>@extends('layouts.app') @section('title', 'Tambah Kategori') @section('content') <div class="card shadow mb-4"> <div class="card-header"> <h6 class="m-0 font-weight-bold text-primary"> <i class="fa fa-clipboard-list fa-fw"></i> Formulir </h6> </div> <div class="card-body"> <form action="{{ route('kategori.store') }}" method="POST"> @csrf <div class="form-group"> <label for="nama">Nama Kategori <x-required/></label> <input type="text" class="form-control @error('nama') is- invalid @enderror" name="nama" id="nama" value="{{ old('nama') }}" placeholder="Masukkan nama kategori" required> @error('nama') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <button type="submit" class="btn btn-primary"> <i class="fa fa-save fa-fw"></i> Simpan </button> </form> </div> </div> @endsection</pre>	



Kemudian edit method store pada KategoriController menjadi kode seperti di bawah ini.

Nama file	app/Http/Controllers/KategoriController.php
Deskripsi	Controller untuk kategori
<pre> use Illuminate\Validation\Rule; ... public function store(Request \$request) { \$request->validate(['nama' => ['required', 'string', 'max:255', Rule::unique(Kategori::class)]]); \$kategori = new Kategori; \$kategori->nama = \$request->nama; \$kategori->save(); return redirect() ->route('kategori.index') ->with('success', 'Data berhasil ditambahkan.');</pre>	

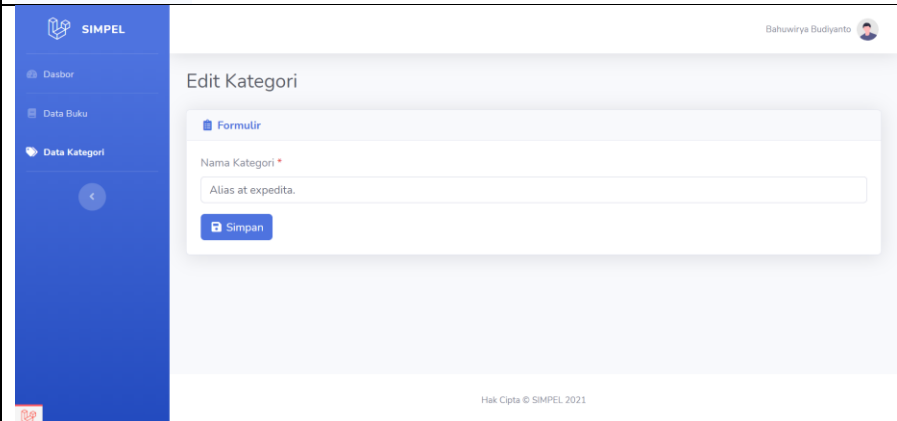
3. Edit Kategori

Edit method edit pada KategoriController sesuai kode di bawah ini.

Nama file	app/Http/Controllers/KategoriController.php
Deskripsi	Controller untuk kategori
<pre> ... public function edit(Kategori \$kategori) { return view('kategori.edit', compact('kategori')); } ... </pre>	

Kemudian buatlah file view edit.blade.php ke dalam folder kategori dan salin kode di bawah ini. Formulir menu edit kategori hanya memiliki 1 kolom saja.

Nama file	resources/views/kategori/edit.blade.php
-----------	---

Deskripsi	View untuk halaman edit kategori
	<pre> @extends('layouts.app') @section('title', 'Edit Kategori') @section('content') <div class="card shadow mb-4"> <div class="card-header"> <h6 class="m-0 font-weight-bold text-primary"> <i class="fa fa-clipboard-list fa-fw"></i> Formulir </h6> </div> <div class="card-body"> <form action="{{ route('kategori.update', \$kategori) }}" method="POST"> @csrf @method('PATCH') <div class="form-group"> <label for="nama">Nama Kategori <x-required/></label> <input type="text" class="form-control @error('nama') is- invalid @enderror" name="nama" id="nama" value="{{ old('nama', \$kategori- >nama) }}" placeholder="Masukkan nama kategori" required> @error('nama') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <button type="submit" class="btn btn-primary"> <i class="fa fa-save fa-fw"></i> Simpan </button> </form> </div> </div> @endsection </pre>
	

Kemudian edit method update pada KategoriController menjadi kode seperti di bawah ini.

Nama file	app/Http/Controllers/KategoriController.php
Deskripsi	Controller untuk kategori
	<pre> ... public function update(Request \$request, Kategori \$kategori) { \$request->validate(['nama' => ['required', 'string', </pre>

```

        'max:255',
        Rule::unique(Kategori::class)->ignore($kategori)
    ]
});

$kategori->nama = $request->nama;
$kategori->update();

return redirect()
    ->route('kategori.index')
    ->with('success', 'Data berhasil diperbarui.');
```

4. Hapus Kategori

Pada penghapusan kategori, hanya kategori yang memiliki 0 buku saja yang dapat dihapus. Oleh karena itu, pengecekan ini menggunakan percabangan. Edit method destroy pada KategoriController menjadi seperti kode di bawah ini.

Nama file	app/Http/Controllers/KategoriController.php
Deskripsi	Controller untuk kategori
<pre> ... public function destroy(Kategori \$kategori){ if(\$kategori->buku()->count() > 0){ return redirect() ->route('kategori.index') ->with('error', 'Data tidak dapat dihapus. Karena masih terdapat buku dengan kategori ini.');</pre>	
<pre> } else{ \$kategori->delete(); return redirect() ->route('kategori.index') ->with('success', 'Data berhasil dihapus.');</pre>	
<pre> } } ... </pre>	

Membuat CRUD Pembelian

Pembelian adalah suatu kegiatan yang dilakukan oleh sebuah toko dengan membeli barang untuk menambah stok. Buatlah controller resource PembelianController dengan menjalankan perintah Artisan berikut.

```
php artisan make:controller PembelianController --resource --model=Pembelian
```

Kemudian tambahkan routes untuk resource controller PembelianController pada file routes/web.php.

Nama file	routes/web.php
Deskripsi	Routes laravel
<pre> ... Route::middleware('auth')->group(function(){ </pre>	

```

....

Route::resource('pembelian', 'PembelianController')->only('index',
'create', 'store', 'destroy');
});
...

```

Tambahkan html sidebar untuk data pembelian di bawah html sidebar data kategori pada file sidebar.blade.php dengan kode sebagai berikut.

Nama file	resources/views/layouts/sidebar.blade.php
Deskripsi	Sidebar
<pre> ... <li class="nav-item @if(Route::is('pembelian.*')) active @endif"> <i class="fas fa-fw fa-warehouse"></i> Data Pembelian ... </pre>	

Buka file PembelianController.php kemudian edit method index yang sudah ada. Halaman data pembelian akan menampilkan nama buku, jumlah stok masuk, dan tanggal masuk. Pada sel nama buku akan terdapat shortcut yang menuju info barang.

Nama file	app/Http/Controllers/PembelianController.php
Deskripsi	Controller untuk pembelian
<pre> ... public function index() { return view('pembelian.index', ['data' => Pembelian::with('buku') ->latest() ->paginate(10)]); } ... </pre>	

Buatlah file index.blade.php ke dalam folder pembelian.


Nama file	resources/views/pembelian/index.blade.php
Deskripsi	View untuk indeks pembelian
<pre> @extends('layouts.app') @section('title', 'Data Pembelian') @section('actions') <i class="fas fa-plus fa-fw"></i> Tambah Data @endsection @section('content') <div class="card shadow mb-4"> <div class="card-body"> </pre>	

```

<div class="table-responsive">
  <table class="table table-bordered">
    <thead>
      <tr>
        <th>No.</th>
        <th>Tanggal</th>
        <th>Nama Buku</th>
        <th>Jumlah Masuk</th>
        <th>Aksi</th>
      </tr>
    </thead>
    <tbody>
      @forelse($data as $pembelian)
        <tr>
          <td>{{ $data->firstItem() + $loop->index }}</td>
          <td>{{ $pembelian->created_at->translatedFormat('d F Y h:i') }}</td>
          <td>
            @isset($pembelian->buku)
              <a href="{{ route('buku.show', $pembelian->buku) }}">{{ $pembelian->buku->judul }}</a>
            @endisset
          </td>
          <td>{{ $pembelian->stok }} Buku</td>
          <td>
            <form action="{{ route('pembelian.destroy', $pembelian) }}" onsubmit="return confirm('Apakah anda yakin?')" method="POST">
              @csrf
              @method('DELETE')
              <button type="submit" class="btn btn-danger btn-sm">
                <i class="fa fa-trash fa-fw"></i>
                <span>Hapus</span>
              </button>
            </form>
          </td>
        </tr>
      @empty
        <tr>
          <td colspan="6" class="text-center">Data Kosong</td>
        </tr>
      @endforelse
    </tbody>
  </table>
</div>

{{ $data->links() }}
</div>
@endsection

```


SIMPEL

Bahariya Budyanto

Dashboard
Data Buku
Data Kategori
Data Pembelian

Data Pembelian
+ Tambah Data

No.	Tanggal	Nama Buku	Jumlah Masuk	Aksi
1	08 Februari 2021 06:45	Molestias doloremque maxime est sint.	80 Buku	Hapus
2	08 Februari 2021 06:45	Molestias doloremque maxime est sint.	99 Buku	Hapus
3	08 Februari 2021 06:45	Molestias doloremque maxime est sint.	44 Buku	Hapus
4	08 Februari 2021 06:45	Molestias doloremque maxime est sint.	86 Buku	Hapus
5	08 Februari 2021 06:45	Minus recusandae iste ut.	87 Buku	Hapus
6	08 Februari 2021 06:45	Molestias doloremque maxime est sint.	39 Buku	Hapus
7	08 Februari 2021 06:45	Molestias doloremque maxime est sint.	88 Buku	Hapus

Kemudian untuk membuat fitur tambah data pembelian, buka PembelianController dan edit method create seperti di bawah ini.

Nama file	app/Http/Controllers/PembelianController.php
Deskripsi	Controller untuk pembelian
<pre> use App\Models\Buku; ... public function create() { return view('pembelian.create', ['data' => Buku::query() ->select('id', 'judul') ->orderBy('judul') ->get()]); } ... </pre>	

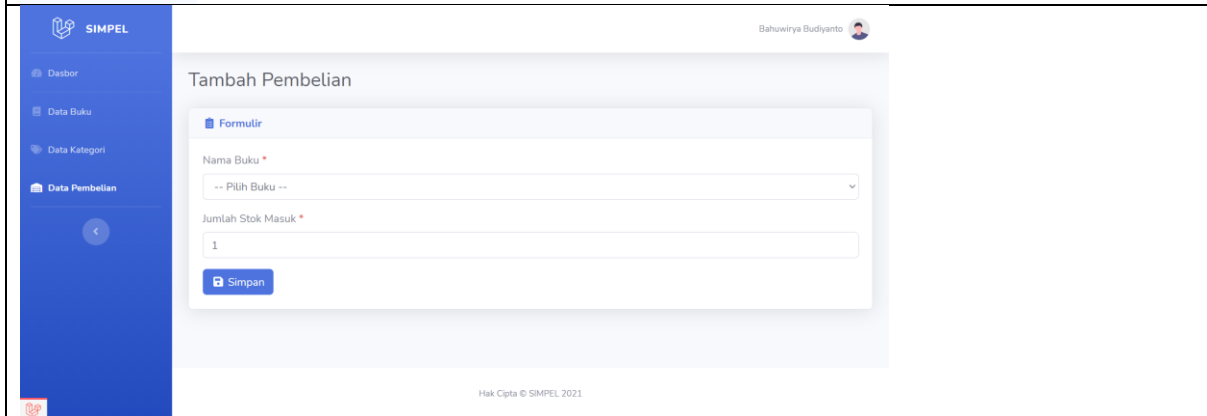
Buatlah file create.blade.php ke dalam folder pembelian.

Nama file	resources/views/pembelian/create.blade.php
Deskripsi	View untuk tambah pembelian
<pre> @extends('layouts.app') @section('title', 'Tambah Pembelian') @section('content') <div class="card shadow mb-4"> <div class="card-header"> <h6 class="m-0 font-weight-bold text-primary"> <i class="fa fa-clipboard-list fa-fw"></i> Formulir </h6> </div> <div class="card-body"> <form action="{{ route('pembelian.store') }}" method="POST"> @csrf <div class="form-group"> <label for="buku_id">Nama Buku <x-required/></label> <select class="form-control @error('buku_id') is-invalid @enderror" name="buku_id" id="buku_id"> <option selected disabled hidden>-- Pilih Buku --</option> @foreach(\$data as \$buku) <option value="{{ \$buku->id }}" @if(old('buku_id') == \$buku->id) selected @endif>{{ \$buku->judul }}</option> @endforeach </select> @error('buku_id') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <div class="form-group"> <label for="stok">Jumlah Stok Masuk <x-required/></label> <input type="number" class="form-control @error('stok') is- invalid @enderror" name="stok" id="stok" value="{{ old('stok', 1) }}" required> @error('stok') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> </form> </div> </div> </pre>	


```

<button type="submit" class="btn btn-primary">
  <i class="fa fa-save fa-fw"></i>
  <span>Simpan</span>
</button>
</form>
</div>
</div>
</div>
@endsection

```



Edit method store pada PembelianController untuk mendefinisikan action dari tambah data pembelian.

Nama file	app/Http/Controllers/PembelianController.php
Deskripsi	Controller untuk pembelian
<pre> use Illuminate\Validation\Rule; ... public function store(Request \$request) { \$request->validate(['buku_id' => ['required', Rule::exists(Buku::class, 'id')], 'stok' => 'required integer min:1']); \$pembelian = new Pembelian; \$pembelian->buku_id = \$request->buku_id; \$pembelian->stok = \$request->stok; \$pembelian->save(); return redirect() ->route('pembelian.index') ->with('success', 'Data berhasil ditambahkan.');</pre>	

Definisikan observer dari model Pembelian yaitu PembelianObserver dengan menjalankan perintah Artisan di bawah ini.

```
php artisan make:observer PembelianObserver --model=Pembelian
```

Kemudian daftarkan observer tersebut ke dalam EventServiceProvider seperti model buku sebelumnya.

Nama file	app/Providers/EventServiceProvider.php
Deskripsi	Service provider laravel
<pre> use App\Models\Pembelian; use App\Observers\PembelianObserver; ... public function boot() { ... Pembelian::observe(PembelianObserver::class); } </pre>	

Buka PembelianObserver dan edit method created. Pada fungsi ini kita menambahkan stok buku data pembelian ditambahkan.

Nama file	app/Observers/PembelianObserver.php
Deskripsi	Observer untuk model pembelian
<pre> ... public function created(Pembelian \$pembelian) { \$pembelian->buku->stok += \$pembelian->stok; \$pembelian->buku->update(); } ... </pre>	

Fitur yang perlu diimplementasikan selanjutnya adalah hapus. Ketika data pembelian terhapus, stok tidak akan berpengaruh. Edit method destroy menjadi seperti di bawah ini.

Nama file	app/Http/Controllers/PembelianController.php
Deskripsi	Controller untuk pembelian
<pre> ... public function destroy(Pembelian \$pembelian) { \$pembelian->delete(); return redirect() ->back() ->with('success', 'Data berhasil dihapus.');</pre>	

Redirect back adalah suatu fitur untuk redirect ke halaman sebelumnya. Hal ini berguna karena pada halaman data pembelian terdapat pagination. Sehingga posisi halaman tidak berubah ke posisi awal.

Membuat CRUD Penjualan

Penjualan adalah suatu aktivitas transaksi yang dilakukan oleh sebuah toko dengan pembeli dimana stok buku akan berkurang setelah transaksi tersebut selesai. Buatlah controller resource PenjualanController dengan menggunakan perintah Artisan.

```
php artisan make:controller PenjualanController --resource --model=Penjualan
```

Kemudian tambahkan routes untuk resource controller PenjualanController pada file routes/web.php.

Nama file	routes/web.php
Deskripsi	Routes laravel
<pre>... Route::middleware('auth')->group(function() { Route::resource('penjualan', 'PenjualanController')->only('index', 'create', 'store', 'destroy'); }); ...</pre>	

Tambahkan html sidebar untuk data penjualan di bawah html sidebar data kategori pada file sidebar.blade.php dengan kode sebagai berikut.

Nama file	resources/views/layouts/sidebar.blade.php
Deskripsi	Sidebar
<pre>... <li class="nav-item @if(Route::is('penjualan.*')) active @endif"> <i class="fas fa-fw fa-cash-register"></i> Data Penjualan ...</pre>	

Buka file PenjualanController.php kemudian edit method index seperti berikut. Halaman data penjualan akan menampilkan nama pembeli, nama buku, jumlah stok keluar, harga jual saat itu, dan tanggal masuk. Pada sel nama buku akan terdapat shortcut yang menuju info barang.

Nama file	app/Http/Controllers/PenjualanController.php
Deskripsi	Controller untuk penjualan
<pre>... public function index() { return view('penjualan.index', ['data' => Penjualan::with('buku') ->latest() ->paginate(10)]); } ...</pre>	

Buka model Penjualan dan tambahkan 3 accessor berikut.

Nama file	app/Models/Penjualan.php
Deskripsi	Model penjualan
<pre>... public function getHargaRupiahAttribute() { return 'Rp. ' . str_replace(',', '.', number_format(\$this->harga)); } </pre>	

```

public function getTotalAttribute() {
    return $this->stok * $this->harga;
}

public function getTotalRupiahAttribute() {
    return 'Rp. ' . str_replace(',', '.', number_format($this->total));
}
...

```

Kemudian buatlah file index.blade.php ke dalam folder penjualan.

Nama file	resources/views/penjualan/index.blade.php
Deskripsi	View untuk indeks penjualan
<pre> @extends('layouts.app') @section('title', 'Data Penjualan') @section('actions') <i class="fa fa-plus fa-fw"></i> Tambah Data @endsection @section('content') <div class="card shadow mb-4"> <div class="card-body"> <div class="table-responsive"> <table class="table table-bordered"> <thead> <tr> <th>No.</th> <th>Nama Pembeli</th> <th>Nama Buku</th> <th>Harga</th> <th>Jumlah</th> <th>Total</th> <th>Aksi</th> </tr> </thead> <tbody> @foreach(\$data as \$penjualan) <tr> <td>{{ \$data->firstItem() + \$loop->index }}</td> <td> {{ \$penjualan->nama }} {{ \$penjualan->created_at->translatedFormat('d F Y h:i') }} </td> <td> @isset(\$penjualan->buku) buku) }}">{{ \$penjualan->buku->judul }} @endisset </td> <td>{{ \$penjualan->harga_rupiah }}</td> <td>{{ \$penjualan->stok }}</td> </tr> @endforeach </tbody> </table> </div> </div> </div> </pre>	

```

<td>{{ $penjualan->total_rupiah }}</td>
<td>
    <form action="{{ route('penjualan.destroy', $penjualan) }}" onsubmit="return confirm('Apakah anda yakin?')" method="POST">
        @csrf
        @method('DELETE')
        <button type="submit" class="btn btn-danger btn-sm">
            <i class="fa fa-trash fa-fw"></i>
            <span>Hapus</span>
        </button>
    </form>
</td>
</tr>
@empty
<tr>
<td colspan="6" class="text-center">Data Kosong</td>
</tr>
@endforelse
</tbody>
</table>
</div>

{{ $data->links() }}
</div>
</div>
@endsection

```

No.	Nama Pembeli	Nama Buku	Harga	Jumlah	Total	Aksi
1	Syahrini Melinda Kuswandari 08 Februari 2021 12:38	Ut asperiores eum odit.	Rp. 62.000	3	Rp. 186.000	Hapus
2	Praba Joko Prayoga S.Pd 08 Februari 2021 12:38	Ut asperiores eum odit.	Rp. 6.000	61	Rp. 366.000	Hapus
3	Anastasia Suryatmi M.Ti. 08 Februari 2021 12:38	Ut asperiores eum odit.	Rp. 42.000	88	Rp. 3.696.000	Hapus
4	Margana Warsa Maulana 08 Februari 2021 12:38	Ut asperiores eum odit.	Rp. 12.000	84	Rp. 1.008.000	Hapus
5	Bakidin Waluyo Pradana S.Gz 08 Februari 2021 12:38	Ut asperiores eum odit.	Rp. 61.000	21	Rp. 1.281.000	Hapus

Kemudian untuk membuat fitur tambah data penjualan, buka PenjualanController dan edit method create seperti di bawah ini.

Nama file	app/Http/Controllers/PenjualanController.php
Deskripsi	Controller untuk penjualan
<pre> use App\Models\Buku; ... public function create() { return view('penjualan.create', ['data' => Buku::query() ->select('id', 'judul') ->orderBy('judul') ->get()]); } ... </pre>	

Buatlah file create.blade.php ke dalam folder penjualan.

Nama file	resources/views/penjualan/create.blade.php
Deskripsi	View untuk halaman tambah penjualan
<pre>@extends('layouts.app') @section('title', 'Tambah Penjualan') @section('content') <div class="card shadow mb-4"> <div class="card-header"> <h6 class="m-0 font-weight-bold text-primary"> <i class="fa fa-clipboard-list fa-fw"></i> Formulir </h6> </div> <div class="card-body"> <form action="{{ route('penjualan.store') }}" method="POST"> @csrf <div class="form-group"> <label for="nama">Nama Pembeli <x-required/></label> <input type="text" class="form-control @error('nama') is- invalid @enderror" name="nama" id="nama" value="{{ old('nama') }}" placeholder="Masukkan nama pembeli" required> @error('nama') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <div class="form-group"> <label for="buku_id">Nama Buku <x-required/></label> <select class="form-control @error('buku_id') is-invalid @enderror" name="buku_id" id="buku_id"> <option selected disabled hidden>-- Pilih Buku --</option> @foreach(\$data as \$buku) <option value="{{ \$buku->id }}" @if(old('buku_id') == \$buku->id) selected @endif>{{ \$buku->judul }} {{ \$buku->harga_rupiah }}</option> @endforeach </select> @error('buku_id') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <div class="form-group"> <label for="stok">Jumlah Barang <x-required/></label> <input type="number" class="form-control @error('stok') is- invalid @enderror" name="stok" id="stok" value="{{ old('stok', 1) }}" required> @error('stok') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <button type="submit" class="btn btn-primary"> <i class="fa fa-save fa-fw"></i> Simpan </button> </form> </div> </div> @endsection</pre>	

Edit method store pada PenjualanController untuk mendefinisikan action dari tambah data penjualan.

Nama file	app/Http/Controllers/PenjualanController.php
Deskripsi	Controller untuk penjualan
<pre> use Illuminate\Validation\Rule; ... public function store(Request \$request) { \$buku = Buku::query()->findOrFail(\$request->buku_id); \$request->validate(['buku_id' => ['required', Rule::exists(Buku::class, 'id')], 'nama' => 'required string max:255', 'stok' => 'required integer min:1 max:'.\$buku->stok]); \$penjualan = new Penjualan; \$penjualan->buku_id = \$request->buku_id; \$penjualan->nama = \$request->nama; \$penjualan->stok = \$request->stok; \$penjualan->harga = \$buku->harga; \$penjualan->save(); return redirect() ->route('penjualan.index') ->with('success', 'Data berhasil ditambahkan.');</pre>	

Definisikan observer dari model Penjualan yaitu PenjualanObserver dengan menjalankan perintah Artisan di bawah ini.

```
php artisan make:observer PenjualanObserver --model=Penjualan
```

Kemudian daftarkan observer tersebut ke dalam EventServiceProvider seperti model buku sebelumnya.

Nama file	app/Providers/EventServiceProvider.php
Deskripsi	Service provider laravel

```

use App\Models\Penjualan;
use App\Observers\PenjualanObserver;

...

public function boot() {

    ...

    Penjualan::observe(PenjualanObserver::class);
}

```

Buka PenjualanObserver dan edit method created. Pada fungsi ini kita menambahkan stok buku data penjualan ditambahkan.

Nama file	app/Observers/PenjualanObserver.php
Deskripsi	Observer untuk penjualan
<pre> ... public function created(Penjualan \$penjualan) { \$penjualan->buku->stok -= \$penjualan->stok; \$penjualan->buku->update(); } ... </pre>	

Fitur yang perlu diimplementasikan selanjutnya adalah hapus. Ketika data penjualan terhapus, stok tidak akan berpengaruh. Edit method destroy menjadi seperti di bawah ini.

Nama file	app/Http/Controllers/PenjualanController.php
Deskripsi	Controller untuk penjualan
<pre> ... public function destroy(Penjualan \$penjualan) { \$penjualan->delete(); return redirect() ->back() ->with('success', 'Data berhasil dihapus.');</pre>	