

LAPORAN PRAKTIKUM PEMROGRAMAN WEB 2

PERTEMUAN 5

ELOQUENT : MODEL



Disusun Oleh:

Wildan Dzaky Ramadhani

22/505766/SV/21917

Dosen Pengampu:

Dinar Nugroho Pratomo, S.Kom., M.IM., M.Cs.

Faza Maula Azif, S.Kom., M.Eng.

PROGRAM STUDI D4 TEKNOLOGI REKAYASA PERANGKAT LUNAK

DEPARTEMEN TEKNIK ELEKTRO DAN INFORMATIKA

SEKOLAH VOKASI

UNIVERSITAS GADJAH MADA

YOGYAKARTA

2023

DAFTAR ISI

DAFTAR ISI.....	2
BAB I.....	4
1.1. Latar Belakang.....	4
1.2. Rumusan Masalah.....	4
BAB II.....	5
2.1. Model.....	5
2.2. Seeder.....	6
2.3. Factory.....	7
BAB III.....	9
3.1. Membuat Model.....	9
3.2. Seeder.....	14
3.3. Factory.....	16
3.4. Pengurutan Data Buku.....	19
3.5. Pembuatan Kolom Nomor.....	20
3.6. Tugas Praktikum.....	21
BAB IV.....	25
4.1. Kesimpulan.....	25
DAFTAR PUSTAKA.....	26

BAB I

PENDAHULUAN

1.1. Latar Belakang

Dalam pengembangan aplikasi web menggunakan Laravel, Model, Seeder, dan Factory adalah komponen-komponen yang saling terkait dan menjadi pondasi penting. Dalam laporan praktikum ini, akan diulas konsep-konsep dasar terkait pengembangan aplikasi web dengan menggunakan framework Laravel. Laporan ini akan menjelaskan bagaimana Model, Seeder, dan Factory saling berinteraksi dalam mengelola data dalam aplikasi Laravel.

Model dalam Laravel memiliki peran krusial sebagai penghubung antara aplikasi dan Database. Model berperan sebagai perantara untuk mengakses dan mengelola data yang tersimpan dalam tabel-tabel Database. Dalam laporan ini, akan dijelaskan bagaimana Model dapat digunakan untuk merepresentasikan entitas dan hubungan antar entitas dalam aplikasi web.

Selain Model, laporan ini juga akan membahas peran Seeder dan Factory dalam mengisi serta mengelola data dalam Database. Seeder adalah alat yang memungkinkan pengembang untuk mengisi data awal yang dibutuhkan dalam tahap pengembangan dan pengujian aplikasi web. Di sisi lain, Factory membantu dalam pembuatan data dummy sesuai dengan kebutuhan aplikasi web. Melalui penggunaan Seeder dan Factory, laporan ini akan menunjukkan bagaimana data dummy dapat dengan cepat dan efisien diintegrasikan ke dalam Database, serta bagaimana data ini dapat digunakan dalam proses pengembangan dan pengujian aplikasi web. Keseluruhan, laporan ini akan memberikan pemahaman mendalam tentang bagaimana Model, Seeder, dan Factory dapat berkolaborasi dalam pengembangan aplikasi web menggunakan Laravel.

1.2. Rumusan Masalah

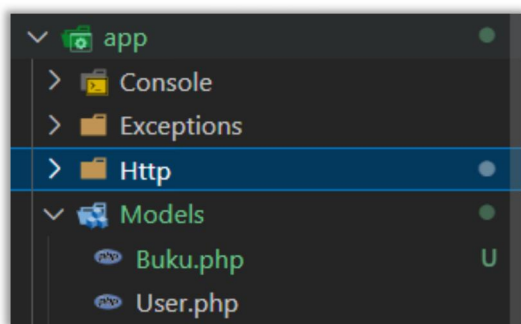
- A. Apakah yang dimaksud dengan Model, Seeder, dan Factory pada MVC?
- B. Bagaimana cara pembuatan dan implementasi Model, Seeder, dan Factory pada suatu project?

BAB II

LandaSAN TEORI

2.1. Model

Model adalah representasi dari tabel dalam basis data yang memungkinkan Kita untuk berinteraksi dengan data dalam tabel tersebut melalui objek-objek dalam kode PHP Kita. Dalam konteks framework Laravel, Model adalah elemen yang sangat penting dalam konsep MVC (Model-View-Controller). Model berperan sebagai perantara yang mengatur komunikasi antara aplikasi dan Database. Dalam kerangka kerja ini, Model berkolaborasi dengan Controller dan View untuk membentuk fitur atau halaman dalam aplikasi Laravel. Peran utama Model adalah untuk mengambil, memproses, dan mengembalikan data dari Database kepada Controller. Kita dapat menemukan Model dalam folder `app\Models` pada Laravel.



Berikut ini adalah beberapa konsep penting tentang Model dalam Laravel:

- a) ORM (Object-Relational Mapping): Model dalam Laravel mengadopsi ORM, yang memungkinkan Kita untuk berinteraksi dengan basis data menggunakan objek-objek PHP. Ini memungkinkan pengelolaan data dalam tabel basis data tanpa perlu menulis kueri SQL secara langsung.
- b) Representasi Data Tabel: Setiap Model dalam Laravel biasanya merepresentasikan satu tabel dalam basis data. Nama Model secara konvensi dihubungkan dengan nama tabel dalam bentuk jamak. Sebagai contoh, Model "User" akan dihubungkan dengan tabel "users".
- c) Eloquent ORM: Eloquent adalah ORM bawaan Laravel yang sangat kuat. Ini menyediakan berbagai metode dan fitur yang memungkinkan Kita menjalankan operasi CRUD (Create, Read, Update, Delete) pada data dalam tabel. Kita juga dapat mendefinisikan hubungan antara Model untuk menggambarkan ketergantungan data yang lebih kompleks.
- d) Migrasi Basis Data: Laravel menyediakan fasilitas migrasi yang memungkinkan Kita

mendefinisikan struktur basis data menggunakan kode PHP. Ini memudahkan pengelolaan struktur basis data dalam kode Kita dan memungkinkan Kita dengan mudah berbagi struktur basis data dengan tim pengembang lainnya.

- e) Validasi Data: Model dalam Laravel juga dapat digunakan untuk menerapkan validasi data sebelum menyimpannya ke dalam basis data. Ini membantu memastikan bahwa data yang dimasukkan adalah benar dan sesuai dengan aturan bisnis Kita.
- f) Event dan Observer: Laravel menyediakan mekanisme untuk menangani event yang terkait dengan Model. Kita dapat mendaftarkan observer untuk Model tertentu dan menjalankan kode tertentu secara otomatis saat Model tersebut diubah atau berinteraksi dengan data lainnya.
- g) Soft Deletes: Laravel mendukung fitur "soft deletes" yang memungkinkan Kita untuk menandai data sebagai dihapus tanpa menghapusnya benar-benar dari basis data. Data yang dihapus secara lunak masih dapat diakses jika diperlukan.
- h) Query Builder: Jika diperlukan, Kita dapat menggunakan Query Builder untuk mengeksekusi kueri kompleks pada Model Kita. Ini memberikan fleksibilitas tambahan saat berurusan dengan data.
- i) Konsep-konsep ini bersama-sama membentuk dasar pemahaman tentang bagaimana Model berperan dalam mengelola data dalam aplikasi Laravel.

2.2. Seeder

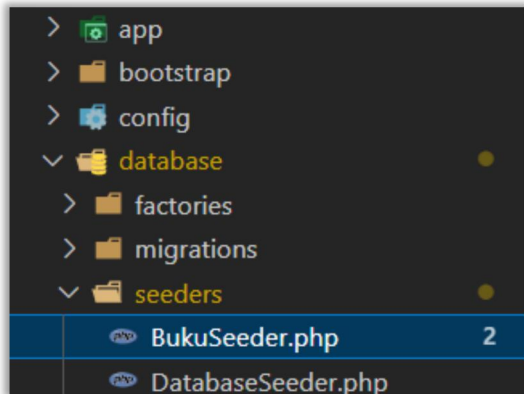
Seeder merupakan salah satu komponen penting dalam pengembangan aplikasi web dengan Laravel. Fungsinya adalah memungkinkan pengembang untuk mengisi basis data dengan data awal atau data uji secara otomatis. Seeder memiliki dua peran utama, yaitu dalam pengembangan dan pengujian aplikasi.

Dalam pengembangan, seeder sangat berguna karena memudahkan pengembang dalam mengisi tabel-tabel dalam basis data dengan data awal yang diperlukan. Contohnya, dalam pengembangan aplikasi e-commerce, Kita dapat menggunakan seeder untuk menambahkan produk-produk awal ke dalam tabel produk. Hal ini memungkinkan Kita untuk melihat bagaimana aplikasi berperilaku dengan data nyata.

Selain itu, dalam konteks pengujian, seeder membantu dalam mengisi basis data dengan data uji yang relevan. Ini memungkinkan pengujian fungsional aplikasi dengan berbagai skenario. Dengan kata lain, Kita dapat memastikan bahwa aplikasi Kita berfungsi dengan baik dalam berbagai situasi dengan menggunakan data yang berbeda.

Secara teknis, seeder dalam Laravel diimplementasikan sebagai kelas PHP yang

ditempatkan di direktori "database/seeder". Seeder ini menggunakan Eloquent, ORM dari Laravel, untuk berinteraksi dengan tabel-tabel dalam basis data. Kita dapat menjalankan seeder dengan perintah Artisan, seperti "php artisan db:seed", untuk mengisi tabel-tabel tersebut dengan data yang telah Kita tentukan dalam seeder.



Seeder adalah salah satu komponen integral dalam pengembangan aplikasi web menggunakan Laravel. Fungsinya adalah memungkinkan pengembang untuk mengisi basis data dengan data awal atau data uji secara otomatis. Seeder memiliki peran yang sangat berguna dalam dua konteks utama, yaitu pengembangan dan pengujian aplikasi.

Dalam pengembangan, seeder sangat mempermudah pengembang dalam mengisi tabel-tabel dalam basis data dengan data awal yang diperlukan. Sebagai contoh, jika Kita sedang mengembangkan sebuah aplikasi e-commerce, Kita dapat menggunakan seeder untuk menambahkan produk-produk awal ke dalam tabel produk. Hal ini memungkinkan Kita untuk melihat bagaimana aplikasi berperilaku dengan data nyata.

Selain itu, dalam konteks pengujian, seeder sangat membantu dalam mengisi basis data dengan data uji yang relevan. Ini memungkinkan pengujian fungsional aplikasi dengan berbagai skenario yang berbeda. Dengan kata lain, Kita dapat memastikan bahwa aplikasi Kita berfungsi dengan baik dalam berbagai situasi dengan menggunakan data yang beragam.

Secara teknis, seeder dalam Laravel diimplementasikan sebagai kelas PHP yang ditempatkan dalam direktori "database/seeder". Seeder ini menggunakan Eloquent, yang merupakan ORM (Object-Relational Mapping) dari Laravel, untuk berinteraksi dengan tabel-tabel dalam basis data. Kita dapat menjalankan seeder dengan perintah Artisan, seperti "php artisan db:seed," untuk mengisi tabel-tabel tersebut dengan data yang telah Kita tentukan dalam seeder. Ini merupakan alat yang sangat berguna untuk mengisi dan mengelola data dalam basis data Kita selama tahap pengembangan dan pengujian aplikasi web.

2.3. Factory

Factory dalam Laravel adalah komponen yang sangat penting dalam pengembangan aplikasi web. Factory digunakan untuk secara otomatis menghasilkan data palsu, yang memiliki banyak kegunaan dalam pengujian (testing) dan pengembangan aplikasi.

Factory bekerja dengan mengaitkan dirinya dengan model Eloquent yang sesuai dalam Laravel, dan kemudian menggunakan pustaka Faker untuk mengisi data palsu ke dalam model tersebut. Ini memungkinkan Kita untuk dengan mudah membuat sejumlah data palsu untuk mengisi tabel-tabel dalam basis data Kita, yang dapat digunakan dalam berbagai skenario pengujian dan pengembangan.

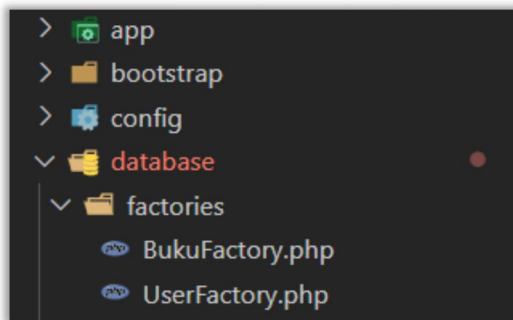
Untuk menggunakan factory dalam Laravel, langkah-langkah umumnya adalah sebagai berikut:

1. Mendefinisikan Factory: Pertama-tama, Kita perlu mendefinisikan factory dengan menggunakan perintah ``php artisan make:factory NamaFactory``. Perintah ini akan membuat file factory di dalam direktori ``database/factories``. Di dalam file factory ini, Kita dapat menentukan model terkait dan cara data palsu harus dibuat.

2. Definisi Factory: Setelah factory terdefinisi, Kita dapat menentukan bagaimana data palsu harus dibuat untuk model tertentu. Kita dapat menentukan jenis data, seperti string, angka, tanggal, dan lainnya, serta cara data ini harus diisi. Factory menggunakan pustaka Faker untuk menghasilkan data palsu yang realistis.

3. Menggunakan Factory: Setelah definisi factory selesai, Kita dapat dengan mudah menghasilkan data palsu menggunakan metode ``factory()`` pada model terkait. Contohnya, dengan menggunakan ``User::factory()->create()``, Kita dapat membuat instansi baru dari model ``User`` dengan data palsu sesuai dengan definisi factory. Kita juga dapat menggunakan metode lain seperti ``make()`` untuk membuat data palsu tanpa menyimpannya dalam basis data.

Factory adalah alat yang sangat berguna dalam pengembangan dan pengujian aplikasi web Laravel karena memungkinkan Kita untuk mengisi basis data dengan data palsu yang realistis dan dapat diandalkan dengan cepat dan mudah. Hal ini mempermudah Kita untuk menguji berbagai fitur dan skenario dalam aplikasi Kita tanpa harus menggunakan data nyata.



Factory juga dapat digunakan bersamaan dengan seeder, yang membantu Kita mengisi data awal ke dalam database. Factory memberikan fleksibilitas untuk menyesuaikan data yang dihasilkan sesuai dengan kebutuhan pengujian atau pengembangan Kita.

Selain itu, factory juga membantu dalam membersihkan data setelah pengujian selesai dengan mudah, sehingga Kita dapat menjaga kebersihan database Kita. Dengan pemahaman tentang konsep dasar factory dalam Laravel, Kita dapat lebih efisien mengelola data palsu dalam proyek Kita.

BAB III

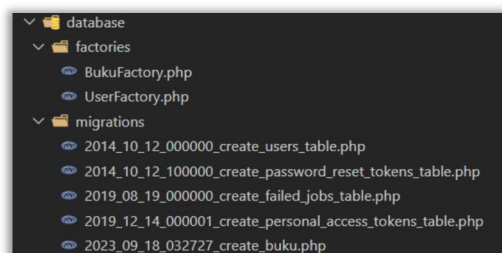
DOKUMENTASI PRAKTIKUM

3.1. Membuat Model

Sebelum kita ke pembuatan model, langkah pertama yang harus kita lakukan adalah membuat database dan melakukan migration terlebih dahulu seperti pada pertemuan sebelumnya dengan perintah sebagai berikut

```
PS C:\xampp\htdocs\ppw2_pertemuan5> php artisan make:migration create_buku
INFO Migration [C:\xampp\htdocs\ppw2_pertemuan5\database\Migrations\2023_09_18_032727_create_buku.php] created successfully.
```

Jika migration telah terbentuk, maka akan muncul folder pada bagian database mengenai file yang telah dibuat.



Kemudian ubah pada bagian file database yang dibuat untuk pembuatan table seperti gambar dibawah ini.

```
<?php

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

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('buku', function (Blueprint $table) {
            $table->id();
            $table->string('judul');
            $table->string('penulis');
            $table->integer('harga');
            $table->date('tgl_terbit');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('buku');
    }
};
```

Kemudian jangan lupa untuk mengganti DB_DATABASE sesuai dengan nama Database yang dibuat pada file .env.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=buku
DB_USERNAME=root
DB_PASSWORD=
```

Selanjutnya, kita dapat menggunakan perintah **php artisan migrate** untuk memigrasikan data yang telah dibuat ke database pada phpMyAdmin. Berikut merupakan tampilan perintah serta tampilan pada phpMyAdmin.

```
PS C:\xampp\htdocs\ppw2_pertemuan5> php artisan migrate

INFO Preparing database.
Creating migration table ..... 33ms DONE
INFO Running migrations.
2014_10_12_000000_create_users_table ..... 31ms DONE
2014_10_12_100000_create_password_reset_tokens_table ..... 38ms DONE
2019_08_19_000000_create_failed_jobs_table ..... 26ms DONE
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0004 seconds.)

```
SELECT * FROM `buku`
```

☐ Profiling [Edit inline] [Edit] [Explain SQL] []

id	judul	penulis	harga	tgl_terbit	created_at	updated_at
----	-------	---------	-------	------------	------------	------------

Query results operations

Create view

Kemudian kita akan membuat model dengan perintah **php artisan make:model Buku**. Usahakan nama model sama dengan nama database dengan diawali huruf kapital. Perintah tersebut akan membuat sebuah file baru Buku.php di dalam folder app/models.

```
PS C:\xampp\htdocs\webpert5> php artisan make:model Buku

INFO Model [C:\xampp\htdocs\webpert5\app\Models\Buku.php] created successfully.
```

```

└─ app
   ├── Console
   ├── Exceptions
   └── Http
       ├── Controllers
       │   ├── BukuController.php
       │   ├── Controller.php
       │   ├── Middleware
       │   └── Kernel.php
       └── Models
           ├── Buku.php
           └── User.php

```

Kemudian kita akan ganti isi dari Buku.php menjadi seperti berikut.

```
<?php

namespace App\Models;

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

5 references | 0 implementations
class Buku extends Model
{
    use HasFactory;

    0 references
    protected $table = "buku";
}
```

Perintah `protected $table = "buku";` dalam model Laravel digunakan untuk secara eksplisit menentukan nama tabel yang akan digunakan oleh model tersebut, yang berguna jika nama Model dan nama tabel tidak mengikuti konvensi baku Laravel. Dalam contoh ini, Model Buku akan berhubungan dengan tabel bernama buku di dalam Database.

Selanjutnya untuk dapat menampilkan tampilan website di browser perlu adanya pembuatan Route, Controller, serta View yang telah dipelajari sebelumnya. Untuk penjelasan lebih detailnya dapat dilihat pada Laporan Praktikum sebelumnya. Langkah singkat pembuatan Route, Controller, serta View adalah sebagai berikut:

1. Tuliskan perintah untuk membuat controller seperti pada gambar dibawah

```
PS C:\xampp\htdocs\ppw2_pertemuan5> php artisan make:controller BukuController
INFO Controller [C:\xampp\htdocs\ppw2_pertemuan5\app\Http\Controllers\BukuController.php] created successfully.
```

2. Buka File Controller dan ubah isinya menjadi seperti berikut

```
<?php

namespace App\Http\Controllers;

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

class BukuController extends Controller
{
    //fungsi index
    public function index(){
        $data_buku = Buku::all();

        return view('buku', compact('data_buku'));
    }
}
```

Kode tersebut merupakan bagian dari Controller Laravel yang mengontrol tindakan terkait data buku. Fungsi `index()` digunakan untuk menampilkan daftar semua buku. Pertama, Model Buku diambil menggunakan `Buku::all()`, yang mengambil semua data buku dari Database. Selanjutnya, data buku tersebut dikirim ke tampilan buku menggunakan metode `view()` bersama dengan data yang dikompresi menggunakan

compact()). Dengan demikian, saat pengguna mengakses halaman terkait, daftar buku akan ditampilkan berdasarkan data yang diambil dari Database. Controller ini mengikuti pola MVC (Model-View-Controller) untuk mengelola logika bisnis dan presentasi dalam aplikasi web Laravel.

3. Pada bagian folder **Routes/web.php** tuliskan perintah berikut

```
<?php

use App\Http\Controllers\BukuController;
use Illuminate\Support\Facades\Route;

/*
|-----
| Web Routes
|-----
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "web" middleware group. Make something great!
|
*/

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

Route::get('/buku', [BukuController::class, 'index']);
```

Perintah `Route::get('/buku', [BukuController::class, 'index']);` digunakan untuk mendefinisikan Route pada aplikasi web Laravel. Ini mengkonfigurasi sebuah rute yang akan ditangani oleh metode `index()` dalam `BukuController`. Jadi, ketika pengguna mengakses URL `/buku` dalam aplikasi, Laravel akan menjalankan metode `index()` dalam `BukuController` untuk menangani permintaan tersebut

4. Folder View yang akan digunakan untuk menampilkan data buku dapat dibuat dengan membuat file `buku.blade.php` yang didalamnya berisi perintah-perintah seperti di bawah ini:

```
<table class="table table-striped">
  <thead>
    <th>id</th>
    <th>Judul Buku</th>
    <th>Penulis</th>
    <th>Harga</th>
    <th>Tgl. Terbit</th>
    <th>Aksi</th>
  </thead>
  <tbody>
    @foreach ($data_buku as $buku)
      <tr>
        <td>{{ $buku->id }}</td>
        <td>{{ $buku->judul }}</td>
        <td>{{ $buku->penulis }}</td>
        <td>{{ "Rp ".number_format($buku->harga, 2, ',', ',') }}</td>
        <td>{{ date('d/m/Y', strtotime($buku->tgl_terbit)) }}</td>
      </tr>
    @endforeach
  </tbody>
</table>
```

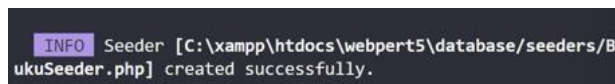
Kode HTML yang diberikan adalah bagian dari tampilan yang digunakan untuk menampilkan data buku dalam bentuk tabel. Tabel ini memiliki baris header (thead) yang menampilkan kolom-kolom seperti "id," "Judul Buku," "Penulis," "Harga,"

"Tgl. Terbit," dan "Aksi." Kemudian, melalui penggunaan perulangan **@foreach**, data buku yang diterima dari Controller dengan nama **\$data_buku** diiterasi. Setiap entri buku kemudian ditampilkan dalam baris (tr) yang sesuai dalam tabel, dan kolom-kolomnya diisi dengan informasi seperti ID buku, judul, penulis, harga dengan format mata uang, dan tanggal terbit yang diformat ulang. Dengan demikian, kode ini menghasilkan tampilan tabel yang menampilkan data buku secara terstruktur untuk ditampilkan kepada pengguna dalam halaman *web*.

3.2. Seeder

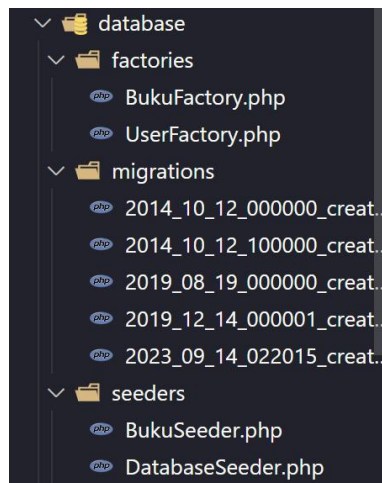
Secara default Laravel menempatkan file Seeder yang dibuat ke dalam direktori **database/seeds**. Untuk membuat Seeder, maka cukup dengan menggunakan fitur artisan *command*. Dengan artisan *command* pengembang dapat meng-generate file Seeder hanya dengan sekali *command* dan ditempatkan pada direktori **database/seeds**. Untuk menjalankannya ikuti langkah-langkah di bawah ini:

1. Tuliskan perintah di bawah ini pada *command line*.
php artisan make:seeder BukuSeeder



```
INFO Seeder [C:\xampp\htdocs\webpert5\database\seeders/BukuSeeder.php] created successfully.
```

2. Buka pada bagian Folder Seeder, lihat apakah *file* Seeder yang baru saja dibuat sudah masuk atau belum.



3. Secara *default* Seeders hanya memiliki satu method yakni **run**. Method ini akan di eksekusi dengan menggunakan artisan command **php artisan db:seed**. Method ini juga dapat digunakan untuk memasukkan *dummy* data yang diinginkan ke dalam Database dengan menggunakan Query Builder atau Eloquent.

```
<?php

use Illuminate\Database\Seeder;

class UsersTableSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        //
    }
}
```

4. Ubah perintah yang ada di *file* Seeder yang baru saja dibuat sesuai dengan yang diinginkan.

```
<?php

namespace Database\Seeders;

use App\Models\Buku;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

1 reference | 0 implementations
class BukuSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    0 references | 0 overrides
    public function run(): void
    {
        //menambahkan isi tabel
        $buku = Buku::create([
            "judul" => "Dariel Sang Pemuda Wanita",
            "penulis" => "Yodhimas",
            "harga" => 1000000,
            "tgl_terbit" => "2023-09-18"
        ]);
    }
}
```

Kode ini adalah bagian dari Seeder dalam Laravel yang digunakan untuk mengisi data awal ke dalam tabel buku dalam Database. Dalam metode `run()`, menggunakan model Buku untuk membuat entri baru dalam tabel tersebut dengan atribut-atribut yang telah ditentukan, seperti judul, penulis, harga, dan tanggal terbit. Dengan menjalankan Seeder ini, data buku dengan nilai-nilai yang telah ditentukan secara manual akan dimasukkan ke

dalam Database, sehingga memungkinkan untuk memiliki data awal atau contoh yang dapat digunakan untuk pengembangan dan pengujian aplikasi. Perlu diingat bahwa Seeder ini hanya contoh, dan dalam penggunaan sebenarnya, Seeder sering digunakan dengan Factory untuk menghasilkan banyak data dummy secara otomatis.

5. Jalankan perintah di bawah ini untuk memasukkan data di atas ke dalam Database.

```
PS C:\xampp\htdocs\webpert5> php artisan db:seed  
  
INFO Seeding database.
```

6. Lihat pada Database apakah sudah masuk atau belum data yang telah dituliskan sebelumnya.

	id	judul	penulis	harga	tgl_terbit	created_at	updated_at
<input type="checkbox"/>	1	Dariel Sang Pemuda Wanita	Yodhimas	1000000	2023-09-18	2023-09-18 03:53:27	2023-09-18 03:53:27

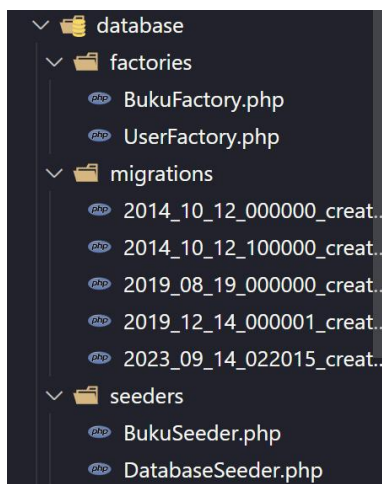
3.3. Factory

Untuk membuat Factory pada Laravel dapat dilakukan dengan mengikuti langkah-langkah di bawah ini:

1. Tuliskan perintah seperti di bawah ini untuk membuat file Factory.

```
PS C:\xampp\htdocs\webpert5> php artisan make:factory BukuFactory
```

2. Selanjutnya, cek pada Folder Factory apakah berhasil ditambahkan atau belum.



3. Ubah perintah Factory sesuai dengan yang diinginkan. Misalnya:


```

<?php

namespace Database\Factories;

use Illuminate\Database\Eloquent\Factories\Factory;

/**
 * @extends \Illuminate\Database\Eloquent\Factories\Factory<\App\Models\Buku>
 */
class BukuFactory extends Factory
{
    /**
     * Define the model's default state.
     *
     * @return array<string, mixed>
     */
    public function definition(): array
    {
        return [
            //
            'judul' => $this->faker->sentence(),
            'penulis' => $this->faker->name(),
            'harga' => $this->faker->numberBetween(30000, 100000),
            'tgl_terbit' => $this->faker->dateTimeThisMonth(),
        ];
    }
}

```

Kode ini merupakan bagian dari Factory Laravel yang bertanggung jawab untuk menghasilkan data palsu atau dummy untuk model Buku. Dalam metode `definition()`, kita mendefinisikan atribut-atribut model yang akan diisi dengan data palsu. Sebagai contoh, judul buku diisi dengan kalimat acak yang dihasilkan oleh Faker, penulis dengan nama acak, harga dengan angka acak antara 30,000 dan 100,000, dan tanggal terbit dengan tanggal acak dalam bulan ini. Dengan menggunakan Factory seperti ini, kita dapat dengan mudah membuat berbagai data dummy untuk pengujian atau pengembangan aplikasi, dan data tersebut akan dihasilkan secara otomatis saat diperlukan.

4. Untuk dapat menggunakan perintah yang telah dibuat pada Factory, maka selanjutnya pada Seeder tuliskan perintah di bawah ini.

```

<?php

namespace Database\Seeders;

use App\Models\Buku;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

















































class BukuSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        Buku::factory(10)->create();
    }
}

```


5. Kemudian, tuliskan perintah di bawah ini untuk memasukkan data-data yang baru.

```
PS C:\xampp\htdocs\webpert5> php artisan db:seed
INFO Seeding database.
```

6. Lihat pada Database apakah telah berhasil ditambahkan atau belum.

		id	judul	penulis	harga	tgl_terbit	created_at	updated_at
<input type="checkbox"/>	  	1	Dariel Sang Pemuda Wanita	Yodhimas	1000000	2023-09-18	2023-09-18 03:53:27	2023-09-18 03:53:27
<input type="checkbox"/>	  	2	Saepe beatae qui quas ut architecto nihil.	Betsy Wintheiser Jr.	785582768	2023-08-23	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	3	Est rem fugit iure perspiciatis omnis veritatis.	Flavio Kreiger	2101529003	2023-08-27	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	4	Autem temporibus et eveniet pariatur ipsa alias au...	Rusty Waters	1499347115	2023-08-25	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	5	Qui maxime in expedita saepe.	Dejon Douglas Jr.	309602568	2023-09-18	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	6	Ut sint nesciunt minima omnis nesciunt perferendis...	Carmela Stiedemann	1880037792	2023-08-23	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	7	Sed totam libero et alias est est provident vel.	Ms. Cayla Grant PhD	1860156915	2023-09-03	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	8	Vel ipsum nam consectetur dignissimos quis error.	Dr. Art Doyle PhD	1856784628	2023-09-18	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	9	Aut cum iusto consequuntur soluta.	Dr. Rafael Murphy PhD	54048885	2023-09-08	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	10	Et molestiae rem qui vero rerum ipsa aspernatur qu...	Prof. Maya Hodkiewicz I	289731689	2023-09-11	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	11	Deleniti aut harum reiciendis assumenda dolor simi...	Valentine Zboncak	1142146023	2023-08-29	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	12	Fugit illum itaque est molestias et provident.	Dr. Walton Rowe	1080926847	2023-08-29	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	13	Quia dolores corrupti veniam totam atque itaque.	Prof. Kariane Johnston Jr.	2144215844	2023-09-12	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	14	Sit dolorum ad culpa cupiditate dignissimos.	Jace Rath	378922134	2023-08-30	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	15	Et sit quibusdam sed accusantium nesciunt.	Iva Murphy Sr.	1159404786	2023-09-14	2023-09-18 04:01:48	2023-09-18 04:01:48
<input type="checkbox"/>	  	16	In reiciendis velit sit consequatur placeat.	Dr. Whitney Carroll V	1694224541	2023-09-06	2023-09-18 04:01:48	2023-09-18 04:01:48

7. Jalankan *web* untuk melihat seperti apa tampilannya dengan menggunakan **php artisan serve** lalu menuliskan **/buku** pada *url* paling belakang

id	Judul Buku	Penulis	Harga	Tgl. Terbit	Aksi
1	Dariel Sang Pemuda Wanita	Yodhimas	Rp 1,000,000,00	18/09/2023	1694995200
2	Saepe beatae qui quas ut architecto nihil.	Betsy Wintheiser Jr.	Rp 785,582,768,00	18/09/2023	1692748800
3	Est rem fugit iure perspiciatis omnis veritatis.	Flavio Kreiger	Rp 2,101,529,003,00	18/09/2023	1693094400
4	Autem temporibus et eveniet pariatur ipsa alias aut quaerat.	Rusty Waters	Rp 1,499,347,115,00	18/09/2023	1692921600
5	Qui maxime in expedita saepe.	Dejon Douglas Jr.	Rp 309,602,568,00	18/09/2023	1694995200
6	Ut sint nesciunt minima omnis nesciunt perferendis natus.	Carmela Stiedemann	Rp 1,880,037,792,00	18/09/2023	1692748800
7	Sed totam libero et alias est est provident vel.	Ms. Cayla Grant PhD	Rp 1,860,156,915,00	18/09/2023	1693699200
8	Vel ipsum nam consectetur dignissimos quis error.	Dr. Art Doyle PhD	Rp 1,856,784,628,00	18/09/2023	1694995200
9	Aut cum iusto consequuntur soluta.	Dr. Rafael Murphy PhD	Rp 54,048,885,00	18/09/2023	1694131200
10	Et molestiae rem qui vero rerum ipsa aspernatur quibusdam.	Prof. Maya Hodkiewicz I	Rp 289,731,689,00	18/09/2023	1694390400
11	Deleniti aut harum reiciendis assumenda dolor similique perspiciatis.	Valentine Zboncak	Rp 1,142,146,023,00	18/09/2023	1693267200
12	Fugit illum itaque est molestias et provident.	Dr. Walton Rowe	Rp 1,080,926,847,00	18/09/2023	1693267200
13	Quia dolores corrupti veniam totam atque itaque.	Prof. Kariane Johnston Jr.	Rp 2,144,215,844,00	18/09/2023	1694476800
14	Sit dolorum ad culpa cupiditate dignissimos.	Jace Rath	Rp 378,922,134,00	18/09/2023	1693353600
15	Et sit quibusdam sed accusantium nesciunt.	Iva Murphy Sr.	Rp 1,159,404,786,00	18/09/2023	1694649600
16	In reiciendis velit sit consequatur placeat.	Dr. Whitney Carroll V	Rp 1,694,224,541,00	18/09/2023	1693958400
17	Sit voluptatem fugit fuga quis pariatur.	Alvina Schmeler	Rp 503,110,632,00	18/09/2023	1694476800
18	Pariatur laudantium suscipit quis odio asperiores.	Lori Tillman Jr.	Rp 104,228,232,00	18/09/2023	1694736000
19	Ab facere ab tempore sit reiciendis dolores inventore perferendis.	Ms. Loren Miller	Rp 379,753,891,00	18/09/2023	1693353600

3.4. Pengurutan Data Buku

Dalam menampilkan data, Laravel memiliki kemampuan untuk mengurutkannya berdasarkan kolom tertentu pada tabel. Misalnya, jika kita ingin mengurutkan buku berdasarkan buku terakhir yang dimasukkan (buku terbaru di perpustakaan). Dengan kata lain, kita ingin mengurutkannya berdasarkan kolom **id** yang paling terakhir. Untuk melakukan ini, kita menggunakan **sortByDesc('id')**. Untuk mengimplementasikannya, kita perlu mengubah kode dalam file **BukuController.php**, yaitu menggantikan **\$data_buku = Buku::all();** dengan **\$data_buku = Buku::all()->sortByDesc('id');**. Dengan langkah ini, data buku akan ditampilkan secara terurut berdasarkan id dengan nilai yang paling terbaru. Berikut adalah *source code* perubahan tersebut:

```
<?php

namespace App\Http\Controllers;

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

class BukuController extends Controller
{
    //fungsi index
    public function index(){
        $data_buku = Buku::all()->sortByDesc('id');

        return view('buku', compact('data_buku'));
    }
}
```

Tampilan yang akan di keluarkan adalah sebagai berikut

id	Judul Buku	Penulis	Harga	Tgl. Terbit Aksi
11	Officiis exercitationem soluta ut non.	Dakota Orn	Rp 92,328,00	12/09/2023
10	Sit repellendus ratione iusto ut sint.	Prof. Dillan Weissnat PhD	Rp 99,272,00	02/09/2023
9	Ut odit quos eos temporibus sed.	Odie Macejkovic Jr.	Rp 62,401,00	20/08/2023
8	Beatae architecto vero et tempore est molestiae.	Ahmad Witting	Rp 85,234,00	16/08/2023
7	Soluta dicta et quidem nesciunt omnis hic dolore.	Jannie Mohr PhD	Rp 45,560,00	03/09/2023
6	Id ipsam saepe porro doloreque eligendi.	Mrs. Elta Morissette Sr.	Rp 37,781,00	28/08/2023
5	Repellat consequuntur maiores recusandae eum aliquid.	Magnolia Deckow	Rp 83,541,00	21/08/2023
4	Ut ea doloreque explicabo.	Laurence Greenfelder DVM	Rp 37,537,00	26/08/2023
3	Voluptatem esse aliquid voluptatem numquam velit.	Skye Turcotte	Rp 95,325,00	11/09/2023
2	Officiis aut consectetur quasi nesciunt est.	Holden Gutmann MD	Rp 95,566,00	17/08/2023
1	Pergi Pagi Pulang Siang	Maritza Angel	Rp 45,000,00	14/09/2003

Selanjutnya, kita coba kembali dengan mengurutkan sesuai dengan Judul Buku.

```
<?php

namespace App\Http\Controllers;

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

class BukuController extends Controller
{
    //fungsi index
    public function index(){
        $data_buku = Buku::all()->sortByDesc('judul');

        return view('buku', compact('data_buku'));
    }
}
```

← → ↺ ⓘ 127.0.0.1:8000/buku

id	Judul Buku	Penulis	Harga	Tgl. Terbit Aksi
3	Voluptatem esse aliquid voluptatem numquam velit.	Skye Turcotte	Rp 95,325,00	11/09/2023
9	Ut odit quos eos temporibus sed.	Odie Macejkovic Jr.	Rp 62,401,00	20/08/2023
4	Ut ea doloreque explicabo.	Laurence Greenfelder DVM	Rp 37,537,00	26/08/2023
7	Soluta dicta et quidem nesciunt omnis hic dolore.	Jannie Mohr PhD	Rp 45,560,00	03/09/2023
10	Sit repellendus ratione iusto ut sint.	Prof. Dillan Weissnat PhD	Rp 99,272,00	02/09/2023
5	Repellat consequuntur maiores recusandae eum aliquid.	Magnolia Deckow	Rp 83,541,00	21/08/2023
1	Pergi Pagi Pulang Siang	Maritza Angel	Rp 45,000,00	14/09/2003
11	Officiis exercitationem soluta ut non.	Dakota Orn	Rp 92,328,00	12/09/2023
2	Officiis aut consectetur quasi nesciunt est.	Holden Gutmann MD	Rp 95,566,00	17/08/2023
6	Id ipsam saepe porro doloreque eligendi.	Mrs. Elta Morissette Sr.	Rp 37,781,00	28/08/2023
8	Beatae architecto vero et tempore est molestiae.	Ahmad Witting	Rp 85,234,00	16/08/2023

3.5. Pembuatan Kolom Nomor

Untuk memperbaiki tampilan kolom No dalam tabel buku yang saat ini tidak menampilkan nomor berurut dengan benar, kita perlu membuat variabel nomor khusus yang akan mengandung nilai nomor yang bertambah seiring dengan data yang ditampilkan. Caranya adalah dengan membuka file BukuController.php dan mengedit fungsi index(). Dalam kode yang telah diberikan, kita membuat variabel \$no yang awalnya diatur ke nilai 0. Kemudian, kita mengirimkan variabel \$no ke tampilan dengan menggunakan fungsi compact(). Untuk melihat perubahan ini dalam tampilan, kita perlu memodifikasi kode yang sebelumnya menggunakan {{ \$buku->id }} menjadi {{ ++\$no }}. Dengan melakukan langkah-langkah ini, tampilan akan menampilkan nomor yang bertambah secara berurutan seperti 1, 2, dan seterusnya untuk setiap entri buku yang ditampilkan dalam tabel.

Tampilan Web:

id	Judul Buku	Penulis	Harga	Tgl. Terbit Aksi
1	Pergi Pagi Pulang Siang	Maritza Angel	Rp 45,000,00	14/09/2003
2	Officiis aut consectetur quasi nesciunt est.	Holden Gutmann MD	Rp 95,566,00	17/08/2023
3	Voluptatem esse aliquid voluptatem numquam velit.	Skye Turcotte	Rp 95,325,00	11/09/2023
4	Ut ea doloremque explicabo.	Laurence Greenfelder DVM	Rp 37,537,00	26/08/2023
5	Repellat consequuntur maiores recusandae eum aliquid.	Magnolia Deckow	Rp 83,541,00	21/08/2023
6	Id ipsam saepe porro doloremque eligendi.	Mrs. Elta Morissette Sr.	Rp 37,781,00	28/08/2023
7	Soluta dicta et quidem nesciunt omnis hic dolore.	Jannie Mohr PhD	Rp 45,560,00	03/09/2023
8	Beatae architecto vero et tempore est molestiae.	Ahmad Witting	Rp 85,234,00	16/08/2023
9	Ut odit quos eos temporibus sed.	Odie Macejkovic Jr.	Rp 62,401,00	20/08/2023
10	Sit repellendus ratione iusto ut sint.	Prof. Dillan Weissnat PhD	Rp 99,272,00	02/09/2023
11	Officiis exercitationem soluta ut non.	Dakota Orn	Rp 92,328,00	12/09/2023

3.6. Tugas Praktikum

Link GitHub : https://github.com/wildandr/https---github.com-wildandr-ppw2Pertemuan5_2.git

1. Silahkan buat sebuah code dalam Controller dan View yang akan menampilkan jumlah data yang dimiliki (tampilkan di bawah tabel).
2. Silahkan sebuah code dalam Controller dan View yang akan menampilkan jumlah total harga semua buku yang ada pada tabel (tampilkan di bawah tabel).
3. TUGAS TAMBAHAN Seeder dan Factory (SUDAH ADA DI BAB III LATIHAN PRAKTIKUM DI ATAS)

http\Controllers\BukuController.php

```
<table border="1 px" class="table table-striped">
  <thead>
    <th>id</th>
    <th>Judul Buku</th>
    <th>Penulis</th>
    <th>Harga</th>
    <th>Tgl. Terbit</th>
    <th>Aksi</th>
  </thead>
  <tbody>
    @foreach ($data_buku as $buku)
      <tr>
        <td>{{ ++$no }}</td>
        <td>{{ $buku->judul }}</td>
        <td>{{ $buku->penulis }}</td>
        <td>{{ "Rp ".number_format($buku->harga, 2, ',', ',') }}</td>
        <td>{{ date('d/m/Y', strtotime($buku->tgl_terbit)) }}</td>
      </tr>
    @endforeach
  </tbody>
</table>
<h3> <p>Jumlah data = {{ $jumlahData }}</p> </h3>
<h3> <p>Total harga = {{ "Rp ".$totalHarga . ",00"}}</p> </h3>
```

Class ini secara umum digunakan untuk mengelola permintaan HTTP terkait data buku

dalam aplikasi. Di dalamnya, terdapat fungsi **index()** yang akan dieksekusi ketika pengguna mengakses URL tertentu yang terkait dengan tampilan data buku. Fungsi **index()** melakukan beberapa hal:

1. Mendapatkan data buku dari model **Buku** menggunakan **Buku::all()** dan mengurutkannya berdasarkan kolom id yang paling baru dengan - **>sortByDesc('id')**.
2. Menghitung jumlah data buku dalam basis data dengan **Buku::all()->count('id')**.
3. Menghitung total harga dari seluruh buku dalam basis data dengan **Buku::all()->sum('harga')**.
4. Mengirimkan data buku, nomor urut, jumlah data, dan total harga ke tampilan "buku" menggunakan **view()** dan **compact()**, sehingga data ini dapat ditampilkan kepada pengguna.

Dengan demikian, class **BukuController** bertindak sebagai perantara antara permintaan pengguna dan model data buku serta mengatur tampilan data buku yang akan ditampilkan kepada pengguna.

resources\views\buku.blade.php

```
<?php

namespace App\Http\Controllers;

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

class BukuController extends Controller
{
    //fungsi index
    public function index(){
        $data_buku = Buku::all()->sortByDesc('id');
        $no = 0;
        $jumlahData = Buku::all()->count('id');
        $totalHarga = Buku::all()->sum('harga');
        return view('buku', compact('data_buku', 'no', 'jumlahData',
        'totalHarga'));
    }
}
```

Kode di atas adalah bagian dari View dalam aplikasi Laravel yang bertujuan untuk menampilkan data buku dalam bentuk tabel HTML. Ini adalah penjelasan singkat dari kode tersebut:

1. `<table>`: Ini adalah elemen HTML yang digunakan untuk membuat tabel.
Atribut
``border="1 px"`` mencoba menentukan ketebalan garis pinggir tabel, tetapi perlu dicatat bahwa penggunaan ini bukan cara yang umum untuk mengatur garis pinggir tabel dalam Bootstrap. Biasanya, dapat mengandalkan kelas CSS Bootstrap seperti ``table-bordered``.
2. `<thead>`: Ini adalah elemen yang digunakan untuk menentukan bagian kepala tabel (header) yang akan berisi judul kolom-kolom.
3. `<th>`: Ini adalah elemen untuk menentukan judul kolom-kolom tabel, seperti "id," "Judul Buku," "Penulis," "Harga," "Tgl. Terbit," dan "Aksi."
4. `<tbody>`: Ini adalah elemen yang digunakan untuk menentukan bagian isi tabel yang akan diisi dengan data buku.
5. `@foreach ($data_buku as $buku)`: Ini adalah perulangan **foreach** yang digunakan untuk mengulang setiap entri buku dalam variabel **\$data_buku** yang diterima dari controller.
6. `<tr>`: Ini adalah elemen untuk mendefinisikan baris dalam tabel.
7. `<td>`: Ini adalah elemen untuk menampilkan data dalam sel-sel tabel. ``$buku->judul``, ``$buku->penulis``, dan sebagainya digunakan untuk menampilkan data buku dari setiap kolom.
8. `<h3>`: Ini adalah elemen untuk mengatur judul-judul yang menunjukkan jumlah data dan total harga buku. Variabel **\$jumlahData** dan **\$totalHarga** disisipkan ke dalam teks menggunakan kurung kurawal ganda `{{ }}`.

Kode tersebut menghasilkan tampilan tabel yang menampilkan data buku dengan kolom-kolom yang sesuai dan juga menampilkan jumlah data serta total harga dari data buku tersebut.

← → ↻ ⓘ 127.0.0.1:8000/buku					
id	Judul Buku	Penulis	Harga	Tgl. Terbit	Aksi
1	Officiis exercitationem soluta ut non.	Dakota Orn	Rp 92,328,00	12/09/2023	
2	Sit repellendus ratione iusto ut sint.	Prof. Dillan Weissnat PhD	Rp 99,272,00	02/09/2023	
3	Ut odit quos eos temporibus sed.	Odie Macejkovic Jr.	Rp 62,401,00	20/08/2023	
4	Beatae architecto vero et tempore est molestiae.	Ahmad Witting	Rp 85,234,00	16/08/2023	
5	Soluta dicta et quidem nesciunt omnis hic dolore.	Jannie Mohr PhD	Rp 45,560,00	03/09/2023	
6	Id ipsam saepe porro doloremque eligendi.	Mrs. Elta Morissette Sr.	Rp 37,781,00	28/08/2023	
7	Repellat consequuntur maiores recusandae eum aliquid.	Magnolia Deckow	Rp 83,541,00	21/08/2023	
8	Ut ea doloremque explicabo.	Laurence Greenfelder DVM	Rp 37,537,00	26/08/2023	
9	Voluptatem esse aliquid voluptatem numquam velit.	Skye Turcotte	Rp 95,325,00	11/09/2023	
10	Officiis aut consectetur quasi nesciunt est.	Holden Gutmann MD	Rp 95,566,00	17/08/2023	
11	Pergi Pagi Pulang Siang	Maritza Angel	Rp 45,000,00	14/09/2003	

Jumlah data = 11

Total harga = Rp 779545,00

BAB IV

KESIMPULAN

4.1. Kesimpulan

Praktikum ini berfokus pada penggunaan "Controller" dan "Migration" dalam Laravel adalah langkah penting dalam memahami pengembangan aplikasi web modern. "Controller" berfungsi sebagai otak aplikasi yang mengatur alur logika bisnis dan merespons permintaan pengguna, sementara "Migration" memungkinkan pengelolaan struktur basis data dengan terstruktur. Dalam praktikum ini, mahasiswa belajar cara mengelola dan mengatur basis data dengan efisien, serta bagaimana memisahkan logika aplikasi menjadi komponen yang terstruktur. Kemampuan ini menjadi fondasi yang kuat untuk mengembangkan aplikasi web yang kuat, modular, dan siap berkembang. Praktikum ini membekali mahasiswa dengan keterampilan penting dalam pengembangan web dan mempersiapkan mereka untuk menghadapi tuntutan dunia kerja yang terus berkembang di bidang teknologi informasi.

DAFTAR PUSTAKA

<https://buildwithangga.com/tips/apa-itu-seeder-pada-laravel>

https://github.com/wildandr/https---github.com-wildandr-ppw2Pertemuan5_2.git