Tutorial Laravel 1: Asas CRUD Menggunakan Eloquent (ORM)

Penulis: Muhamad Said Nizamuddin Bin Nadim

Pada tutorial kali ini kita akan membina asas CRUD (Create,Read,Update,Delete) menggunakan konsep Eloquent (ORM) Laravel 8. Kaedah yang ditunjukkan di dalam tutorial ini sangat membantu bagi pemula yang ingin membangunkan aplikasi web menggunakan Laravel 8. mereka. Kita akan membina aplikasi ini bermula daripada awal sehingga selesai.

Langkah 1: Cipta Projek Laravel

Dalam langkah pertama, kita perlu mencipta projek Laravel 8 di dalam folder***localhost*** komputer anda. Kita perlu mengaktifkan terminal terlebih dahulu dan memasukkan c***omposer command*** berikut di terminal.

```
composer create-project laravel/laravel:~8.0 example-app --prefer-dist
```

Langkah 2: Hubung Pangkalan Data

Kita perlu mencipta satu pangkalan data (***Database***) menggunakan perisian pengurusan pangkalan data. Sebagai contoh pangkalan data bernama ***tutorial_week4*** kemudian kita akan mengubah tetapan yang berkaitan pangkalan data. Caranya adalah kita membuka folder tersebut menggunakan ***editor*** dan mengubah fail ***.env***.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=tutorial_week4 // ubah maklumat di sini
DB_USERNAME=root // ubah jika tetapan lalai anda bukan root
DB_PASSWORD= // ubah jika tetapan lalai mempunyai kata laluan
```

Langkah 3: Cipta Model, Migration, Controller

Pada versi ***Laravel 8.70*** keatas**,** Kita dapat mencipta ***model*** berserta ***migration*** dan ***controller*** menggunakan ***comand artisan*** seperti berikut dalam projek kita.

```
php artisan make:model example -mcr
```

Langkah 4: Tambah maklumat Migration

Migration merupakan kawalan kepada pangkalan data kita, ianya membolehkan kita mengubah suai skema pangkalan data dengan mudah. Laravel telah terpasang **Laravel's schema Builder** secara lalai boleh terus digunakan.

```
    public function up()
    {
        Schema::create('example', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('number');
            $table->text('address');
            $table->timestamps();
        });
    }
```

Peringatan: Fail ***Migration*** berada di project/database/migrations

Setelah selesai mengatur nama kolum baharu yang di dalam fail ***Migration***, kita perlu menjalankan ***command artisan*** seperti berikut.

```
php artisan migrate
```

Langkah 5: Tambah Route

Menggunakan ***Routing***, kita boleh membuat ***Request URL***. terdapat beberapa bentuk set permintaan ***HTTP*** seperti ***POST Request***, ***GET Request***, **PUT Request** dan **DELETE Request** menggunakan **Routing** Laravel 8. Anda boleh membuat **Route** dalam web dengan mudah.

Kita akan menambah satu set permintaan ***HTTP*** yang diperlukan dalam satu ***Controller*** dan kaedah ini perlu diulang disetiap ***Controller*** atau pautan yang ingin digunakan.

```
use App\Http\Controllers\ExampleController; //letak diatas

Route::group(['prefix' => 'example', 'as' => 'example.'], function () {
    Route::get('/', [example::class, 'index'])->name('index');
    Route::get('/create', [example::class, 'create'])->name('create');
    Route::post('store', [example::class, 'store'])->name('store');
    Route::get('/{item}/show', [example::class, 'show'])->name('show');
    Route::get('/{item}/edit', [example::class, 'edit'])->name('edit');
    Route::put('/{item}/update', [example::class, 'update'])->name('update');
    Route::delete('/{item}/destroy', [example::class, 'destroy'])->name('destroy');
});
```

Peringatan: Route berada di project/routes/web.php

Langkah 6: Ubah Model

Model dalam Laravel 8 menyediakan hubungan untuk berinteraksi dengan pangkalan data.***Model Event*** merupakan kitaran hayat M**odel** yang boleh kita gunakan untuk menjalankan aktiviti seperti rekod pangkalan data untuk menyimpan data, mengemaskini atau memadam data.

```
    protected $table = "example";

    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [

        'name',
        'number',
        'address',
    ];
```

Peringatan: Model berada di project/app/models

Langkah 7: Ubah AppServiceProvider

Seterusnya, jika kita ingin menggunakan ***Framework Bootstrap***, kita perlu mengubah tetapan Paginationpergi ke fail dan import dan gunakan ***Paginator::useBootstrap()*** dalam ***function boot()***:

```
use Illuminate\Pagination\Paginator; //letak diatas

public function boot()
{
     Paginator::useBootstrap();
}
```

Peringatan: fail berada di project/app/Providers/AppServiceProvider.php

Langkah 8: Tambah Fungsi Controller

```
use App\Models\Example; // letak diatas
```

```
/**
 * Display a listing of the resource.
 *
 * @return \Illuminate\Http\Response
 */
public function index()
{
    $example = Example::latest()->paginate(5);

    return view('example.index',compact('example'))
        ->with('i', (request()->input('page', 1) - 1) * 5);
}

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

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

    $example = new PhoneBook1();
    $example->name = $request->name;
    $example->name = $request->number;
    $example->name = $request->address;
    $example->save();

    return redirect()->route('example.index')
        ->with('success','Example created successfully.');
}

/**
 * Display the specified resource.
 *
 * @param  \App\Models\Example  $example
 * @return \Illuminate\Http\Response
 */
public function show(Example $example)
{
    return view('example.show',compact('example'));
}

/**
 * Show the form for editing the specified resource.
 *
 * @param  \App\Models\Example  $example
```

```
     * @return \Illuminate\Http\Response
     */
    public function edit(Example $example)
    {
        return view('example.edit',compact('example'));
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \App\Models\Example  $Example
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, Example $example)
    {
        $request->validate([
            'name' => 'required',
            'number' => 'required',
            'address' => 'required',
        ]);

        $example = Example::find($example->id);
        $example->name = $request->name;
        $example->number = $request->number;
        $example->address = $request->address;
        $example->save();

        return redirect()->route('example.index')
            ->with('success','Example updated successfully.');
    }

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

        return redirect()->route('example.index')
            ->with('success','Example deleted successfully');
    }
```

Peringatan: Controller berada di project/app/Http/Controllers

Langkah 9: Tambah View

layout.blade.php

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.cs

    <title>Laravel Tutorial - Week 4</title>
  </head>
  <body>
    <!-- Begin page content -->
    @yield('content')

    <!-- Optional JavaScript; choose one of the two! -->

    <!-- Option 1: jQuery and Bootstrap Bundle (includes Popper) -->
    <script src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js" integrity="sha384-D
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js" integrit

    <!-- Option 2: Separate Popper and Bootstrap JS -->
    <!--
    <script src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js" integrity="sha384-D
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js" integrity="sha38
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.min.js" integrity="sha3
    -->
  </body>
</html>
```

index.blade.php

```
@extends('phonebook1.layout')

@section('content')
<main role="main" class="container">
    <h1 class="text-center mt-5">Laravel Tutorial - Week 4</h1>
    <div class="d-flex flex-row-reverse bd-highlight mb-3">
        <a class="btn btn-success" href="{{ route('phonebook1.create') }}"> Create New Contact</a>
    </div>
        @if ($message = Session::get('success'))
        <div class="alert alert-success">
            <p>{{ $message }}</p>
        </div>
    @endif
    </div>
    <table class="table">
        <thead class="thead-dark">
            <tr>
                <th scope="col">#</th>
                <th scope="col">Name</th>
                <th scope="col">Number</th>
                <th scope="col">Address</th>
                <th scope="col">Action</th>
            </tr>
        </thead>
        <tbody>
            @foreach ($phoneBook1s as $phoneBook1)
                <tr>
                    <th scope="row">{{ ++$i }}</th>
                    <td>{{ $phoneBook1->name }}</td>
                    <td>{{ $phoneBook1->number }}</td>
                    <td>{{ $phoneBook1->address }}</td>
                    <td>
                        <a href="{{ route('phonebook1.show', $phoneBook1->id) }}" class="btn btn-info">
                        <a href="{{ route('phonebook1.edit', $phoneBook1->id) }}" class="btn btn-warnin
                        <form action="{{ route('phonebook1.destroy', $phoneBook1->id) }}" method="post"
                            @method('Delete')
                            @csrf
                            <button type="submit" class="btn btn-danger">DELETE</button>
                        </form>
                    </td>
                </tr>
            @endforeach
        </tbody>
    </table>
    <div class="d-flex flex-row-reverse bd-highlight mb-3">
        {!! $phoneBook1s->links() !!}
    </div>
</main>
@endsection
```

create.blade.php