

Sepecial Edition
For Ayang Syma



Tutorial

LARAVEL

#CRUD



By Your Futures Husband

1. Buat project baru

```
composer create-project laravel/laravel nama_project
```

2. Buka project laravel dengan vs code

```
cd nama_project
```

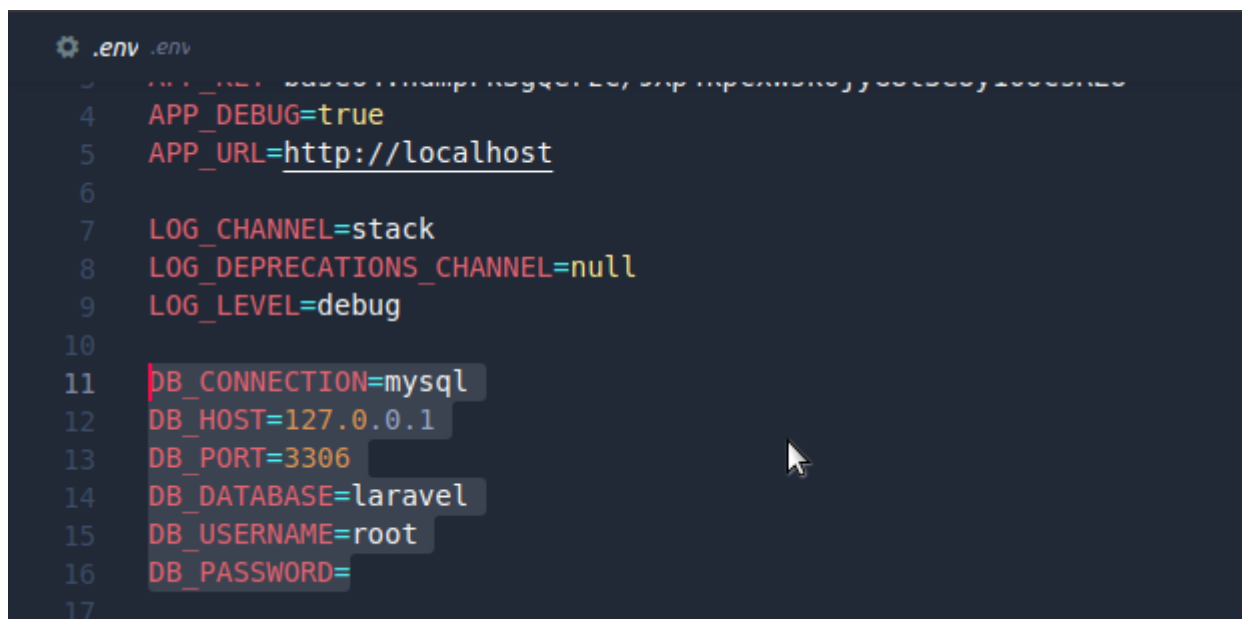
```
code .
```

3. Sekarang kita akan membuat database nya di php myadmin / xampp.



4. Edit konfigurasi database di .ENV lokasi file tepat berada di dalam project.

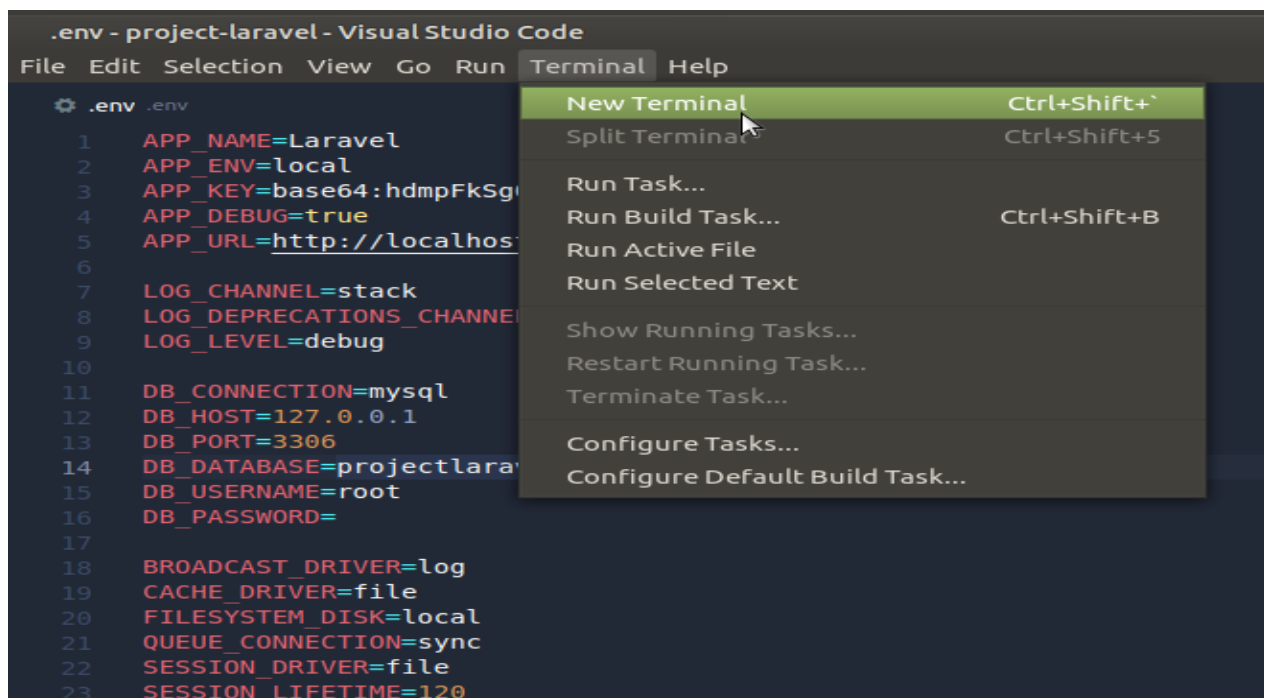
Ubah pada **DB_DATABASE=NAMA_DATABASE**, nama database di sesuaikan dengan nama database yang tadi di buat.



Jadi seperti ini :

```
.env .env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:hdmpFkSgQerzC/9Xp4kpcxWsk0jyG8t3eUy10oCsREU=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_DEPRECATIONS_CHANNEL=null
9 LOG_LEVEL=debug
10
11 DB_CONNECTION=mysql
12 DB_HOST=127.0.0.1
13 DB_PORT=3306
14 DB_DATABASE=projectlaravel
15 DB_USERNAME=root
16 DB_PASSWORD=
```

5. Setelah itu install **laravel/ui** dengan membuka terminal di vs code di bagian atas/ **ctr + `** :



Setelah terminal nya terbuka ketik :

```
composer require laravel/ui
```

Kemudian setelahnya jalankan:

```
php artisan ui bootstrap --auth
npm install
```

```
npm run dev
```

```
php artisan migrate
```

```
php artisan serve
```

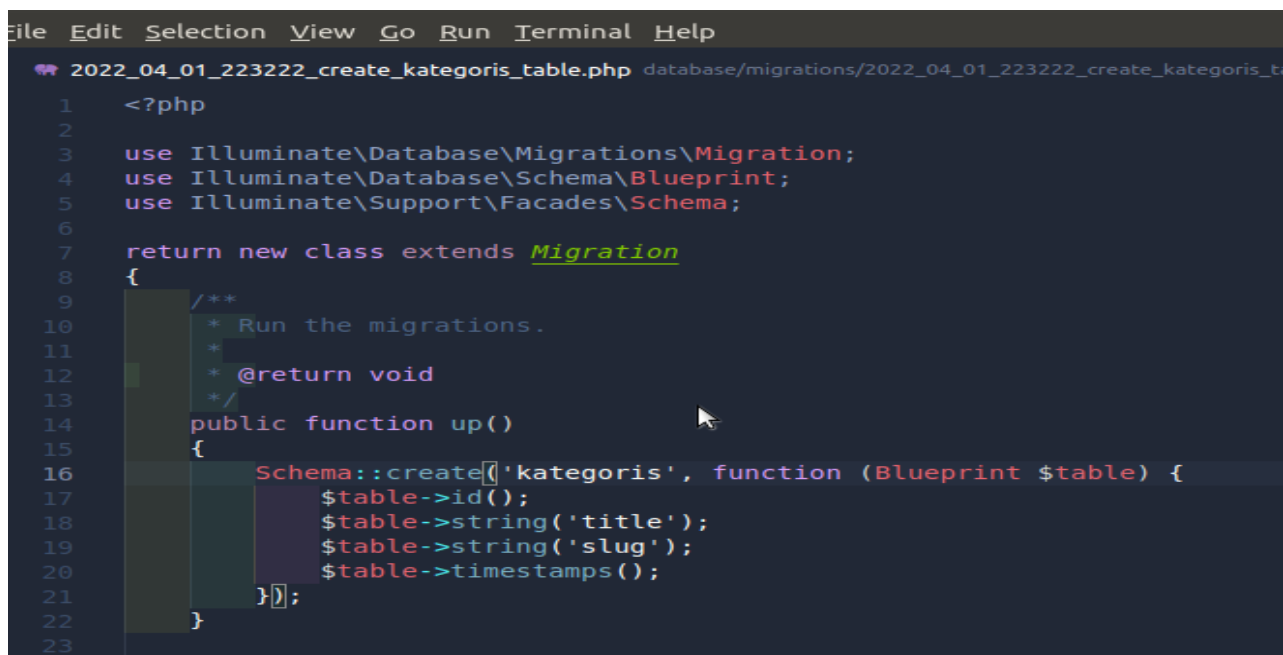
Setelah menjalankan development server kemudian buka di browser <http://127.0.0.1:8000> dan register.

6. Sekarang kita buat model , migration dan controller, buka terminal baru kemudian ketik

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

7. Edit file migration kategori:

Tambahkan field title dan slug



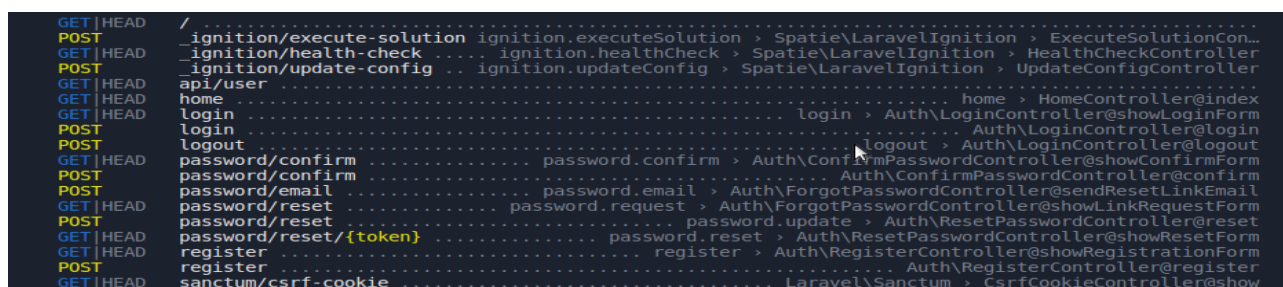
```
1 <?php
2
3 use Illuminate\Database\Migrations\Migration;
4 use Illuminate\Database\Schema\Blueprint;
5 use Illuminate\Support\Facades\Schema;
6
7 return new class extends Migration
8 {
9     /**
10      * Run the migrations.
11      *
12      * @return void
13      */
14     public function up()
15     {
16         Schema::create('kategoris', function (Blueprint $table) {
17             $table->id();
18             $table->string('title');
19             $table->string('slug');
20             $table->timestamps();
21         });
22     }
23 }
```

```
php artisan migrate
```

8. Buka file route / web.php dan tambahkan route resource. Pertama kita cek terlebih dahulu apa saja route yang kita punya dengan :

(bagian ini ngga wajib di lakukan cuma buat cek aja)

```
php artisan route:list
```



```
GET|HEAD / ..... ignition.executeSolution ..... Spatie\LaravelIgnition > ExecuteSolutionCon...
POST / ..... ignition.healthCheck ..... Spatie\LaravelIgnition > HealthCheckController
GET|HEAD / ..... ignition/update-config ..... ignition.updateConfig > Spatie\LaravelIgnition > UpdateConfigController
GET|HEAD api/user .....
GET|HEAD home ..... home > HomeController@index
GET|HEAD login ..... login > Auth\LoginController@showLoginForm
POST login ..... Auth\LoginController@login
POST logout ..... Auth\LoginController@logout
GET|HEAD password/confirm ..... password.confirm > Auth\ConfirmPasswordController@showConfirmForm
POST password/confirm ..... Auth\ConfirmPasswordController@confirm
POST password/email ..... Auth\ForgotPasswordController@sendResetLinkEmail
GET|HEAD password/reset ..... password.request > Auth\ForgotPasswordController@showLinkRequestForm
POST password/reset ..... password.update > Auth\ResetPasswordController@reset
GET|HEAD password/reset/{token} ..... password.reset > Auth\ResetPasswordController@showResetForm
GET|HEAD register ..... register > Auth\RegisterController@showRegistrationForm
POST register ..... Auth\RegisterController@register
GET|HEAD sanctum/csrf-cookie ..... Laravel\Sanctum > CsrfCookieController@show
```

Kita tambahkan route resource seperti ini :

```
Route::resource('kategori',KategoriController::class);
```

web.php - project-laravel - Visual Studio Code
File Edit Selection View Go Run Terminal Help

```
15 */
16
17 Route::get('/', function () {
18     return view('welcome');
19 });
20
21 Auth::routes();
22
23 Route::get('/home', [App\Http\Controllers\HomeController::class, 'index']->name('home'));
24
25 Route::resource('kategori',KategoriController::class);
```

Tips saat memanggil controller menggunakan sugestion dari extension **php intelephense** kalo belum di install, install terlebih dulu:

```
Auth::routes();
Route::get('/home', [App\Http\Controllers\HomeController::class, 'index']->name('home'));
Route::resource('kategori',KategoriController::class);
```

kategori
KategoriController::class use App\Http\Controllers\KategoriController::class

Kita akan mencoba memanggil **php artisan route:list** lagi disini kita wajib menjalankan untuk melihat route dan tujuan routnya mau kemana misal :

```
POST      kategori ..... kategori.store >
KategoriController@store
```

Pada bagian depan adalah method yang digunakan sedangkan pada bagian belakang route **KategoriController@store** maksudnya, Jika kita akses <http://127.0.0.1:8000/kategori> dengan method **POST** maka kita akan di arahkan ke **KategoriController.php** pada function **store**

```
ivan@ivan-desktop:~/Project/Laravel/project-laravel$ php artisan route:list
GET|HEAD / ..... ignition/execute-solution ignition.executeSolution > Spatie\LaravelIgnition > ExecuteSoluti...
POST _ignition/health-check ignition.healthCheck > Spatie\LaravelIgnition > HealthCheckController
POST _ignition/update-config ignition.updateConfig > Spatie\LaravelIgnition > UpdateConfigControl...
GET|HEAD api/user .....
GET|HEAD home ..... home > HomeController@index
GET|HEAD kategori ..... kategori.index > KategoriController@index
POST kategori ..... kategori.store > KategoriController@store
GET|HEAD kategori/create ..... kategori.create > KategoriController@create
GET|HEAD kategori/{kategori} ..... kategori.show > KategoriController@show
PUT|PATCH kategori/{kategori} ..... kategori.update > KategoriController@update
DELETE kategori/{kategori} ..... kategori.destroy > KategoriController@destroy
GET|HEAD kategori/{kategori}/edit ..... kategori.edit > KategoriController@edit
GET|HEAD login ..... login > Auth\LoginController@showLoginForm
POST login ..... Auth\LoginController@login
POST logout ..... Auth\LoginController@logout
GET|HEAD password/confirm ..... password.confirm > Auth\ConfirmPasswordController@showConfirmForm
POST password/confirm ..... Auth\ConfirmPasswordController@confirm
POST password/email ..... password.email > Auth\ForgotPasswordController@sendResetLinkEmail
GET|HEAD password/reset ..... password.request > Auth\ForgotPasswordController@showLinkRequestForm
POST password/reset ..... password.update > Auth\ResetPasswordController@reset
GET|HEAD password/reset/{token} ..... password.reset > Auth\ResetPasswordController@showResetForm
GET|HEAD register ..... register > Auth\RegisterController@showRegistrationForm
POST register ..... Auth\RegisterController@register
GET|HEAD sanctum/csrf-cookie ..... Laravel\Sanctum > CsrfCookieController@show
```

Pada bagian ini juga jika ada :

```
PUT|PATCH kategori/{kategori} ..... kategori.update > KategoriController@update
```

Jika ada simbol `{kategori}` berarti kita harus memasukan id ketika menggunakan route tersebut

Misal : <http://127.0.0.1:8000/kategori/8>

Disini saya akan menjelaskan kegunaan masing2 function:

Function	Fungsi
index	Untuk menampilkan halaman utama
store	Untuk menyimpan data dari form tambah data
create	Untuk menampilkan form tambah data
show	Untuk menampilkan data per item
update	Untuk mengupdate data dari form update data
destroy	Untuk menghapus data
edit	Untuk menampilkan form update data

Disini kita tidak akan menggunakan route index (karena halaman utama sudah ada di route /home) dan show (karena kita tidak akan menampilkan item nya satu persatu).

9. Kita menambahkan fillable pada model fungsinya untuk memberitahu model field apa saja yang bisa di isi.

```
protected $fillable=['title','slug'];
```

```

kategori.php app/Models/kategori.php/ kategori
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class kategori extends Model
9  {
10     protected $fillable=['title','slug'];
11     use HasFactory;
12 }

```

Menampilkan data dari database

Karena database masih kosong kamu bisa mengisi nya secara manual lewat phpmyadmin / xampp. Kemudian kita ambil data dari model lalu kita kirim ke view (home), buka file HomeController karena view home di tampilkan dari HomeController. Tambahkan array di dalam view():

```
return view('home',['kategori'=>kategori::all()]);
```

```

HomeController.php app/Http/Controllers/HomeController.php/ HomeController/ index
2
3 namespace App\Http\Controllers;
4
5 use App\Models\kategori;
6 use Illuminate\Http\Request;
7
8 class HomeController extends Controller
9 {
10     /**
11      * Create a new controller instance.
12      *
13      * @return void
14      */
15     public function __construct()
16     {
17         $this->middleware('auth');
18     }
19
20     /**
21      * Show the application dashboard.
22      *
23      * @return \Illuminate\Contracts\Support\Renderable
24      */
25     public function index()
26     {
27         return view('home', ['kategori'=>kategori::all()]);
28     }
29 }

```

Lalu buat tabel di file home.blade.php , di dalam div class= card-body hapus dan buat tabel. Tag yang di kasih warna adalah tag yang harus kalian hafal dan pahami.

Tag <table> untuk membungkus tabelnya

Tag <thead> untuk membungkus bagian kepala tabel

Tag <tbody> untuk membungkus bagian body dari tabel

Tag <tr> (tabel row) untuk membuat baris

Tag <td> (table data) untuk membuat sel

Tag <th> (table head) untuk membuat judul pada header

Kalian buat table nya seperti ini dengan 4 colom dan pada tag table tambahkan class="table"

```
<table class="table">
  <thead>
    <tr>
      <th>id</th>
      <th>title</th>
      <th>slug</th>
      <th colspan="2" class="text-center">action</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td></td>
      <td></td>
      <td></td>
      <td></td>
    </tr>
  </tbody>
</table>
```

Di dalam div card-header hapus isinya lalu tambahkan button untuk insert seperti ini :

```
<a href="/kategori/create" class="btn btn-success">insert</a>
```

```

6 <div class="col-md-8">
7   <div class="card">
8     <div class="card-header">{{ __('Dashboard') }}</div>
9
10    <div class="card-body">
11      <table class="table">
12        <thead>
13          <tr>
14            <th>id</th>
15            <th>title</th>
16            <th>slug</th>
17            <th>action</th>
18          </tr>
19        </thead>
20        <tbody>
21          <tr>
22            <td></td>
23            <td></td>
24            <td></td>
25            <td></td>
26          </tr>
27        </tbody>
28      </table>
29    </div>
30  </div>
31</div>
32</div>
33</div>
34@endsection

```

Kemudian kita akan menggunakan perulangan foreach lebih detail bisa baca di [sini](#). untuk menampilkan data, Kita membuat perulangan di antara tag tr sehingga kita akan mendapatkan baris data sesuai data yang kita punya.

```

@foreach ($kategori as $item)

    <tr>

        <td>{{ $item->id }}</td>

        <td>{{ $item->title }}</td>

        <td>{{ $item->slug }}</td>

        <td>

        </td>

        <td>

        </td>

    </tr>

@endforeach

```

Setelah itu kita tambahkan button di dua tag <td> terakhir.

```

<td>

    <button type="button" class="btn btn-warning">edit</button>

</td>

<td>

    <button type="button" class="btn btn-danger">delete</button>

</td>

```

Kemudian kita buat supaya button ya berfungsi. Karena pada halaman sebelumnya kita sudah menjabarkan function dan juga fungsinya. Untuk edit method nya menggunakan get dan url nya seperti ini /kategori/{kategori}/edit. karena menggunakan get jadi kita bisa menggunakan tag <a> jadi button edit kita ubah menjadi seperti ini:

```
<a href="{{'/kategori/' . $item->id . '/edit'}}" class="btn btn-warning">edit</a>
```

Dan untuk button delete nya menggunakan method delete dan url nya seperti ini /kategori/{kategori} karena menggunakan method delete jadi button harus kita bungkus dengan tag form dan ganti typenya menjadi type="submit" seperti ini :

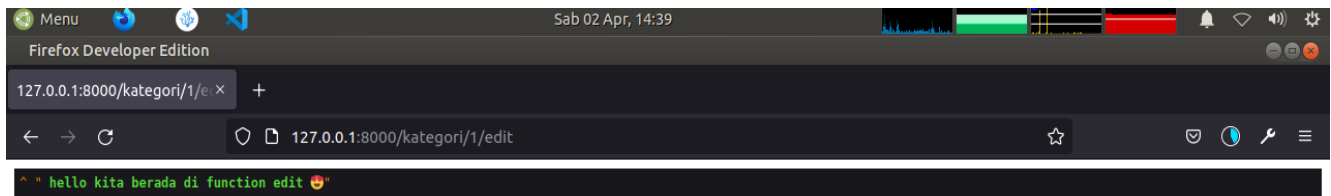
```
<form action="{{'/kategori/' . $item->id}}" method="POST">
    @method('DELETE')
    @csrf
    <button type="submit" class="btn btn-danger">delete</button>
</form>
```

Kita juga menambahkan @method('DELETE') gunanya untuk membuat form nya menggunakan method delete (mungkin ada yang bertanya apakah harus menggunakan method delete jawabannya ya karna harus sesuai dengan yang ada di route list) dan @csrf ini wajib di tambahkan ketika membuat form di laravel kalo tidak, akan muncul peringatan **page expired** .

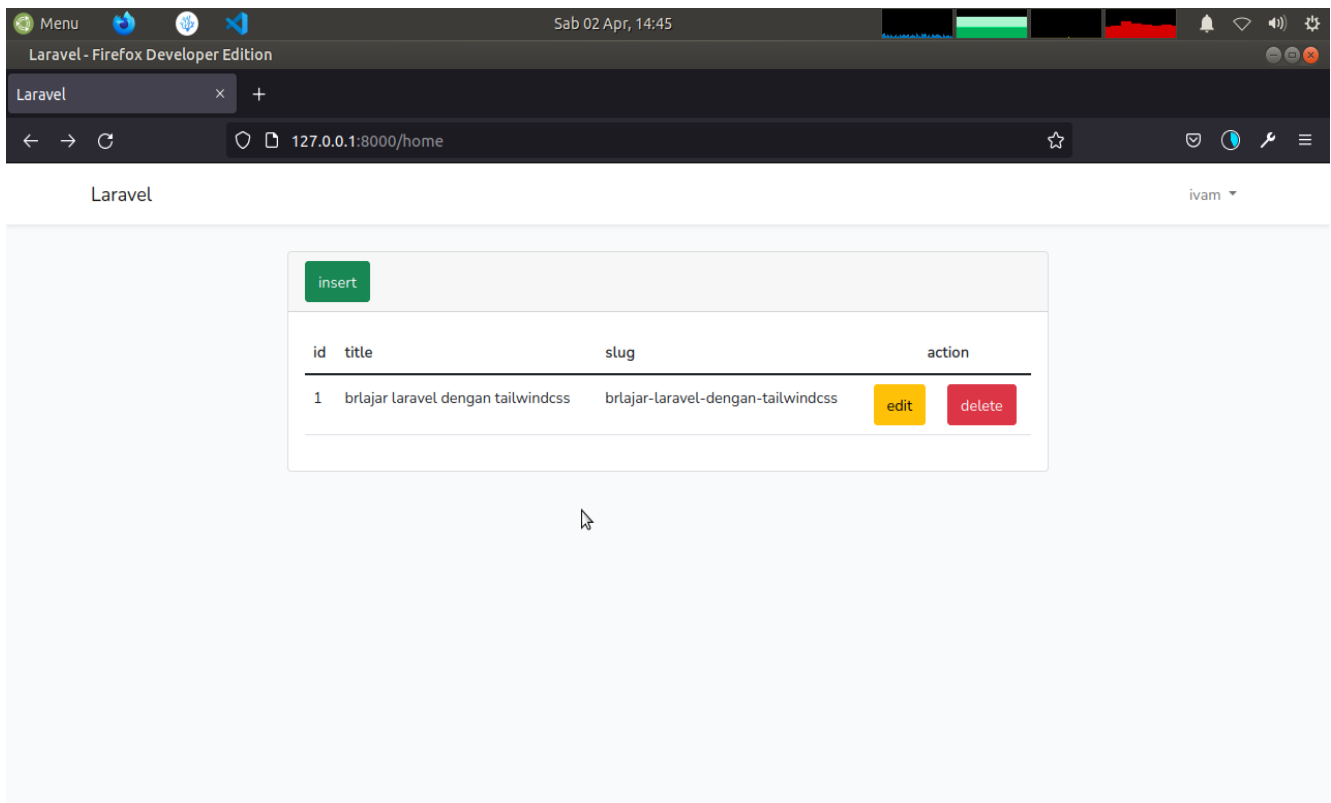
Untuk mengeceknya kita bisa menambahkan dump die / dd() pada tiap function di kategoriController misal:

```
KategoriController.php app/Http/Controllers/KategoriController.php KategoriController
54      * Show the form for editing the specified resource.
55      *
56      * @param \App\Models\kategori $kategori
57      * @return \Illuminate\Http\Response
58      */
59      public function edit(kategori $kategori)
60      {
61          dd(" hello kita berada di function edit 🥰");
62      }
63
64      /**
65       * Update the specified resource in storage.
66       *
67       * @param \Illuminate\Http\Request $request
68       * @param \App\Models\kategori $kategori
69       * @return \Illuminate\Http\Response
70       */
71      public function update(Request $request, kategori $kategori)
72      {
73          dd(" hello kita berada di function update 🥰");
74      }
75
76      /**
77       * Remove the specified resource from storage.
78       *
79       * @param \App\Models\kategori $kategori
80       * @return \Illuminate\Http\Response
81       */
82      public function destroy(kategori $kategori)
83      {
```

Fungsinya adalah supaya kita tau bahwa kita sudah mengases ke function yang benar. Hasilnya menjadi seperti ini jika function edit berjalan dengan benar :



Hasilnya view home akan menjadi seperti ini :



insert data ke database

Pertama buat view baru di folder view dengan nama `form-kategori-insert.blade.php` , lalu tambahkan kode berikut didalamnya :

```
@extends('layouts.app')

@section('content')

<div class="row justify-content-center">

    <div class="col-md-8">

        <div class="card">

            <h3 class="text-center">form insert</h3>

            <div class="card-body">

                <div>

                </div>

            </div>

        </div>

    </div>

</div>

</div>

@endsection
```

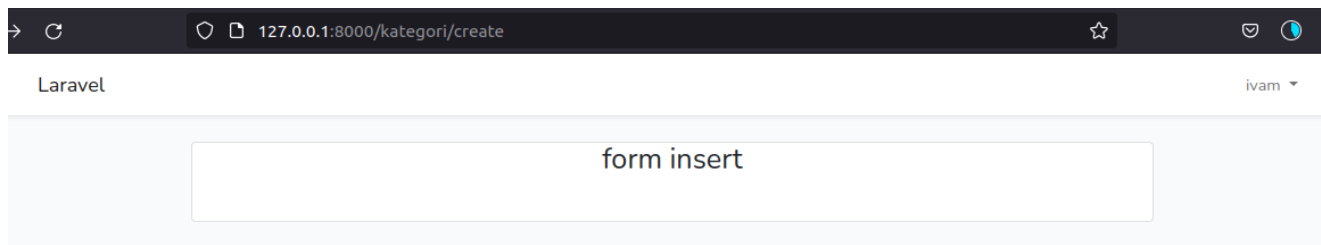
Kemudian kita ke `kategoriController` lalu pada function `create` tambahkan kode:

```
return view('form-kategori-insert');
```

Jika menampilkan `hello kita berada di function create` berarti sudah benar dan perintah `dd()` bisa kita comment .

```
24  */
25  public function create()
26  {
27      // dd(" hello kita berada di function create 🍕");
28      return view('form-kategori-insert');
29  }
30
```

Kita coba jalankan tombol insert yang berada di home page, maka akan tampil seperti ini:



Kemudian kita buat form didalamnya tapi sebelum itu harap pelajari terlebih dahulu tentang form [disini](#).

Kita tambahkan kode ini didalam `div card-body` :

```

<form action="/kategori" method="POST">

    @csrf

    <div class="mb-3">

        <label class="form-label">title</label>

        <input type="text" name="title" class="form-control" required>

    </div>

    <button type="submit" class="btn btn-success"> submit</button>

</form>

```

```

form-kategori-insert.blade.php resources/views/form-kategori-insert.blade.php div.row.justify-content-center div.col-md-8 div.card div.card-body form
1 @extends('layouts.app')
2
3 @section('content')
4 <div class="row justify-content-center">
5     <div class="col-md-8">
6         <div class="card">
7             <h3 class="text-center">form insert</h3>
8             <div class="card-body">
9                 <form action="/kategori" method="POST">
10                     @csrf
11                     <div class="mb-3">
12                         <label class="form-label">title</label>
13                         <input type="text" class="form-control" >
14                     </div>
15                     <button type="submit" class="btn btn-success"> submit</button>
16                 </form>
17             </div>
18         </div>
19     </div>
20 </div>
21 @endsection

```

Pada bagian action kita ingat kembali, apa yang telah kita pelajari sebelumnya. Karena kita akan mengakses function store maka url nya /kategori dengan method post. Jika form di submit muncul text seperti hello anda berada di function store maka method dd() bisa di comment.

Kemudian kita ke kategoriController pada function store tambahkan kode berikut :

```

kategori::create([

    'title'=>$request->title,

    'slug' => Str::slug($request->title,'-')

]);

return redirect('/home');

```

Karena kita menggunakan slug maka tambahkan code ini pada bagian atas file.

```

use Illuminate\Support\Str;

```

Ok saatnya kita melakukan uji coba insert data dan jika kamu mengikuti langkah2 nya dengan benar maka sudah bisa untuk insert data dan kembali ke home page.

Menu Sab 02 Apr, 15:52

Laravel - Firefox Developer Edition

Laravel Belajar HTML #11: Cara Form controls - Bootstrap Helpers - Laravel - The PH

127.0.0.1:8000/home

Laravel ivam

insert

id	title	slug	action	
1	brlajar laravel dengan tailwindcss	brlajar-laravel-dengan-tailwindcss	edit	delete
2	kiuj	kiuj	edit	delete

Update data pada database.

Karena pada saat membuat table kita juga membuat button edit yang sudah terhubung dengan kategoriController pada function edit. Maka dari itu kita sekarang akan membuat **form-kategori-edit.blade.php** untuk isinya kita bisa copy paste dari file form-kategori-insert.blade.php karena sebenarnya file insert dan edit tidak banyak berbeda.

Kemudian kita tampilkan view nya di function edit dan kita kirim data yang mau di edit seperti ini :

```
return view('form-kategori-edit',['edit'=>$kategori]);
```

```
public function edit(kategori $kategori)
{
    // dd(" hello kita berada di function edit 🥰");
    return view('form-kategori-edit',['edit'=>$kategori]);
}
```

Lalu pada bagian view edit kita ubah menjadi seperti ini :

```
<h3 class=" text-center">form edit</h3>

<div class="card-body">

<form action="{{'/kategori/'.$edit->id}}" method="POST">

    @csrf

    @method('PUT')

    <div class="mb-3">

        <label class="form-label">title</label>

        <input type="text" value="{{ $edit->title }}" name="title" class="form-control" required>

    </div>

    <button type="submit" class="btn btn-success"> submit</button>

</form>

</div>
```

Pada dasarnya kita hanya perlu mengubah pada bagian **form action** , **@method('PUT')** dan **attribut value pada input**.

Kemudian kita ubah function update pada kategoriController jadi seperti ini :

```
$kategori->update([

    'title'=>$request->title,

    'slug' => Str::slug($request->title,'-')

]);

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

Pada function update juga mirip dengan function store:


```

public function update(Request $request, kategori $kategori)
{
    // dd("hello kita berada di function update 🥰");
    $kategori->update([
        'title'=>$request->title,
        'slug' => Str::slug($request->title, '-')
    ]);

    return redirect('/home');
}

public function store(Request $request)
{
    // dd("hello kita berada di function create 🥰");
    kategori::create([
        'title'=>$request->title,
        'slug' => Str::slug($request->title, '-')
    ]);

    return redirect('/home');
}

```

Jika sudah web yang kita buat sudah dapat melakukan update.

Delete data pada database.

Sama seperti tombol edit, tombol delete juga sudah kita hubungkan dengan functin destroy yang ada di kategoriController.

```

$kategori->delete();

return back();

```

```

96     public function destroy(kategori $kategori)
97     {
98         // dd("hello kita berada di function destroy 🥰");
99         $kategori->delete();
100         return back();
101     }

```

Dengan ini maka project crud kita telah selesai 🎉.

ok pinter.

Jika ada perasaan yang ingin di sampaikan hubungi 083106253736

Trima Kasih

