# Resource功能 – RESTful路由

php artisan make:controller controller\_name --resource

設置完畢會出現RESTful路由，讓你可以輕鬆地處理 CRUD 操作。

下面是各個動作的功用：

* index - 列出資源的列表頁面
* create - 顯示新增資源的表單頁面
* store - 儲存新增資源的資料
* show - 顯示特定資源的詳細資訊頁面
* edit - 顯示編輯特定資源的表單頁面
* update - 儲存編輯後的資源資料
* destroy - 刪除特定的資源



# web.php使用resource

1. 第一種方式

use App\Http\Controllers\PostsController;

Route::resource('/posts', PostsController::class);

1. 第二種方式

Route::resource('/posts', 'App\Http\Controllers\PostsController');

## 部分路由

* only功用

Route::resource('/posts', PostsController::class, ['only' => [

    'create'

]]);

透過only限定該路徑只執行create()的內容

Route::resource('/posts', PostsController::class)->only(['create']);

另一種用法

* except功用

Route::resource('/posts', PostsController::class, ['except' => [

    'create'

]]);

透過except排除create()，只執行其他功能

Route::resource('/posts', PostsController::class)->except(['create']);

另一種用法

# (Create)簡單操作

@extends('layout.app')

@section('content')

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

        {{csrf\_field()}}

        <input type="text" name="title" placeholder="Enter title">

        <input type="text" name="content" placeholder="Enter content">

        <input type="submit" name="submit">

    </form>

@endsection

@section('footer')

@endsection

建立簡單的Web page => 預設有使用csrf的加密功能，因此要加上csrf\_field()

    public function create()

    {

        //

        return view('posts/create');

    }

透過create取用該頁面

    public function store(Request $request)

    {

        //

        return 'Title :'.$request->title."<br>".'Content :'.$request->content;

    }

store獲取使用者輸入的資料

# (store)儲存資料進資料庫

    use App\Models\Post;

public function store(Request $request)

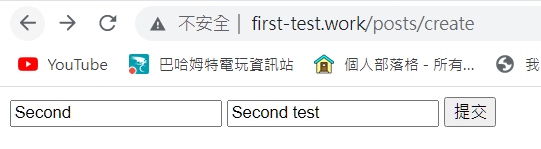
    {

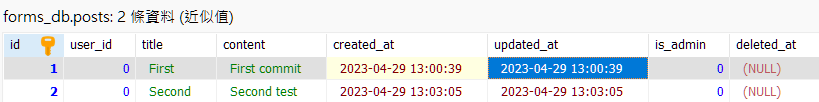
        //

        Post::create($request->all());

    }

引入要儲存的Model，並且將資料存入資料庫中





成功存入!!!!

    public function store(Request $request)

    {

        $post = new Post;

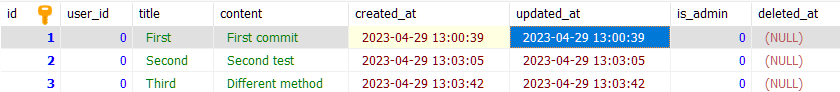
        $post->title = $request->title;

        $post->content = $request->content;

        $post->save();

    }

另一種儲存方法



成功存入

# (index)重新導向

    public function store(Request $request)

    {

        $post = new Post;

        $post->title = $request->title;

        $post->content = $request->content;

        $post->save();

        return redirect('posts');

    }

使用者輸入完資料後，重新導向到posts頁面，而預設會到posts中的index()

    public function index()

    {

        return view('posts.index');

    }

因此設置使用者到posts後會顯示的畫面

# (index)頁面顯示

    public function index()

    {

        $posts = Post::all();

        return view('posts.index', compact('posts'));

    }

將先前存進posts表格中的所有資料給取出，並且compact()傳入字串符(相當於變數)

    public function index()

    {

        $posts = Post::all();

        return view('posts.index', ['posts'=>$posts]);

    }

另一種寫法

    public function index()

    {

        $posts = Post::all();

        return view('posts.index')->with('posts',$posts);

    }

再另一種寫法

@extends('layout.app')

@section('content')

    <ul>

        @foreach ($posts as $post)

            <li>

                <h1>{{$post->title}}</h1>

                <p>{{$post->content}}</p>

            </li>

        @endforeach

    </ul>

@endsection

給blade引擎取用

# (show)顯示資料

    public function show(string $id)

    {

        $post = Post::findOrFail($id);

        return view('posts.show', compact('post'));

    }

$post儲存使用者搜尋並且符合post\_id的文章

透過view傳遞post變數給posts.show的blade做取用

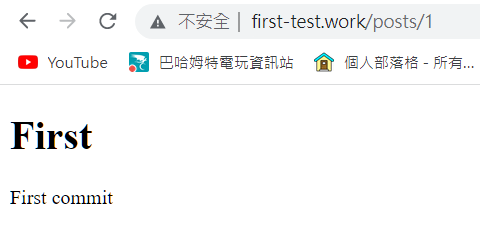
@extends('layout.app')

@section('content')

    <h1>{{$post->title}}</h1>

    <p>{{$post->content}}</p>

@endsection



* **(index->show)透過連結到文章頁面**

@extends('layout.app')

@section('content')

    <ul>

        @foreach ($posts as $post)

            <li>

                <a href="{{route('posts.show', $post->id)}}">

                    <h1>{{$post->title}}</h1>

                </a>

                <p>{{$post->content}}</p>

            </li>

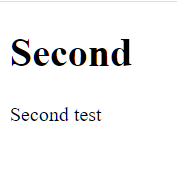
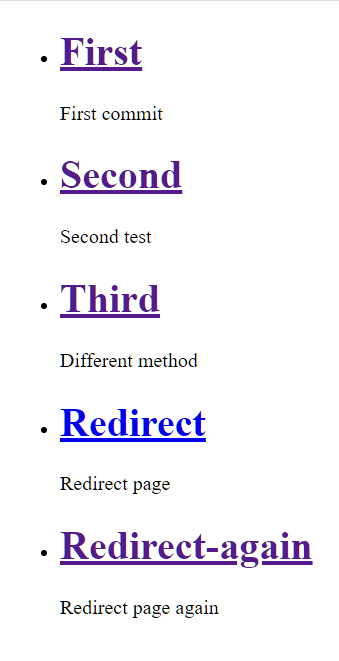
        @endforeach

    </ul>

@endsection

到index的blade設置以上內容，新增<a>超連結

route(‘posts.show’) => 導向到posts.show功能的路徑，並且傳入$post->id => 當點擊某個文章，會將該文章的id傳給posts.show的blade中



點擊後的結果

# (edit->update)更新資料

    public function edit(string $id)

    {

        $post = Post::findOrFail($id);

        return view('posts.edit', compact('post'));

    }

edit的路徑是posts/{$id}/edit

因此$post儲存要編輯的post資料，並且將post資料傳遞給edit的blade

@extends('layout.app')

@section('content')

    <h1>Edit post</h1>

    <form action="/posts/{{$post->id}}" method="POST">

        @csrf

        <input type="hidden" name="\_method" value="PUT">

        <input type="text" name="title" value={{$post->title}}>

        <input type="text" name="content" value={{$post->content}}>

        <input type="submit" name="submit">

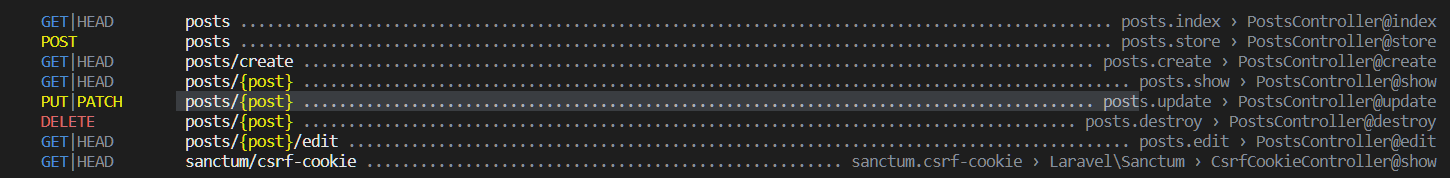
    </form>

@endsection

@section('footer')

@endsection

注意: php artisan route:list顯示，update使用的是PUT和PATCH的功能



因此當使用者編輯好資料後，透過表單的action posts/{{$post->id}}將資料傳遞給update的頁面

接下來要設置當使用者edit完畢後，該筆資料應該要更新，因此到update()

    public function update(Request $request, string $id)

    {

        $post = Post::findOrFail($id);

        $post -> update($request->all());

        return redirect('posts');

    }

先查找該筆資料，並且將使用者先前輸入的資料，更新到資料庫中，最後重新導向到/posts頁面

# (show)增加導向連結

@extends('layout.app')

@section('content')

    <h1><a href="{{route('posts.edit', $post->id)}}">{{$post->title}}</a></h1>

    <p>{{$post->content}}</p>

@endsection

當使用者點擊h1時，會重新導向到/posts/{$id}/edit的頁面

* **(destroy)刪除資料**

    public function destroy(string $id)

    {

        $post = Post::findOrFail($id);

        $post -> delete();

        return redirect('posts');

    }

當使用者觸發到該連結時，會刪除資料庫符合ID的資料

@section('content')

    <h1>Edit post</h1>

    <form action="/posts/{{$post->id}}" method="POST">

        @csrf

        <input type="hidden" name="\_method" value="PUT">

        <input type="text" name="title" value={{$post->title}}>

        <input type="text" name="content" value={{$post->content}}>

        <input type="submit" name="submit" value="UPDATE">

    </form>

    <form action="/posts/{{$post->id}}" method="POST">

        @csrf

        <input type="hidden" name="\_method" value="DELETE">

        <input type="submit" value="DELETE">

    </form>

@endsection

因此在posts.edit的blade中，新增一個form為delete用的表單