Laravel安裝：於安裝目錄輸入指令 composer create-project laravel/laravel **laravel 5.1.\*** --prefer-dist (粗體為**創建目錄** **版本號**)

一、路由控制器與視圖

|  |
| --- |
| 1.路由設置 (app/Http/routes.php)  //回傳文字  Route::get('test',function () {return 'test'; }); //test目錄，回傳文字  //回傳視圖 (/表示根目錄)  Route::get('/', function () {return view('welcome'); }); //根目錄，回傳視圖  //回傳控制器class getIndex (隱式控制路由)  Route::controller('home','Home'); //home目錄，控制器Home.php的class getIndex  //回傳控制器指定類別 (帶子目錄)  Route::get('home/{name}/{id?}','Home@index'); //home目錄，控制器Home.php的class index  //帶子目錄 name(必要) id?(非必要+?)  //請求方法控制  Route::any('home', 'Home@index'); //任何請求方法  Route::match(['get','post'], 'home', 'Home@index'); //複數請求方法 |
| 2.控制器設置 (app/Http/controllers/Home.php (文件名自定))  <?php  namespace App\Http\Controllers;  use Illuminate\Routing\Controller as BaseController;  class Home extends BaseController  {  public function index() //控制器Home.php的class index  {  $data = ['a'=>'顯示資料']; //view只接受組合陣列，會使用extract拆開陣列  return view('Home', $data); //回傳視圖  }  } |
| 3.視圖基本設置 (resources/views/Home.blade.php (Home為呼叫名，必須加.blade))  <html>  全是html語法，{{$test}} <--相當於echo $test，變數的顯示  全部的邏輯語法皆需要在前面加老鼠@，且不能敘述大括號，要用end做結尾  例如： @foreach ($test = $key=>$value)  變數{{$key}}=>{{$value }} <br>  **@endforeach**  </html> |
| 4.繼承視圖布局  概念：主架構**布局**文件(包括<header><main><footer>)、**子視**圖文件(僅main裡的section之類內容)   1. 任意引入視圖： @include('視圖呼叫名') 2. 子視繼承布局： @extends('布局呼叫名') 3. 布局呼叫子視區塊： @yield('子視區塊名') 4. 布局子視交叉區塊： (由子視覺定整體區塊排版)  |  |  | | --- | --- | | **布局：**  @section('同區塊名')  布局區塊內容  @show <!—布局區塊結束--> | **子視：**  @section('同區塊名')  子視區塊內容  @parent <!—布局區塊內容-->  子視區塊內容  @stop <!—子視區塊結束，stop同endsection--> |  1. 子視區塊格式： @extends('布局呼叫名') @section 子視為數個區塊所組成，透過布局安插 @stop |

二、服務的創建與使用

|  |
| --- |
| 1.服務創造：app下創造服務資料夾/服務文件 (EX App\ServiceTest\GeneralService.php)  <?php  namespace App\ServiceTest;  class GeneralService{} |
| 2-1.提供者綁定服務：服務資料夾下創造提供綁定文件 (EX App\ServiceTest\TestServiceProvider.php)  <?php  namespace App\ServiceTest;  use Illuminate\Support\ServiceProvider;  class TestServiceProvider extends ServiceProvider  {  protected $defer = true; //緩載，讓沒用到的服務不會載入  public function boot(){  //boot方法會在所有提供者的register完成後才註冊  }  public function register()  {  $this->app->bind(GeneralService::class,function($app){  return new GeneralService("111"); }); //普通回調函數綁定  $this->app->singleton(SingleService::class,function($app){  return new SingleService(); }); //單例回調函數綁定  $instance = new InstanceService();  $this->app->instance('instanceService',$instance); //實例對象服務綁定  }  public function provides() //緩載時要提供所有的註冊名  {  return [  GeneralService::class,  SingleService::class,  'instanceService'  ];  }  } |
| 2-2.註冊服務提供者：在config/app.php中的'providers'進行註冊  App\ServiceTest\TestServiceProvider::class, |
| 3.控制器載入服務：控制器文件函數的初始參數中設置  public function \_\_construct (\App\ServiceTest\GeneralService $generalService){} |
| 2-補充.在bootstrap/app.php中，將服務綁至服務容器。(非服務提供者)  $app->bind(App\ServiceTest\GeneralService::class,function($app){  return new App\ServiceTest\GeneralService(); }); //普通回調函數綁定  $app->singleton(App\ServiceTest\SingleService::class,function($app){  return new App\ServiceTest\SingleService(); }); //單例回調函數綁定  $instance = new App\ServiceTest\InstanceService();  $app->instance('instanceService',$instance); //實例對象服務綁定 |

三、表單使用

|  |
| --- |
| 1.路由設置 (app/Http/routes.php)  Route::match(['get','post'],'welcome','WelcomeController@index'); //將收到表單資料的文件修改為POST呼叫 |
| 2.表單發送 (控制器+視圖)  use Illuminate\Http\Request; //獲取請求函數  ..class..  public function \_\_construct (Request $request){} //從函數初始載入中獲取接收請求  $request->session()->save(); //記錄session狀態  <!--所有表單資料欄位都必須有name，表單用post發送-->  ..form..  <input type="hidden" name="\_token" value="{{ csrf\_token() }}"> <!--post必須要加入csrf驗正參數--> |
| 3.控制器接收請求 (控制器)  use Illuminate\Http\Request; //獲取請求函數  ..class..  public function \_\_construct (Request $request) //從函數初始載入中獲取接收請求  {  //表單內容資料  **$request->all(); //全部資料(含post、get)**  $request->input(); //檔案類型以外全部資料(含post、get)  $request->query(); //僅get全部資料  $request->input('name'); //單一名稱資料  $request->only('name1', 'name2'); //複數名稱資料  $request->except('name1', 'name2'); //除此之外(複數)名稱資料  //其他資料  $request->method(); //方法  $request->url(); //網址(不含#或get參數)  $request->fullUrl(); //完整網址(含#或get參數)  $request->path(); //網址僅路徑(不含網域、#或get參數)  } |
| 4.接收請求session一次存儲 (控制器)  use Illuminate\Http\Request; //獲取請求函數  //存儲方法  **$request->flash(); //所有資料存儲**  $request->flashOnly('name1', 'name2'); //(複數)名稱資料存儲  $request->flashExcept('name1', 'name2'); //除此之外(複數)名稱資料存儲  //獲取方法  **$request->old(); //所有資料獲取**  $request->old('name'); //單一資料獲取 |

四、模組資料庫使用：僅資料表數據管理 (Laravel存取的資料表強制必須要有一個AI的id)

|  |
| --- |
| 1.資料庫登入資訊：根目錄 .env 檔案 |
| 2.創造資料表模組：app/Models/Student.php (默認檔案名稱隨資料表名稱，檔名首小寫+s = 資料表名(students)，一表一模組)  <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Model;  class Student extends Model  {  protected $table = 'students'; //自定表名 (若不隨默認表名對應設置)  public $timestamps = false; //關閉自動時間戳，時間戳為表內的DATETIME格式created\_at、updated\_at參數  protected $fillable = ['name', 'age']; //允許大量賦值 (迴圈更新資料必須開啟)  protected $guarded = ['name', 'age']; //拒絕大量賦值  } |
| 3.讀取資料 (控制器，若將讀取資料函數寫在模組，則把Student::改成$this->)  <?php  use App\Models\Student; //讀取資料表模組  ..class..  $student = Student::all(); //讀取所有數據 (**陣列**)  $student = Student::find(1); //讀取id=1資料  $student = Student::where('id','=','1')->get(); //讀取條件數據 (**陣列**) (當有first lists等最後位時，get會被取代)  $student = Student::whereRaw("(age >= 0) or (id > 200)")->get(); //讀取字串條件數據 (**陣列**)  //**陣列**可在最後加入 ->take(數量) ，限制提取之最大數量  //**陣列**可在最後加入 ->count() ，將會回傳資料筆數  //**陣列**可在最後加入 ->skip(數量) ，將會忽略資料  //**陣列**可在最後加入 ->first() ，將回傳第一筆資料  //可在最後加入　　 ->lists(V欄,K欄)，將會把K欄當鍵做組合陣列  //可在中間加入　　 ->select('a', 'b') ，將會只取a b兩欄的值  //可在中間加入　　 ->distinct() ，將會抓取不重複的值  //可在中加加入　　 ->orderBy('name', 'desc')，或是asc進行排序  //以下是屬性獲取方法  $data = $student->toArray(); //將資料以組合陣列顯示 |
| 4.寫入資料  <?php  //插入數據  $student = new Student;  $student->name = '小企鵝';  $student->age = '28';  $student->save(); //儲存插入法  Student::create(['name'=>'John' , 'age'=>'30']); //創造插入法  $Student = Student::firstOrCreate(['name' => 'John']); //創造插入法，若重複則不插入，回傳第一筆發現重複的資料  //修改數據  $user->touch(); //僅更新時間戳  $student = Student::find(1);  $student->name = 'john';  $student->save(); //儲存修改法  student::find(3)->update(['name' => 'hahahaha']); //更新修改法 find  student::where('age','>','10')->update(['name' => 'aloha']); //更新修改法 where  student::whereRaw("name = 'aloha'")->update(['name' => 'kkr']); //更新修改法 whereRaw    //刪除數據  $student = Student::find(1); $student->delete(); //刪除id=1資料  User::destroy(1); User::destroy([1, 2, 3]); User::destroy(1, 2, 3); //刪除id符合資料  student::where('age','>','10')->delete(); //刪除條件數據  student::whereRaw("name = 'aloha'")->delete(); //刪除字串條件數據 |

|  |
| --- |
| 5.高級語法  //join語法  DB::table('users')  ->join('contacts', function($join)  {  $join->on('users.id', '=', 'contacts.user\_id')  ->where('contacts.user\_id', '>', 5) ->where('contacts.user\_id', '>', 5);  })  ->get(); |
|  |
|  |
|  |

五、Session的使用

|  |
| --- |
| 1.Session啟用 (控制器)  use Illuminate\Http\Request; //獲取請求  ..class..  public function index(Request $request) {... //從函數的初始參數中載入請求 |
| 2.Session讀取 (控制器、視圖)  $session = $request->session()->all(); //讀取所有值  $session = $request->session()->get('\_token'); //讀取單一值  $session = session('\_token'); //讀取單一值  $session = $request->session()->has('\_token '); //是否有此鍵 |
| 3.Session修改 (控制器)  $request->session()->put('name','小芳'); //依鍵的值新增或修改 (值)  $request->session()->push('stack', '123'); //依鍵創造或新增陣列 (陣列)  $request->session()->forget('name'); //依鍵刪除  $value = $request->session()->pull('name', 'default'); //依鍵刪除，同時取出刪除資料 (若無刪除資料，回傳'default')  $request->session()->flush(); //清空Session  $request->session()->regenerate(); //重置整個Session ID (安全性考量)  $request->session()->save(); //\*\*必要\*\* 所有修改執行後必須儲存 |

零、其他功能使用

|  |
| --- |
| 1.其它功能函數  dd(變數); //return顯示變數的結構 |
| 2.重導向函數  use Illuminate\Http\RedirectResponse;  ..class..  redirect()->action('HomeController@index');  redirect('http://www.google.com'); |
|  |