# Laravel5.4

出自：兄弟连IT教育

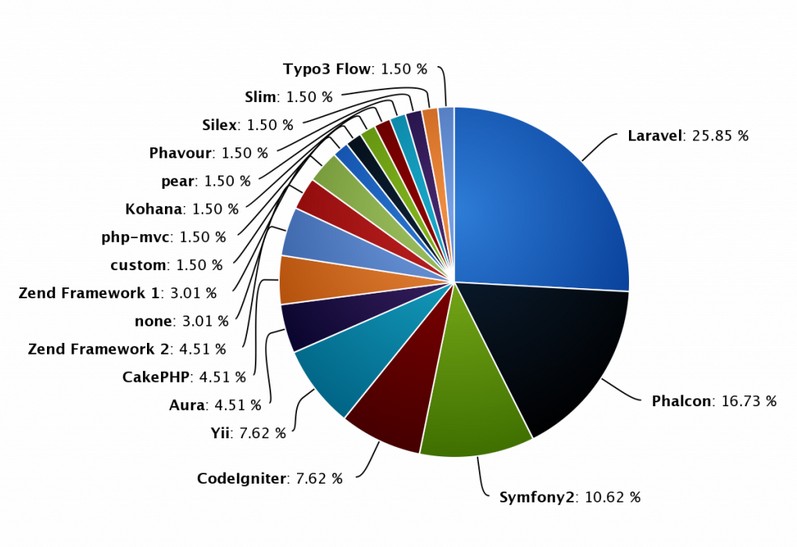
转载请注明出处注意:\*\* 代表重点

**第一章 Laravel框架介绍及安装 目录结构\*\***

* 1. **框架定义\*\***

框架是可以快速实现项目需求的工具

* 1. **框架排名**



* 1. **Laravel框架安装\*\***
     1. 运行环境要求

(1)PHP 版本 >= 5.6.4 windows 版本的wamp 默认开启所有扩展 linux 默认(3)没开

(2)Mcrypt PHP 扩展

php的加密扩展,提供多种加密算法(3)openssl扩展

对传输的数据进行加密

1. mbstring 扩展

mbstring 提供了针对多字节字符串的函数，能够帮你处理 PHP 中的多字节编码

1. Tokenizer PHP 扩展

能够对php代码进行分割

1. XML扩展

xml是一种可扩展的标记性语言,开启xml的目的是传输数据以及存储数据

* + 1. composer安装

composer create-project --prefer-dist laravel/laravel 项目名字

* 1. **配置虚拟主机 模仿工作时候的效果** 
     1. apache ->httpd.conf 开启配置信息

Include conf/extra/httpd-vhosts.conf（php7默认支持虚拟主机）

* + 1. 在httpd-vhosts.conf配置文件(C:\wamp\bin\apache\apache2.4.18\conf\extra)下 配置根目录 服务器名字

1. <VirtualHost \*:80>
2. ServerAdmin chenxiaoyu@lampbrother.net
3. DocumentRoot "C:\wamp\www\demo42\project\public"
4. ServerName www.php42.com
5. ErrorLog "logs/www.php42.com-error.log"
6. CustomLog "logs/www.php42.com-access.log" common
7. </VirtualHost>
   * 1. 解析域名

Windows-》System32-》drivers-》etc-》hosts文件下写入:

127.0.0.1 www.php42.com

* + 1. 重启apache 服务
    2. 测试

cmd里执行命令:ping lamp.com

* 1. **Laravel框架目录结构\*\***
     1. 主要目录结构\*\*

入口文件:project/public/index.php 路由文件:project/routes/web.php

中间件目录 过滤:

project/app/http/Middleware

控制器目录:project/app/http/Controllers 视图目录:project/resources/views

配置文件:project/conﬁg app.php 核心配置文件

database.php 数据库配置文件cache.php 缓存配置文件mail.php 邮件配置文件view.php 视图配置文件session.php session配置文件

根目录:public

js img 静态资源

index.php 入口文件

第三方资源临时目录:project/storage

第三方扩展类库目录 project/vendor composer 安装的三方类 例如:验证码三方类快速配置文件:project/.env 配置数据库连接信息 发送的邮件配置信息

artisan 工具匠 创建控制器 中间件...工具

安装工具记录信息文件 project/composer.json

* + 1. 总体目录结构

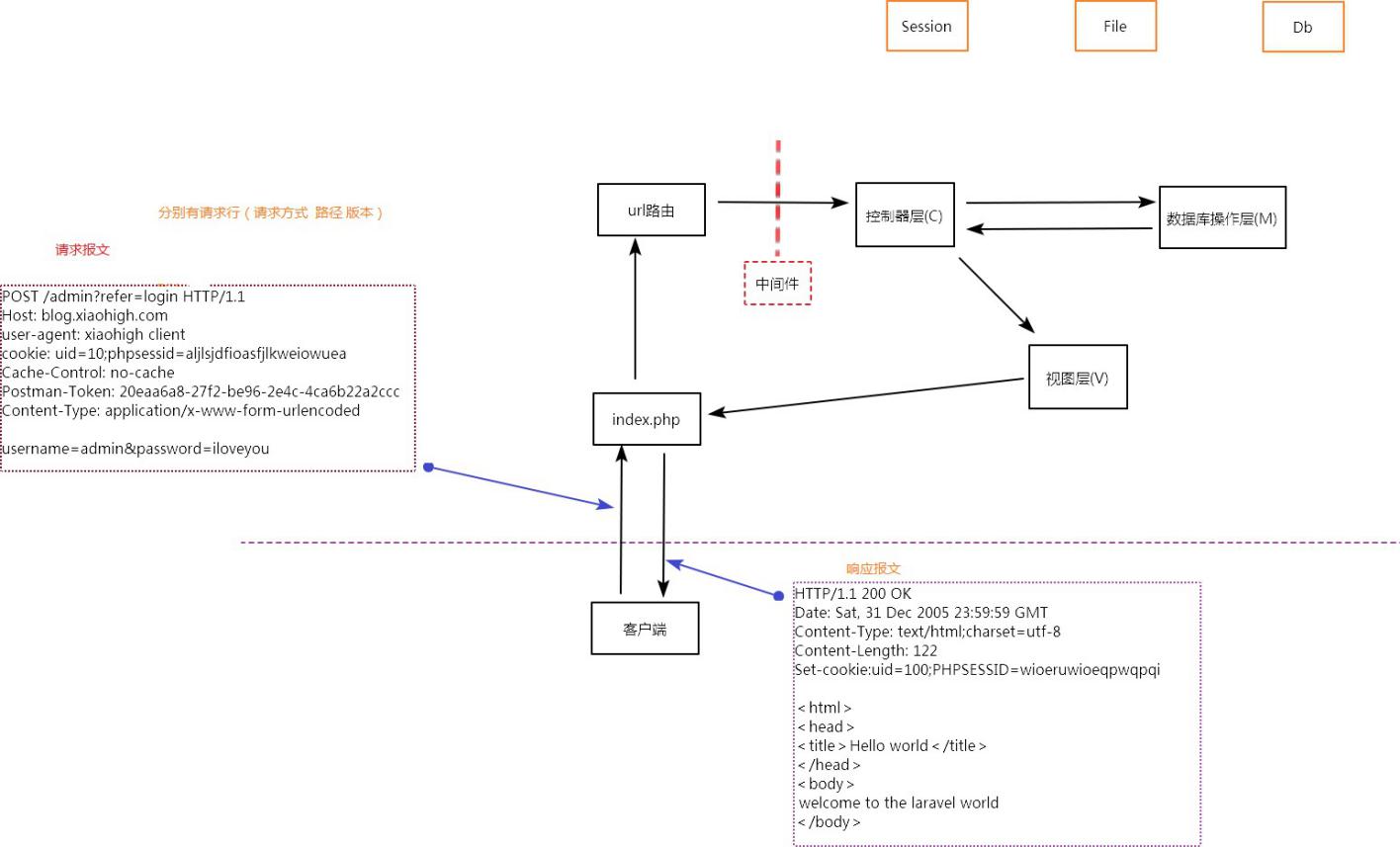
1. 目录或文件 说明
2. ｜– app 包含Controller、Model、路由等在内的应用目录，大部分业务将在该目录下进行
3. ｜ ｜– Console 命令行程序目录
4. ｜ ｜ ｜– Kernel.php 命令调用内核文件，包含commands变量(命令清单，自定义的命令需加入到这里) 和schedule方法(用于任务调度，即定时任务)
5. ｜ ｜– Exceptions 包含了自定义错误和异常处理类
6. ｜ ｜– Handlers 事件处理目录
7. ｜ ｜– Http HTTP传输层相关的类目录
8. ｜ ｜ ｜– Controllers 控制器目录
9. ｜ ｜ ｜– Kernel.php 包含http中间件和路由中间件的内核文件
10. ｜ ｜ ｜– Middleware 中间件目录
11. ｜ ｜– Jobs 该目录下包含队列的任务类
12. ｜ ｜– Providers 服务提供者目录
13. ｜ ｜– User.php 自带的模型实例，我们新建的Model默认也存储在该目录
14. ｜– bootstrap 框架启动载入目录
15. ｜ ｜– app.php 创建框架应用实例
16. ｜ ｜– cache 存放框架启动缓存，web服务器需要有该目录的写入权限
17. ｜– composer.json 存放依赖关系的文件
18. ｜– composer.lock 锁文件，存放安装时依赖包的真实版本
19. ｜– config 各种配置文件的目录
20. ｜ ｜– app.php 系统级配置文件
21. ｜ ｜– auth.php 用户身份认证配置文件，指定好table和model就可以很方便地用身份认证功能了
22. ｜ ｜– broadcasting.php 事件广播配置文件
23. ｜ ｜– cache.php 缓存配置文件
24. ｜ ｜– database.php 数据库配置文件
25. ｜ ｜– filesystems.php 文件系统配置文件，这里可以配置云存储参数
26. ｜ ｜– mail.php 电子邮件配置文件
27. ｜ ｜– queue.php 消息队列配置文件
28. ｜ ｜– services.php 可存放第三方服务的配置信息
29. ｜ ｜– session.php 配置session的存储方式、生命周期等信息
30. ｜ ｜– view.php 模板文件配置文件，包含模板目录和编译目录等
31. ｜– database 数据库相关目录
32. ｜ ｜– factories 工厂类目录，也是用于数据填充
33. ｜ ｜ ｜– UserFactory.php 在该文件可定义不同Model所需填充的数据类型
34. ｜ ｜– migrations 存储数据库迁移文件
35. ｜ ｜– seeds 存放数据填充类的目录
36. ｜ ｜– DatabaseSeeder.php 执行php artisan db:seed命令将会调用该类的run方法。该方法可调用执行该目录下其他Seeder类，也可调用factories方法生成ModelFactory里定义的数据模型
37. ｜– public 网站入口，应当将ip或域名指向该目录而不是根目录。可供外部访问的css、js和图片等资源皆放 置于此
38. ｜ ｜– index.php 入口文件
39. ｜ ｜– .htaccess Apache服务器用该文件重写URL
40. ｜ ｜– web.config IIS服务器用该文件重写URL
41. ｜– resources 资源文件目录
42. ｜ ｜– assets 可存放包含LESS、SASS、CoffeeScript在内的原始资源文件
43. ｜ ｜– lang 本地化文件目录
44. ｜ ｜– views 视图文件就放在这啦
45. | - routes 路由文件
46. |- web.php 路由文件
47. ｜– storage 存储目录。web服务器需要有该目录及所有子目录的写入权限
48. ｜ ｜– app 可用于存储应用程序所需的一些文件？待补充
49. ｜ ｜– framework 该目录下包括缓存、sessions和编译后的视图文件
50. ｜ ｜– logs 日 志 目 录
51. ｜– tests 测试目录
52. ｜– vendor 该目录下包含Laravel源代码和第三方依赖包

53

1. ｜– .env 环境配置文件。config目录下的配置文件会使用该文件里面的参数，不同生产环境使用不同的.env文件即可。
2. ｜– .env.example 样例文件
3. ｜– .gitattribute 用于设置文件的对比方式（常用非文本文件）
4. ｜– .gitigonre 你想要忽略的文件或者目录
5. ｜– artisan 强大的命令行接口，你可以在app/Console/Commands下编写自定义命令
6. ｜– package.json gulp配置文件
7. ｜– phpunit.xml phpunit（一种PHP测试框架）配置文件
8. ｜– server.php PHP内置的Web服务器将把这个文件作为入口。以public/index.php为入口的可以忽略掉该文 件

62

* 1. **Laravel框架访问原理\*\***



**第二章配置 路由 中间件\*\***

* 1. **配置\*\***
     1. 开发前必须要做的
        1. 修改时区

conﬁg/app.php >>>>> 'timezone' => 'PRC',

* + - 1. storage 和 vendor 目录要让服务器有写入的权限,windows下没有问题,linux下需要给权限 chmod -R 777 (3)程序秘钥

<1 程序秘钥 目录在.env文件里

APP\_KEY=X2MaUIfj8wWi8njF1rC4xkh1NRyAqFkm

<2 作用

安全秘钥 别人不能够随随便便 侵入到laravel 拿不到的密码 即使拿到加密密码 也拿不到明文密码模式,提供加密算法

<3 程序秘钥生成（注意用git拉取的laravel程序秘钥文件默认没有）

php artisan key:generate (4)artisan是什么?

其实就是一个工具匠,其实就是php文件，里面有一堆php文件，在命令行下用php命令实现php文件，实现功能,例如：创建控制器 请求类 中间件 生成程序秘钥

* + 1. 开发过程中要用到的
       1. 读取和设置配置

<1 读取配置

Conﬁg::get('app.timezone');

<2 设置配置

Conﬁg::set('app.locale', 'cn'); (2)获取环境配置

env('DB\_HOST')

1. 关闭和启动应用

<1 关闭应用

php artisan down

模板配置 resources/views/errors/503.blade.php

写入代码

1

2

3

4

5

6

7

8

9

10

11

<!doctype html>

<html lang="{{ app()->getLocale() }}">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>404 not found</title>

<!-- Fonts -->

<link href="https://fonts.googleapis.com/css?family=Raleway:100,600" rel="stylesheet" type="text/css">

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

<!-- Styles -->

<style>

html, body {

background-color: #fff; color: #636b6f;

font-family: 'Raleway', sans-serif; font-weight: 100;

height: 100vh; margin: 0;

}

.full-height {

height: 100vh;

}

.flex-center {

29

30

31

32 }

33

align-items: center; display: flex;

justify-content: center;

* 1. .position-ref {
  2. position: relative; 36 }

37

1. .top-right {
2. position: absolute;
3. right: 10px;
4. top: 18px;

42 }

43

1. .content {
2. text-align: center; 46 }

47

1. .title {
2. font-size: 84px;

50 }

51

1. .links > a {
2. color: #636b6f;
3. padding: 0 25px;
4. font-size: 12px;
5. font-weight: 600;
6. letter-spacing: .1rem;
7. text-decoration: none;
8. text-transform: uppercase; 60 }

61

1. .m-b-md {
2. margin-bottom: 30px; 64 }
3. </style>
4. </head>
5. <body>
6. <center>

69 <h1>404 not founded...</h1>

70 </center> 71

1. </body>
2. </html> 74

<2 启动应用

php artisan up

1. URL重写

<1 Url 重写的目的是提高网站的收录排名 增加用户的点击量 SEO

<2 在laravel里可以把index.php入口文件里直接重写掉

<3 apache

在网站根目录下创建.htaccess文件 Options +FollowSymLinks RewriteEngine On RewriteCond %

{REQUEST\_FILENAME} !-d RewriteCond %{REQUEST\_FILENAME} !-f RewriteRule ^ index.php [L]

<4 nginx

在ngnix配置文件中添加 location / { try\_ﬁles $uri $uri/ /index.php?$query\_string; }

* 1. **路由\*\***

文件位置:routes/web.php

* + 1. **路由定义**

当一个请求访问控制器里面的方法的时候,我们的路由充当了一个指导的角色,指导的请求方式( get post)

* + 1. **基本路由**
       1. 路由get

Route::get('/love', function(){echo '这是get路由'}) //love 是路由规则,get是请求方式,只要请求路径匹配到路由规则,就会自动调用匿名函数

实例代码:

1. //路由基本使用
2. Route::get("/index",function(){
3. echo "这是路由的基本使用"; 4 });
   * + 1. 路由post

Route::post('/insert', function(){}) //csrf 学完毕后再看这个,因为浏览器里默认的是get请求

* + 1. **带参数的路由**
       1. 普通使用

Route::get('/goods/{id}', function($id){ echo $id; });

* + - 1. 限制参数类型

Route::get('/param/{name}', function($name){ return $name; })->where('name','[a-z]+');

* + 1. **传递多个参数**

1. //传递多个参数路由
2. Route::get("/user/{name}-{id}",function($name,$id){
3. echo $name.":".$id; 4 });
   * 1. **路由别名**
        1. 路由别名设置
4. //路由别名
5. Route::get("/home/person",['as'=>'person',function(){
6. echo "这是前台的个人中心页面"; 4 }]);

5

* + - 1. 通过路由别名获取到原有的路由规则

1. //通过路由别名获取到原有的路由规则
2. Route::get("/a",function(){
3. //route 路由函数 通过别名获取路由规则
4. echo route('person'); 5 });
   * 1. **路由组设置**
5. //路由组
6. Route::group([],function(){
7. //子路由
8. Route::get("/b",function(){
9. echo "这是子路由1"; 6 });
10. //子路由2
11. Route::get("/c",function(){
12. echo "这是子路由2"; 10 });

11 });

* + 1. **404页面设置**
       1. 当匹配一个不存在的路由规则时,会触发404页面
       2. 404页面设置

模板里设置404页面，模板位置:resources/views/errors/404.blade.php

* + 1. **CSRF保护**
       1. 什么是CSRF?

跨站请求伪造（英语：Cross-site request forgery），也被称为one-click attack或者session riding，通常缩写为CSRF或者XSRF， 是一种挟制用户在当前已登录的Web应用程序上执行非本意的操作的攻击方法，一旦有了跨站伪造请求攻击,数据时相当不安全的

* + - 1. CSRF保护(防止跨站伪造请求攻击) CSRF（跨站请求伪造）式攻击。

Laravel框架里在表单里防止跨站伪造请求攻击,内置辅助函数:{{csrf\_ﬁeld()}}，防止跨站伪造请求攻击，保护 数据安全,确认请求的确来自于客户,而不是恶意的第三方

实例演示

路由文件代码

* + - * 1. 加载表单

在routes/web.php里写入如下代码

1. //加载form表单
2. Route::get("/add",function(){
3. return view("form"); 4 });
   * + - 1. 在resources/views 下创建 form.blade.php 写入如下代码

</form>

</body>

</html>

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>form</title>

</head>

<body>

<form action="/insert" method="post">

用户名:<input type="text" name="name"><br>

密码:<input type="password" name="password"><br>

{{csrf\_field()}}

<input type="submit" value="提交">

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

* + - * 1. 在routes/web.php里写入如下代码

1. //post路由 MethodNotAllowedHttpException 没有匹配到路由规则
2. Route::post("/insert",function(){
3. echo "this is insert"; 4 });
   * + 1. Ajax里CSRF保护
4. 如果是普通的Ajax get请求,不需要加csrf保护,和我们平时使用的Ajax get方式一样代码演示
   1. 在routes/web.php 里写入路由代码
5. //Ajax 处理get请求
6. Route::get("/ajax",function(){
7. return view("ajax"); 4 });
   1. 在resources/views 下创建ajax.blade.php 模板
8. <!doctype html>
9. <html>
10. <head>
11. <meta charset="utf-8">
12. <title>Ajax get</title>
13. <script type="text/javascript" src="/static/js/jquery-1.8.3.min.js"></script>
14. </head>
15. <body>
16. <button>获取Ajax响应数据</button>
17. </body>
18. <script type="text/javascript"> 12 // alert($);
19. //获取按钮 表单单击事件
20. $("button").click(function(){
21. //普通get请求
22. // $.get("/doajax",{},function(data){
23. // alert(data);

18 // });

19 });

1. </script>
2. </html>
   1. 在routes/web.php 里写入路由代码
3. //Ajax 处 理 get
4. Route::get("/doajax",function(){ 3 echo "666";

4 });

1. 如果是Ajax post 请求,需要加CSRF保护
   1. csrf保护方法

<1 先去用meta存储token字符串

<meta name="csrf-token" content="{{ csrf\_token() }}">

<2 用jquery类库把token字符串写入请求标头里

$.ajaxSetup({ headers: {

'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')

}

});

* 1. 代码演示

<1 在routes/web.php 里写入路由代码

1. //Ajax 处理post
2. Route::get("/ajaxs",function(){
3. return view("ajaxs"); 4 });

<2 在resources/views 下创建ajaxs.blade.php 模板

</body>

<script type="text/javascript">

//用jquery类库将token字符串写入到请求的标头里

$.ajaxSetup({ headers: {

'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')

}

});

// alert($);

//获取按钮 表单单击事件

$("button").click(function(){

//普通post请求

$.post("/doajaxs",{},function(data){ alert(data);

});

});

</script>

</html>

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<meta name="csrf-token" content="{{csrf\_token()}}">

<title>Ajax post</title>

<script type="text/javascript" src="/static/js/jquery-1.8.3.min.js"></script>

</head>

<body>

<button>post获取Ajax响应数据</button>

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

<3 在routes/web.php里写入如下代码

1. //Ajax 处理post
2. Route::post("/doajaxs",function(){
3. echo "this is post Ajax response"; 4 });
   1. **中间件\*\***
      1. **中间件目录**

app/http/middleware

* + 1. **中间件作用**

筛子 起到一个过滤的作用 主要做的是权限操作处理

* + 1. **中间件使用**
       1. 创建中间件

打开cmd黑窗口,找到自己的项目,执行命令:php artisan make:middleware LoginMiddleware

* + - 1. 代码操作处理

操作app/http/middleware/LoginMiddleware 类

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

<?php

namespace App\Http\Middleware;

use Closure;

class LoginMiddleware

{

/\*\*

* Handle an incoming request.

\*

* @param \Illuminate\Http\Request $request
* @param \Closure $next
* @return mixed

\*/

public function handle($request, Closure $next)

{

// $request 请求报文

//检测用户是否具有登录的session

if($request->session()->has('id')){ return $next($request);

}else{

//跳转到登录界面

return redirect("/login");

}

26

27 }

28 }

29

* + - 1. 路由注册

在kernel.php（app/http/kernel.php）添加如下代码:

'login' => \App\Http\Middleware\LoginMiddleware::class, //其中login是中间件规则如下:

1 <?php 2

3 namespace App\Http; 4

5 use Illuminate\Foundation\Http\Kernel as HttpKernel; 6

7 class Kernel extends HttpKernel 8 {

9 /\*\*

10 \* The application's global HTTP middleware stack. 11 \*

12 \* These middleware are run during every request to your application. 13 \*

14 \* @var array

15 \*/

1. protected $middleware = [
2. \Illuminate\Foundation\Http\Middleware\CheckForMaintenanceMode::class,
3. \Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,
4. \App\Http\Middleware\TrimStrings::class,
5. \Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,
6. \App\Http\Middleware\TrustProxies::class, 22 ];

23

24 /\*\*

25 \* The application's route middleware groups. 26 \*

27 \* @var array

28 \*/

29 protected $middlewareGroups = [ 30 'web' => [

1. \App\Http\Middleware\EncryptCookies::class,
2. \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
3. \Illuminate\Session\Middleware\StartSession::class,
4. // \Illuminate\Session\Middleware\AuthenticateSession::class,
5. \Illuminate\View\Middleware\ShareErrorsFromSession::class,
6. \App\Http\Middleware\VerifyCsrfToken::class,
7. \Illuminate\Routing\Middleware\SubstituteBindings::class, 38 ],

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

'api' => [

'throttle:60,1', 'bindings',

],

];

/\*\*

* The application's route middleware.

\*

* These middleware may be assigned to groups or used individually.

\*

* @var array

\*/

protected $routeMiddleware = [

'auth' => \Illuminate\Auth\Middleware\Authenticate::class,

'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class, 'bindings' => \Illuminate\Routing\Middleware\SubstituteBindings::class,

'can' => \Illuminate\Auth\Middleware\Authorize::class,

'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,

//中间件注册 login 中间件的规则名

'login' => \App\Http\Middleware\LoginMiddleware::class,

'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,

];

}

* + - 1. 中间件结合路由使用
         1. 在routes/web.php 里写入路由代码

1. //加载登录界面
2. Route::get("/login",function(){
3. return view("login"); 4 });
   * + - 1. 在resources/views 下创建login.blade.php 模板

1

2

3

4

5

6

7

8

9

10

11

12

13

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>登录界面</title>

</head>

<body>

<form action="" method="post">

用户名:<input type="text" name="name"><br>

密码:<input type="password" name="password"><br>

<input type="submit" value="登录">

</form>

1. </body>
2. </html>
   * + - 1. 中间件结合路由使用

<1 中间件普通的使用 login是中间件规则

1. //中间件普通的使用
2. Route::get("/adminss/user",function(){
3. echo "这是后天用户模块";
4. })->middleware('login');

<2 中间件结合路由组使用 （项目里推荐用法） login是中间件规则

1. //中间件结合路由组使用 （项目里推荐用法）
2. Route::group(["middleware"=>"login"],function(){
3. //子路由
4. Route::get("/shop",function(){
5. echo "这是后台商品模块"; 6 });
6. //子路由2
7. Route::get("/order",function(){
8. echo "这是后台订单模块"; 10 });

11 });

**第三章 控制器 请求 响应\*\***

* 1. **控制器\*\***
     1. **控制器目录**

app/http/controllers

* + 1. **创建普通控制器**

在cmd里执行如下命令,创建普通控制器

php artisan make:controller Admin/UserController

* + 1. **控制器使用**
       1. 在routes/web.php 写入如下代码

1. Route::get('/shop/index','UserController@index');
2. Route::get("/usersadd","Admin\UserController@add");
   * + 1. 控制器里代码

}

//添加方法

public function add(){

echo "这是会员模块的添加方法";

}

class UserController extends Controller

{

//方法index

public function index(){ echo "这是会员模块列表";

}

namespace App\Http\Controllers\Admin;

use Illuminate\Http\Request;

use App\Http\Controllers\Controller;

<?php

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

* + 1. **带有参数的控制器的访问**
       1. 在routes/web.php 写入如下代码

1. //带参数的控制器访问操作
2. Route::get("/usersdelete/{id}","Admin\UserController@delete");
   * + 1. 控制器里代码

1

2

3

4

5

6

7

8

9

10

<?php

namespace App\Http\Controllers\Admin;

use Illuminate\Http\Request;

use App\Http\Controllers\Controller;

class UserController extends Controller

{

//方法index

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

public function index(){

echo "这是会员模块列表";

}

//添加方法

public function add(){

echo "这是会员模块的添加方法";

}

//删除方法

public function delete($id){ echo "这是删除数据的id".$id;

}

//详情方法

public function info(){ echo "这是会员详情";

}

//index1

public function index1(){ echo "index1";

}

public function index2(){ echo "index2";

}

}

* + 1. **带有中间件的控制器的访问**
       1. 普通控制器结合中间件使用一
          1. 在routes/web.php 写入如下代码

1. //普通控制器结合中间件使用一
2. Route::get("/usersinfo","Admin\UserController@info")->middleware('login');
   * + 1. 普通控制器结合路由组使用

});

//普通控制器结合路由组使用

Route::group(["middleware"=>"login"],function(){

//子路由Route::get("/usersindex1","Admin\UserController@index1"); Route::get("/usersindex2","Admin\UserController@index2");

1

2

3

4

5

6

7

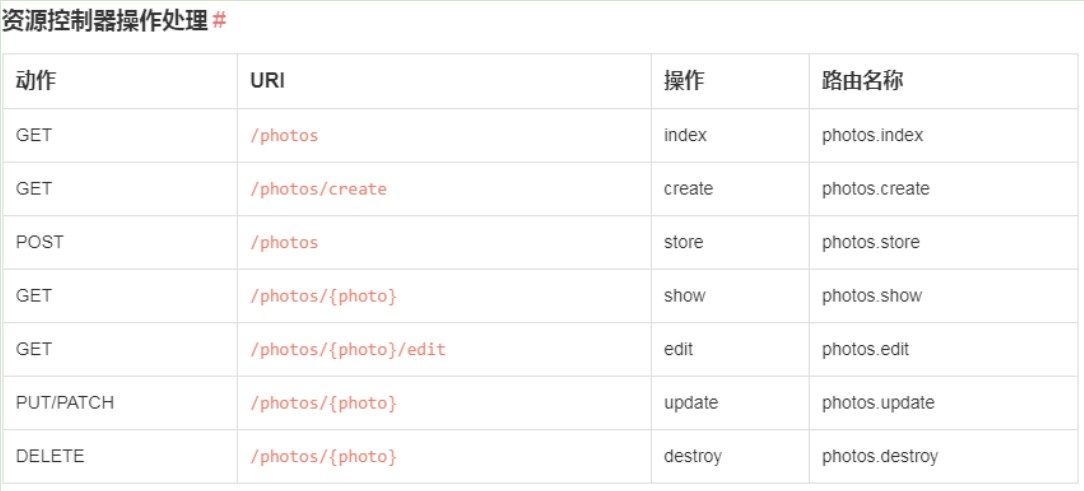
* + 1. **资源控制器**
       1. 作用

资源控制器 UsersController 里所有方法都可以统统交给 以userss开头的路由处理

* + - 1. 创建资源控制器

在cmd执行如下命令创建资源控制器

php artisan make:controller Admin/UsersController --resource

* + - 1. 资源控制器使用(1)Route::resource('/users','UserController'); (2)用法

(3)注意

伪造表单方法,例如 PUT，PATCH 或者 DELETE 请求，你需要添加一个 \_method 隐藏域字段来伪造 HTTP 动作。method\_ﬁeld 辅助函数可以为你创建这个字段：

{{ method\_ﬁeld('PUT') }}

* + - 1. 示例代码 模拟用户模块
         1. 创建Users 资源控制器

php artisan make:controller Admin/UsersController --resource (2)路由代码

1 Route::resource('/users','UserController');

1. 控制器代码

1 <?php 2

3 namespace App\Http\Controllers\Admin; 4

* 1. use Illuminate\Http\Request;
  2. use App\Http\Controllers\Controller; 7

8 class UsersController extends Controller 9 {

10 /\*\*

11 \* Display a listing of the resource. 12 \*

13 \* @return \Illuminate\Http\Response 14 \*/

15 public function index()

16 {

1. // echo "这是Users列表";
2. //加载列表模板
3. return view("Admin.Users.index"); 20 }

21

22 /\*\*

23 \* Show the form for creating a new resource. 24 \*

25 \* @return \Illuminate\Http\Response 26 \*/

27 public function create() 28 {

1. //添加界面
2. // echo "添加界面";
3. //加载添加界面视图
4. return view("Admin.Users.add"); 33 }

34

35 /\*\*

36 \* Store a newly created resource in storage. 37 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @return \Illuminate\Http\Response 40 \*/

41 public function store(Request $request) 42 {

1. //执行添加
2. echo "这是执行添加操作";

45 }

46

47 /\*\*

48 \* Display the specified resource. 49 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 52 \*/

53 public function show($id) 54 {

55 //

56 }

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100 } 101

/\*\*

* Show the form for editing the specified resource.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function edit($id)

{

// echo "这是修改数据的id:".$id;

//加载修改模板

return view("Admin.Users.edit",['id'=>$id]);

}

/\*\*

* Update the specified resource in storage.

\*

* @param \Illuminate\Http\Request $request
* @param int $id
* @return \Illuminate\Http\Response

\*/

//执行修改

public function update(Request $request, $id)

{

echo "这是执行数据修改,id为:".$id;

}

/\*\*

* Remove the specified resource from storage.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function destroy($id)

{

echo "这是删除数据的id".$id;

}

//自定义方法

public function index1(){ echo "this is index1";

}

1. 列表页模板
2. <!doctype html>
3. <html>
4. <head>
5. <meta charset="utf-8">

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

<title>列表模板</title>

</head>

<body>

<center>

<table border="1px" width="400px">

<tr>

<th>ID</th>

<th>NAME</th>

<th>PHONE</th>

<th>操作</th>

</tr>

<tr>

<td>1</td>

<td>科比</td>

<td>123</td>

<td><a href="">删除</a><a href="">修改</a></td>

</tr>

<tr>

<td>2</td>

<td>c罗</td>

<td>1234</td>

<td><form action="/userss/2" method="post">

{{csrf\_field()}}

{{method\_field('DELETE')}}

<button>删除</button>

</form><a href="/userss/2/edit">修改</a></td>

</tr>

</table>

</center>

</body>

</html>

1. 添加模板
2. <!doctype html>
3. <html>
4. <head>
5. <meta charset="utf-8">
6. <title>添加模板</title>
7. </head>
8. <body>
9. <form action="/userss" method="post">
10. 用户名:<input type="text" name="name"><br>
11. 密码:<input type="password" name="password"><br>
12. {{csrf\_field()}}
13. <input type="submit" value="提交"> 13
14. </form>
15. </body>
16. </html>
17. 修改模板

</form>

</body>

</html>

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>修改模板</title>

</head>

<body>

<form action="/userss/{{$id}}" method="post">

用户名:<input type="text" name="name" value="taoge"><br>

密码:<input type="password" name="password" value="123"><br>

{{csrf\_field()}}

{{method\_field('PUT')}}

<input type="submit" value="修改">

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

* + - 1. 资源控制器结合中间件

1. 资源控制器结合中间件一Route::resource("/adminorder","Admin\OrderController")->middleware('login');
2. 资源控制器结合中间件二(推荐使用) Route::group(['middleware'=>'login'],function(){

Route::resource("/usersss","Admin\UsersController"); Route::resource("/adminorders","Admin\OrderController");

});

* 1. **请求\*\***
     1. **请求对象声明**

在控制器方法里 Request $request 声明请求对象

1 public function index(Request $request) 2 {

1. echo "<pre>";
2. var\_dump($request); 5 }
   * 1. **基本信息获取**

dd() 方法打印数据的同时 终止脚本代码

1 public function index(Request $request) 2 {

1. // echo "<pre>";
2. // var\_dump($request);
3. //获取请求路径
4. $path=$request->path();
5. //获取完整的url地址
6. $url=$request->url();
7. //获取请求方式
8. $method=$request->method();
9. //检测请求方式
10. $res=$request->isMethod('post');
11. //dd() 打印数据的同时 终止脚本代码
12. dd($data1); 15 }
    * 1. **提取请求参数**
         1. 提取所有的参数

$input = $request->all();

* + - 1. 提取单个参数

$name = $request->input('name');

* + - 1. 提取部分 参数
         1. 获取指定的参数

$request->only(['username', 'password']); (2)获取除了某个参数之外的参数

$request->except(['credit\_card']);

* + - 1. 设置默认值

$request->input('name', 'Sally');

* + - 1. 检测是否存在

$request->has('name')

* + 1. **闪存信息**
       1. 什么是闪存:

稍纵即逝 当重新请求服务器 闪存信息就会被清除掉

* + - 1. 将所有的请求参数写入闪存中

$request->ﬂash()

* + - 1. 将部分参数写入闪存中

$request->ﬂashOnly('title','price')

* + - 1. 除去某些参数之外的参数

$request->ﬂashExcept('\_token');

* + - 1. 简便使用

return back()->withInput(); //阻止页面提交的同时,将全部的参数信息写入闪存里

* + - 1. 代码实例
         1. 模板代码

</form>

</body>

</html>

{{csrf\_field()}}

<input type="submit" value="提交">

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>req模板</title>

</head>

<body>

<form action="/req" method="post">

用户名:<input type="text" name="name"><br>

电话:<input type="text" name="phone" value="{{old('phone')}}"><br>

邮箱:<input type="text" name="email" value="{{old('email')}}"><br>

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

* + - * 1. 控制器代码

1

2

3

4

5

6

7

8

9

<?php

namespace App\Http\Controllers;

//导入请求类

use Illuminate\Http\Request;

class ReqController extends Controller

{

/\*\*

10 \* Display a listing of the resource. 11 \*

12 \* @return \Illuminate\Http\Response 13 \*/

1. // Request $request 用Reuest 请求类约束请求对象 $request
2. public function index(Request $request) 16 {

17

18 }

19

20 /\*\*

21 \* Show the form for creating a new resource. 22 \*

23 \* @return \Illuminate\Http\Response 24 \*/

25 public function create() 26 {

1. //加载模板
2. return view("req"); 29

30 }

31

32 /\*\*

33 \* Store a newly created resource in storage. 34 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @return \Illuminate\Http\Response 37 \*/

38 public function store(Request $request) 39 {

1. //获取所有参数
2. // $allparam=$request->all();
3. // //去除某个参数
4. // $data=$request->except(['\_token']);
5. // dd($data);
6. // echo "这是执行添加操作";
7. //获取用户名
8. $name=$request->input('name');
9. //将所有的参数写在闪存里
10. // $request->flash();
11. //将部分参数写在闪存里
12. // $request->flashOnly('email');
13. //除去某些参数写在闪存里
14. // $request->flashExcept('email');
15. //判断
16. if($name==null){

56 // echo "1";

1. //阻止表单提交 将所有的参数写入到闪存里
2. return back()->withInput(); 59 }

60 }

61

62 /\*\*

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106 } 107

* Display the specified resource.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function show($id)

{

//

}

/\*\*

* Show the form for editing the specified resource.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function edit($id)

{

//

}

/\*\*

* Update the specified resource in storage.

\*

* @param \Illuminate\Http\Request $request
* @param int $id
* @return \Illuminate\Http\Response

\*/

public function update(Request $request, $id)

{

//

}

/\*\*

* Remove the specified resource from storage.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function destroy($id)

{

//

}

* + 1. **文件操作**
       1. 检测是否有文件上传

$request->hasFile('photo')

* + - 1. 将文件移动到指定位置

$request->ﬁle('photo')->move('./uploads/', 'iloveyou.jpg');

* + - 1. 获取图片后缀

$requst->ﬁle(‘参数’)->getClientOriginalExtension()

* + - 1. 代码实例
         1. 模板

</form>

</body>

</html>

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>file模板</title>

</head>

<body>

<form action="/file" method="post" enctype="multipart/form-data">

文件:<input type="file" name="pic"><br>

{{csrf\_field()}}

<input type="submit" value="提交">

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

* + - * 1. 控制器

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class FileController extends Controller

{

/\*\*

* Display a listing of the resource.

\*

* @return \Illuminate\Http\Response

\*/

public function index(Request $request)

{

}

/\*\*

* Show the form for creating a new resource.

\*

* @return \Illuminate\Http\Response

\*/

public function create()

24 {

25

26

27 }

28

29 /\*\*

//

return view("file");

30 \* Store a newly created resource in storage. 31 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @return \Illuminate\Http\Response 34 \*/

35 public function store(Request $request) 36 {

37 //

1. // dd($request->all());
2. //判断是否具有文件上传
3. if($request->hasFile('pic')){
4. //初始化名字

42 $name=time()+rand(1,10000);

1. //获取上传文件后缀
2. // $ext=$request->file('pic')->extension();
3. $ext=$request->file("pic")->getClientOriginalExtension(); 46 // dd($ext);

47

1. //移动到指定的目录下（提前在public下新建uploads目录）
2. $request->file("pic")->move("./uploads",$name.".".$ext); 50 }

51 }

52

53 /\*\*

54 \* Display the specified resource. 55 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 58 \*/

59 public function show($id) 60 {

61 //

62 }

63

64 /\*\*

65 \* Show the form for editing the specified resource. 66 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 69 \*/

70 public function edit($id) 71 {

72 //

73 }

74

75 /\*\*

76 \* Update the specified resource in storage.

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

\*

* @param \Illuminate\Http\Request $request
* @param int $id
* @return \Illuminate\Http\Response

\*/

public function update(Request $request, $id)

{

//

}

/\*\*

* Remove the specified resource from storage.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function destroy($id)

{

//

}

}

### cookie操作

* + - 1. 设置cookie一

return response('kalaok')->withCookie(cookie名字,cookie值,过期时间单位为分);

* + - 1. 设置cookie二

\Cookie::queue(cookie名字,cookie值,过期时间单位为分);//其中\Cookie 获取的是系统Cookie类

* + - 1. 获取cookie一

$request->cookie(cookie名字);

* + - 1. 获取cookie二

\Cookie::get(cookie名字);

* + 1. **session操作**

laravel框架里session自动开启,不需要session\_start()

* + - 1. 存储一条数据至 Session 中

session(['key' => 'value']);

* + - 1. 获取 Session 中的一条数据

$value = session('key');

* + - 1. **获取所有 Session 数据**

**$data = $request->session()->all();**

* + - 1. **判断某个 Session 值是否存在**

$request->session()->has('session名字')

* + - 1. **push 方法可以将一个新的值加入至一个 Session 数组内**

$request->session()->push('session名', 'session值');

* + - 1. **从 Session 中取出并删除数据**

$request->session()->pull('session名字');

* 1. **响应\*\***

1. 返回字符串

return '哈哈哈';

1. 设置cookie

return response('ssss')->withCookie('id',20,60);

1. 显示模板

return view('goods.add');

1. 返回json

return response()->json(['a'=>100,'b'=>2000]);

1. 下载文件

return response()->download('web.conﬁg');

1. 页面跳转

return redirect('/goods/add');

**第四章 视图\*\***

* 1. **视图目录**

resources/views

* 1. **加载模板**

return view('User.add');//其中User为模板目录,add即为add.blade.php

* 1. **分配数据到模板**

return view('User.add',['变量'=>值,'变量2'=>值2]);

* 1. **模板引擎blade**
     1. 解析变量

1 {{$username}}

* + 1. 使用函数

{{time()}}

* + 1. 设置默认值

{{$username or '默认值'}} //如果$username有值则解析,否则拿默认值

* + 1. 显示html代码

{!! $name !!}

* + 1. 引入子视图

@include('header')

* + 1. 流程控制

@if (count($records) === 1) have one record!

@elseif (count($records) > 1) Ihave multiple records! @else

I don't have any records! @endif

* + 1. 循环控制
       1. for 循环

@for ($i = 0; $i < 10; $i++) The current value is {{ $i }} @endfor

* + - 1. foreach循环

@foreach ($users as $user) This is user {{ $user->id }} @endforeach

* + 1. 占位符
       1. 第一种方式

@yield('占位符名字') (2)第二种方式

@section('占位符名字') @show

* + 1. 新模板内容
       1. 模板继承标签

@extends('index') (2)给标题赋值

@section('title', 'new Title')

(3)给主体内容赋值@section('content')

new contents is here @endsection

* 1. **模板继承代码演示**
     1. 创建资源控制器AdminController

在cmd 项目目录下执行 php artisan make controller Admin/AdminController --resource

* + 1. 路由代码

Route::resource("/admin","Admin\AdminController");

* + 1. 控制器代码

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

<?php

namespace App\Http\Controllers\Admin;

use Illuminate\Http\Request;

use App\Http\Controllers\Controller;

class AdminController extends Controller

{

/\*\*

* Display a listing of the resource.

\*

* @return \Illuminate\Http\Response

\*/

public function index()

{

//

//加载后台模板

return view("Admin.Admin.index");

}

/\*\*

* Show the form for creating a new resource.

\*

* @return \Illuminate\Http\Response

\*/

public function create()

{

//

}

/\*\*

33 \* Store a newly created resource in storage. 34 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @return \Illuminate\Http\Response 37 \*/

38 public function store(Request $request) 39 {

40 //

41 }

42

43 /\*\*

44 \* Display the specified resource. 45 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 48 \*/

49 public function show($id) 50 {

51 //

52 }

53

54 /\*\*

55 \* Show the form for editing the specified resource. 56 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 59 \*/

60 public function edit($id) 61 {

62 //

63 }

64

65 /\*\*

66 \* Update the specified resource in storage. 67 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @param int $id
3. \* @return \Illuminate\Http\Response 71 \*/

72 public function update(Request $request, $id) 73 {

74 //

75 }

76

77 /\*\*

78 \* Remove the specified resource from storage. 79 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 82 \*/

83 public function destroy($id) 84 {

85 //

86 }

87 }

88

* + 1. 在resource/Views/Admin目录下新建模板Admin/index.blade.php 模板代码如下

1. @extends("Admin.AdminPublic.public")
2. @section('admin') 3 <h1>后台首页</h1>
3. @endsection
4. @section("title",'后台首页')
   * 1. 在resource/Views/Admin目录下新建AdminPublic/public.blade.php代码如下
        1. <!DOCTYPE html>
        2. <!--[if lt IE 7]> <html class="lt-ie9 lt-ie8 lt-ie7" lang="en"> <![endif]-->
        3. <!--[if IE 7]> <html class="lt-ie9 lt-ie8" lang="en"> <![endif]-->
        4. <!--[if IE 8]> <html class="lt-ie9" lang="en"> <![endif]--> 5 <!--[if gt IE 8]><!--><html lang="en"><!--<![endif]-->
5. <head>
6. <meta charset="utf-8"> 8

9 <!-- Viewport Metatag -->

10 <meta name="viewport" content="width=device-width,initial-scale=1.0"> 11

1. <!-- Plugin Stylesheets first to ease overrides -->
2. <link rel="stylesheet" type="text/css" href="/static/admins/plugins/colorpicker/colorpicker.css" media="screen">
3. <link rel="stylesheet" type="text/css" href="/static/admins/custom- plugins/wizard/wizard.css" media="screen">

15

1. <!-- Required Stylesheets -->
2. <link rel="stylesheet" type="text/css" href="/static/admins/bootstrap/css/bootstrap.min.css" media="screen">
3. <link rel="stylesheet" type="text/css" href="/static/admins/css/fonts/ptsans/stylesheet.css" media="screen">
4. <link rel="stylesheet" type="text/css" href="/static/admins/css/fonts/icomoon/style.css" media="screen">

20

1. <link rel="stylesheet" type="text/css" href="/static/admins/css/mws-style.css" media="screen">
2. <link rel="stylesheet" type="text/css" href="/static/admins/css/icons/icol16.css" media="screen">
3. <link rel="stylesheet" type="text/css" href="/static/admins/css/icons/icol32.css" media="screen">

24

1. <!-- Demo Stylesheet -->
2. <link rel="stylesheet" type="text/css" href="/static/admins/css/demo.css" media="screen">

27

1. <!-- jQuery-UI Stylesheet -->
2. <link rel="stylesheet" type="text/css" href="/static/admins/jui/css/jquery.ui.all.css" media="screen">
3. <link rel="stylesheet" type="text/css" href="/static/admins/jui/jquery-ui.custom.css" media="screen">

31

1. <!-- Theme Stylesheet -->
2. <link rel="stylesheet" type="text/css" href="/static/admins/css/mws-theme.css" media="screen">
3. <link rel="stylesheet" type="text/css" href="/static/admins/css/themer.css" media="screen">
4. <link rel="stylesheet" type="text/css" href="/static/admins/css/my.css" media="screen">

36

37

38 <title>@yield('title')</title> 39

40 </head> 41

42 <body>

43 <!-- Header -->

44 <div id="mws-header" class="clearfix"> 45

1. <!-- Logo Container -->
2. <div id="mws-logo-container"> 48
3. <!-- Logo Wrapper, images put within this wrapper will always be vertically centered -->
4. <div id="mws-logo-wrap">
5. <img src="/static/admins/images/mws-logo.png" alt="mws admin">
6. </div>
7. </div>

54

1. <!-- User Tools (notifications, logout, profile, change password) -->
2. <div id="mws-user-tools" class="clearfix"> 57
3. <!-- Notifications -->
4. <div id="mws-user-notif" class="mws-dropdown-menu">
5. <a href="#" data-toggle="dropdown" class="mws-dropdown-trigger"><i class="icon-exclamation-sign"></i></a>

61

1. <!-- Unread notification count -->
2. <span class="mws-dropdown-notif">35</span> 64
3. <!-- Notifications dropdown -->
4. <div class="mws-dropdown-box">
5. <div class="mws-dropdown-content">
6. <ul class="mws-notifications">
7. <li class="read">

70 <a href="#">

71

72

adipiscing elit, et al commore

73

74

75

76

77

</a>

<span class="message">

Lorem ipsum dolor sit amet consectetur

</span>

<span class="time"> January 21, 2012

</span>

78 </li>

79 <li class="read">

80 <a href="#">

1. <span class="message">
2. Lorem ipsum dolor sit amet
3. </span>
4. <span class="time">
5. January 21, 2012
6. </span>

87 </a>

88 </li>

89 <li class="unread">

90 <a href="#">

1. <span class="message">
2. Lorem ipsum dolor sit amet
3. </span>
4. <span class="time">
5. January 21, 2012
6. </span>

97 </a>

98 </li>

99 <li class="unread">

100 <a href="#">

1. <span class="message">
2. Lorem ipsum dolor sit amet
3. </span>
4. <span class="time">
5. January 21, 2012
6. </span>

107 </a>

108 </li>

109 </ul>

1. <div class="mws-dropdown-viewall">
2. <a href="#">View All Notifications</a>
3. </div>
4. </div>
5. </div>
6. </div>
7. <!-- Messages -->
8. <div id="mws-user-message" class="mws-dropdown-menu">
9. <a href="#" data-toggle="dropdown" class="mws-dropdown-trigger"><i class="icon-envelope"></i></a>
10. <!-- Unred messages count -->

|  |  |
| --- | --- |
| 122 | <span class="mws-dropdown-notif">35</span> |
| 123 |  |
| 124 | <!-- Messages dropdown --> |
| 125 | <div class="mws-dropdown-box"> |
| 126 | <div class="mws-dropdown-content"> |
| 127 | <ul class="mws-messages"> |
| 128 | <li class="read"> |
| 129 | <a href="#"> |
| 130 | <span class="sender">John Doe</span> |
| 131 | <span class="message"> |
| 132 | Lorem ipsum dolor sit amet consectetur  adipiscing elit, et al commore |
| 133 | </span> |
| 134 | <span class="time"> |
| 135 | January 21, 2012 |
| 136 | </span> |
| 137 | </a> |
| 138 | </li> |
| 139 | <li class="read"> |
| 140 | <a href="#"> |
| 141 | <span class="sender">John Doe</span> |
| 142 | <span class="message"> |
| 143 | Lorem ipsum dolor sit amet |
| 144 | </span> |
| 145 | <span class="time"> |
| 146 | January 21, 2012 |
| 147 | </span> |
| 148 | </a> |
| 149 | </li> |
| 150 | <li class="unread"> |
| 151 | <a href="#"> |
| 152 | <span class="sender">John Doe</span> |
| 153 | <span class="message"> |
| 154 | Lorem ipsum dolor sit amet |
| 155 | </span> |
| 156 | <span class="time"> |
| 157 | January 21, 2012 |
| 158 | </span> |
| 159 | </a> |
| 160 | </li> |
| 161 | <li class="unread"> |
| 162 | <a href="#"> |
| 163 | <span class="sender">John Doe</span> |
| 164 | <span class="message"> |
| 165 | Lorem ipsum dolor sit amet |
| 166 | </span> |
| 167 | <span class="time"> |
| 168 | January 21, 2012 |
| 169 | </span> |
| 170 | </a> |
| 171 | </li> |
| 172 | </ul> |
| 173 | <div class="mws-dropdown-viewall"> |

1. <a href="#">View All Messages</a>
2. </div>
3. </div>
4. </div>
5. </div>
6. <!-- User Information and functions section -->
7. <div id="mws-user-info" class="mws-inset"> 182
8. <!-- User Photo -->
9. <div id="mws-user-photo">
10. <img src="/static/admins/example/profile.jpg" alt="User Photo">
11. </div>
12. <!-- Username and Functions -->
13. <div id="mws-user-functions">
14. <div id="mws-username">
15. Hello, John Doe
16. </div>
17. <ul>
18. <li><a href="#">Profile</a></li>
19. <li><a href="#">Change Password</a></li>
20. <li><a href="index.html">Logout</a></li>

197 </ul>

1. </div>
2. </div>
3. </div>
4. </div> 202
5. <!-- Start Main Wrapper -->
6. <div id="mws-wrapper"> 205
7. <!-- Necessary markup, do not remove -->
8. <div id="mws-sidebar-stitch"></div>
9. <div id="mws-sidebar-bg"></div> 209
10. <!-- Sidebar Wrapper -->
11. <div id="mws-sidebar"> 212
12. <!-- Hidden Nav Collapse Button -->
13. <div id="mws-nav-collapse">
14. <span></span>
15. <span></span>
16. <span></span>
17. </div>
18. <!-- Searchbox -->
19. <div id="mws-searchbox" class="mws-inset">
20. <form action="typography.html">
21. <input type="text" class="mws-search-input" placeholder="Search...">
22. <button type="submit" class="mws-search-submit"><i class="icon- search"></i></button>
23. </form>
24. </div>
25. <!-- Main Navigation -->
26. <div id="mws-navigation">
27. <ul>
28. <li>
29. <a href="#"><i class="icon-user"></i> 用户管理</a>
30. <ul class="closed">
31. <li><a href="/adminusers/create">用户添加</a></li>
32. <li><a href="/adminusers">用户列表</a></li>

236 </ul>

237 </li>

238

1. <li>
2. <a href="#"><i class="icon-th-list"></i> 分类管理</a>
3. <ul class="closed">
4. <li><a href="">分类添加</a></li>
5. <li><a href="">分类列表</a></li>

244 </ul>

245 </li>

246

1. <li>
2. <a href="#"><i class="icon-file"></i> 文章管理</a>
3. <ul class="closed">
4. <li><a href="">文章添加</a></li>
5. <li><a href="">文章列表</a></li>

252 </ul>

253 </li>

254

1. <li>
2. <a href="#"><i class="icon-file"></i> 商品管理</a>
3. <ul class="closed">
4. <li><a href="">商品添加</a></li>
5. <li><a href="">商品列表</a></li>

260 </ul>

261 </li>

262 </ul>

1. </div>
2. </div>
3. <!-- Main Container Start -->
4. <div id="mws-container" class="clearfix"> 268
5. <div class="container">
6. @section('admin')
7. @show
8. <!-- Panels End -->
9. </div>

276 <!-- footer -->

277 <div id="mws-footer">

278

279

280

</div>

Copyright Your Website 2012. All Rights Reserved.

</div>

281 <!-- Main Container End --> 282

283 </div> 284

1. <!-- JavaScript Plugins -->
2. <script src="/static/admins/js/libs/jquery-1.8.3.min.js"></script>
3. <script src="/static/admins/js/libs/jquery.mousewheel.min.js"></script>
4. <script src="/static/admins/js/libs/jquery.placeholder.min.js"></script>
5. <script src="/static/admins/custom-plugins/fileinput.js"></script> 290
6. <!-- jQuery-UI Dependent Scripts -->
7. <script src="/static/admins/jui/js/jquery-ui-1.9.2.min.js"></script>
8. <script src="/static/admins/jui/jquery-ui.custom.min.js"></script>
9. <script src="/static/admins/jui/js/jquery.ui.touch-punch.js"></script> 295
10. <!-- Plugin Scripts -->
11. <script src="/static/admins/plugins/datatables/jquery.dataTables.min.js"></script> 298 <!--[if lt IE 9]>

299 <script src="js/libs/excanvas.min.js"></script> 300 <![endif]-->

1. <script src="/static/admins/plugins/flot/jquery.flot.min.js"></script>
2. <script src="/static/admins/plugins/flot/plugins/jquery.flot.tooltip.min.js">

</script>

1. <script src="/static/admins/plugins/flot/plugins/jquery.flot.pie.min.js"></script>
2. <script src="/static/admins/plugins/flot/plugins/jquery.flot.stack.min.js">

</script>

1. <script src="/static/admins/plugins/flot/plugins/jquery.flot.resize.min.js">

</script>

1. <script src="/static/admins/plugins/colorpicker/colorpicker-min.js"></script>
2. <script src="/static/admins/plugins/validate/jquery.validate-min.js"></script>
3. <script src="/static/admins/custom-plugins/wizard/wizard.min.js"></script> 309
4. <!-- Core Script -->
5. <script src="/static/admins/bootstrap/js/bootstrap.min.js"></script>
6. <script src="/static/admins/js/core/mws.js"></script> 313
7. <!-- Themer Script (Remove if not needed) -->
8. <script src="/static/admins/js/core/themer.js"></script> 316
9. <!-- Demo Scripts (remove if not needed) -->
10. <script src="/static/admins/js/demo/demo.dashboard.js"></script> 319
11. </body>
12. </html>

# 第五章 数据库操作\*\*

1. 支持的数据库类型

MySQL Postgres SQLite SQL Server

1. 数据库连接配置

在.env环境快速配置 ,配置如下:

DB\_CONNECTION=mysql//数据库类型DB\_HOST=127.0.0.1//主机

DB\_PORT=3306//端口号DB\_DATABASE=lampbrotherss//数据库名字DB\_USERNAME=root//用户名DB\_PASSWORD=//密码

1. 数据库操作结果类型

结果集的返回类型:默认的是对象格切换数组结果集返回类型步骤如下 :

* 1. 打开app/Providers/EventServiceProvier.php
  2. 加入代码

use Illuminate\Database\Events\StatementPrepared //引入类(3)然后在boot方法中加入如下代码

Event::listen(StatementPrepared::class,function($event){ $event->statement-

>setFetchMode(\PDO::FETCH\_ASSOC); }); (4)代码如下

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

<?php

namespace App\Providers;

use Illuminate\Support\Facades\Event;

use Illuminate\Foundation\Support\Providers\EventServiceProvider as ServiceProvider; use Illuminate\Database\Events\StatementPrepared //引入类

class EventServiceProvider extends ServiceProvider

{

/\*\*

* The event listener mappings for the application.

\*

* @var array

\*/

protected $listen = [ 'App\Events\Event' => [

'App\Listeners\EventListener',

],

];

/\*\*

* Register any events for your application.

\*

* @return void

\*/

26

27

28

29

30

31

public function boot()

{

parent::boot();

32

33

34

//

Event::listen(StatementPrepared::class,function($event){ $event->statement-

>setFetchMode(\PDO::FETCH\_ASSOC); });

}

}

1. 数据库基本操作

先在控制器里引入DB类,use DB; (1)查询

DB::select(查询sql语句);

(2)插入

DB::insert(插入sql语句); (3)更新

DB::update(更新sql语句);

(4)删除

DB::delete(删除sql语句); (5)一般语句

DB::statement('drop table users'); //删除表

1. 构造器
   1. 增删改查

<1 插入

单条

DB::table('users')->insert( ['email' => '[john@example.com](mailto:john@example.com)', 'votes' => 0] );

多条

[DB::table('users')->insert([ ['email' => 'taylor@example.com', 'votes' => 0], ['email' => 'dayle@ex ample.com', 'votes' => 0] ]);](mailto:dayle@example.com)

获取id插入

$id = DB::table('users')->insertGetId( ['email' => '[john@example.com](mailto:john@example.com)', 'votes' => 0] );

<2 更新

DB::table('users') ->where('id','=', 1) ->update(['votes' => 1]);

<3 删除

DB::table('users')->where('votes', '<', 100)->delete();

<4 查询

查询所有

DB::table('users')->get()

查询单条

DB::table('users')->where("id","=",1)->ﬁrst()

查询单条结果中的某个字段

DB::table('user')->where("id","=",308)->value('username')

获取一列数据

DB::table('user')->pluck('username') (2)连贯操作

<1 设置字段

DB::table('users')->select('name', 'email as user\_email')->get()

<2 条件

DB::table('users') ->where('name', 'like', '%T%') ->get()

DB::table('users') ->where('votes', '>', 100) ->orWhere('name','=', 'John') ->get(); DB::table('users') ->whereBetween('votes', [1, 100])->get()

DB::table('users') ->whereIn('id', [1, 2, 3]) ->get()

<3 排序

orderBy('name', 'desc')

<4 分页

DB::table('users')->paginate(3);

<5 连接表(join)

DB::table('users') ->join('contacts', 'users.id', '=', 'contacts.user\_id') ->join('orders', 'users.id', '=', 'orders.user\_id') ->select('users.\*', 'contacts.phone', 'orders.price') ->get();

<6 计算

总数

DB::table('users')->count()

最大值

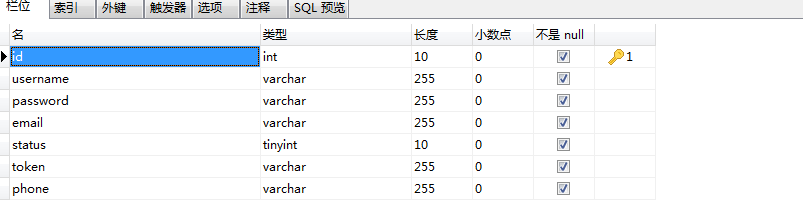
DB::table('orders')->max('price')

平均值

DB::table('orders')->avg('price')

**第六章 用户模块实战开发\*\***

* 1. **创建数据表 users 结构如下**



* 1. **新建控制器Users**

php artisan make controller Admin/UsersController --resource

* 1. **路由代码**

Route::resource("/adminusers","Admin\UsersController");

* 1. **控制器代码**

redirect("路由地址")->with('success','添加成功'); //在跳转的时候 设置session信息

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

<?php

namespace App\Http\Controllers\Admin;

use Illuminate\Http\Request;

use App\Http\Controllers\Controller; use DB;

//导入Hash类

use Hash;

//导入校验类

use App\Http\Requests\UserInsert;

class UsersController extends Controller

{

/\*\*

* Display a listing of the resource.

\*

* @return \Illuminate\Http\Response

\*/

public function index(Request $request)

{

//获取搜索关键词

$k=$request->input('keywords');

23

24

25

26

27 }

28

29 /\*\*

//获取列表数据

$user=DB::table("users")->where("username",'like',"%".$k."%")->paginate(3);

//加载模板

return view("Admin.Users.index",['user'=>$user,'request'=>$request->all()]);

30 \* Show the form for creating a new resource. 31 \*

32 \* @return \Illuminate\Http\Response 33 \*/

34 public function create() 35 {

1. //添加界面
2. // echo "this is add";
3. return view("Admin.Users.add"); 39

40 }

41

42 /\*\*

43 \* Store a newly created resource in storage. 44 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @return \Illuminate\Http\Response 47 \*/

48 public function store(UserInsert $request) 49 {

1. // dd($request->all());
2. $data=$request->except(['repassword','\_token']);
3. //对密码做加密
4. $data['password']=Hash::make($data['password']);
5. $data['status']=1;
6. $data['token']=str\_random(50);
7. // dd($data);
8. if(DB::table("users")->insert($data)){
9. return redirect("/adminusers")->with('success','数据添加成功');
10. }else{
11. return redirect("/adminusers/create")->with('error','数据添加失败'); 61 }

62

63 }

64

65 /\*\*

66 \* Display the specified resource. 67 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 70 \*/

71 public function show($id) 72 {

73 //

74 }

75

76 /\*\*

77 \* Show the form for editing the specified resource. 78 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 81 \*/

82 public function edit($id) 83 {

1. // echo "这是修改数据的id:".$id;
2. //获取需要修改的数据
3. $user=DB::table("users")->where("id",'=',$id)->first();
4. return view("Admin.Users.edit",['user'=>$user]); 88

89 }

90

91 /\*\*

92 \* Update the specified resource in storage. 93 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @param int $id
3. \* @return \Illuminate\Http\Response 97 \*/

98

1. //执行修改
2. public function update(Request $request, $id)

101 {

1. // dd($request->all());
2. $data=$request->except(['\_token','\_method']);
3. //密码加密
4. $data['password']=Hash::make($data['password']);
5. if(DB::table("users")->where("id","=",$id)->update($data)){
6. return redirect("/adminusers")->with('success',"修改成功");
7. }else{
8. return redirect("/adminusers/$id",'数据修改失败');

110 }

111 }

112

113 /\*\*

114 \* Remove the specified resource from storage.

115 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 118 \*/

119 public function destroy($id)

120 {

1. // echo $id;
2. if(DB::table("users")->where("id",'=',$id)->delete()){
3. return redirect("/adminusers")->with('success','数据删除成功');
4. }else{
5. return redirect("/adminusers")->with('error','数据删除失败');

126 }

127 }

128

129

130

131

132

133

134

135

136

137

138

139

140

//删除

public function del(Request $request){

$id=$request->input('id');

// echo $id;

if(DB::table("users")->where("id",'=',$id)->delete()){ echo 1;

}else{

echo 0;

}

}

}

* 1. **模板代码**
     1. 列表页模板

{{$user->appends($request)->render()}} //显示分页

* + - 1. @extends("Admin.AdminPublic.public")
      2. @section("admin")
      3. <html>
      4. <head></head>
      5. <script type="text/javascript" src="/static/js/jquery-1.8.3.min.js"></script>
      6. <body>
      7. <div class="mws-panel grid\_8">
      8. <div class="mws-panel-header">
      9. <span><i class="icon-table"></i>会员列表</span>
      10. </div>
      11. <div class="mws-panel-body no-padding">
      12. <div id="DataTables\_Table\_1\_wrapper" class="dataTables\_wrapper" role="grid">
      13. <form action="/adminusers" method="get">
      14. <div class="dataTables\_filter" id="DataTables\_Table\_1\_filter">
      15. <label>搜索: <input type="text" aria-controls="DataTables\_Table\_1" name="keywords" value="{{$request['keywords'] or ''}}"/></label><input type="submit" value="搜索">
      16. </div>
      17. </form>
      18. <table class="mws-datatable-fn mws-table dataTable" id="DataTables\_Table\_1" aria- describedby="DataTables\_Table\_1\_info">
      19. <thead>
      20. <tr role="row">
      21. <th class="sorting\_asc" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-sort="ascending" aria- label="Rendering engine: activate to sort column descending" style="width: 157px;">ID</th>
      22. <th class="sorting" role="columnheader" tabindex="0" aria-

controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Browser: activate to sort column ascending" style="width: 209px;">会员名</th>

* + - 1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Platform(s): activate

to sort column ascending" style="width: 191px;">邮箱</th>

* + - 1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Engine version:

activate to sort column ascending" style="width: 137px;">电话</th>

* + - 1. <th class="sorting" role="columnheader" tabindex="0" aria-

controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="CSS grade: activate to sort column ascending" style="width: 101px;">操作</th>

26 </tr>

1. </thead>
2. <tbody role="alert" aria-live="polite" aria-relevant="all">
3. @foreach($user as $row)
4. <tr class="odd">
5. <td class=" sorting\_1">{{$row->id}}</td>
6. <td class=" ">{{$row->username}}</td>
7. <td class=" ">{{$row->email}}</td>
8. <td class=" ">{{$row->phone}}</td>
9. <td class=" ">
10. <form action="/adminusers/{{$row->id}}" method="post">
11. {{csrf\_field()}}
12. {{method\_field("DELETE")}}
13. <button class="btn btn-info">删除</button>
14. </form>
15. <a href="javascript:void(0)" class="btn btn-success del" >删除</a><a href="/adminusers/{{$row->id}}/edit" class="btn btn-danger">修改</a></td>

42 </tr>

1. @endforeach
2. </tbody>
3. </table>
4. <div class="dataTables\_paginate paging\_full\_numbers" id="pages">
5. {{$user->appends($request)->render()}}
6. </div>
7. </div>
8. </div>
9. </div>
10. </body>
11. <script type="text/javascript"> 54 // alert($);
12. //获取删除按钮 绑定单击事件
13. $(".del").click(function(){
14. //Ajax
15. id=$(this).parents("tr").find("td:first").html();
16. s=$(this).parents("tr");
17. // alert(id);
18. //Ajax
19. $.get('/adminusersdel',{id:id},function(data){
20. // alert(data);
21. if(data==1){
22. alert("删除成功");
23. //把删除数据所在的tr从dom里移除
24. s.remove();
25. }else{

69

70

71

72

73

74

75

76

77

alert("删除失败");

}

});

});

</script>

</html> @endsection

@section('title','后台用户列表')

* + 1. 添加模板

1. @extends("Admin.AdminPublic.public")
2. @section("admin")
3. <html>
4. <head></head>
5. <body>
6. <div class="mws-panel grid\_8">
7. <div class="mws-panel-header">
8. <span>会员添加</span>
9. </div>
10. <div class="mws-panel-body no-padding">
11. <form class="mws-form" action="/adminusers" method="post">
12. @if (count($errors) > 0)
13. <div class="alert alert-danger">
14. <div class="mws-form-message error">
15. <ul>
16. @foreach ($errors->all() as $error) 17 <li>{{ $error }}</li>

18 @endforeach

19 </ul>

1. </div>
2. </div>
3. @endif
4. <div class="mws-form-inline">
5. <div class="mws-form-row">
6. <label class="mws-form-label">会员名</label>
7. <div class="mws-form-item">
8. <input type="text" name="username" class="large" value="{{old('username')}}" />
9. </div>
10. </div>
11. <div class="mws-form-row">
12. <label class="mws-form-label">密码</label>
13. <div class="mws-form-item">
14. <input type="password" name="password" class="large" />
15. </div>
16. </div>
17. <div class="mws-form-row">
18. <label class="mws-form-label">重复密码</label>
19. <div class="mws-form-item">
20. <input type="password" name="repassword" class="large" />

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

</div>

</div>

<div class="mws-form-row">

<label class="mws-form-label">邮箱</label>

<div class="mws-form-item">

<input type="text" name="email" class="large" value="{{old('email')}}"/>

</div>

</div>

<div class="mws-form-row">

<label class="mws-form-label">电话</label>

<div class="mws-form-item">

<input type="text" name="phone" class="large" value="{{old('phone')}}"/>

</div>

</div>

</div>

<div class="mws-button-row">

{{csrf\_field()}}

<input type="submit" value="Submit" class="btn btn-danger" />

<input type="reset" value="Reset" class="btn " />

</div>

</form>

</div>

</div>

</body>

</html> @endsection

@section('title','后台用户添加')

* + 1. 修改模板

1. @extends("Admin.AdminPublic.public")
2. @section("admin")
3. <html>
4. <head></head>
5. <body>
6. <div class="mws-panel grid\_8">
7. <div class="mws-panel-header">
8. <span>会员修改</span>
9. </div>
10. <div class="mws-panel-body no-padding">
11. <form class="mws-form" action="/adminusers/{{$user->id}}" method="post">
12. <div class="mws-form-inline">
13. <div class="mws-form-row">
14. <label class="mws-form-label">会员名</label>
15. <div class="mws-form-item">
16. <input type="text" name="username" class="large" value="{{$user->username}}" />
17. </div>
18. </div>
19. <div class="mws-form-row">

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

<label class="mws-form-label">密码</label>

<div class="mws-form-item">

<input type="password" name="password" class="large" value="{{$user-

>password}}"/>

</div>

</div>

<div class="mws-form-row">

<label class="mws-form-label">邮箱</label>

<div class="mws-form-item">

<input type="text" name="email" class="large" value="{{$user->email}}"/>

</div>

</div>

<div class="mws-form-row">

<label class="mws-form-label">电话</label>

<div class="mws-form-item">

<input type="text" name="phone" class="large" value="{{$user->phone}}"/>

</div>

</div>

</div>

<div class="mws-button-row">

{{csrf\_field()}}

{{method\_field("PUT")}}

<input type="submit" value="Submit" class="btn btn-danger" />

<input type="reset" value="Reset" class="btn " />

</div>

</form>

</div>

</div>

</body>

</html> @endsection

@section('title','后台用户修改')

* 1. **插入表单请求校验类操作**
     1. 创建表单校验类

php artisan make:request AdminUserinsert

* + 1. 给表单校验授权

public function authorize()

{

return true;

}

* + 1. 设置规则

public function rules()

{

return [

//会员名做规则设置'username'=>'required'

];

}

* + 1. 在Users导入校验类

use App\Http\Requests\AdminUserInsert;

* + 1. 显示错误信息

1. @if (count($errors) > 0)
2. <div class="alert alert-danger"> 3 <ul>

4 @foreach ($errors->all() as $error) 5 <li>{{ $error }}</li>

6 @endforeach

7 </ul>

8 </div>

* + 1. 自定义错误信息

1. public function messages(){
2. return[
3. 'username.required'=>'用户名不能为空'

4 ];

5 }

* + 1. 整体代码演示

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest; class UserInsert extends FormRequest

{

/\*\*

* Determine if the user is authorized to make this request.

\*

* @return bool

\*/

//给表单校验授权

public function authorize()

{

return true;

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62 }

63

}

/\*\*

* Get the validation rules that apply to the request.

\*

* @return array

\*/

//校验规则

public function rules()

{

return [

//会员名做规则设置'username'=>'required|regex:/\w{4,8}/|unique:users',

//密码和重复密码

'password'=>'required|regex:/\w{8,16}/', 'repassword'=>'required|regex:/\w{8,16}/|same:password',

//邮箱

'email'=>'required|email',

//电话'phone'=>'required|regex:/\d{11}/'

];

}

//自定义错误消息

public function messages(){ return[

'username.required'=>'用户名不能为空',

'username.regex'=>'用户名必须为4-8位的任意数字字母下划线', 'username.unique'=>'用户名重复',

'password.required'=>' 密 码 不 能 为 空 ', 'password.regex'=>'密码必须为8-16位的任意数字字母下划线', 'repassword.required'=>'重复密码不能为空',

'repassword.regex'=>'重复密码必须为8-16位的任意数字字母下划线', 'repassword.same'=>'两次密码不一致',

//邮箱

'email.required'=>'邮箱不能为空', 'email.email'=>'邮箱格式不符合',

//电话

'phone.required'=>'电话不能为空', 'phone.regex'=>'电话不符合规则'

];

}

# 第七章模型使用\*\*

## 创建模型

php artisan make:model User

* 1. **在模型类里定义属性**

与模型关联的数据表

protected $table = 'my\_ﬂights';

该模型是否被自动维护时间戳

public $timestamps = false;

可以被批量赋值的属性。

protected $ﬁllable = ['name'];

* 1. **修改器**

在获取数据的字段值后自动进行处理例如我们要对状态值进行转换

public function getStatusAttribute($value){

$status=[1=>"禁用",2=>"开启"]; return $status[$value];

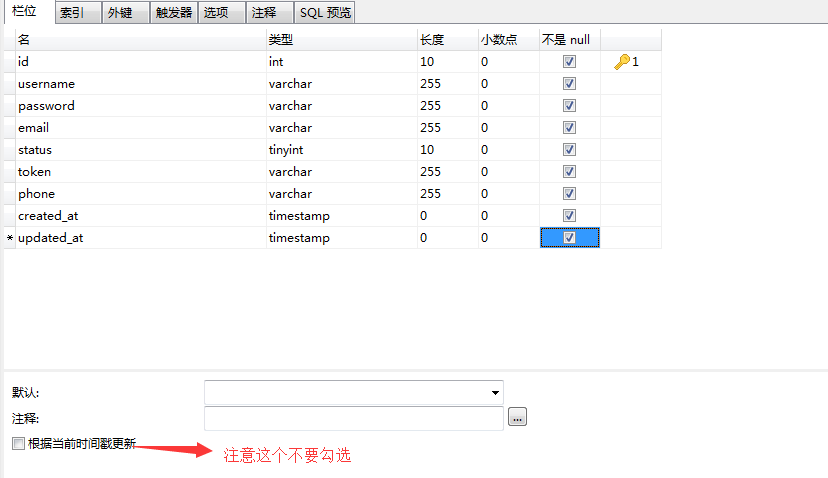
}

* 1. **模型关联**

一对一(例如会员和会员详情模块) hasOne(其他模型,关联字段)

一对多(例如会员和会员收货地址模块) hasMany(其他模型,关联字段)

* 1. **模型会员模块实战**
     1. 设计会员表



* + 1. 创建模型类Userss

php artisan make:model Models/Userss

* + 1. 模型类Userss代码

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model; class Userss extends Model

{

//定义与模型关联的数据表protected $table="users";

//主键

protected $primaryKey="id";

//设置是否需要自动维护时间戳 created\_at updated\_at public $timestamps =true;

/\*\*

* 可以被批量赋值的属性。

\*

* @var array

\*/

protected $fillable = ['username','password','email','status','token','phone'];

//修改器的方法

//对完成的数据(状态 status)做自动处理

public function getStatusAttribute($value){

$status=[1=>'禁用',2=>'激活']; return $status[$value];

26 }

27

28 }

29

* + 1. 控制器代码

1 <?php 2

3 namespace App\Http\Controllers\Admin; 4

1. use Illuminate\Http\Request;
2. use App\Http\Controllers\Controller;
3. //导入Hash类
4. use Hash;
5. //导入校验类
6. use App\Http\Requests\UserInsert;
7. //导入要调用的模型类
8. use App\Models\Userss;
9. class UsersController extends Controller 14 {

15 /\*\*

16 \* Display a listing of the resource. 17 \*

18 \* @return \Illuminate\Http\Response 19 \*/

1. //列表
2. public function index(Request $request) 22 {
3. //获取搜索关键词
4. $k=$request->input('keywords');
5. //获取列表数据
6. $user=Userss::where("username",'like',"%".$k."%")->paginate(3);
7. //加载模板
8. return view("Admin.Users.index",['user'=>$user,'request'=>$request->all()]); 29 }

30

31 /\*\*

32 \* Show the form for creating a new resource. 33 \*

34 \* @return \Illuminate\Http\Response 35 \*/

36 public function create() 37 {

1. //添加界面
2. // echo "this is add";
3. return view("Admin.Users.add"); 41

42 }

43

44 /\*\*

45 \* Store a newly created resource in storage. 46 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @return \Illuminate\Http\Response 49 \*/

50 public function store(UserInsert $request) 51 {

1. // dd($request->all());
2. $data=$request->except(['repassword','\_token']);
3. //对密码做加密
4. $data['password']=Hash::make($data['password']);
5. $data['status']=1;
6. $data['token']=str\_random(50);
7. // dd($data);
8. if(Userss::create($data)){
9. return redirect("/adminusers")->with('success','数据添加成功');
10. }else{
11. return redirect("/adminusers/create")->with('error','数据添加失败'); 63 }

64

65 }

66

67 /\*\*

68 \* Display the specified resource. 69 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 72 \*/

73 public function show($id) 74 {

1. //获取用户对应得详情信息
2. // echo $id;
3. $info=Userss::find($id)->info;
4. // dd($data);
5. //加载模板 分配数据
6. return view("Admin.Users.info",['info'=>$info]); 81 }

82

1. //获取会员地址
2. public function address($id){
3. // echo $id;
4. $address=Userss::find($id)->useraddress;
5. // dd($address);
6. return view("Admin.Users.address",['address'=>$address]); 89 }

90

91 /\*\*

92 \* Show the form for editing the specified resource. 93 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 96 \*/

97 public function edit($id)

98

99

100

101

102

103

104

105

106

107

{

}

/\*\*

// echo "这是修改数据的id:".$id;

//获取需要修改的数据

$user=Userss::find($id);

// dd($user);

return view("Admin.Users.edit",['user'=>$user]);

108 \* Update the specified resource in storage.

109 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @param int $id
3. \* @return \Illuminate\Http\Response 113 \*/

114

1. //执行修改
2. public function update(Request $request, $id)

117 {

1. // dd($request->all());
2. $data=$request->except(['\_token','\_method']);
3. //密码加密
4. $data['password']=Hash::make($data['password']);
5. if(Userss::where("id","=",$id)->update($data)){
6. return redirect("/adminusers")->with('success',"修改成功");
7. }else{
8. return redirect("/adminusers/$id",'数据修改失败');

126 }

127 }

128

129 /\*\*

130 \* Remove the specified resource from storage.

131 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 134 \*/

135 public function destroy($id)

136 {

1. // echo $id;
2. if(Userss::destroy($id)){
3. return redirect("/adminusers")->with('success','数据删除成功');
4. }else{
5. return redirect("/adminusers")->with('error','数据删除失败');

142 }

143 }

144

145 // //删除

1. // public function del(Request $request){
2. // $id=$request->input('id');
3. // // echo $id;
4. // if(DB::table("users")->where("id",'=',$id)->delete()){
5. // echo 1;

151

152

153

154

155

156

// }else{

// echo 0;

// }

// }

}

## 命名空间

* + 1. 命名空间定义

命名空间是一种封装事物的方法,一般包含三大类型数据，常量 函数 类

* + 1. 命名空间基本使用

class A{

public $name="admin";

}

echo \one\PI;

\one\test();

实例化类A

$a=new \one\A();

echo $a->name;

namespace one;

const PI=3.1415926;

function test(){

echo "this is test";

}

1

2

3

4

5

6

7

8

9

10

11

12

13

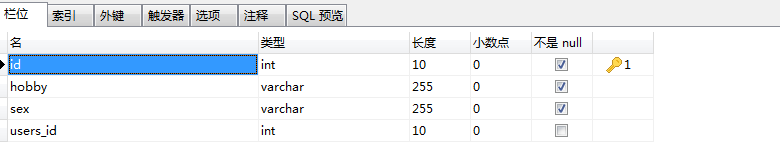
14

15

* + 1. Laravel里命名空间使用

在框架里命名空间所在的路径和文件所在路径是一致的,例如: project\app\Models\Userss =>use App\Models\Userss;

* 1. **会员模块之关联用户详情模块 用户地址模块实战**
     1. **会员模块之关联用户详情模块**
        1. 设计用户详情表 users\_info 表结构如下



* + - 1. 创建Userinfo模型类

php artisan make:model Models/Userinfo

* + - 1. Userinfo模型类代码

namespace App\Models;

use Illuminate\Database\Eloquent\Model; class Userinfo extends Model

{

//定义与模型关联的数据表protected $table="users\_info";

//主键

protected $primaryKey="id";

//设置是否需要自动维护时间戳 created\_at updated\_at public $timestamps =false;

/\*\*

* 可以被批量赋值的属性。

\*

* @var array

\*/

protected $fillable = ['hobby','sex','users\_id'];

}

<?php

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

* + - 1. Userss模型类代码

1

2

3

4

5

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

class Userss extends Model

{

//定义与模型关联的数据表protected $table="users";

//主键

protected $primaryKey="id";

//设置是否需要自动维护时间戳 created\_at updated\_at public $timestamps =true;

/\*\*

* 可以被批量赋值的属性。

\*

* @var array

\*/

protected $fillable = ['username','password','email','status','token','phone'];

//修改器的方法

//对完成的数据(状态 status)做自动处理

public function getStatusAttribute($value){

$status=[1=>'禁用',2=>'激活']; return $status[$value];

}

//获取与用户关联的详情信息

public function info(){

return $this->hasOne('App\Models\Userinfo','users\_id');

}

}

* + - 1. Users/index.blade.php模板

1. @extends("Admin.AdminPublic.public")
2. @section("admin")
3. <html>
4. <head></head>
5. <script type="text/javascript" src="/static/js/jquery-1.8.3.min.js"></script>
6. <body>
7. <div class="mws-panel grid\_8">
8. <div class="mws-panel-header">
9. <span><i class="icon-table"></i>会员列表</span>
10. </div>
11. <div class="mws-panel-body no-padding">
12. <div id="DataTables\_Table\_1\_wrapper" class="dataTables\_wrapper" role="grid">
13. <form action="/adminusers" method="get">
14. <div class="dataTables\_filter" id="DataTables\_Table\_1\_filter">
15. <label>搜索: <input type="text" aria-controls="DataTables\_Table\_1" name="keywords" value="{{$request['keywords'] or ''}}"/></label><input type="submit" value="搜索">
16. </div>
17. </form>
18. <table class="mws-datatable-fn mws-table dataTable" id="DataTables\_Table\_1" aria- describedby="DataTables\_Table\_1\_info">
19. <thead>
20. <tr role="row">
21. <th class="sorting\_asc" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-sort="ascending" aria- label="Rendering engine: activate to sort column descending" style="width: 157px;">ID</th>
22. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Browser: activate to

sort column ascending" style="width: 209px;">会员名</th>

1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Platform(s): activate

to sort column ascending" style="width: 191px;">邮箱</th>

1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Platform(s): activate

to sort column ascending" style="width: 191px;">状态</th>

25

1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Engine version: activate to sort column ascending" style="width: 137px;">电话</th>
2. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Engine version: activate to sort column ascending" style="width: 137px;">添加时间</th>
3. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Engine version: activate to sort column ascending" style="width: 137px;">修改时间</th>
4. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="CSS grade: activate to sort column ascending" style="width: 101px;">操作</th>

30 </tr>

1. </thead>
2. <tbody role="alert" aria-live="polite" aria-relevant="all">
3. @foreach($user as $row)
4. <tr class="odd">
5. <td class=" sorting\_1">{{$row->id}}</td>
6. <td class=" ">{{$row->username}}</td>
7. <td class=" ">{{$row->email}}</td>
8. <td class=" ">{{$row->status}}</td>
9. <td class=" ">{{$row->phone}}</td>
10. <td class=" ">{{$row->created\_at}}</td>
11. <td class=" ">{{$row->updated\_at}}</td>
12. <td class=" ">
13. <a href="/adminusers/{{$row->id}}" class="btn btn-success">用户详情</a>
14. <form action="/adminusers/{{$row->id}}" method="post">
15. {{csrf\_field()}}
16. {{method\_field("DELETE")}}
17. <button class="btn btn-info">删除</button>
18. </form>
19. <a href="javascript:void(0)" class="btn btn-success del" >删除</a><a href="/adminusers/{{$row->id}}/edit" class="btn btn-danger">修改</a></td>

50 </tr>

51 @endforeach

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

</tbody>

</table>

<div class="dataTables\_paginate paging\_full\_numbers" id="pages">

{{$user->appends($request)->render()}}

</div>

</div>

</div>

</div>

</body>

<script type="text/javascript">

// alert($);

//获取删除按钮 绑定单击事件

$(".del").click(function(){

//Ajax id=$(this).parents("tr").find("td:first").html(); s=$(this).parents("tr");

// alert(id);

//Ajax

$.get('/adminusersdel',{id:id},function(data){

// alert(data); if(data==1){

alert("删除成功");

//把删除数据所在的tr从dom里移除

s.remove();

}else{

alert("删除失败");

}

});

});

</script>

</html> @endsection

@section('title','后台用户列表')

* + - 1. Users控制器代码

1

2

3

4

5

6

7

8

9

10

11

12

13

14

<?php

namespace App\Http\Controllers\Admin;

use Illuminate\Http\Request;

use App\Http\Controllers\Controller;

//导入Hash类use Hash;

//导入校验类

use App\Http\Requests\UserInsert;

//导入要调用的模型类

use App\Models\Userss;

class UsersController extends Controller

{

15 /\*\*

16 \* Display a listing of the resource. 17 \*

18 \* @return \Illuminate\Http\Response 19 \*/

1. //列表
2. public function index(Request $request) 22 {
3. //获取搜索关键词
4. $k=$request->input('keywords');
5. //获取列表数据
6. $user=Userss::where("username",'like',"%".$k."%")->paginate(3);
7. //加载模板
8. return view("Admin.Users.index",['user'=>$user,'request'=>$request->all()]); 29 }

30

31 /\*\*

32 \* Show the form for creating a new resource. 33 \*

34 \* @return \Illuminate\Http\Response 35 \*/

36 public function create() 37 {

1. //添加界面
2. // echo "this is add";
3. return view("Admin.Users.add"); 41

42 }

43

44 /\*\*

45 \* Store a newly created resource in storage. 46 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @return \Illuminate\Http\Response 49 \*/

50 public function store(UserInsert $request) 51 {

1. // dd($request->all());
2. $data=$request->except(['repassword','\_token']);
3. //对密码做加密
4. $data['password']=Hash::make($data['password']);
5. $data['status']=1;
6. $data['token']=str\_random(50);
7. // dd($data);
8. if(Userss::create($data)){
9. return redirect("/adminusers")->with('success','数据添加成功');
10. }else{
11. return redirect("/adminusers/create")->with('error','数据添加失败'); 63 }

64

65 }

66

67 /\*\*

68 \* Display the specified resource. 69 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 72 \*/

73 public function show($id) 74 {

1. //获取用户对应得详情信息
2. // echo $id;
3. $info=Userss::find($id)->info;
4. // dd($data);
5. //加载模板 分配数据
6. return view("Admin.Users.info",['info'=>$info]); 81 }

82

83 /\*\*

84 \* Show the form for editing the specified resource. 85 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 88 \*/

89 public function edit($id) 90 {

1. // echo "这是修改数据的id:".$id;
2. //获取需要修改的数据
3. $user=Userss::find($id);
4. // dd($user);
5. return view("Admin.Users.edit",['user'=>$user]); 96

97 }

98

99 /\*\*

100 \* Update the specified resource in storage.

101 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @param int $id
3. \* @return \Illuminate\Http\Response 105 \*/

106

1. //执行修改
2. public function update(Request $request, $id)

109 {

1. // dd($request->all());
2. $data=$request->except(['\_token','\_method']);
3. //密码加密
4. $data['password']=Hash::make($data['password']);
5. if(Userss::where("id","=",$id)->update($data)){
6. return redirect("/adminusers")->with('success',"修改成功");
7. }else{
8. return redirect("/adminusers/$id",'数据修改失败');

118 }

119 }

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

/\*\*

* Remove the specified resource from storage.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function destroy($id)

{

// echo $id; if(Userss::destroy($id)){

return redirect("/adminusers")->with('success','数据删除成功');

}else{

return redirect("/adminusers")->with('error','数据删除失败');

}

}

// //删除

// public function del(Request $request){

//

//

//

//

//

//

//

// }

$id=$request->input('id');

// echo $id;

if(DB::table("users")->where("id",'=',$id)->delete()){ echo 1;

}else{

echo 0;

}

}

* + - 1. 创建Users/info.blade.php 模板 代码如下

1. @extends("Admin.AdminPublic.public")
2. @section("admin")
3. <html>
4. <head></head>
5. <script type="text/javascript" src="/static/js/jquery-1.8.3.min.js"></script>
6. <body>
7. <div class="mws-panel grid\_8">
8. <div class="mws-panel-header">
9. <span><i class="icon-table"></i>会员详情</span>
10. </div>
11. <div class="mws-panel-body no-padding">
12. <div id="DataTables\_Table\_1\_wrapper" class="dataTables\_wrapper" role="grid">
13. <table class="mws-datatable-fn mws-table dataTable" id="DataTables\_Table\_1" aria- describedby="DataTables\_Table\_1\_info">
14. <thead>
15. <tr role="row">
16. <th class="sorting\_asc" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-sort="ascending" aria- label="Rendering engine: activate to sort column descending" style="width: 157px;">ID</th>
17. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Browser: activate to

sort column ascending" style="width: 209px;">兴趣</th>

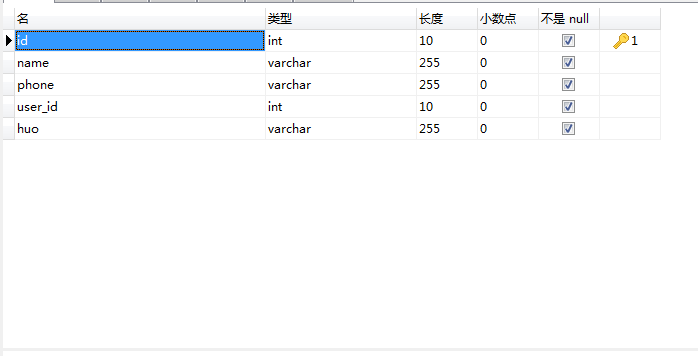
1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Platform(s): activate

to sort column ascending" style="width: 191px;">性别</th>

19 </tr>

1. </thead>
2. <tbody role="alert" aria-live="polite" aria-relevant="all">
3. <tr class="odd">
4. <td class=" sorting\_1">{{$info->id}}</td>
5. <td class=" ">{{$info->hobby}}</td>
6. <td class=" ">{{$info->sex}}</td>

26 </tr>

1. </tbody>
2. </table>
3. <div class="dataTables\_paginate paging\_full\_numbers" id="pages">
4. </div>
5. </div>
6. </div>
7. </div>
8. </body>
9. </html>
10. @endsection
11. @section('title','后台会员详情')
    * 1. **会员模块之关联用户地址模块**
         1. 设计用户地址表 address 表结构如下
         2. 创建Useraddress模型类

php artisan make:model Models/Useraddress

* + - 1. Useraddress模型类代码

namespace App\Models;

use Illuminate\Database\Eloquent\Model; class Useraddress extends Model

{

//

//定义与模型关联的数据表protected $table="address";

//主键

protected $primaryKey="id";

//设置是否需要自动维护时间戳 created\_at updated\_at public $timestamps =false;

/\*\*

* 可以被批量赋值的属性。

\*

* @var array

\*/

protected $fillable = ['name','phone','users\_id','huo'];

}

<?php

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

* + - 1. Userss模型类代码

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model; class Userss extends Model

{

//定义与模型关联的数据表protected $table="users";

//主键

protected $primaryKey="id";

//设置是否需要自动维护时间戳 created\_at updated\_at public $timestamps =true;

/\*\*

* 可以被批量赋值的属性。

\*

* @var array

\*/

protected $fillable = ['username','password','email','status','token','phone'];

//修改器的方法

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

//对完成的数据(状态 status)做自动处理

public function getStatusAttribute($value){

$status=[1=>'禁用',2=>'激活']; return $status[$value];

}

//获取与用户关联的详情信息public function info(){

return $this->hasOne('App\Models\Userinfo','users\_id');

}

//获取与用户关联的地址信息(1对多) public function useraddress(){

return $this->hasMany('App\Models\Useraddress','user\_id');

}

}

* + - 1. Users/index.blade.php模板
         1. @extends("Admin.AdminPublic.public")
         2. @section("admin")
         3. <html>
         4. <head></head>
         5. <script type="text/javascript" src="/static/js/jquery-1.8.3.min.js"></script>
         6. <body>
         7. <div class="mws-panel grid\_8">
         8. <div class="mws-panel-header">
         9. <span><i class="icon-table"></i>会员列表</span>
         10. </div>
         11. <div class="mws-panel-body no-padding">
         12. <div id="DataTables\_Table\_1\_wrapper" class="dataTables\_wrapper" role="grid">
         13. <form action="/adminusers" method="get">
         14. <div class="dataTables\_filter" id="DataTables\_Table\_1\_filter">
         15. <label>搜索: <input type="text" aria-controls="DataTables\_Table\_1" name="keywords" value="{{$request['keywords'] or ''}}"/></label><input type="submit" value="搜索">
         16. </div>
         17. </form>
         18. <table class="mws-datatable-fn mws-table dataTable" id="DataTables\_Table\_1" aria- describedby="DataTables\_Table\_1\_info">
         19. <thead>
         20. <tr role="row">
         21. <th class="sorting\_asc" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-sort="ascending" aria- label="Rendering engine: activate to sort column descending" style="width: 157px;">ID</th>
         22. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Browser: activate to

sort column ascending" style="width: 209px;">会员名</th>

* + - * 1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Platform(s): activate

to sort column ascending" style="width: 191px;">邮箱</th>

* + - * 1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Platform(s): activate

to sort column ascending" style="width: 191px;">状态</th>

25

1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Engine version: activate to sort column ascending" style="width: 137px;">电话</th>
2. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Engine version: activate to sort column ascending" style="width: 137px;">添加时间</th>
3. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Engine version: activate to sort column ascending" style="width: 137px;">修改时间</th>
4. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="CSS grade: activate to sort column ascending" style="width: 101px;">操作</th>

|  |  |  |
| --- | --- | --- |
| 30 |  | </tr> |
| 31 |  | </thead> |
| 32 |  | <tbody role="alert" aria-live="polite" aria-relevant="all"> |
| 33 |  | @foreach($user as $row) |
| 34 |  | <tr class="odd"> |
| 35 |  | <td class=" sorting\_1">{{$row->id}}</td> |
| 36 |  | <td class=" ">{{$row->username}}</td> |
| 37 |  | <td class=" ">{{$row->email}}</td> |
| 38 |  | <td class=" ">{{$row->status}}</td> |
| 39 |  | <td class=" ">{{$row->phone}}</td> |
| 40 |  | <td class=" ">{{$row->created\_at}}</td> |
| 41 |  | <td class=" ">{{$row->updated\_at}}</td> |
| 42 |  | <td class=" "> |
| 43 |  | <a href="/adminusers/{{$row->id}}" class="btn btn-success">用户详情</a> |
| 44 |  | <a href="/adminuseraddress/{{$row->id}}" class="btn btn-success">用户收货地址 |
|  | </a> |  |
| 45 |  |  |
| 46 |  | <form action="/adminusers/{{$row->id}}" method="post"> |
| 47 |  | {{csrf\_field()}} |
| 48 |  | {{method\_field("DELETE")}} |
| 49 |  | <button class="btn btn-info">删除</button> |
| 50 |  | </form> |
| 51  52  53  54  55  56  57  58  59  60  61 | <a href="javascript:void(0)" class="btn btn-success del" >删除</a><a href="/adminusers/{{$row->id}}/edit" class="btn btn-danger">修改</a></td>  </tr> @endforeach  </tbody>  </table>  <div class="dataTables\_paginate paging\_full\_numbers" id="pages">  {{$user->appends($request)->render()}}  </div>  </div>  </div>  </div> | |

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

</body>

<script type="text/javascript">

// alert($);

//获取删除按钮 绑定单击事件

$(".del").click(function(){

//Ajax id=$(this).parents("tr").find("td:first").html(); s=$(this).parents("tr");

// alert(id);

//Ajax

$.get('/adminusersdel',{id:id},function(data){

// alert(data); if(data==1){

alert("删除成功");

//把删除数据所在的tr从dom里移除

s.remove();

}else{

alert("删除失败");

}

});

});

</script>

</html> @endsection

@section('title','后台用户列表')

* + - 1. Users控制器代码

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

<?php

namespace App\Http\Controllers\Admin;

use Illuminate\Http\Request;

use App\Http\Controllers\Controller;

//导入Hash类use Hash;

//导入校验类

use App\Http\Requests\UserInsert;

//导入要调用的模型类

use App\Models\Userss;

class UsersController extends Controller

{

/\*\*

* Display a listing of the resource.

\*

* @return \Illuminate\Http\Response

\*/

//列表

public function index(Request $request)

{

23

24

25

26

27

28

29 }

30

31 /\*\*

//获取搜索关键词

$k=$request->input('keywords');

//获取列表数据

$user=Userss::where("username",'like',"%".$k."%")->paginate(3);

//加载模板

return view("Admin.Users.index",['user'=>$user,'request'=>$request->all()]);

32 \* Show the form for creating a new resource. 33 \*

34 \* @return \Illuminate\Http\Response 35 \*/

36 public function create() 37 {

1. //添加界面
2. // echo "this is add";
3. return view("Admin.Users.add"); 41

42 }

43

44 /\*\*

45 \* Store a newly created resource in storage. 46 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @return \Illuminate\Http\Response 49 \*/

50 public function store(UserInsert $request) 51 {

1. // dd($request->all());
2. $data=$request->except(['repassword','\_token']);
3. //对密码做加密
4. $data['password']=Hash::make($data['password']);
5. $data['status']=1;
6. $data['token']=str\_random(50);
7. // dd($data);
8. if(Userss::create($data)){
9. return redirect("/adminusers")->with('success','数据添加成功');
10. }else{
11. return redirect("/adminusers/create")->with('error','数据添加失败'); 63 }

64

65 }

66

67 /\*\*

68 \* Display the specified resource. 69 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 72 \*/

73 public function show($id) 74 {

75 //获取用户对应得详情信息

76

77

78

79

80

81 }

82

// echo $id;

$info=Userss::find($id)->info;

// dd($data);

//加载模板 分配数据

return view("Admin.Users.info",['info'=>$info]);

1. //获取会员地址
2. public function address($id){
3. // echo $id;
4. $address=Userss::find($id)->useraddress;
5. // dd($address);
6. return view("Admin.Users.address",['address'=>$address]); 89 }

90

91 /\*\*

92 \* Show the form for editing the specified resource. 93 \*

1. \* @param int $id
2. \* @return \Illuminate\Http\Response 96 \*/

97 public function edit($id) 98 {

1. // echo "这是修改数据的id:".$id;
2. //获取需要修改的数据
3. $user=Userss::find($id);
4. // dd($user);
5. return view("Admin.Users.edit",['user'=>$user]); 104

105 }

106

107 /\*\*

108 \* Update the specified resource in storage.

109 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @param int $id
3. \* @return \Illuminate\Http\Response 113 \*/

114

1. //执行修改
2. public function update(Request $request, $id)

117 {

1. // dd($request->all());
2. $data=$request->except(['\_token','\_method']);
3. //密码加密
4. $data['password']=Hash::make($data['password']);
5. if(Userss::where("id","=",$id)->update($data)){
6. return redirect("/adminusers")->with('success',"修改成功");
7. }else{
8. return redirect("/adminusers/$id",'数据修改失败');

126 }

127 }

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

/\*\*

* Remove the specified resource from storage.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function destroy($id)

{

// echo $id; if(Userss::destroy($id)){

return redirect("/adminusers")->with('success','数据删除成功');

}else{

return redirect("/adminusers")->with('error','数据删除失败');

}

}

// //删除

// public function del(Request $request){

//

//

//

//

//

//

//

// }

$id=$request->input('id');

// echo $id;

if(DB::table("users")->where("id",'=',$id)->delete()){ echo 1;

}else{

echo 0;

}

}

* + - 1. 创建Users/address.blade.php 模板 代码如下

1. @extends("Admin.AdminPublic.public")
2. @section("admin")
3. <html>
4. <head></head>
5. <script type="text/javascript" src="/static/js/jquery-1.8.3.min.js"></script>
6. <body>
7. <div class="mws-panel grid\_8">
8. <div class="mws-panel-header">
9. <span><i class="icon-table"></i>会员收货地址</span>
10. </div>
11. <div class="mws-panel-body no-padding">
12. <div id="DataTables\_Table\_1\_wrapper" class="dataTables\_wrapper" role="grid">
13. <table class="mws-datatable-fn mws-table dataTable" id="DataTables\_Table\_1" aria- describedby="DataTables\_Table\_1\_info">
14. <thead>
15. <tr role="row">
16. <th class="sorting\_asc" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-sort="ascending" aria- label="Rendering engine: activate to sort column descending" style="width: 157px;">ID</th>
17. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Browser: activate to

sort column ascending" style="width: 209px;">收货人</th>

1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Platform(s): activate

to sort column ascending" style="width: 191px;">电话</th>

1. <th class="sorting" role="columnheader" tabindex="0" aria- controls="DataTables\_Table\_1" rowspan="1" colspan="1" aria-label="Platform(s): activate

to sort column ascending" style="width: 191px;">收货地址</th>

20

21 </tr>

1. </thead>
2. <tbody role="alert" aria-live="polite" aria-relevant="all">
3. @foreach($address as $row)
4. <tr class="odd">
5. <td class=" sorting\_1">{{$row->id}}</td>
6. <td class=" ">{{$row->name}}</td>
7. <td class=" ">{{$row->phone}}</td>
8. <td class=" ">{{$row->huo}}</td>

30 </tr>

1. @endforeach
2. </tbody>
3. </table>
4. <div class="dataTables\_paginate paging\_full\_numbers" id="pages">
5. </div>
6. </div>
7. </div>
8. </div>
9. </body>
10. </html>
11. @endsection
12. @section('title','后台会员收货地址')

# 第八章 自定义函数和自定义类以及debug安装\*\*

## 设置自定义函数

* + 1. 创建文件 app/Library/helper.php 代码如下

1. <?php
2. function fun(){
3. echo "这是自定义函数"; 4 }

5 ?>

* + 1. 在项目下的composer.json中添加信息

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

"autoload": {

"classmap": [

"database"

],

"psr-4": {

"App\\": "app/"

},

"files":[

"app/Library/helper.php"

]

},

* + 1. 项目里cmd里执行 composer dump-autoload
    2. 控制器调用

在控制器方法里直接 调用

fun()

* 1. **设置自定义类**
     1. 创建类文件 app/Library/A.php 代码如下

?>

}

<?php

class A{

//方法 发送短信校验码

public function sendphone(){ echo "this is phone";

}

1

2

3

4

5

6

7

8

9

* + 1. 在项目下的composer.json中的classmap中添加信息

1 "autoload": { 2

3 "classmap": [

4

5

6

7

8

9

10

11

12

13

14

15

16

"database",

"app/Library/A.php"

],

"psr-4": {

"App\\": "app/"

}

},

* + 1. 项目里cmd里执行 composer dump-autoload
    2. 控制器调用

}

$class->sendphone();

//实例化类A

$class = new A();

{

public function xxoo()

use A;//导入自定义类A

1

2

3

4

5

6

7

8

9

10

* 1. **debugbar 调试工具安装**
     1. 在conﬁg/app.php里面的providers添加 如下代码

Barryvdh\Debugbar\ServiceProvider::class, 如 下

App\Providers\AppServiceProvider::class,

App\Providers\AuthServiceProvider::class,

// App\Providers\BroadcastServiceProvider::class, App\Providers\EventServiceProvider::class, App\Providers\RouteServiceProvider::class, Barryvdh\Debugbar\ServiceProvider::class,

1

2

3

4

5

6

7

* + 1. 在项目cmd下执行 命令

composer require barryvdh/laravel-debugbar

* 1. **Laravel 框架结合云之讯 短信接口调用**
     1. 将云之讯三方类Ucpaas.php放入app/Library 下
     2. 在项目下的composer.json中的classmap中添加信息

1. "classmap": [
2. "database/seeds",
3. "database/factories",
4. "app/Library/A.php",
5. "app/Library/Ucpaas.php" 6 ],
   * 1. 项目里cmd里执行 composer dump-autoload
     2. 在app/Library/helper.php 里写入如下代码

?>

}

echo $ucpass->SendSms($appid,$templateid,$param,$mobile,$uid);

//70字内（含70字）计一条，超过70字，按67字/条计费，超过长度短信平台将会自动分割为多条 发送。分割后的多条短信将按照具体占用条数计费。

19

20

21

22

23

$mobile = $p;

$uid = "";

15

16

17

18

留空

$param = rand(1,10000); //多个参数使用英文逗号隔开（如：param=“a,b,c”），如为参数则

14

板ID

//初始化 $options必填

$ucpass = new Ucpaas($options);

$appid = "xxxxxxxxxxxxxxxxxxxxxxx"; //应用的ID，可在开发者控制台内的短信产品下查看

$templateid = "xxx"; //可在后台短信产品→选择接入的应用→短信模板-模板ID，查看该模

<?php

//发送短信校验码

function sendsphone($p){

//初始化必填

//填写在开发者控制台首页上的Account Sid

$options['accountsid']='xxxxxxxxxxx';

//填写在开发者控制台首页上的Auth Token

$options['token']='xxxxxxxxxxx';

1

2

3

4

5

6

7

8

9

10

11

12

13

* + 1. 控制器方法里直接调用

sendsphone(电话号码);

* 1. **Laravel 框架支付宝接口调用**
     1. 将支付宝demo所有文件放在Library文件夹下
     2. 在app/Library/helper.php下加入如下代码

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

<?php

//图片的缩放

// function fun(){

// echo "这是公共的图片缩放函数";

// }

//发送短信校验码

function sendsphone($p){

//初始化必填

//填写在开发者控制台首页上的Account Sid

$options['accountsid']='xxxxxxxxxxx';

//填写在开发者控制台首页上的Auth Token

$options['token']='xxxxxxxxxxx';

//初始化 $options必填

$ucpass = new Ucpaas($options);

$appid = "xxxxxxxxxxxxxxxxxxxxxxx"; //应用的ID，可在开发者控制台内的短信产品下查看

$templateid = "xxx"; //可在后台短信产品→选择接入的应用→短信模板-模板ID，查看该模

板ID

19

$param = rand(1,10000); //多个参数使用英文逗号隔开（如：param=“a,b,c”），如为参数则

留空

20

21

22

$mobile = $p;

$uid = "";

23 //70字内（含70字）计一条，超过70字，按67字/条计费，超过长度短信平台将会自动分割为多条 发送。分割后的多条短信将按照具体占用条数计费。

24

25 echo $ucpass->SendSms($appid,$templateid,$param,$mobile,$uid); 26

27 }

28

29 function pay(){ 30 /\* \*

31 \* 功能：即时到账交易接口接入页

32 \* 版本：3.4

33 \* 修改日期：2016-03\*08

1. \* 说明：
2. \* 以下代码只是为了方便商户测试而提供的样例代码，商户可以根据自己网站的需要，按照技术文档编写,并非一定要使用该代码。
3. \* 该代码仅供学习和研究支付宝接口使用，只是提供一个参考。

37

38 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*注意\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

39

1. \*如果您在接口集成过程中遇到问题，可以按照下面的途径来解决
2. \*1、开发文档中心（https://doc.open.alipay.com/doc2/detail.htm? spm=a219a.7629140.0.0.KvddfJ&treeId=62&articleId=103740&docType=1）
3. \*2、商户帮助中心（https://cshall.alipay.com/enterprise/help\_detail.htm?help\_id=473888）
4. \*3、支持中心（https://support.open.alipay.com/alipay/support/index.htm）

44

45 \*如果想使用扩展功能,请按文档要求,自行添加到parameter数组即可。 46 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

47 \*/

48

1. require\_once("alipay.config.php");
2. require\_once("lib/alipay\_submit.class.php"); 51

52 /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*请求参数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

53

1. //商户订单号，商户网站订单系统中唯一订单号，必填
2. $out\_trade\_no = rand(1,10000); 56
3. //订单名称，必填
4. $subject = '电影测试';

59

60 //付款金额，必填

61 $total\_fee = '0.01';

62

1. //商品描述，可空
2. $body = '描述';

65

66

67

68

69

70 /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

71

1. //构造要请求的参数数组，无需改动
2. $parameter = array(
3. "service" => $alipay\_config['service'],
4. "partner" => $alipay\_config['partner'],
5. "seller\_id" => $alipay\_config['seller\_id'],
6. "payment\_type" => $alipay\_config['payment\_type'],
7. "notify\_url" => $alipay\_config['notify\_url'],
8. "return\_url" => $alipay\_config['return\_url'], 80
9. "anti\_phishing\_key"=>$alipay\_config['anti\_phishing\_key'],
10. "exter\_invoke\_ip"=>$alipay\_config['exter\_invoke\_ip'],
11. "out\_trade\_no" => $out\_trade\_no,
12. "subject" => $subject,
13. "total\_fee" => $total\_fee,
14. "body" => $body,
15. "\_input\_charset" => trim(strtolower($alipay\_config['input\_charset']))
16. //其他业务参数根据在线开发文档，添加参数.文档地址:https://doc.open.alipay.com/doc2/detail.htm?

spm=a219a.7629140.0.0.kiX33I&treeId=62&articleId=103740&docType=1 89 //如"参数名"=>"参数值"

90

91 );

92

1. //建立请求
2. $alipaySubmit = new AlipaySubmit($alipay\_config);
3. $html\_text = $alipaySubmit->buildRequestForm($parameter,"get", "确认");
4. echo $html\_text; 97 }

98 ?>

* + 1. 在app/Library/alipay.conﬁg.php 配置调用接口参数秘钥和通知页面路径

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

<?php

/\* \*

* 配置文件

\* 版本：3.4

\* 修改日期：2016-03-08

* 说明：
* 以下代码只是为了方便商户测试而提供的样例代码，商户可以根据自己网站的需要，按照技术文档编写,并 非一定要使用该代码。
* 该代码仅供学习和研究支付宝接口使用，只是提供一个参考。
* 安全校验码查看时，输入支付密码后，页面呈灰色的现象，怎么办？
* 解决方法：
* 1、检查浏览器配置，不让浏览器做弹框屏蔽设置
* 2、更换浏览器或电脑，重新登录查询。

\*/

//↓↓↓↓↓↓↓↓↓↓请在这里配置您的基本信息↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓

//合作身份者ID，签约账号，以2088开头由16位纯数字组成的字符串，查看地址：

https://b.alipay.com/order/pidAndKey.htm

18

19

$alipay\_config['partner']

= 'xxxxxxxx';

1. //收款支付宝账号，以2088开头由16位纯数字组成的字符串，一般情况下收款账号就是签约账号
2. $alipay\_config['seller\_id'] = $alipay\_config['partner']; 22
3. // MD5密钥，安全检验码，由数字和字母组成的32位字符串，查看地址：

https://b.alipay.com/order/pidAndKey.htm

1. $alipay\_config['key'] = 'xxxxxxx'; 25
2. // 服务器异步通知页面路径 需[http://格式的完整路径](http://格式的完整路径/)，不能加?id=123这类自定义参数，必须外网可以

正常访问

1. $alipay\_config['notify\_url'] = ["http://商户网址/create\_direct\_pay\_by\_user-PHP-UTF-](http://商户网址/create_direct_pay_by_user-PHP-UTF-)

8/notify\_url.php";

28

1. // 页面跳转同步通知页面路径 需[http://格式的完整路径](http://格式的完整路径/)，不能加?id=123这类自定义参数，必须外网可以正常访问
2. $alipay\_config['return\_url'] = ["http://www.o2o28laravel.com/returnurl](http://www.o2o28laravel.com/returnurl)"; 31
3. //签名方式
4. $alipay\_config['sign\_type'] = strtoupper('MD5'); 34
5. //字符编码格式 目前支持 gbk 或 utf-8
6. $alipay\_config['input\_charset']= strtolower('utf-8'); 37
7. //ca证书路径地址，用于curl中ssl校验
8. //请保证cacert.pem文件在当前文件夹目录中
9. $alipay\_config['cacert'] = getcwd().'\\cacert.pem'; 41
10. //访问模式,根据自己的服务器是否支持ssl访问，若支持请选择https；若不支持请选择http
11. $alipay\_config['transport'] = 'http'; 44
12. // 支付类型 ，无需修改
13. $alipay\_config['payment\_type'] = "1"; 47
14. // 产品类型，无需修改
15. $alipay\_config['service'] = "create\_direct\_pay\_by\_user"; 50

51 //↑↑↑↑↑↑↑↑↑↑请在这里配置您的基本信息↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑

52

53

54 //↓↓↓↓↓↓↓↓↓↓ 请在这里配置防钓鱼信息，如果没开通防钓鱼功能，为空即可 ↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓

55

1. // 防钓鱼时间戳 若要使用请调用类文件submit中的query\_timestamp函数
2. $alipay\_config['anti\_phishing\_key'] = ""; 58
3. // 客户端的IP地址 非局域网的外网IP地址，如：221.0.0.1
4. $alipay\_config['exter\_invoke\_ip'] = ""; 61

62 //↑↑↑↑↑↑↑↑↑↑请在这里配置防钓鱼信息，如果没开通防钓鱼功能，为空即可 ↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑

63

64 ?>

* + 1. 创建控制器Pay

php artisan make controller Home/PayController --resource

* + 1. 路由文件代码

1. //支付宝接口调用
2. Route::get("/pays","Home\PayController@pays");
3. //通知给客户端的界面
4. Route::get("/returnurl","Home\PayController@returnurl");
   * 1. 控制器代码

1 <?php 2

3 namespace App\Http\Controllers\Home; 4

1. use Illuminate\Http\Request;
2. use App\Http\Controllers\Controller; 7

8 class PayController extends Controller 9 {

10 /\*\*

11 \* Display a listing of the resource. 12 \*

13 \* @return \Illuminate\Http\Response 14 \*/

15 public function index()

16 {

17 }

18

19 /\*\*

20 \* Show the form for creating a new resource. 21 \*

22 \* @return \Illuminate\Http\Response 23 \*/

24 public function create() 25 {

26 //

27 }

28

29 /\*\*

30 \* Store a newly created resource in storage. 31 \*

1. \* @param \Illuminate\Http\Request $request
2. \* @return \Illuminate\Http\Response 34 \*/

35 public function store(Request $request) 36 {

37 //

38 }

39

40 /\*\*

41 \* Display the specified resource.

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94 }

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function show($id)

{

//

}

/\*\*

* Show the form for editing the specified resource.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function edit($id)

{

//

}

/\*\*

* Update the specified resource in storage.

\*

* @param \Illuminate\Http\Request $request
* @param int $id
* @return \Illuminate\Http\Response

\*/

public function update(Request $request, $id)

{

//

}

/\*\*

* Remove the specified resource from storage.

\*

* @param int $id
* @return \Illuminate\Http\Response

\*/

public function destroy($id)

{

//

}

//支付宝 接口调用

public function pays(){ pay();

}

//通知界面

public function returnurl(){ echo "支付成功";

}

95