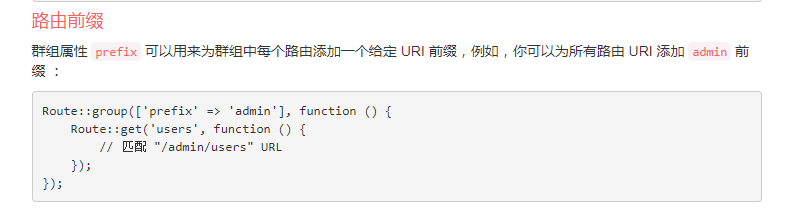
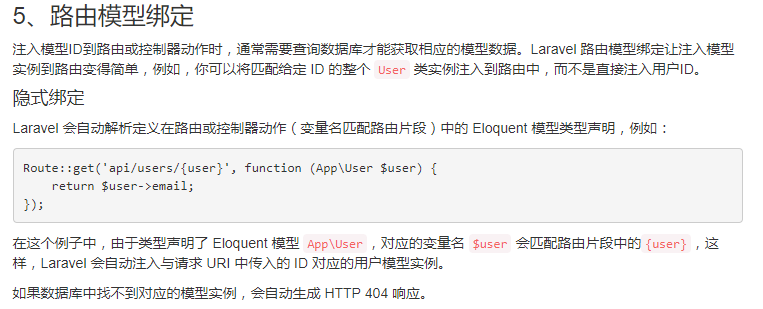


路由群组中间件





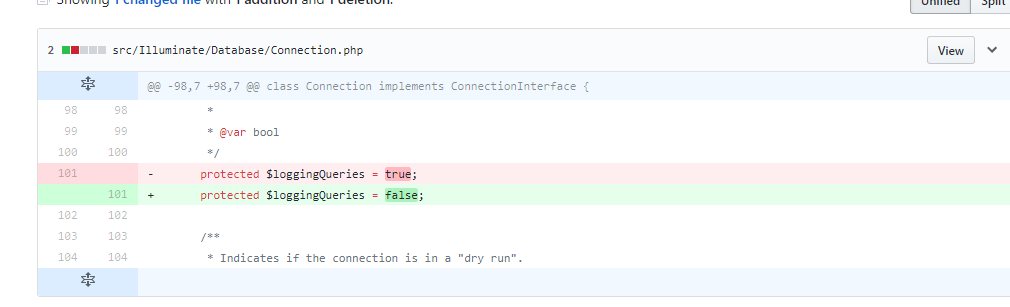




Blade 多字段渲染



php artisan route:list 查看当前路由



有时候打开也不行.需要:

DB::connection()->enableQueryLog(); // 开启查询日志—放在执行语句前面

DB::table('xxx'); // 要查看的sql

$queries = DB::getQueryLog(); // 获取查询日志

print\_r($queries); // 即可查看执行的sql，传入的参数等等

也可以通过事件监听的方法来获取：

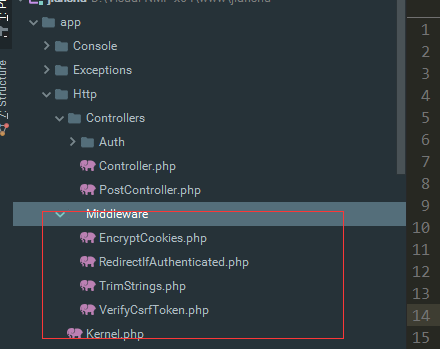
Event::listen('illuminate.query', function($query){

var\_dump($query);

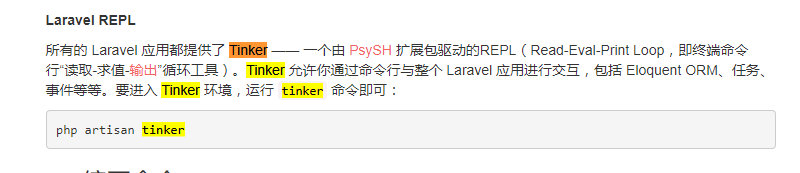
});

配置debugbar : <https://www.cnblogs.com/timeisaway/p/7131711.html>

Laravel ORM 模型



Tinker的使用



工厂模式

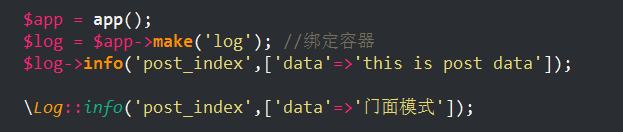
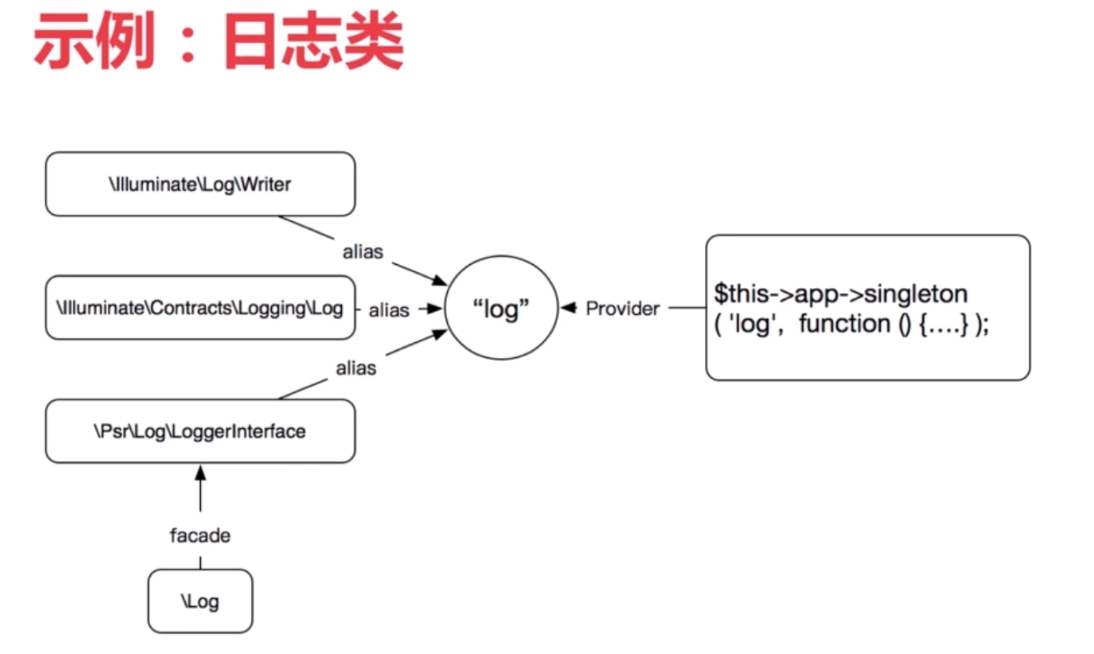
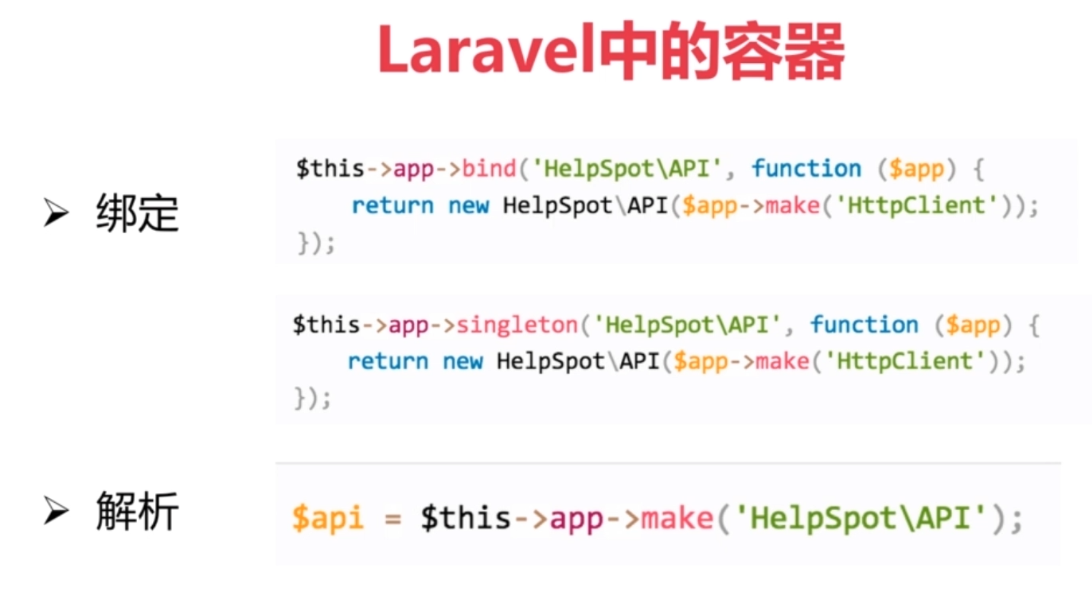
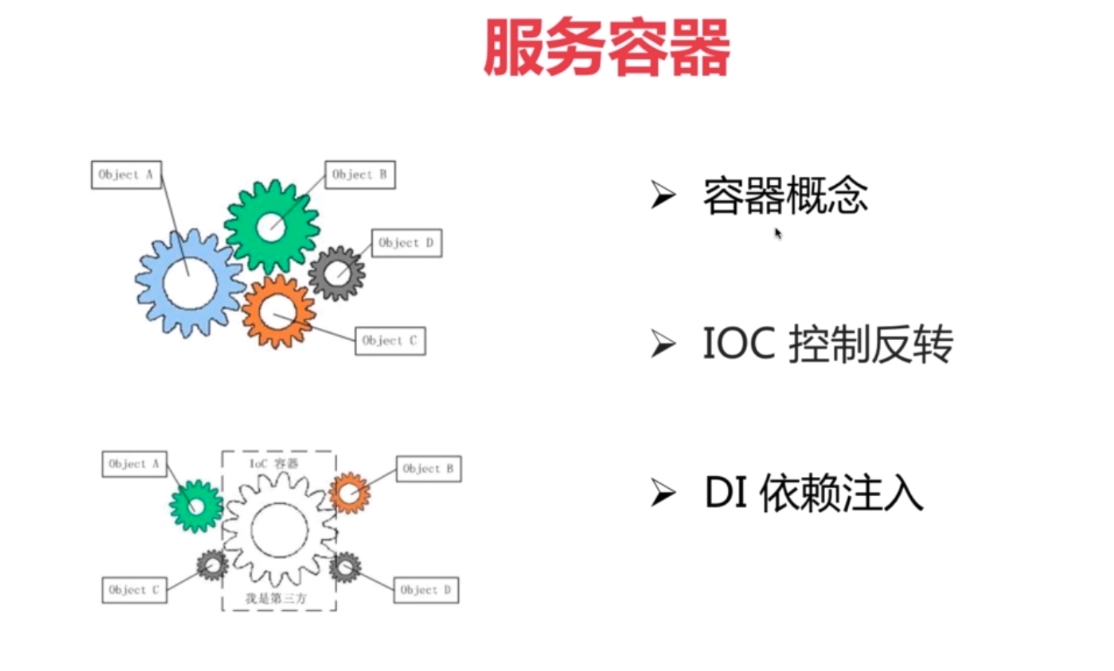
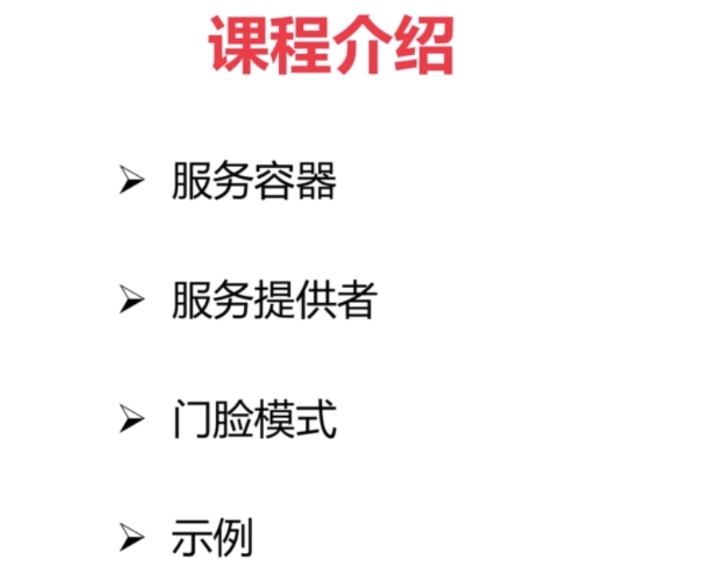




<https://laravel-china.org/topics/438/l5-validation-message-chinese-localization>



核心思想







**模型关联**



模型绑定 如果没有数据会报错







搜索功能

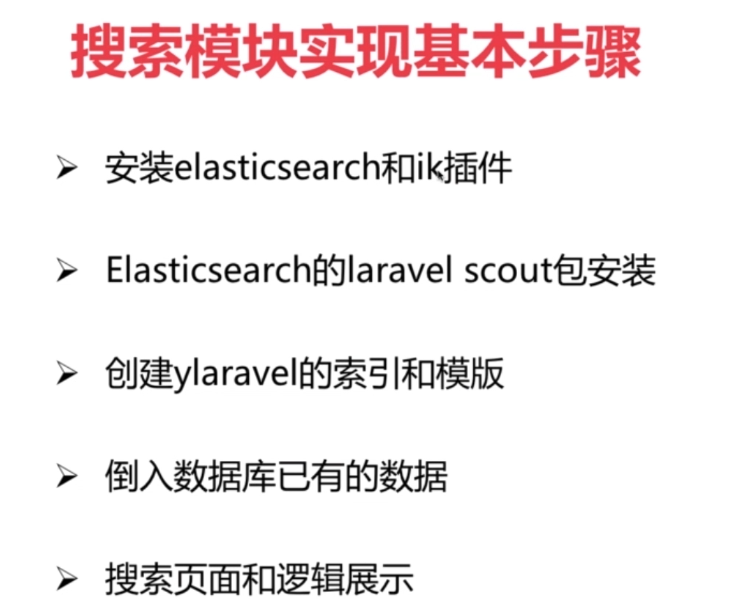


中文搜索插件

<https://elasticsearch.cn/> 中文论坛

<https://github.com/amikey/elasticsearch-analysis-ik> 支持中文分词分析

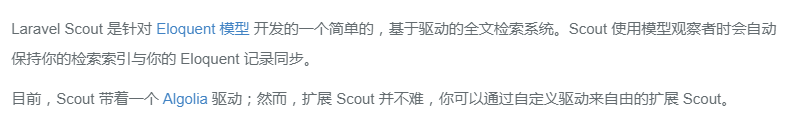




<https://github.com/medcl/elasticsearch-rtf>

java版本必须大于1.8





安装laravel scoutc omposer require laravel/scout



再安装<https://github.com/ErickTamayo/laravel-scout-elastic>

composer require tamayo/laravel-scout-elastic

You must add the Scout service provider and the package service provider in your app.php config:

// config/app.php

'providers' => [

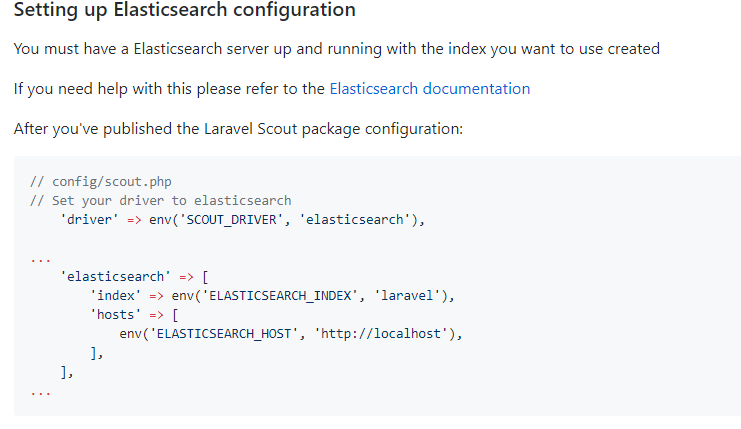
...

Laravel\Scout\ScoutServiceProvider::class,

...

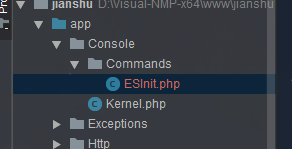
ScoutEngines\Elasticsearch\ElasticsearchProvider::class,

],









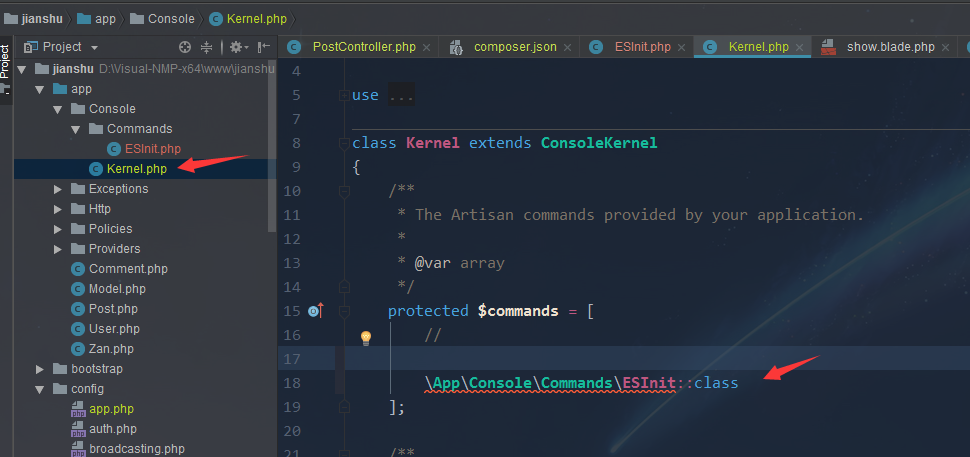


Laravel 有例子

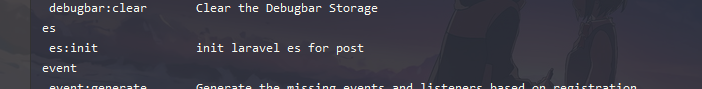


自己定义的ESInit类

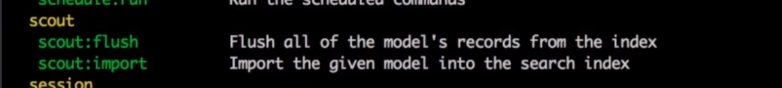
在Kermel.php引入

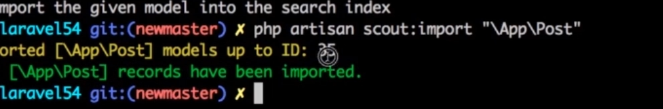


Php srtisan









向数据库添加内容,会自动向索引添加内容

**个人中心**





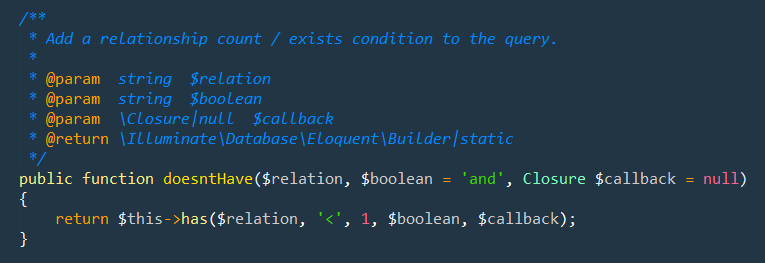


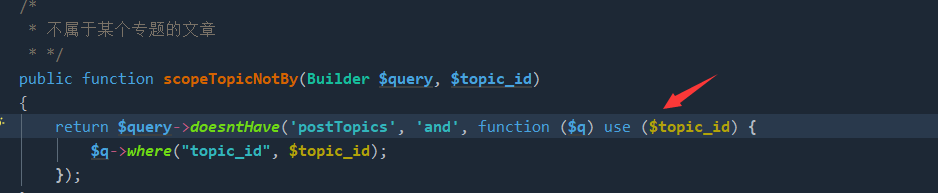
使用ajax添加csrf\_token



专题列表

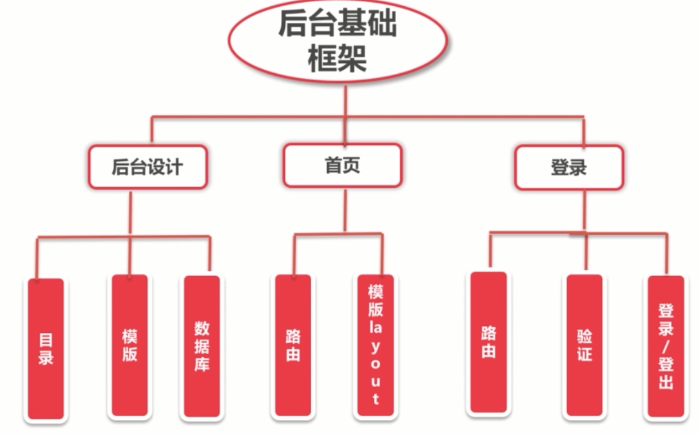


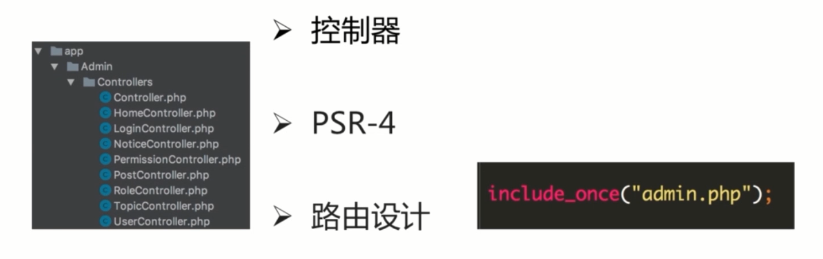




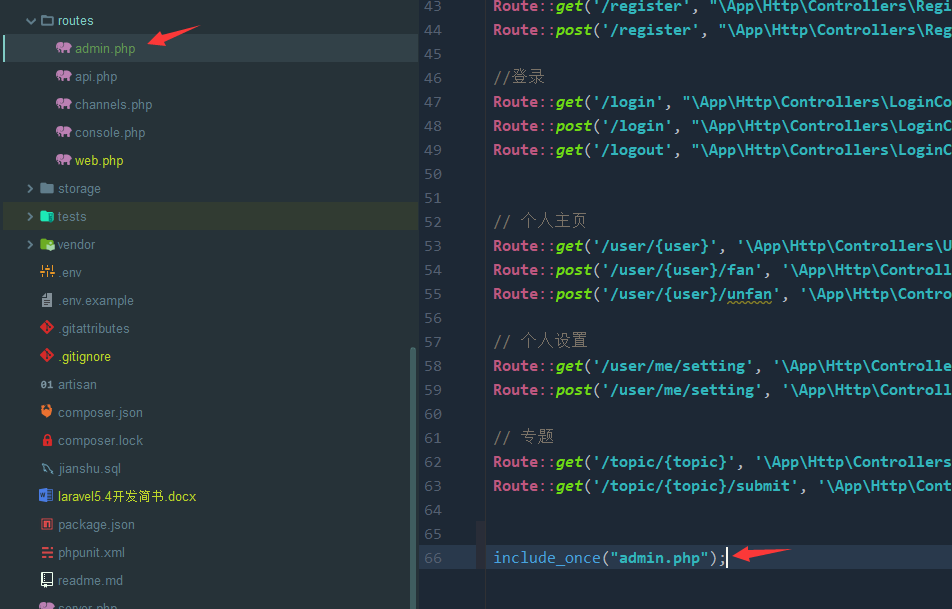
Php匿名函数传递参数

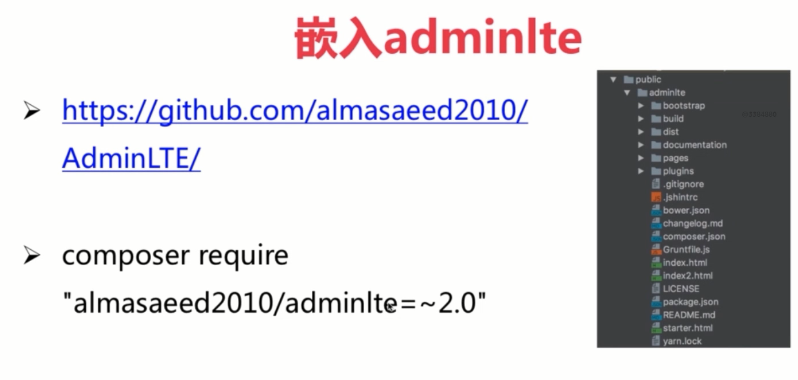






前后台路由分文件管理





### composer安装[AdminLTE](https://blog.6ag.cn/tag/adminlte/)

使用[AdminLTE](https://blog.6ag.cn/tag/adminlte/)的方式有很多，最直接的方式就是下载模板源文件到public目录，然后在views目录下的blade模板文件中引入对应的精通资源文件即可。

**当然，我们这里不用这么简单的方式，我们要用更加复杂的方式来使用AdminLTE。**

使用composer安装AdminLTE，在项目根目录执行下面命令：

|  |  |
| --- | --- |
|  | composer require "almasaeed2010/adminlte=~2.0" |

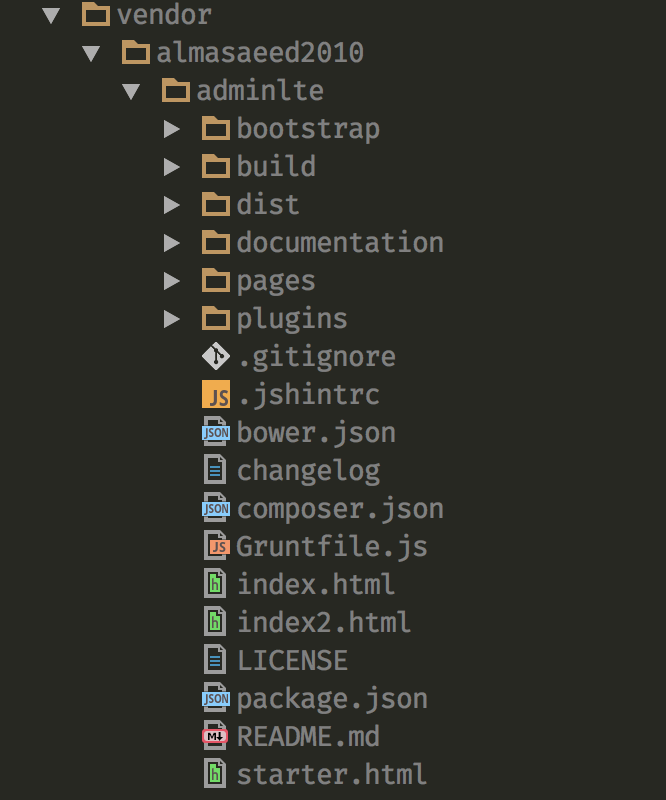
也可以先在composer.json文件中，添加下面配置：

|  |  |
| --- | --- |
|  | "almasaeed2010/adminlte": "~2.0" |

然后在项目根目录执行下面命令进行安装：

composer update

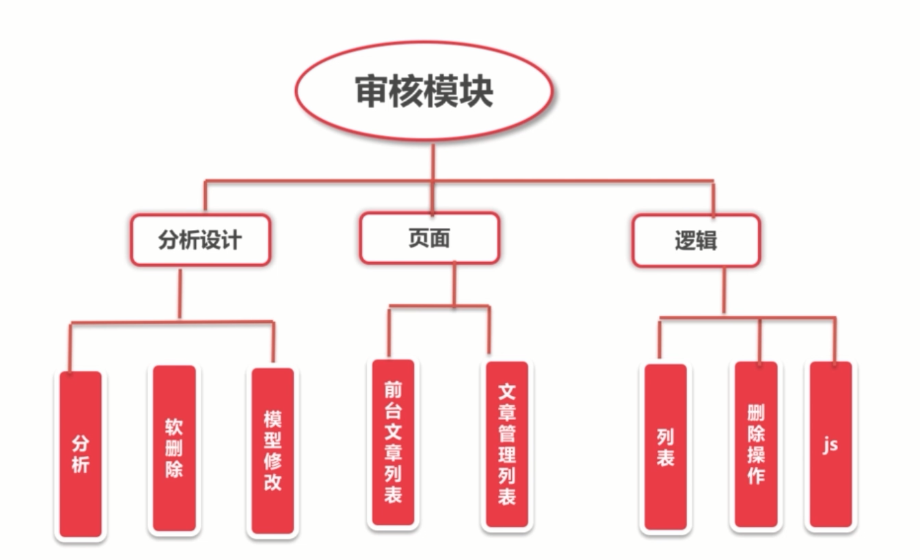
这样AdminLTE模板就会被下载到 vendor/almasaeed2010 目录下。

[](https://blog.6ag.cn/wp-content/uploads/2016/07/QQ20160719-0@2x.png)

<https://blog.6ag.cn/1597.html>

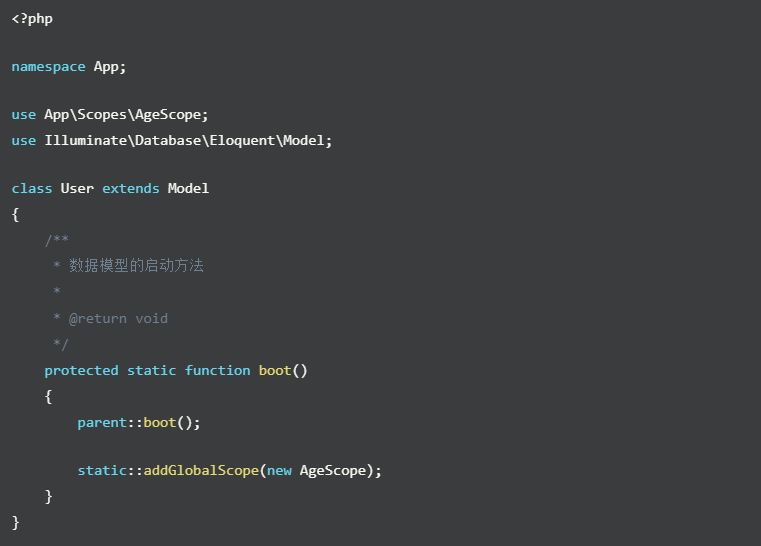
下载完会在vendor目录

然后将adminlte完整copy到public目录下 直接访问xxx/adminlte/index就可以查看



#### 应用全局作用域

要将全局作用域分配给模型，需要重写给定模型的 boot 方法并使用 addGlobalScope 方法：

添加作用域后，如果使用 User::all() 查询则会生成如下SQL语句：

#### 匿名的全局作用域

Eloquent 还允许我们使用闭包定义全局作用域，这在实现简单作用域的时候特别有用，这样的话，我们就没必要定义一个单独的类了：

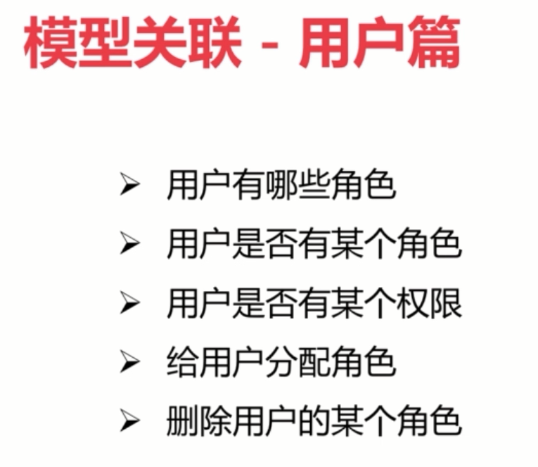
我们还可以通过以下方式，利用 age 标识符来移除全局作用：

#### 移除全局作用域

如果想要在给定查询中移除指定全局作用域，可以使用 withoutGlobalScope：

如果你想要移除某几个或全部全局作用域，可以使用 withoutGlobalScopes 方法：

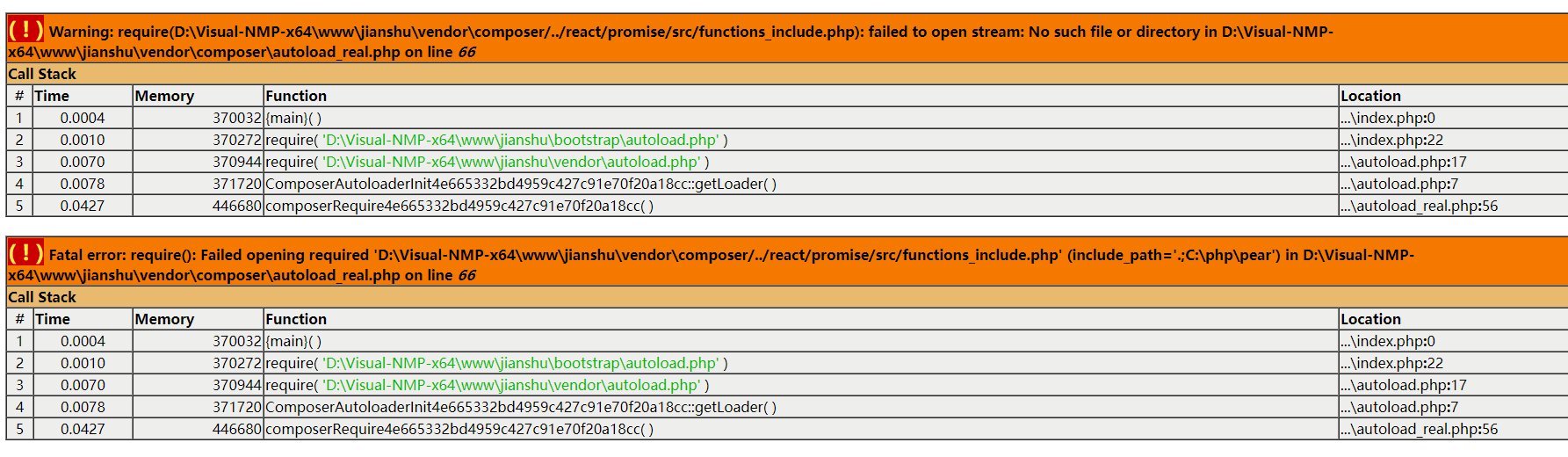
### 





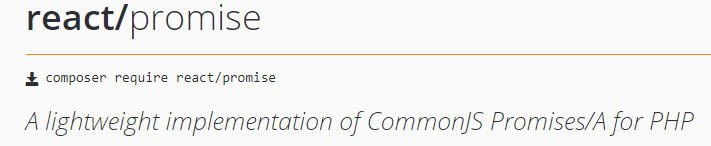






缺扩展

composer require react/promise



Laravel安装whoops报错美化页面

Whoops 的界面比 PHP 默认的错误和异常输出好多了。Whoops 也易于集成，首先我们需要通过 Composer 安装该组件：

composer require filp/whoops

然后在应用的引导文件中注册 Whoops 提供的错误及异常处理程序：

$whoops = new \Whoops\Run;

$whoops->pushHandler(new \Whoops\Handler\PrettyPageHandler);

$whoops->register();

这样，当 PHP 脚本触发了 PHP 错误及异常，就会看到 Whoops 诊断页面。在 Laravel 中我们只需将上述代码放到 App\Exceptions\Handler 类的 render 方法中替换原来的代码即可：



这样，当我们访问页面报错时，就会看到 Whoops 的报错页面：

但是此项目失败,

需要在

Handle类更改

1. */\*\**
2. *\* Render an exception into an HTTP response.*
3. *\**
4. *\* @param \Illuminate\Http\Request $request*
5. *\* @param \Exception $e*
6. *\* @return \Illuminate\Http\Response*
7. *\*/*
8. public function render($request, Exception $e)
9. {
10. if ($this->isHttpException($e))
11. {
12. return $this->renderHttpException($e);
13. }
15. if (config('app.debug'))
16. {
17. return $this->renderExceptionWithWhoops($e);
18. }
20. return parent::render($request, $e);
21. }
23. */\*\**
24. *\* Render an exception using Whoops.*
25. *\**
26. *\* @param \Exception $e*
27. *\* @return \Illuminate\Http\Response*
28. *\*/*
29. protected function renderExceptionWithWhoops(Exception $e)
30. {
31. $whoops = new \Whoops\Run;
32. $whoops->pushHandler(new \Whoops\Handler\PrettyPageHandler());
34. return new \Illuminate\Http\Response(
35. $whoops->handleException($e),
36. $e->getStatusCode(),
37. $e->getHeaders()
38. );
39. }





