技术文档汇总

目录

[一、Laravel 2](#_Toc501014766)

[1.Vagrant+Virtual+HomeStead环境搭建 2](#_Toc501014767)

[2.Laravel文件存储问题 9](#_Toc501014768)

[3.laravel-mix及模板调用问题 9](#_Toc501014769)

[4. Vue组件化-关注/取关按钮 13](#_Toc501014770)

[5.axios 14](#_Toc501014771)

[6.Laravel5.4 Auth::user()无法获取认证用户信息问题 14](#_Toc501014772)

[7.Vue2生命周期函数中this指向不是组件实例的问题 15](#_Toc501014773)

[8.Vue组件报错解决方案 15](#_Toc501014774)

[9.关于Laravel5表单验证错误后Session中的$errors和$old不显示的问题 16](#_Toc501014775)

[10.laravel项目移植到apache服务器上CSS与文件丢失问题 17](#_Toc501014776)

[11.上传图片报错500问题 17](#_Toc501014777)

[12.HomeStead项目移植Apache 18](#_Toc501014778)

[13.Laravel5.4用户授权-policies/Gate 18](#_Toc501014779)

[二、MySQL 20](#_Toc501014780)

[1.MySql数据库更新5.7版本后datetime不支持的解决办法 20](#_Toc501014781)

[2.数据库更新修复 21](#_Toc501014782)

[3.mysql权限 21](#_Toc501014783)

[三、前端 21](#_Toc501014784)

[1.颜色搭配 21](#_Toc501014785)

[2.UEditor插入视频源码过滤问题 22](#_Toc501014786)

[3.Bootstracp模态框弹出时导航条宽度增加导致页面左移 22](#_Toc501014787)

[4.头像上传，服务器报错500 23](#_Toc501014788)

[5.undefined index action 23](#_Toc501014789)

[6.验证码图片无法显示 24](#_Toc501014790)

[四、服务器 24](#_Toc501014791)

[1.多站点配置 24](#_Toc501014792)

[2.路由失效 25](#_Toc501014793)

一、Laravel

1.Vagrant+Virtual+HomeStead环境搭建

本文在Windows操作系统下搭建环境，其他操作系统请仅作参考

一、下载工具

1.virtualBox

Oracle 公司的虚拟机软件, 能运行在当前大部分流行的系统上

网址：<https://www.virtualbox.org/wiki/Downloads>

2.vagrant

命令行接口，提供一种命令行接口, 允许自动化安装虚拟机, 并且因为是脚本编写成的 provision 文本文件, 给共享虚拟机配置提供了可能, Homestead 正是构建在 Vagrant 之上

网址：<https://www.vagrantup.com/downloads.html>

二、安装

1.将vagrant和VirtualBox安装完成后打开Gitbash命令窗口进入本地路径**（没有Git请先下载安装Git）**

cd ~

2.克隆 homestead 项目

git clone https://github.com/laravel/homestead.git Homestead

3.在virtualbox虚拟机中加入 homestead box

有两种方法：

（1）执行以下命令添加 box，这种方法下载很慢，容易被墙，我即使开着VPN也下不了，所以我果断用第二种方法

vagrant box add laravel/homestead

（2）命令行下下载速度太慢的话可以利用工具下载以下链接加速，即先将资源下载到本地，再添加到虚拟机中**（我用迅雷下载的）**

资源地址：

<https://atlas.hashicorp.com/laravel/boxes/homestead/versions/2.1.0/providers/virtualbox.box>

下载完毕我们得到了一个virtualbox.box文件，将它放在C：根目录下**（环境搭建完后可以删除这个文件）**

创建一个metadata.json文件放置在本地目录下（C:用户/当前用户/）**（很重要）**

metadata.json内容：

{

"name": "laravel/homestead",

"versions": [{

"version": "2.1.0",

"providers": [{

"name": "virtualbox",

"url": "file://C:/virtualbox.box"

}]

}]

}

**（我的盒子是2.1.0版所以version是2.1.0，virtualbox，vagrant最好都用最新版避免出现兼容和再更新问题）**

Gitbash：

cd ~

vagrant box add metadata.json

三、配置

1.生成配置文件

cd homestead

bash init.sh

打开Homestead目录，可以找到Homestead.yaml文件

这时开始修改Homestead.yaml文件**（我用的是Sublimt text）**

folders表示主机和虚拟机的共享文件

sites表示域名访问虚拟机位置

我的修改如下：

**（注意：请在源文件上修改，不要复制代码，yaml文件里多一个空格就可能安装失败！）**

ip: "192.168.10.10"

memory: 2048

cpus: 2

provider: virtualbox

authorize: ~/.ssh/id\_rsa.pub

keys:

- ~/.ssh/id\_rsa

folders:

- map: ~/Code/TaskManager

to: /home/vagrant/Code/TaskManager

sites:

- map: taskmanager.app

to: /home/vagrant/Code/TaskManager/public

databases:

- homestead

为了对应共享的文件，在本地目录下新建目录Code，在Code下新建子目录TaskManager

2.域名重定向

修改主机hosts文件添加域重定向

位置：C:\Windows\System32\drivers\etc\hosts

添加一条重定向代码：

192.168.10.10 taskmanager.app

**（如果保存不了，右键hosts-属性-安全-编辑，给当前用户分配此文件的权限）**

3.SSH密钥

在gitbash窗口输入：

ssh-keygen

根据提示输入你要设置的密码**（我设置为空，全部enter跳过）**

然后打开homestead/scripts/homestead.rb，查看版本限制：

config.vm.box\_version = settings["version"] ||= ">= 2.0.0"

如果不符则修改

**（注意：不要看网上的执行vagrant init laravel/homestead 命令，否则Vagrantfile会被初始化，vagrant up的时候不会执行yaml配置！）**

4.安装环境

cd homestead

vagrant up

等待一段时间安装完成

**（命令执行后，显示信息第二行如果是import...laravel/homeatead才是正确的）**

vagrant ssh

进入虚拟机

ls

看是否有个Code文件，如果有，则yaml配置文件执行成功了

四、Box内部署

1.composer

修改虚拟机内composer下载地址转到中国镜像

composer config -g repo.packagist composer https://packagist.phpcomposer.com

下载laravel项目

cd Code

sudo composer create-project laravel/laravel TaskManager

在主机浏览器输入域名：taskmanager.app

访问到Laravel首页则配置成功！

2.修改laravel配置

修改.env文件内数据库名

...

3.安装phpmyadmin

网址：[www.phpmyadmin.net](http://www.phpmyadmin.net)

下载安装解压改名为phpMyAdmin放在主机的Code目录下**（注意：phpMyAdmin目录下就是doc、examples等子目录）**

修改yml文件：

在共享文件后添加

- map: ~/Code/phpMyAdmin

to: /home/vagrant/Code/phpMyAdmin

在域名重定向后添加：

- map: phpmyadmin.app

to: /home/vagrant/Code/phpMyAdmin

在hosts文件添加：

192.168.10.10 phpmyadmin.app

4.重载yaml

登录虚拟机

vagrant up

重新加载配置

vagrant provision

vagrant ssh

主机浏览器url输入phpmyadmin.app，出现phpmyadmin页面，部署完成！

2.Laravel文件存储问题

Laravel有多种文件存储方式，包括：local、public、第三方云存储等，存储方式定义在config/filesystem.php中

本地存储选择local或者public方式

旧版本中：

本地存储磁盘直接放在/public下，用户可以直接访问到

新版本中：

本地磁盘在/storage/app/public下，用户上传的文件不会被直接访问到，在项目根目录下执行：

php artisan storage:link

建立一个/storage/app/public到/public/storage的软链接

在filesystem中设置url即可

**（但是在virtualbox中出于安全考虑，共享文件夹中禁止建立软链接，这样最好将文件系统部署在云端）**

3.laravel-mix及模板调用问题

Laravel中composer用于管理php依赖，而npm用于管理前端依赖

1.升级npm

确保下载依赖不出错

npm install npm -g

如果有权限问题则

sudo npm install npm -g

检查node和npm版本是否最新：

node -v

npm -v

发生权限错误则给出权限，输入：

sudo chown -R $USER:$(id -gn $USER) /home/vagrant/.config

2.安装package.json中的前端依赖

cd到项目根目录下

npm install

如果在windows虚拟机中，由于禁止共享文件建立软链接，所以需要增加选项

npm install --no-bin-links

**（如果安装出错，可以删除项目根目录下的node\_moudles目录，重新安装）**

3.使用Laravel-mix编译整合前端资源

Laravel5.4中mix抛弃了gulp改用webpack，后者对API更加友好

npm run dev

如果报错cross-env：not found，建议用旧版的指令替换，具体操作为，将package.json中script中的内容修改为：

**"prod"**: **"npm run production"**,  
**"dev"**: **"node node\_modules/cross-env/dist/bin/cross-env.js NODE\_ENV=development node\_modules/webpack/bin/webpack.js --progress --hide-modules --config=node\_modules/laravel-mix/setup/webpack.config.js"**,  
**"watch"**: **"node node\_modules/cross-env/dist/bin/cross-env.js NODE\_ENV=development node\_modules/webpack/bin/webpack.js --watch --progress --hide-modules --config=node\_modules/laravel-mix/setup/webpack.config.js"**,  
**"watch-poll"**: **"node node\_modules/cross-env/dist/bin/cross-env.js NODE\_ENV=development node\_modules/webpack/bin/webpack.js --watch --watch-poll --progress --hide-modules --config=node\_modules/laravel-mix/setup/webpack.config.js"**,  
**"hot"**: **"node node\_modules/cross-env/dist/bin/cross-env.js NODE\_ENV=development node\_modules/webpack-dev-server/bin/webpack-dev-server.js --inline --hot --config=node\_modules/laravel-mix/setup/webpack.config.js"**,  
**"production"**: **"node node\_modules/cross-env/dist/bin/cross-env.js NODE\_ENV=production node\_modules/webpack/bin/webpack.js --progress --hide-modules --config=node\_modules/laravel-mix/setup/webpack.config.js"**

再执行npm run dev

4.下载并导入前端资源

cd /resources/assets/

mkdir css

cd css

wget 前端css资源网址

cd ../../js

wget 前端js资源网址

资源文件存放在assets下的css和js文件夹中，我们需要将其导入到项目中

导入js文件：

打开assets/js/bootstrap.js，在require(jquery)下添加资源名，如：

**window**.**$** = **window**.**jQuery** = **require**(**'jquery'**);  
**require**(**'bootstrap-sass'**);  
**require**(**'./select2.min'**);

导入css文件：

打开assets/sass/app.scss，import资源，如：

**@import "./../css/select2.min"**;

5.mix使用

如果要让前端资源随着开发改动自动编译且避免样式缓存问题，则在webpack.mix.js中添加.version()，如下：

**mix**.**js**(**'resources/assets/js/app.js'**, **'public/js'**)  
 .**sass**(**'resources/assets/sass/app.scss'**, **'public/css'**)  
 .version();

执行监听

npm run watch

在public/css和js下会生成带哈希码的css和js文件，这个就是我们要导入视图的资源

一般把编译整合后的资源放在app.blade.php中，css放开头，js放结尾，如:

<**link href="**{{ mix(**'css/app.css'**) }}**" rel="stylesheet"**>

<**script src="**{{ mix(**'js/app.js'**) }}**"**></**script**>

**（注意：子视图的css和js样式一定要放在section中，通过yield引用，而且要放在公用样式之后，否则会被覆盖！）**

4. Vue组件化-关注/取关按钮

<question-follow-button question="{{$question->id}}">

</question-follow-button>

<template>  
 <button class="btn btn-default"  
 v-bind:class="{'btn-success':followed}"  
 v-text="text"  
 v-on:click="follow"  
 >  
 </button>  
</template>  
  
<script>  
 export default {  
 props:['question'],  
 mounted() {  
 this.$http.post('/api/question/isFollow',{'question':this.question}).then(response => {  
 this.followed = response.data.followed;  
 console.log("question followed is "+response.data.followed);  
 })  
 },  
 data(){  
 return {  
 followed: false,  
 }  
 },  
 methods:{  
 follow(){  
 this.$http.post('/api/question/follow',{'question':this.question}).then(response => {  
 this.followed = response.data.followed;  
 console.log(response.data);  
 })  
 }  
 },  
 computed: {  
 text(){  
 return this.followed ? '已关注' : '关注问题';  
 }  
 }  
 }  
</script>

5.axios

vuejs2.0已停止对vue-resource的更新，开始使用axios实现vue组件与后端的异步数据交互

6.Laravel5.4 Auth::user()无法获取认证用户信息问题

原因：未开启Session

在Kernel.php中

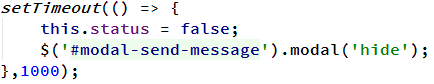
protected $middleware = [  
 \Illuminate\Foundation\Http\Middleware\CheckForMaintenanceMode::class,  
 \Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,  
 \App\Http\Middleware\TrimStrings::class,  
 \Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,  
 \Illuminate\Session\Middleware\StartSession::class,  
 \Illuminate\Session\Middleware\AuthenticateSession::class,  
 \Illuminate\View\Middleware\ShareErrorsFromSession::class,  
];

'web' => [  
 \App\Http\Middleware\EncryptCookies::class,  
 \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,  
 \Illuminate\Session\Middleware\StartSession::class,  
 \Illuminate\Session\Middleware\AuthenticateSession::class,  
 \Illuminate\View\Middleware\ShareErrorsFromSession::class,  
 \App\Http\Middleware\VerifyCsrfToken::class,  
 \Illuminate\Routing\Middleware\SubstituteBindings::class,  
],

7.Vue2生命周期函数中this指向不是组件实例的问题

在生命周期函数的function(){}中this指向上下文

要使this指向组件实例，需用（）=>{}，这样做相当于function(){}.bind(this)将函数的this改变指向



8.Vue组件报错解决方案

Avoid mutating a prop directly since the value will be overwritten whenever the parent component re-renders. Instead, use a data or computed property based on the prop's value. Prop being mutated:

Vue2中，父组件传入的props属性值不允许直接改变，否则会报以上错误

解决方案：

将props要改变的属性赋给data中建立一个变量值，将该值与组件变量绑定

<**template**>  
 <**button class="btn btn-default"  
 v-bind:class="**{**'btn-primary'**:voted}**"  
 v-on:click="**vote**"** ><**span class="glyphicon glyphicon-thumbs-up" aria-hidden="true"**></**span**>**&ensp;**{{ Mycount }}  
 </**button**>  
</**template**>  
  
<**script**>  
 **export default** {  
 **props**:[**'answer'**,**'count'**],  
 mounted() {  
 axios.post(**'/api/answer/'**+**this**.answer+**'/votes/users'**).then(response => {  
 **this**.**voted** = response.data.voted;  
 })  
 },  
 data(){  
 **return** {  
 **voted**: **false**,  
 **Mycount**:**this**.count,  
 }  
 },  
 **methods**:{  
 vote(){  
 axios.post(**'/api/answer/vote'**,{**'answer'**:**this**.**answer**}).then(response => {  
 **this**.**voted** = response.data.voted;  
 response.data.voted ? **this**.**Mycount**++ : **this**.**Mycount**--;  
 })  
 }  
 }  
 }  
</**script**>

9.关于Laravel5表单验证错误后Session中的$errors和$old不显示的问题

方法1：

将该表单发送的路由中间件由

Route::group(['middleware' => 'web'], function () {}

改为

Route::group(['middlewareGroups' => 'web'], function () {}

方法2：

将Kernel.php中

protected $middlewareGroups ’web’中的\Illuminate\Session\Middleware\StartSession::class,

\Illuminate\View\Middleware\ShareErrorsFromSession::class,

移动到$middleware中

10.laravel项目移植到apache服务器上CSS与文件丢失问题

原因：路径不对

解决办法：将apache根目录设置为laravel的public

11.上传图片报错500问题

Unable to guess the mime type as no guessers are available (Did you enable t

问题原因：php\_fileinfo未开启

解决办法：进入php.ini文件

将;extension=php\_fileinfo.dll的分号去掉

12.HomeStead项目移植Apache

修改将Apache根目录设置为laravel/public

修改.env文件数据库信息

laravel-mix npm run watch报错则更新node-sass：

npm rebuild node-sass

npm run watch不监听编译则npm run watch-poll

13.Laravel5.4用户授权-policies/Gate

（1）创建策略类

映射策略类：注册模型与策略类的映射

在\App\Providers\AuthServiceProvider.php文件的protected $policies属性中添加映射，如：

\App\User::class => \App\Policies\UserPolicy::class,

使用Artisan命令生成改模型的policy文件：

php artisan make:policy UserPolicy

这样我们就可以在这个policy文件中编写对应的授权策略

（2）编写策略

创建策略方法，传入当前用户实例和发送实例做为参数，如：在\App\Policies\ UserPolicy.php文件中添加方法

public function update(User $currentUser,User $user)

{

return $currentUser->id === $user->id;

}

这是一个更新用户信息的授权策略

传入当前用户实例和要更改的用户实例

返回是一个bool值，当当前登录的用户的id是要更改的用户id时，返回true，这个操作允许进行

▲当策略为create类型方法时只需要传入当前用户实例一个参数

（3）使用策略

策略一般在控制器中使用，如：

在UsersController控制器的用户信息更改控制器中

$user=User::where('id',$request->get('userId'))->first();

$this->authorize('update', $user);

$request是用户信息变更的表单请求

$user是请求中的用户id对应的用户实例

通过传入User类实例$user找到provider中映射的policy

用过该policy中的’update’方法，返回的值判断当前用户的操作是否可以继续

如果返回false，则自动跳转到403页面

403页面可以在\resources\views\errors\403.blade.php中自定义编写

▲create类策略使用时只需要传入模型类，如：

$this->authorize('create',Lesson::class);

（4）策略过滤

在policy类中创建before方法

在该策略所有方法执行之前都会先执行这个方法，如：

public function before($user, $ability)

{

if ($user->isSuperAdmin()) {

return true;

}

}

（5）不虚传入模型实例的策略可以用gate编写

在AuthServiceProvider文件boot方法中添加策略，如：

Gate::define('memberShow', function ($user) {

return $user->is\_admin === 2;

});

定义策略名，回调函数第一个参数默认为当前登录的用户实例，其他参数自定义

在blade中调用：

@can('memberShow')

<li>

<a href="{{ url('/manage/memberShow') }}">管理</a>

</li>

@endcan

14.路由admin

普通路由不可用/admin

会导致页面样式变化

二、MySQL

1.MySql数据库更新5.7版本后datetime不支持的解决办法

MySQL5.7版本兼容问题

具体请参考：

<https://dev.mysql.com/doc/refman/5.7/en/sql-mode.html#sql-mode-changes>

解决办法：

修改datetime数据类型的默认值，比如直接无定义，勾选可以为空

然后在自己的项目里修改相应的含有datetime字段的程序

**（注意：如果原来多个datatime类型字段默认值都是：0000-00-00 00:00:00，那么修改字段的时候需要多个字段同时修改，否则仍会报错！）**

2.数据库更新修复

Native table 'performance\_schema'.'session\_variables' has the wrong structure

mysql\_upgrade -u -p

问题原因：mysql版本升级，数据表结构更新

解决办法：

在本地找到mysql\_upgrade.exe文件位置

root权限下进入mysql\_upgrade目录

cd到该目录

mysql\_upgrade -u root -p

将任务管理器中的MySQL服务重启

3.mysql权限

创建用户+host：

create user ‘username’@’host’ identified by ‘password’;

授权：

grant all privileges(select,insert,update,delete) on databasename.\* to username@’host’;

更新grant表：

flush privileges;

三、前端

1.颜色搭配

蓝灰色：#304356

深蓝色：#3a5775

淡金色：#9c9987

1140x350

219x162

240x178

2.UEditor插入视频源码过滤问题

UEditor插入视频时会自动过滤img的url

并且返回一个视频img

解决办法，在UEditorXXS白名单中添加代码：

补充：

**img**: [**'src'**, **'alt'**, **'title'**, **'width'**, **'height'**, **'id'**, **'\_src'**, **'\_url'**,**'loadingclass'**, **'class'**, **'data-latex'**],

**video**: [**'autoplay'**, **'controls'**, **'loop'**, **'preload'**, **'src'**, **'height'**, **'width'**, **'class'**, **'style'**],

新增：

**source**: [**'src'**, **'type'**],  
**embed**: [**'type'**, **'class'**, **'pluginspage'**, **'src'**, **'width'**, **'height'**, **'align'**, **'style'**, **'wmode'**, **'play'**, **'autoplay'**,**'loop'**, **'menu'**, **'allowscriptaccess'**, **'allowfullscreen'**, **'controls'**, **'preload'**],  
**iframe**: [**'src'**, **'class'**, **'height'**, **'width'**, **'max-width'**, **'max-height'**, **'align'**, **'frameborder'**, **'allowfullscreen'**]

清空浏览器缓存

3.Bootstracp模态框弹出时导航条宽度增加导致页面左移

问题原因：

bootstracp模态框弹出后，页面滚动条隐藏，body添加样式padding-right:21px;但是导航栏display为fixed，添加的样式对它无效，导致模态框弹出后导航栏右移

解决方法：

添加.modal-open类样式，使模态框弹出后滚动条不隐藏且页面不添加padding-right

.**modal-open** {  
 **overflow**: **auto !important**;  
 **padding-right**: 0**px !important**;  
}

4.头像上传，服务器报错500

 TokenMismatchException in **VerifyCsrfToken.php** (line 68)

问题原因：Ajax CsrfToken未传入后端，请求被Middleware阻止

解决办法：

（1）在Ajax表单中添加token

（2）在App\Http\Middleware\VerifyCsrfToken中添加except项



5.undefined index action

错误原因：判断未定义的超级全局变量

解决办法：在判断时添加isset（）&&

6.验证码图片无法显示

在验证码生成函数的header前添加ob\_clean();

四、服务器

1.多站点配置

（1）hosts文件添加重定向

127.0.0.1 [www.test1.com](http://www.test1.com)

127.0.0.1 [www.test2.com](http://www.test2.com)

（2）httpd.conf

修改监听端口：

Listen 8080

Listen 8081

Listen 8082

注释默认访问路径：

#DocumentRoot "F:/AppServ/www/MachineEduSystem/public/"

启用apache虚拟主机功能，去掉注释

LoadModule vhost\_alias\_module modules/mod\_vhost\_alias.so

引入虚拟主机配置文件，去掉注释

Include conf/extra/httpd-vhosts.conf

（3）httpd-vhosts.conf

【方案一】同一域名不同端口映射不同虚拟主机，不同端口映射不同站点

【方案二】同一端口映射不同域名，不同域名映射不同站点

#配置我们自己的虚拟主机

#端口号

<VirtualHost \*:8081>

#开发者邮箱

ServerAdmin 863403035@qq.com

#站点目录

DocumentRoot "F:/AppServ/www/EducationSystem2.0"

#域名

ServerName www.test1.com

#错误访问文件

ErrorLog "logs/dummy-host2.example.com-error.log"

CustomLog "logs/dummy-host2.example.com-access.log" common

<Directory "F:/AppServ/www/EducationSystem2.0">

Options FollowSymLinks

#不允许他人修改页面

AllowOverride None

Order deny,allow

#允许所有人访问

Allow from all

</Directory>

</VirtualHost>

2.路由失效

（1）检查apache rewrite\_module是否开启

具体方法：httpd.conf文件中找到#LoadModule rewrite\_module modules/mod\_rewrite.so这一行，去掉#号

（2）端口映射AllowOverride

具体方法：在httpd.conf（未开启多站点）或者httpd-vhosts.conf（开启了多站点）

该站点端口对应的配置代码段中添加：AllowOverride All