****

Web应用开发

实 验 报 告

|  |  |
| --- | --- |
| **学 院** |  |
| **专 业** |  |
| **班 级** |  |
| **学 号** |  |
| **学生姓名** |  |
| **教师姓名** |  |
| **完成日期** |  |
| **成 绩** |  |

**引用须知：**

**本文档已删除各类敏感信息，所有内容均可引用，但仅限学习交流。祝大家能顺利通过HWT的Web课程！**

**——F001**

**作业4 RESTful API的实现**

1. **实验目的和要求**
2. 实验目的：基于Laravel框架和RESTful设计准则，按照前后端分离的开发形式来实现简单的用户注册和登录。
3. **实验要求**：

（1）实现Email形式的注册功能和相应的登录功能，注册部分具备邮件激活功能（使用Laravel的邮件发送机制或第三方组件）；

（2）实现忘记密码时通过重置密码邮件设置新密码（使用Laravel的邮件发送机制或第三方组件）；

（3）包含对某个物品（自己选择）的操作，以RESTful API风格进行；

（4）如果使用JWT认证（JSON Web Tokens），后面会酌情加分；

（5）不要求提供前端方面的实现，即不需要做网站或APP，此外，对API的实际使用在Postman上进行，并进行截图。

注意：要求1和要求2，不能按照作业3的方式来做。

1. **实验内容**

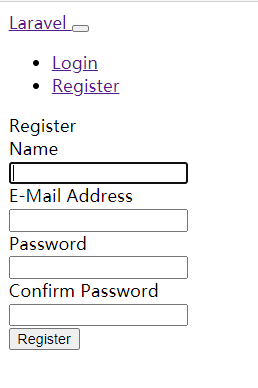
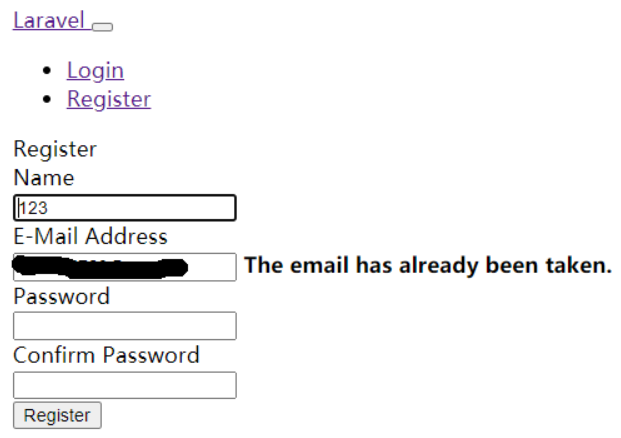
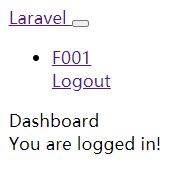
根据**实验要求中的每条规定**，进行实验。

（1）实现Email形式的注册功能和相应的登录功能

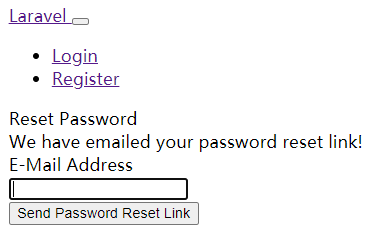
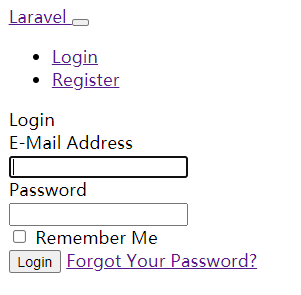
首先需要构建相关的视图，使用composer require laravel/ui进行。

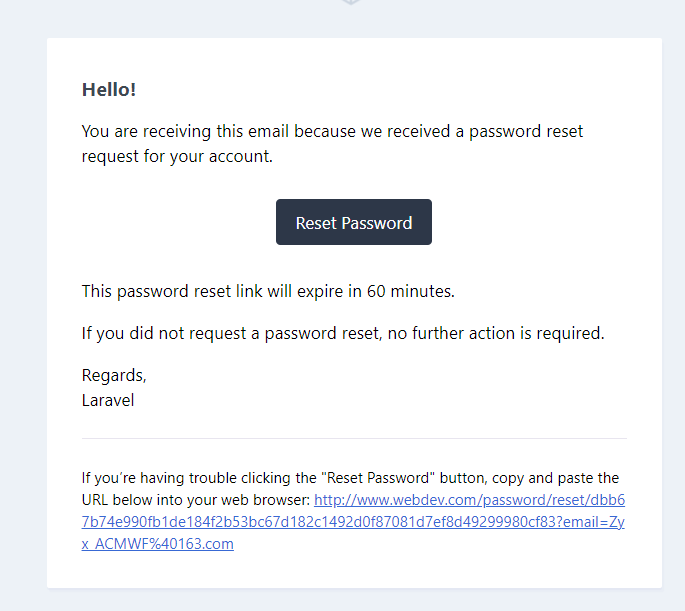
再通过脚手架命令php artisan ui vue –auth搭建相应具体的auth

接下来在env中进行相关数据库凭证的配置即可完成登录，主要是账号密码。



以上三张图片实现了注册及登录的相关情况。

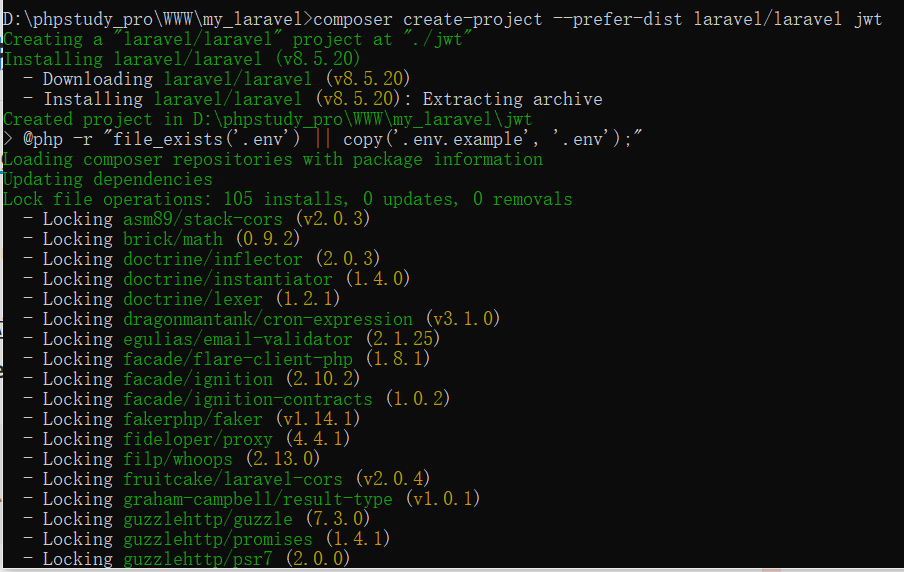
（2）实现忘记密码时通过重置密码邮件设置新密码 



以上三张图，前两张为重新设置密码的操作，最后一张为邮箱中收到更改密码的邮件。可见相关操作是正确的。

直接点击Forgot Your Password即可，并且设置了相关的交互界面。

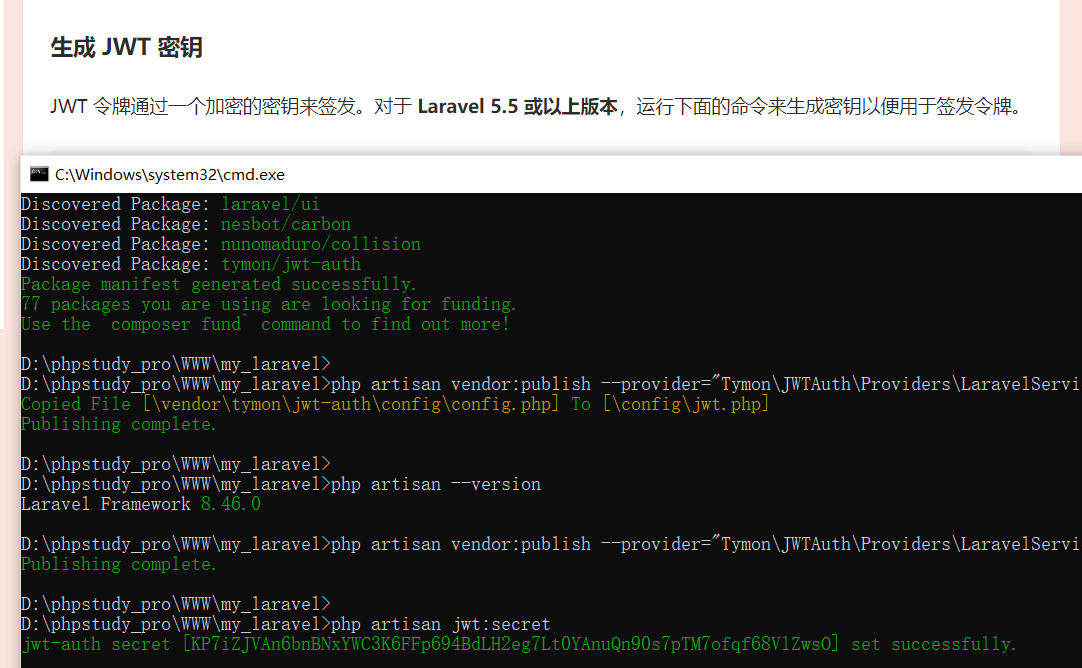
（3）包含对某个物品（自己选择）的操作，以RESTful API风格进行，并且使用了JWT认证（JSON Web Tokens）；



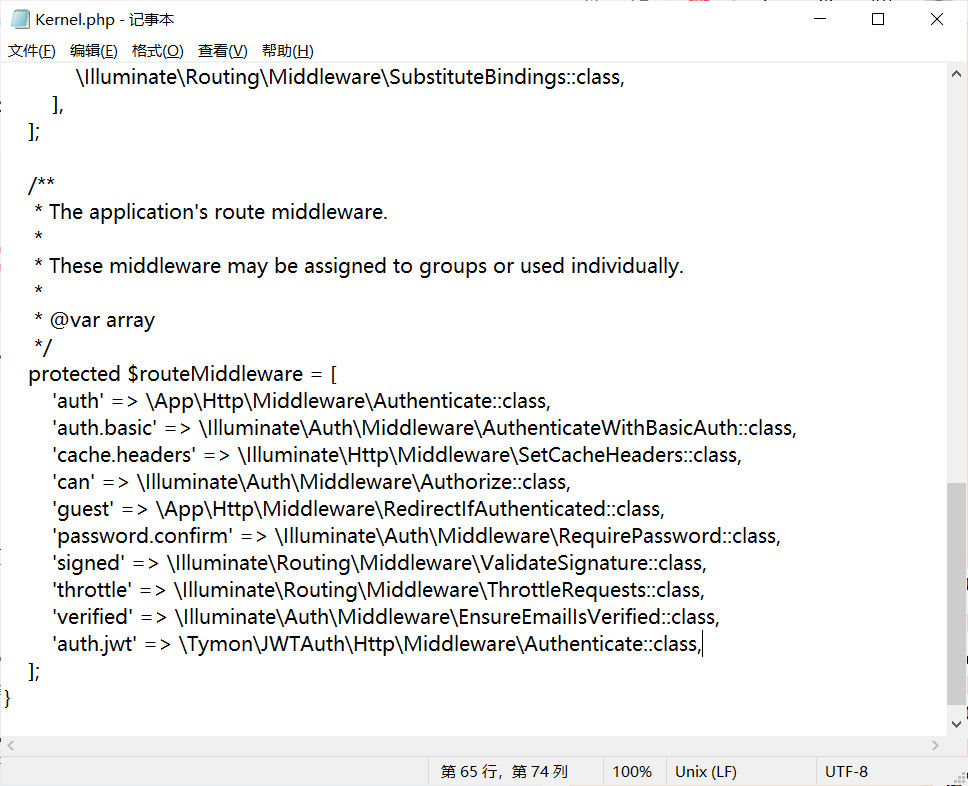
首先新建一个文件夹，便于区分，防止不同版本不同内容的文件混淆



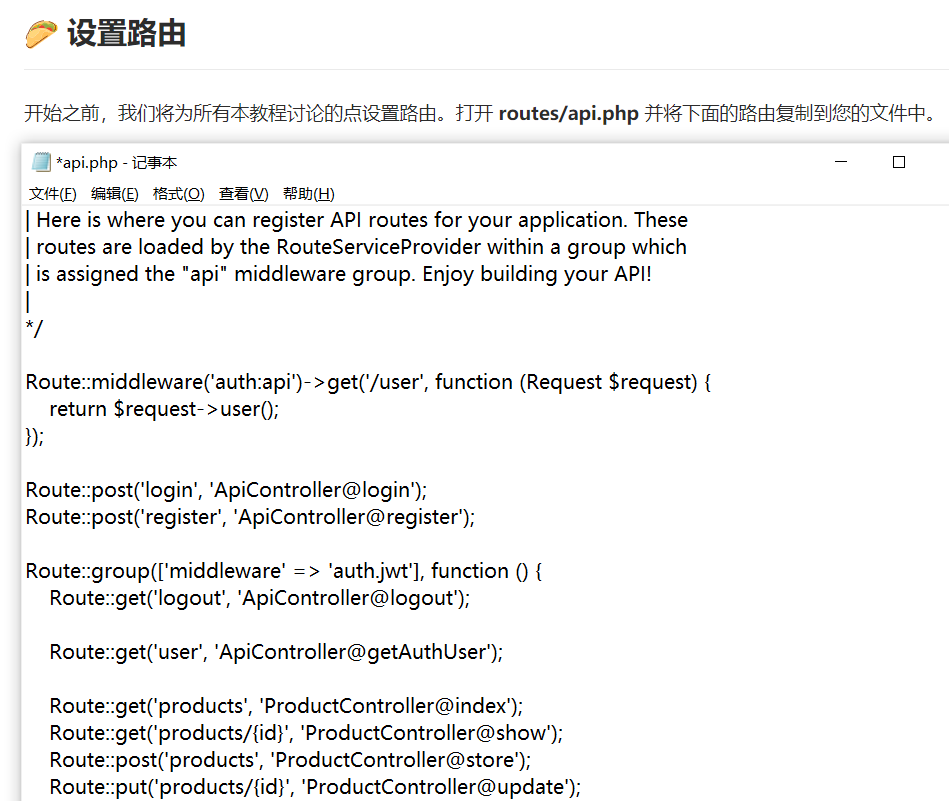
接下来配置相关的扩展包，这里使用了tymon/jwt-auth的扩展包，这里可以更加方便地使用JWT。



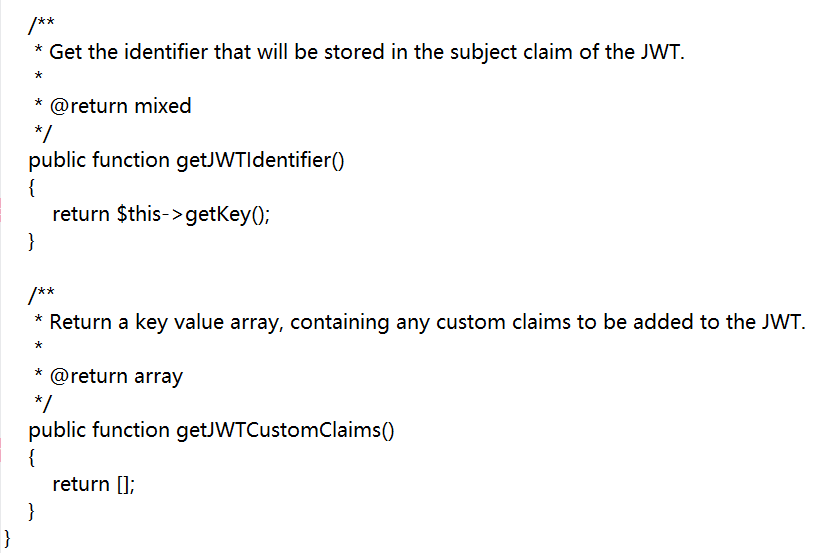
这里需要生成JWT密钥，进行相关地签发，其实相对于其他类型的数据而言，这里体现了web开发过程中安全性的考虑。



更改kernel文件，实现相关路径的定义，为接下来调用jwt时更加方便。



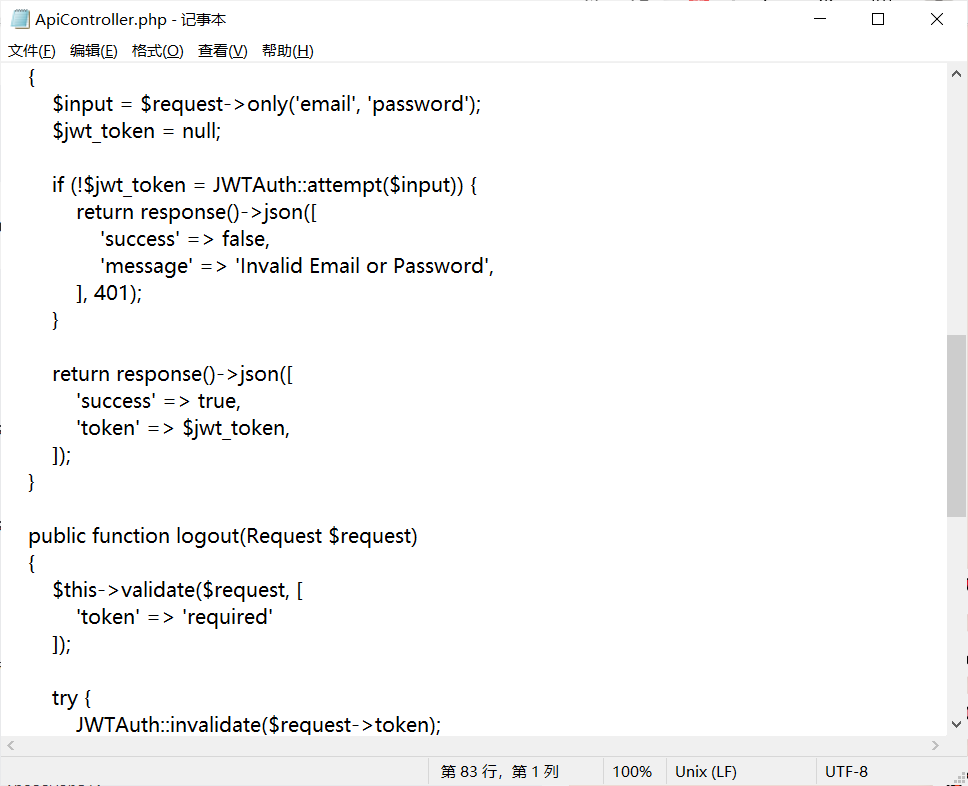
修改了相关的路由设置，均在productcontroller和apicontroller下实现。



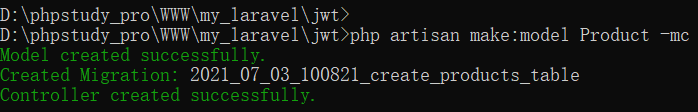
将model下的user进行更改，引入了JWT的相关方法。



创建新的Apicontroller

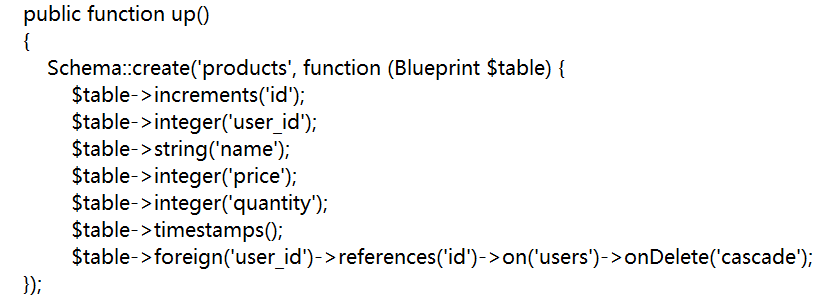


在APIcontroller中实现JWT的相关功能，以上部分就完成了身份验证的部分

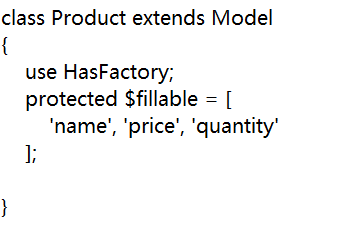


使用脚手架命令更新版本，并且创建相应的控制器（-mc功能带有的相关方式）

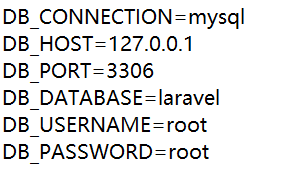
更改migration的up函数为



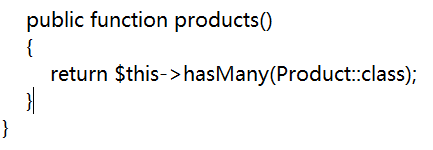
更改product的文件，设置相应的值



下一步需要配置.env中的数据库凭证，主要是用户名和密码两栏。

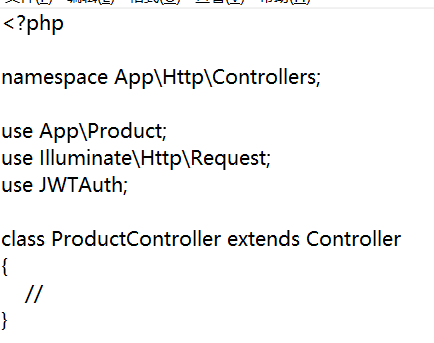


设置好.env里面的登录密码就能migration了，完成后，在user.php里加入对products的相关设置。





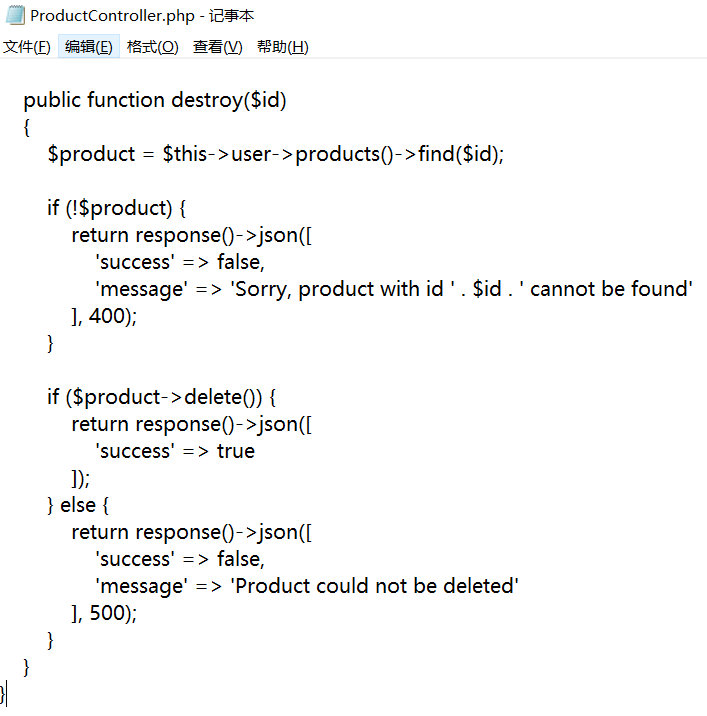
之前的准备工作都已经完成了，接下来需要对productcontroller进行修改即可。首先需要增添头文件：



头文件添加完成后，以Restful API的格式实现五个功能，分别为：

* index, 为经过身份认证的用户获取所有产品列表
* show, 根据 ID 获取特定的产品
* store, 将新产品存储到产品列表中
* update, 根据 ID 更新产品详情
* destroy, 根据 ID 从列表中删除产品

三个API均有参数，另外两个没有参数。可以看到，API部分没有edit view和create view两个方法，这与我们之前所提到的七个方法对资源进行操作是吻合的。



以上的文件就实现了五个方法

提示：

1. 可以参考网络资料，但不能照搬照抄。

2. 作业报告的格式尽量整齐，字体要统一，内容要层次分明。

1. **程序设计思路**

本次实验的一开始过程中，尝试按照老师文档中附带的相关链接进行操作，但是在使用Seed随机生成数据的过程中，产生了相关的问题，无法正常继续。在重新深入学习Laravel及相关知识后，发现该步骤是可以略过的。因为前两个要求与实验3的要求一致，因此这里仅将路由配置中的代码更改为Restful API风格即可，再加上登陆后对相关信息修改的操作，同样按照Restful API风格进行。

Restful风格的要点是通过资源的角度看待问题，实现起来的格式主要通过资源\*7种操作的方式进行，分别是crud（增删查改）+list+create+edit。由于存在相关共用的情况，因此还需要加上http的方法用以区分，即GET POST PUT DELETE。按照上述要求，以对blog操作为例，URL补全后就是：

Create：POST blogs

List：GET blogs

Delete：DELETE blogs/:id

Update：PUT blogs:/id

Read：GET blogs/:id

Create：GET blogs/create

Edit：GET blogs/:id/edit

了解了上述知识即思路后，我们就可以开始编写自己的程序了。

1. **程序源代码**

**注意源代码要有详细的注释。同学们提交的每个程序都应该遵循Honor Code（诚实代码保证）的要求。**

**请大家特别注意一定要在每个程序首部的注释中加上以下保证：**

**# 我真诚地保证：**

**# 我自己独立地完成了整个程序从分析、设计到编码的所有工作。**

**# 如果在上述过程中，我遇到了什么困难而求教于人，那么，我将在程序实习报告中**

**# 详细地列举我所遇到的问题，以及别人给我的提示。**

**# 在此，我感谢 XXX, …, XXX对我的启发和帮助。下面的报告中，我还会具体地提到**

**# 他们在各个方法对我的帮助。**

**# 我的程序里中凡是引用到其他程序或文档之处，**

**# 例如教材、课堂笔记、网上的源代码以及其他参考书上的代码段,**

**# 我都已经在程序的注释里很清楚地注明了引用的出处。**

**# 我从未没抄袭过别人的程序，也没有盗用别人的程序，**

**# 不管是修改式的抄袭还是原封不动的抄袭。**

**# 我编写这个程序，从来没有想过要去破坏或妨碍其他计算机系统的正常运转。**

**# 张逸轩**

1. **源代码（包含数据库表设计代码）**

此处不进行复制粘贴，将会放在附件的压缩包中一并上传

注意：

(1) 可以在此处贴关键代码，也可以放在压缩包中跟本文档一起上传。

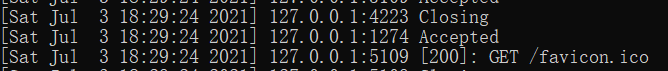
(2) 如果是压缩项目工程代码，建议把vendor目录排除掉，以减小压缩包大小。

1. **程序测试方法及测试结果记录（不能光截图，要有相应的文字说明）**
2. **测试方法**

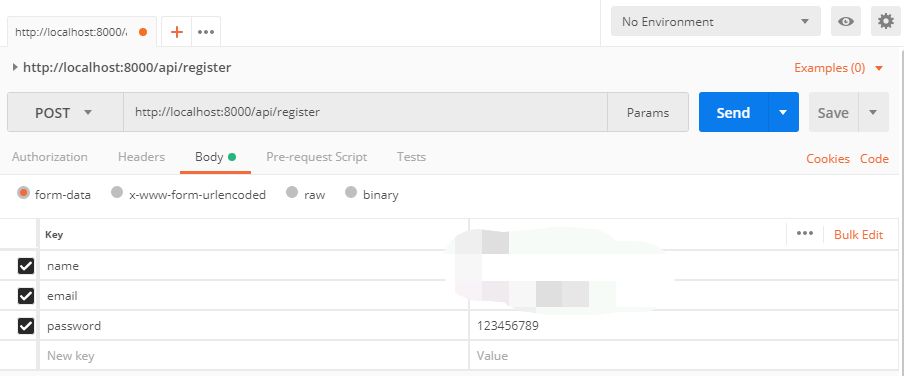
使用Postman进行Restful API的测试。在网页中进行相应的操作。

1. **测试流程**

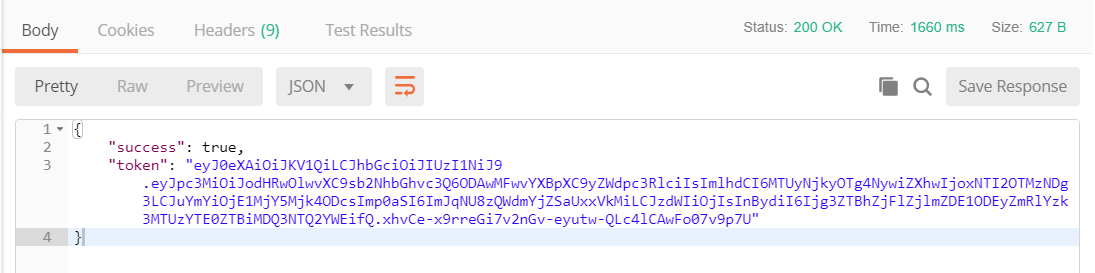
使用php artisan serve命令，在命令行中监听localhost:8000



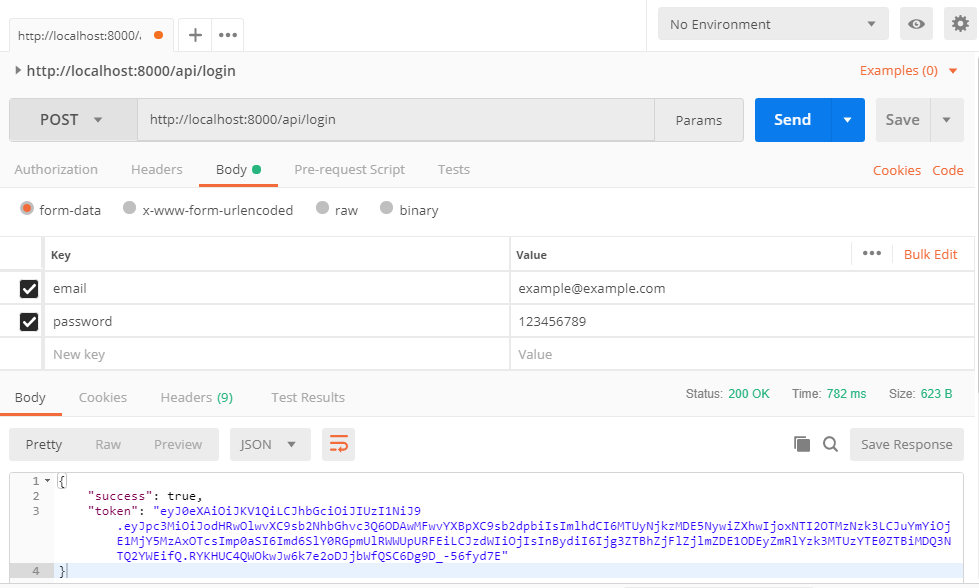
因为在代码的编写中，有写当传送成功时会返回一个200的正确提示码，因此可以认为我们的编写正确，正常地返回了相关参数



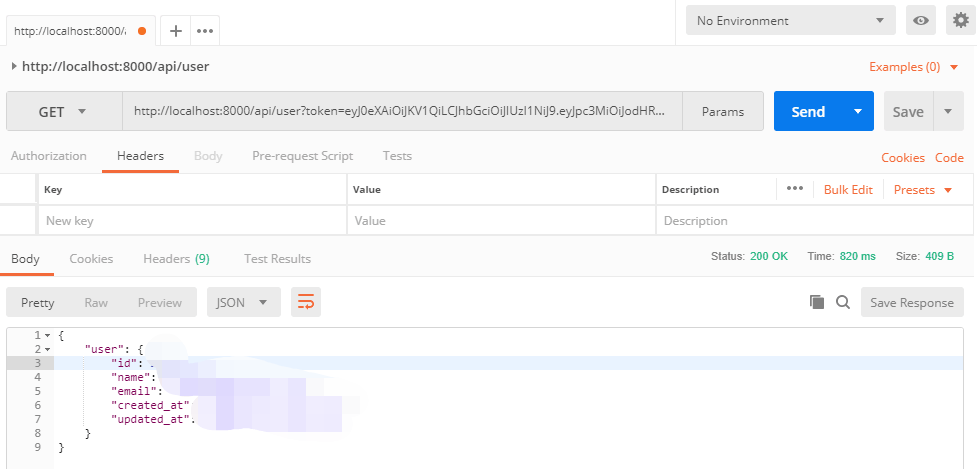
通过register发送请求，获得token令牌。



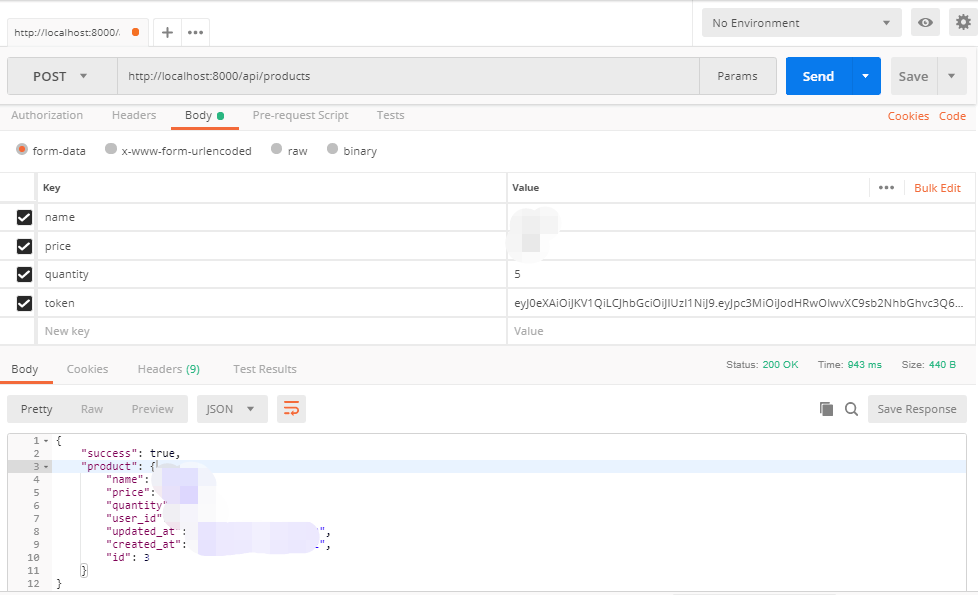
通过发送另一个请求检测login路由，返回200和令牌



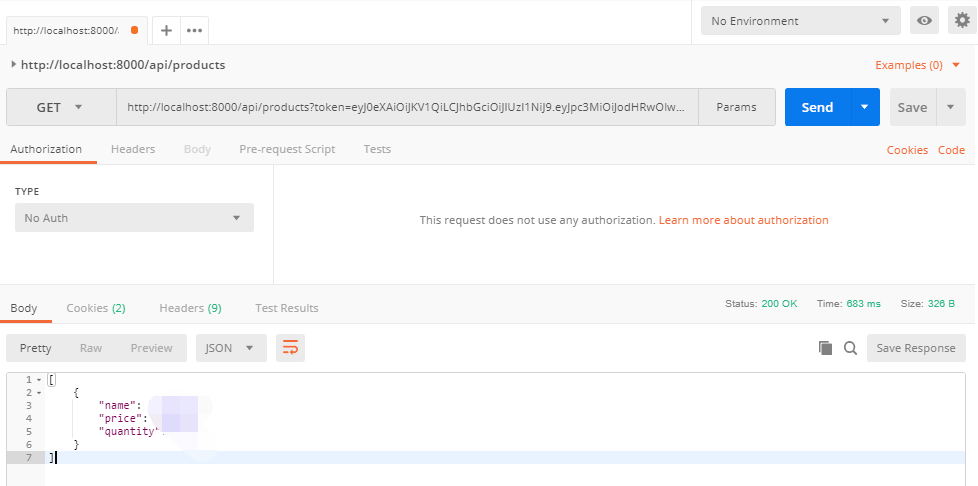
获取用户的详细信息：



以上为身份的相关测试，接下来测试产品部分，首先我们创建一个产品。



接下来通过index的方法获取我们的产品



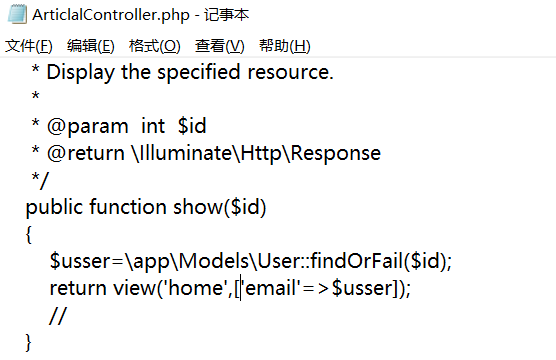
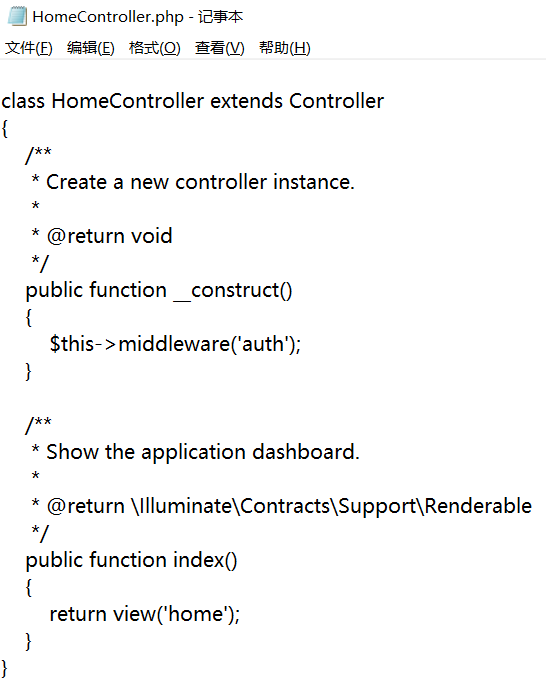
可见，相关的路由均可正常工作，本次实验成功。

1. **实验分析总结及心得（该部分也是评分的一个重点）**

（结合所学知识对实验过程中观察到的实验结果进行分析总结，以便加深对知识的理解，并总结通过实验学到的知识或技术）

特别推荐学生写出做实验遇到的问题，以及从原理分析得到解决方案的过程。

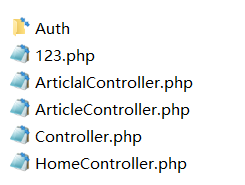
在一开始的分析完成后，由于与实验三的需求存在重叠，因此原想法是在实验三的基础上进行一些代码的重构，例如尝试将原有的article文件按原有步骤命名为articlal文件进行区分尝试做重构，但是最终发现相关代码逻辑十分混乱，难以继续进行，因此重新新建了项目的目录，在jwt文件夹下继续进行相关的尝试。这启示我们在进行开发的过程中，需要有良好的项目书写规范，该新建就新建，不要依靠沉没成本做决策。



Artilalcontroller与homecontroller乱作

一团，相互引用，增添了很多不必要的麻

烦。



多个版本放在一起的文件夹，也显得很凌乱。

另外，之前在学习的过程中，一直没有对Laravel所谓的MVC框架有一个更好的了解，单纯通过修改单个文件，感觉非常麻烦。在这次实验完成之后，我会尝试使用PHPstorm等更多更好地工具提升web开发的工作效率。在学习资源的寻找上，这次实验相对于实验三而言更加困难，因此需要提升我们的“搜商”，更好地解决相关问题。本次实验用时相对较长，有很大一部分原因是找不到合适的学习资料所导致的。但通过这次实验，也提高了我对于MVC框架的基本理解，提高了对Web的相关兴趣，在暑假中也同样会继续深入学习，希望未来能够取得更好地成绩。

1. **参考文献**

(1)https://zhuanlan.zhihu.com/p/97978097

(2)https://learnku.com/articles/19172

(3)https://laravelacademy.org/post/6171.html

(4)https://laravelacademy.org/post/9153.html

(5)https://www.extendfeature.com/laravel-8-target-class-seeder-does-not-exist/

(6)https://www.cnblogs.com/hhyl/p/13488424.html

(7)https://www.bilibili.com/video/BV1Vz4y1y7Do?

(8)https://blog.csdn.net/qq\_30682027/article/details/78225074

(9)https://laravelacademy.org/post/9283

(10)https://learnku.com/docs/laravel/5.5

(11)https://segmentfault.com/a/1190000013474769

(12)https://blog.csdn.net/createno\_1/article/details/81738665

(13)https://www.php.cn/php-weizijiaocheng-403288.html