此次改动是某冬子突然改需求才改的，为了应对后续可能的更多无理的需求改动几乎完全重构了安全模块

本次项目中的身份安全认证是以JWT为标准的token认证

关于JWT和token认证参考<https://www.jianshu.com/p/e88d3f8151db>

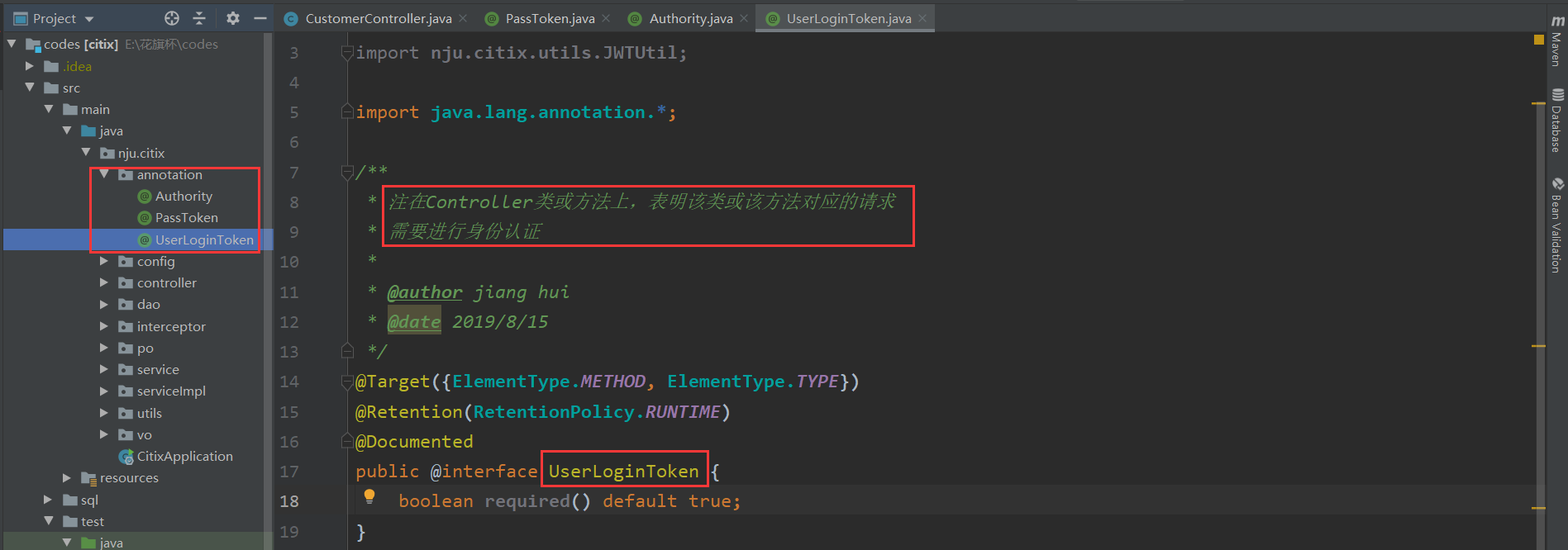
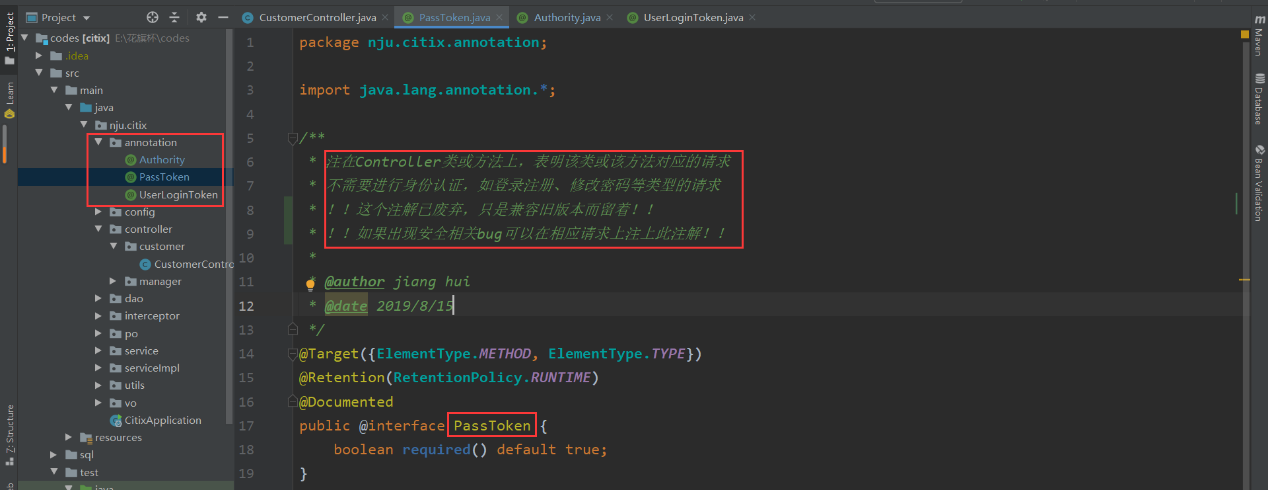
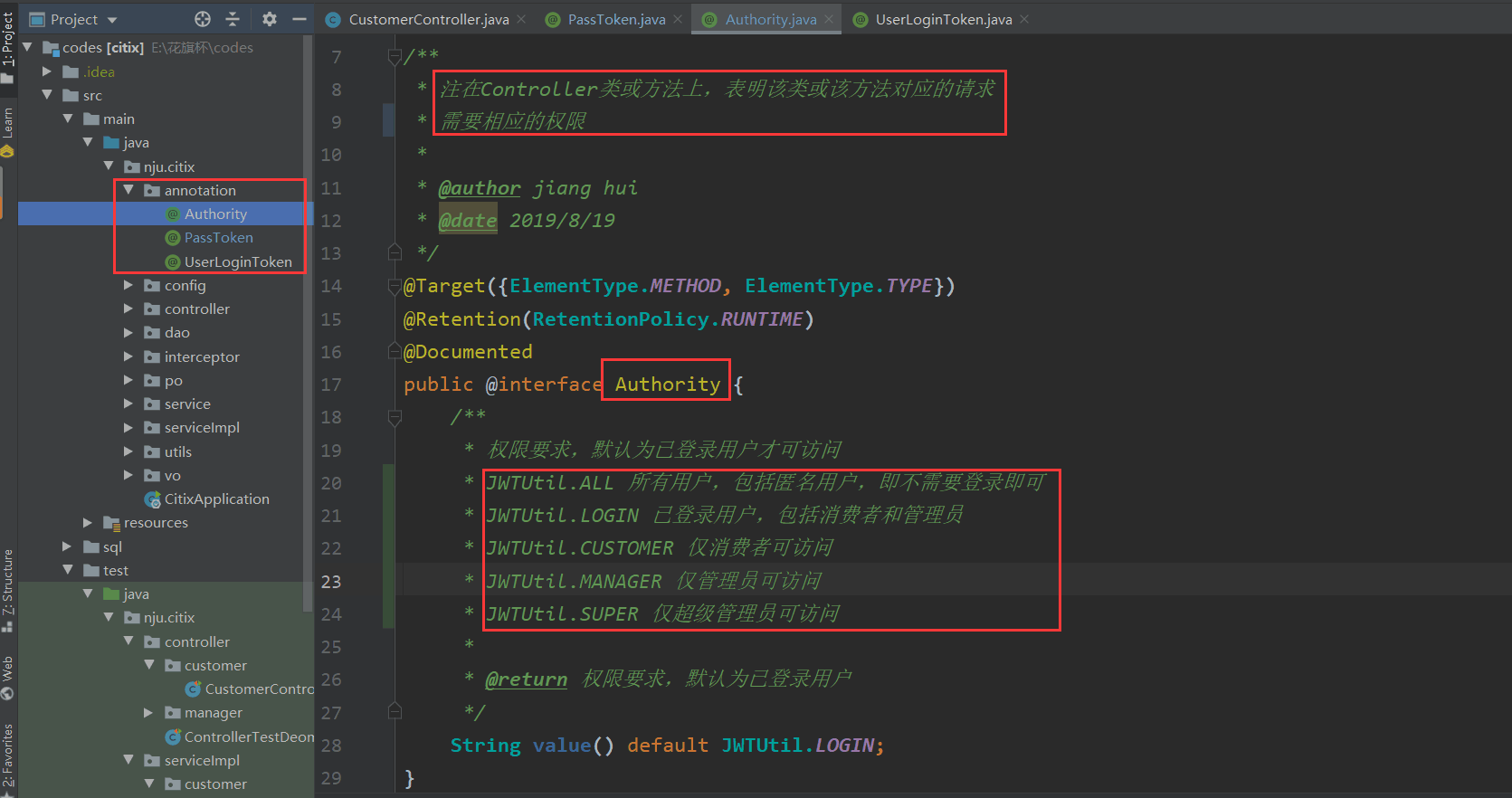
在实际代码中也是在上述网站里的源码中把轮子修改到符合我们的需求

<https://github.com/JinBinPeng/springboot-jwt>

感兴趣的同学可以自己去看看，这里我就简述一下项目中安全认证模块的用法

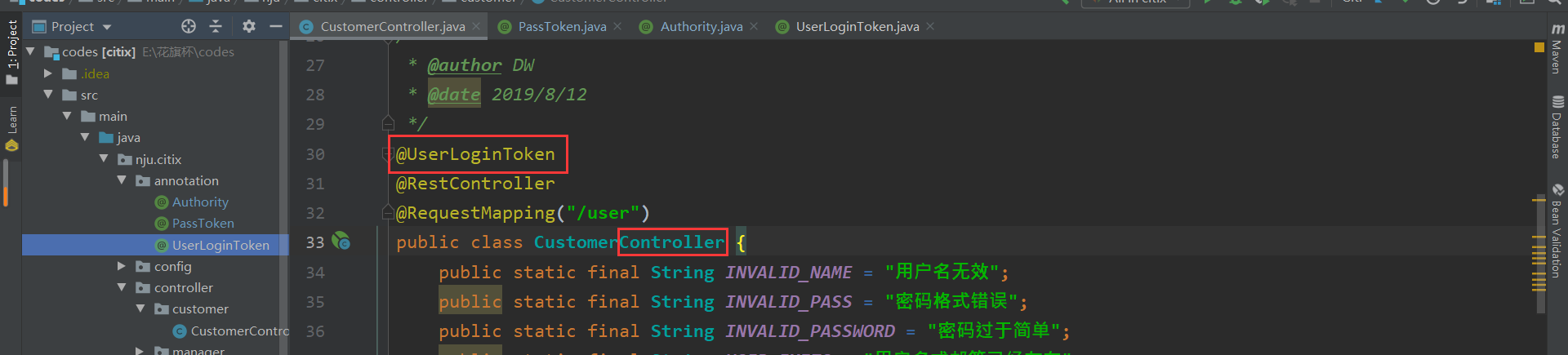
**自定义注解使用方式**

这里是**三个自定义注解**，用法在注释里说了，用来标注一个请求是否要进行安全认证，**注的位置和@RequestMapping、@PostMapping、@GetMapping一样**

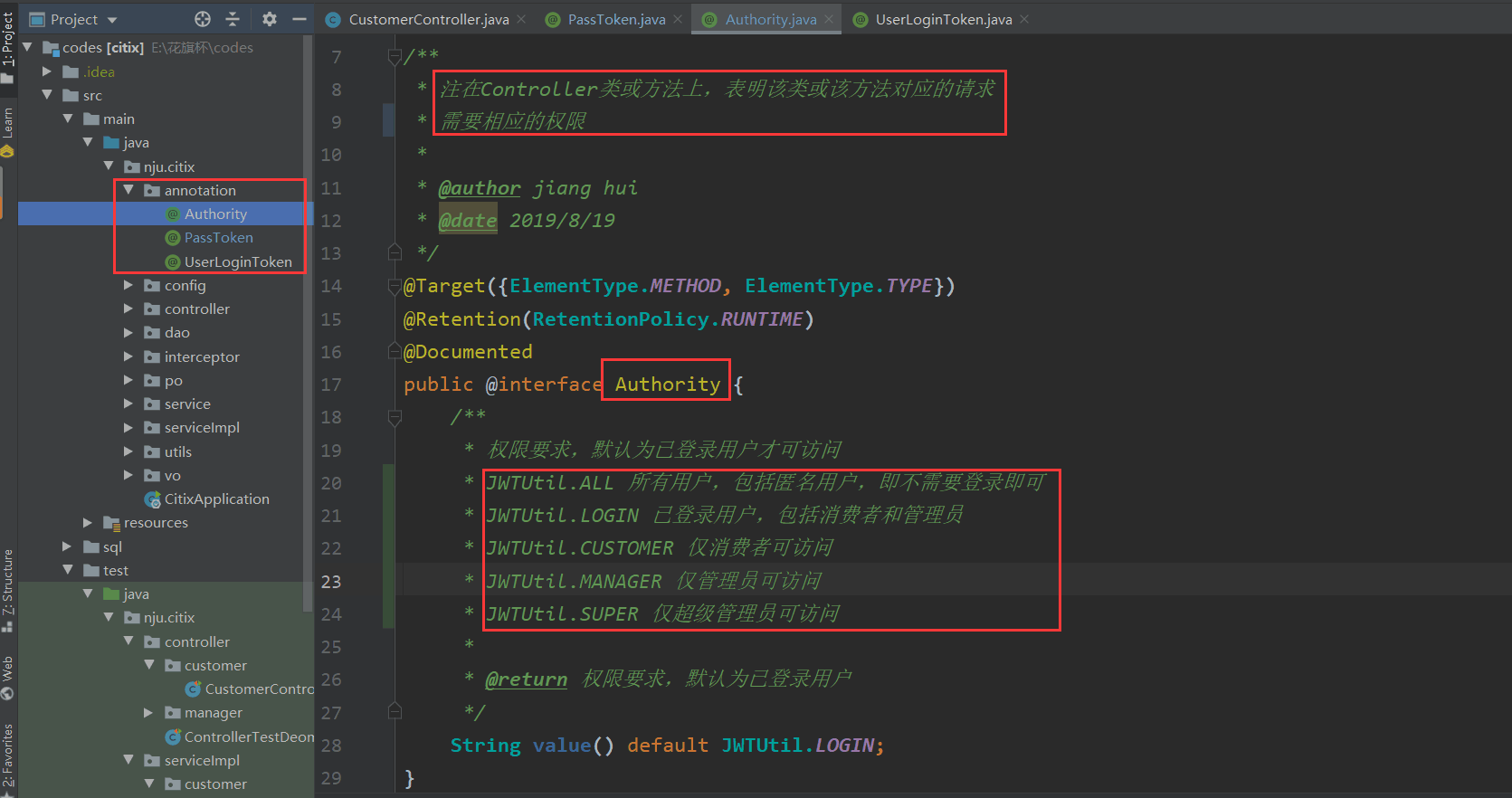


**建议用法！！！**

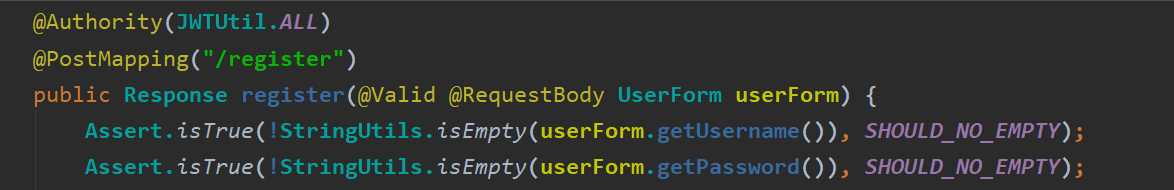
**@UserLoginToken**注在**controller**上面表明该controller下所有请求都需要安全认证，在开发中可以不加，等确认没问题后再加上安全认证

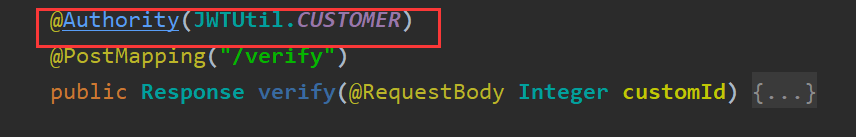


**@Authority**注在请求对应的**方法**上修饰对应请求需要的权限

目前共有5中权限级别，默认是需要登录后访问，后续有要求尽早在群里提出

建议注在请求对应的方法上





**最好每个controller类上都注@UserLoginToken**

**每个请求对应的方法都注上@Authority(JWTUtil.XXX)**

如果不知道相应请求权限或暂时不想考虑认证就先用@Authority(JWTUtil.ALL)

**！！！@Authority(参数)的参数调用请使用引用，不要使用生字符串！！！**

**其他用法(不推荐)**

这里就不截图了，最好不要这么用，只是我保留了旧版本设计中的更方便的用法

1、@UserLoginToken和@Authority其实可以注在方法和类上，表面该类下所有请求或该方法对应的请求需要安全认证和需要相应的权限，注在方法上的优先级高于注在类上的

2、@PassToken的优先级最高，不过其作用和不注@UserLoginToken或注上@Authority(JWTUtil.ALL)的效果一样

3、@Authority不加参数时默认值为@Authority(JWTUtil.LOGIN)

4、在需要安全验证的请求中如果不注@Authority默认权限需求为@Authority(JWTUtil.LOGIN)

5、在开发过程中可以先不考虑安全认证问题，不注我的自定义注解时我默认通过验证，但是注意在开发完成后记得加上相应的注解

！！！@PassToken是上个版本的注解，实际作用和@Authority(JWTUtil.ALL)一样！！！

！！！这里保留来兼容旧版本！！！

！！！在安全认证出bug时可以用来跳过安全监测！！！

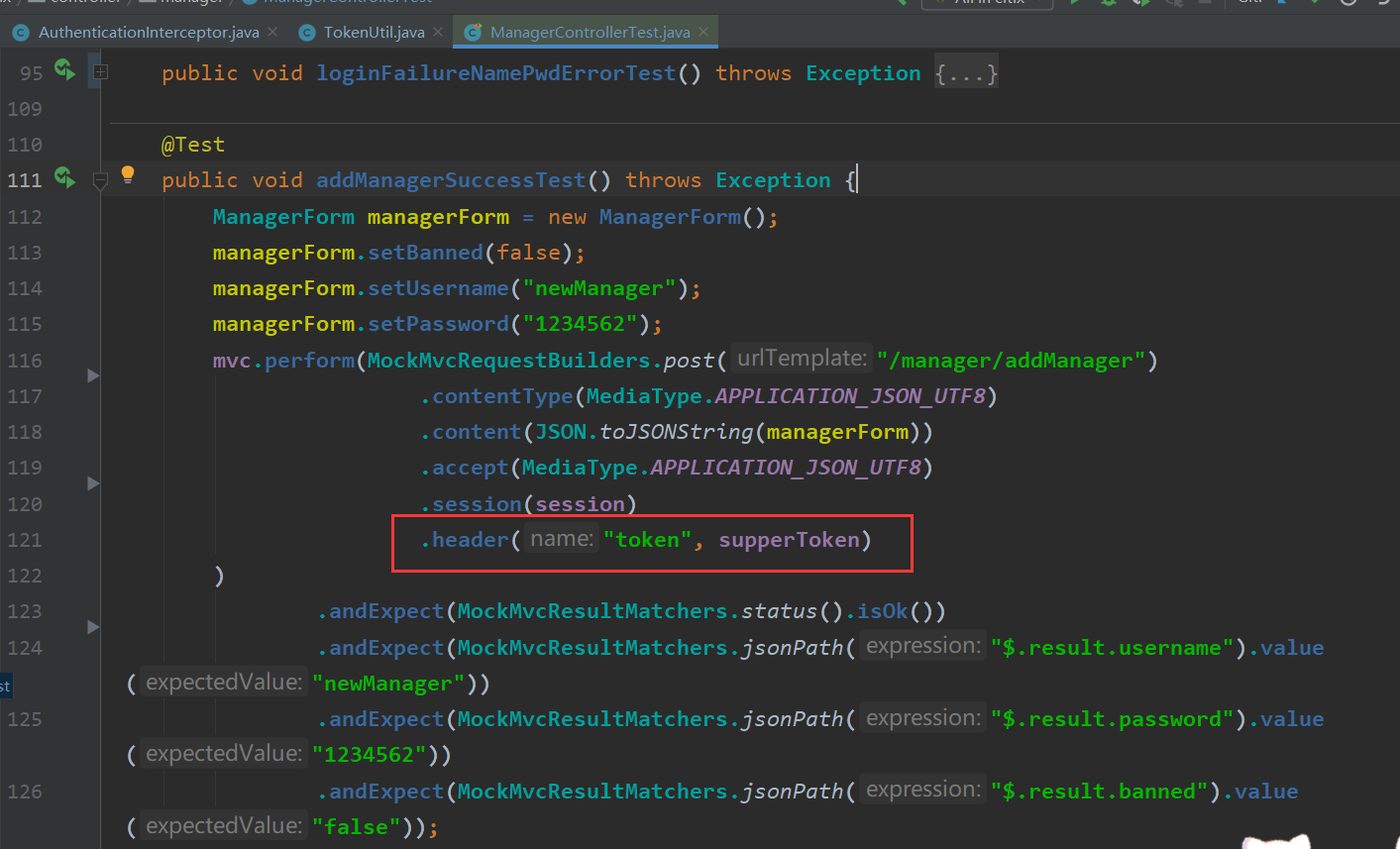
！！！不过暂时没发现和@Authority(JWTUtil.ALL)有区别！！！

**token的使用问题**

token的使用较上个版本没有变化，亲测结果是以前能过的测试一行代码都不需要修改

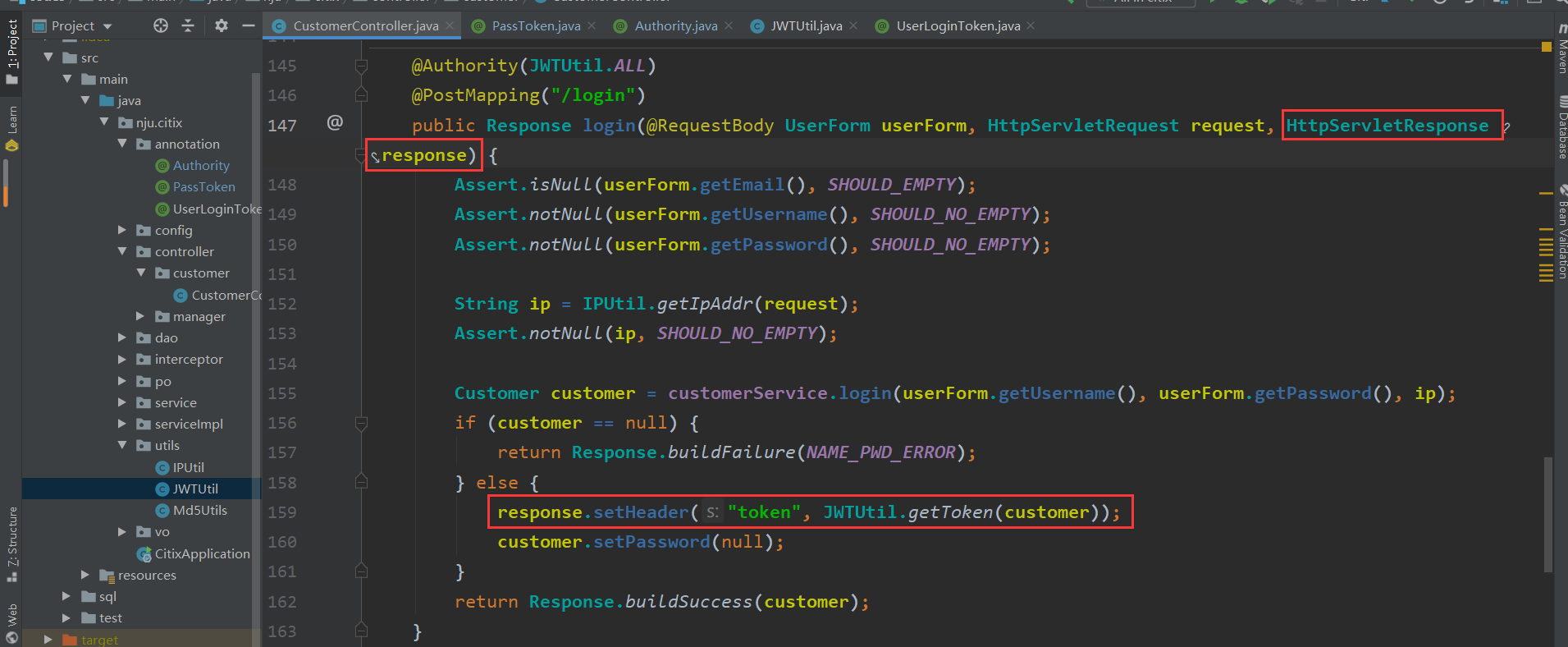
**token是放在请求头里的**

写测试的时候如果是**controller层的测试要在需要安全认证的请求里加上相应用户的token，**不对应用户的token会返回错误消息。下面例子里是预定义超级管理员增加一个管理员的测试

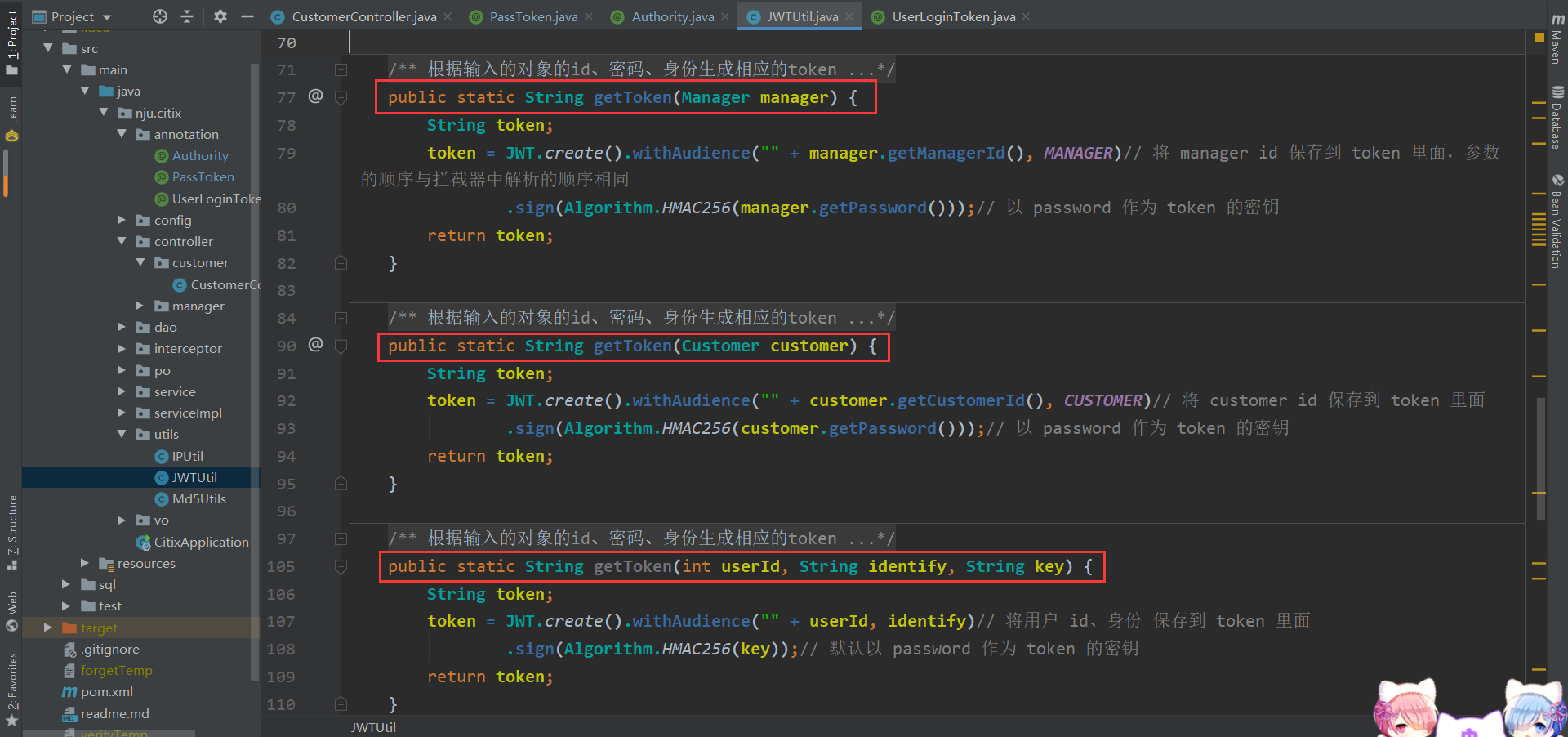


**token是放在请求头里的**

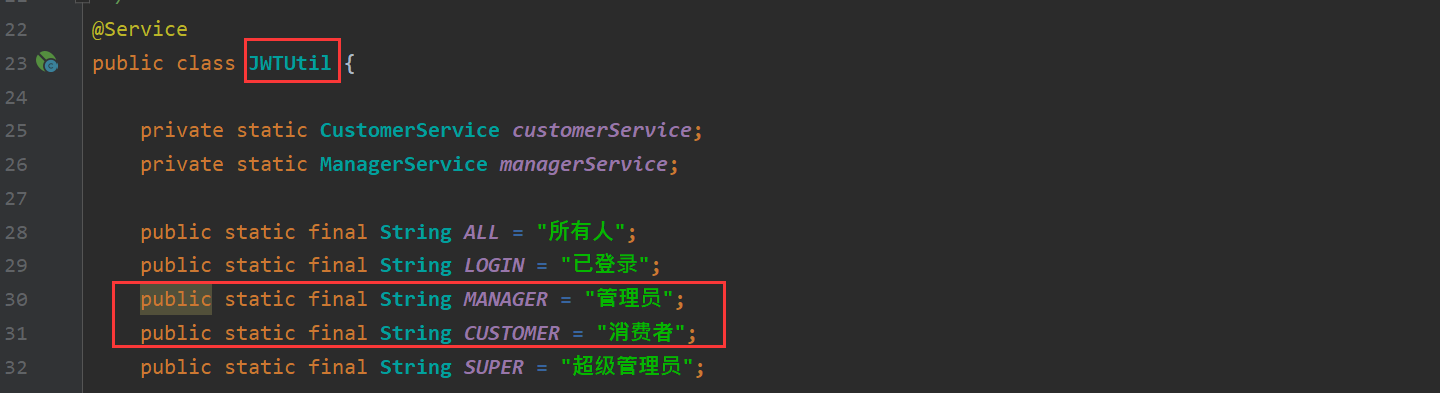
如果是登录注册等会返回token的方法里，**要将token写在header里返回去**，而不是写在Response里面，同时注意不要将用户密码返回，可以设置为null，这里的HttpServletResponse如果不熟悉没关系，它就是你的接口实际返回的东西，直接setHeader()就可以将token放进去



对应用户的**token生成方法在TokenUtil类里面**可以看到，我写成了**静态方法，需要的参数目前就是用户的id，可以传入一个对象，也可以传入id和相应的身份**，方便后续修改token的生成方式



Token用到的身份只有两个，调用时**请使用引用**，不要使用生字符串



**前端的同学注意一下**

**Token是放在请求头里的**

发送请求的时候注意看接口文档相应接口是否要求身份验证，最好可以写一个全局的设置，在所有请求开始先确认一下token的有效性，比如是否过期等，token时效性我暂时没有写，就是说获取了一次对应用户的token，那个用户的token就不会变了，在每个请求的请求头里加上就行

Ps：上个版本我说明了我的实现方式，这次改动几乎完全重构了代码，个人感觉写的挺优雅的了，就懒得写说明了，有兴趣的同学可以私戳我，我们可以交♂流一下更好的实现