1 实验10 SpringBoot集成Spring Security

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学号 |  | 姓名 |  | 班级 |  |
| 上课地点 |  | 教师 |  | 上课时间 |  |

**Notice：**

**1、实验内容都是老师手打，有错误难免，请自行更正。**

一． 实验目的

1. 了解和掌握vue+SpringBoot+MyBatis的开发方法

2. 了解SpringBoot和SpringSecurity的集成方法

二、实验要求：

MySQL 8.0

Eclipse 2019-09

VS Code

三． 实验内容

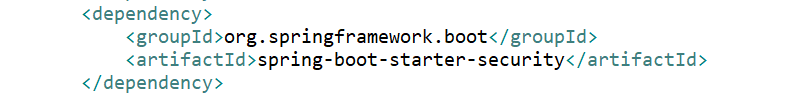
**本实验在实验9的基础上增加了SpringBoot和Spring Secutiy的集成，所使用代码和实验9一样，重复部分不再详述。**

**参考资料：**

[**https://blog.csdn.net/yuanlaijike/article/details/80249235**](https://blog.csdn.net/yuanlaijike/article/details/80249235)

[**https://www.tqwba.com/x\_d/jishu/388428.html**](https://www.tqwba.com/x_d/jishu/388428.html)

1. 修改pom.xml文件

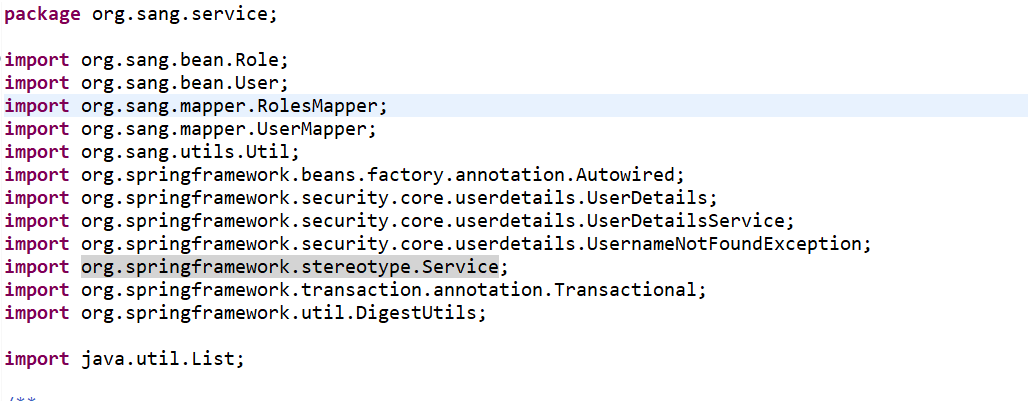


1. 实现UserService类，该类继承自UserDetailsService，同时需要在该类中重写loadUserByUsername方法，注意：

1）数据库中至少需要保存User和Role的关系，通常通过1对多的表关系来实现

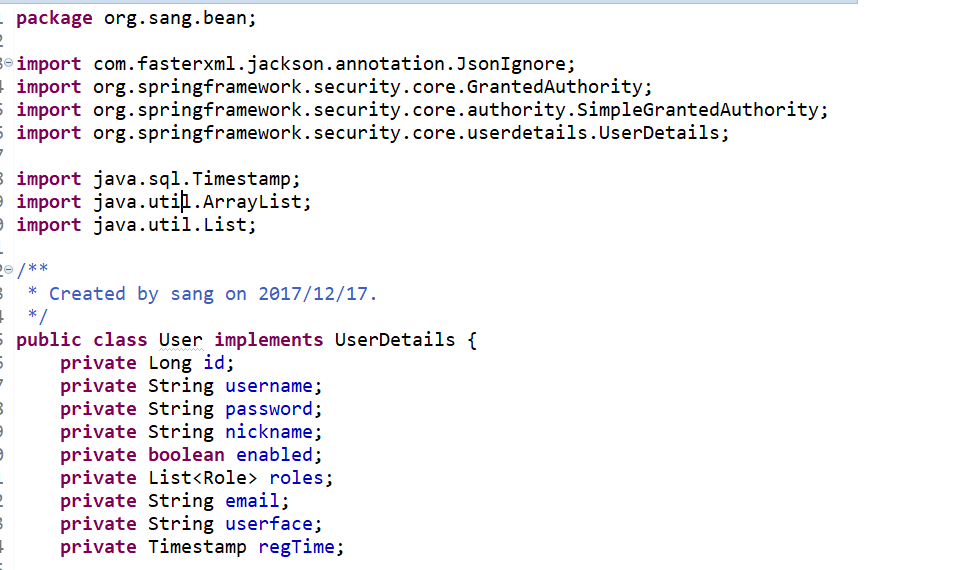
2）在该方法中需要调用user.setRoles初始化当前user的role属性



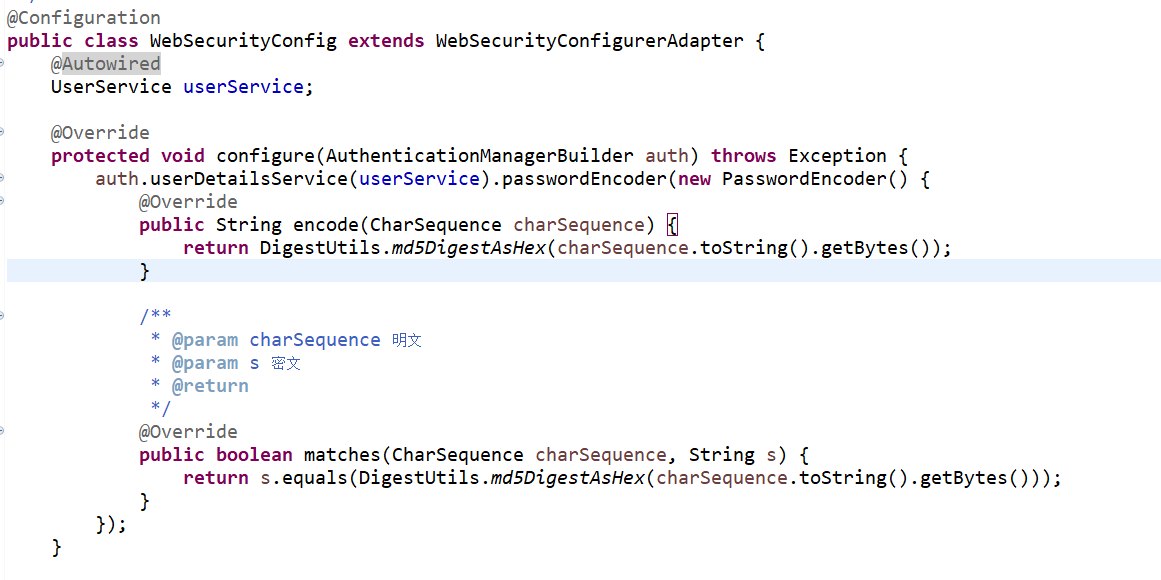




1. 实现User类，该类继承与UserDetails，并与类Role为1对多关系



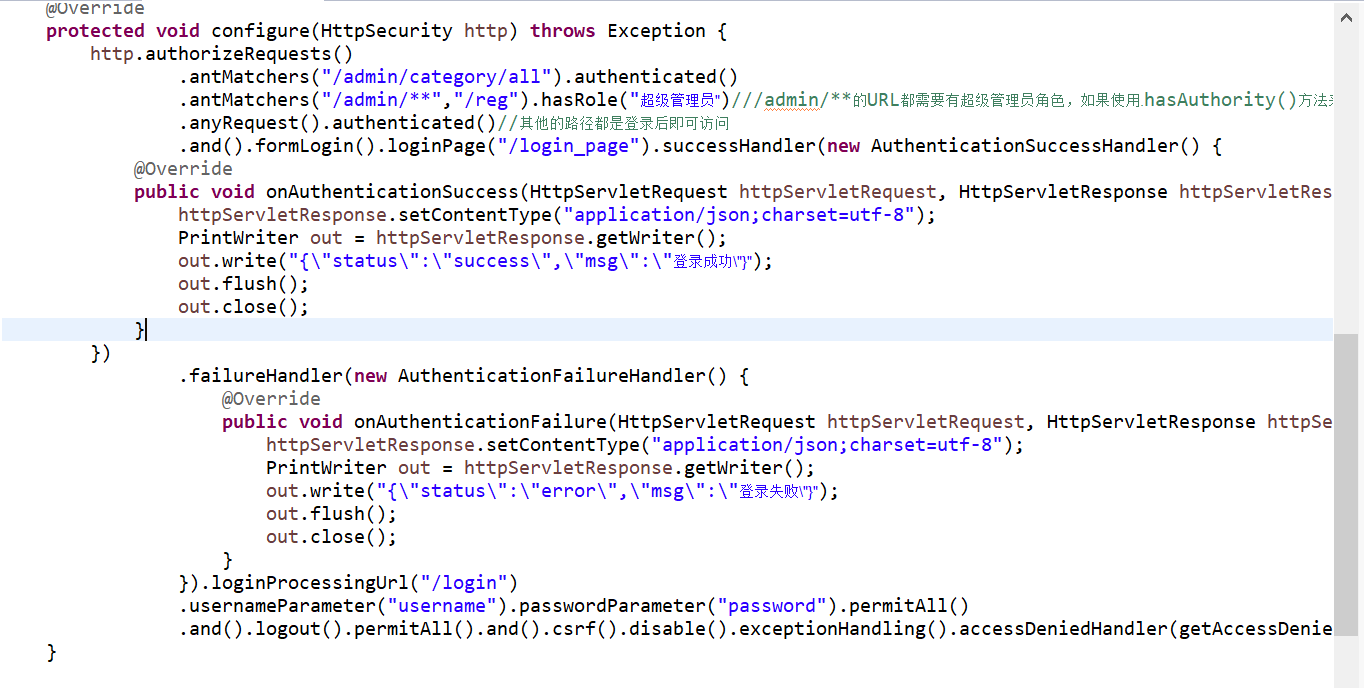
1. 继承WebSecurityConfigurerAdapter类并重写其中的几个方法，上一步中实现的UserService类需要注入到这个类中
2. 重写configue方法注入user并对密码进行加密



1. 重写configue方法实现相应的url权限规则

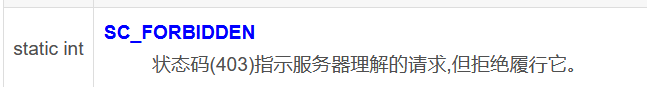
下面的意思是:/admin/\*\* 的URL需要超级管理员

其他URL需要登录



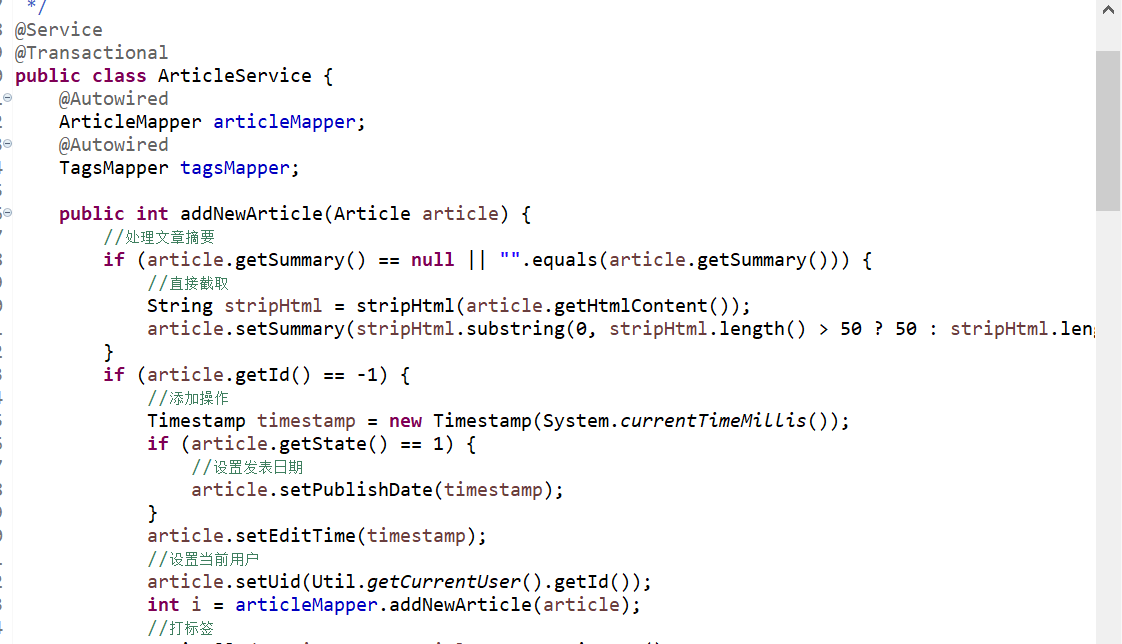
1. 实现AuthenticationAccessDeniedHandler

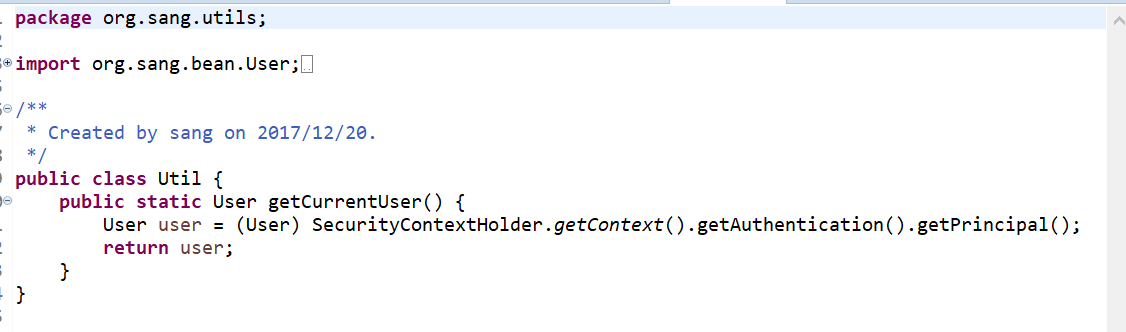
https://blog.csdn.net/goolejuck/article/details/24328527





通过SpringSecurity得到当前登录用户的方法：





四、练习（提交作业时只需要交练习）

注：本实验使用代码和实验9的代码一致

根据实验内容完成以下内容：

1. 根据本实验内容，将你在实验9中实现的功能同本节课的内容进行整合，要求在登录后的页面增加几个菜单，对这些菜单需要进行权限访问限制，部分URL只允许特定的用户访问 （如下图所示）
2. 文件要求如下：
   1. UserService.java中的主要代码为文字
   2. WebSecurityConfig.java中的主要代码为文字
3. 其他文件可以为截图



示例： 为避免重复，下面红色字体部分可以删除，可以增加少量文字说明

UserService文件

**public** **class** UserService **implements** UserDetailsService {

@Autowired

UserMapper userMapper;

@Autowired

RolesMapper rolesMapper;

@Override

**public** UserDetails loadUserByUsername(String s) **throws** UsernameNotFoundException {

User user = userMapper.loadUserByUsername(s);

**if** (user == **null**) {

//避免返回null，这里返回一个不含有任何值的User对象，在后期的密码比对过程中一样会验证失败

**return** **new** User();

}

//查询用户的角色信息，并返回存入user中

List<Role> roles = rolesMapper.getRolesByUid(user.getId());

user.setRoles(roles);

**return** user;

}

WebSecurityConfig.java

@Override

**protected** **void** configure(HttpSecurity http) **throws** Exception {

http.authorizeRequests()

.antMatchers("/admin/category/all").authenticated()

.antMatchers("/admin/\*\*","/reg").hasRole("超级管理员")///admin/\*\*的URL都需要有超级管理员角色，如果使用.hasAuthority()方法来配置，需要在参数中加上ROLE\_,如下.hasAuthority("ROLE\_超级管理员")

.anyRequest().authenticated()//其他的路径都是登录后即可访问

.and().formLogin().loginPage("/login\_page").successHandler(**new** AuthenticationSuccessHandler() {

@Override