目录

[一、 技术框架 3](#_Toc514228140)

[1. 后端 3](#_Toc514228141)

[2. 前端 3](#_Toc514228142)

[二、 配置文件说明 4](#_Toc514228143)

[1. 一览配置文件 4](#_Toc514228144)

[2. application.yml 4](#_Toc514228145)

[3. bootstrap.yml 4](#_Toc514228146)

[4. spring security配置 5](#_Toc514228147)

[5. logback-admin.xml 5](#_Toc514228148)

[6. 跨域配置 6](#_Toc514228149)

[7. 全局统一的异常处理 7](#_Toc514228150)

[三、 登陆逻辑 8](#_Toc514228151)

[1. 登陆第一步 8](#_Toc514228152)

[2. 登陆成功 9](#_Toc514228153)

[3. 登陆失败 9](#_Toc514228154)

[4. 校验密码 9](#_Toc514228155)

[四、 Token实现类 11](#_Toc514228156)

[1. redis实现 11](#_Toc514228157)

[2. 数据库实现 12](#_Toc514228158)

[3. 切换到数据库存储 12](#_Toc514228159)

[五、 密码加密 13](#_Toc514228160)

[六、 详细配置 14](#_Toc514228161)

[4. 数据源配置 14](#_Toc514228162)

[5. Mybatis配置 14](#_Toc514228163)

[6. 上传文件 15](#_Toc514228164)

[7. 日志配置 16](#_Toc514228165)

[8. 端口号配置 16](#_Toc514228166)

[9. Redis配置 17](#_Toc514228167)

[10. 异步、线程池配置 17](#_Toc514228168)

[11. 邮件配置 18](#_Toc514228169)

[七、 接口开发访问说明 18](#_Toc514228170)

[八、 代码生成介绍 20](#_Toc514228171)

[1. 输入表名 20](#_Toc514228172)

[2. 预览 20](#_Toc514228173)

[九、 定时任务管理介绍 22](#_Toc514228174)

[1. 添加任务 22](#_Toc514228175)

# 技术框架

## 后端

Jdk 1.8

核心框架：Spring boot

安全框架：spring security

持久层框架：Mybatis

数据库连接池：Alibaba Druid

缓存框架：redis

日志框架：SL4J

前后端交互：json

## 前端

Jquery

Html

样式、弹窗：Layui

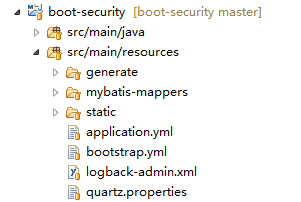
样式：Bootstrap

树形：Ztree

分页：Datatables

# 配置文件说明

## 一览配置文件



## application.yml

此文件为spring boot的基础配置文件，配置了数据源、上传文件大小、mybatis、上传文件路径、日志级别、日志目录、token过期时间、端口号，以及自定义配置参数。

## bootstrap.yml

此配置同为spring boot的基础配置文件，加载优先于application.xml可以理解为，此文件配置系统级参数，application.yml配置应用级参数。

## spring security配置

com.boot.security.server.config.SecurityConfig



com.boot.security.server.config.SecurityHandlerConfig

该类配置登陆成功、登陆失败、未登录返回401、退出，处理器。

## logback-admin.xml

日志配置，此文件中的变量引用application.yml中的自定义参数。



将日志分为了三类



admin-server.log 可输出业务参数，可根据业务场景进行再分包

admin-server.log.error 所有的异常日志

admin-server.log.sql 通过mybatis生成的sql语句、参数、返回个数

## 跨域配置



## 全局统一的异常处理

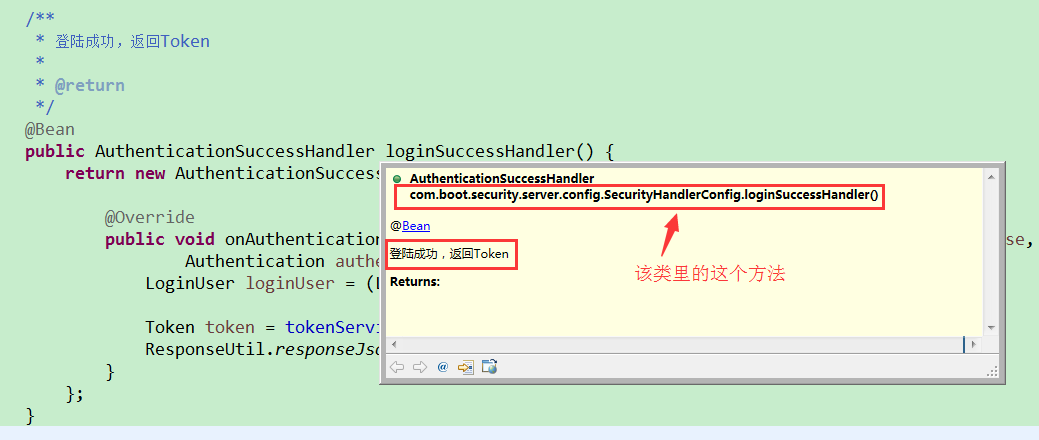


# 登陆逻辑

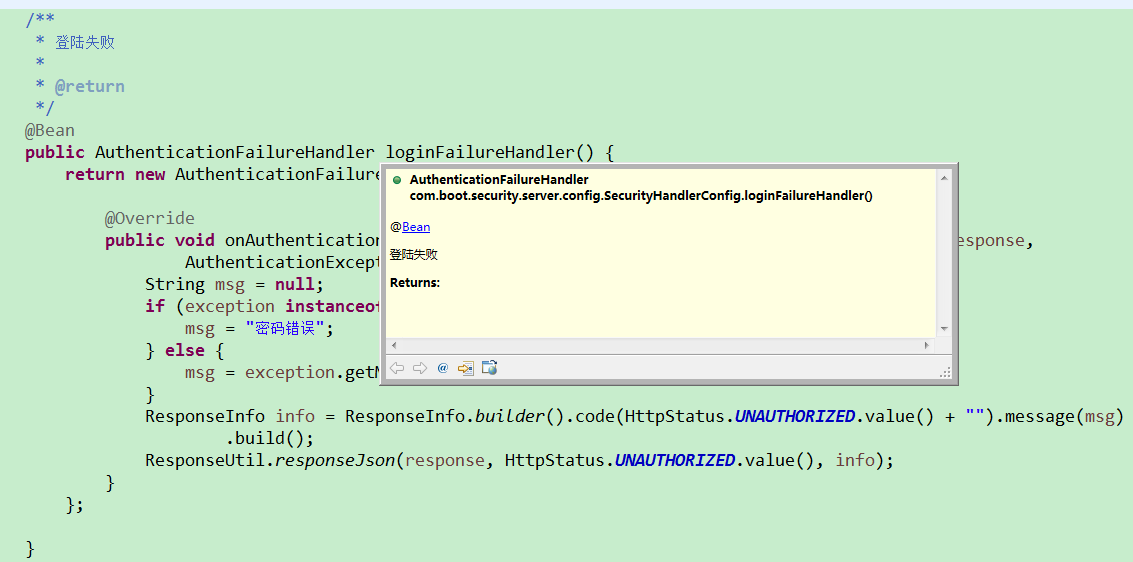
## 登陆第一步



## 登陆成功



## 登陆失败



## 校验密码

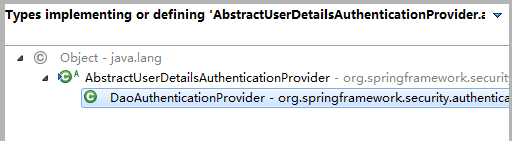
很多同学没发现密码校验在哪，其实这块是spring security底层做的校验，

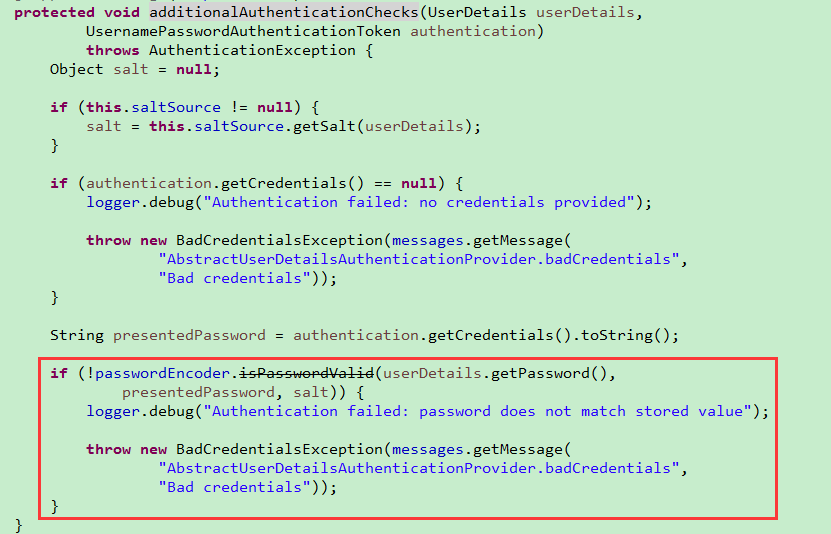
可以沿着源码查看类

[org](eclipse-javadoc:%E2%98%82=0-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/4.2.3.RELEASE%5C/spring-security-core-4.2.3.RELEASE.jar%3Corg).[springframework](eclipse-javadoc:%E2%98%82=0-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/4.2.3.RELEASE%5C/spring-security-core-4.2.3.RELEASE.jar%3Corg.springframework).[security](eclipse-javadoc:%E2%98%82=0-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/4.2.3.RELEASE%5C/spring-security-core-4.2.3.RELEASE.jar%3Corg.springframework.security).[authentication](eclipse-javadoc:%E2%98%82=0-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/4.2.3.RELEASE%5C/spring-security-core-4.2.3.RELEASE.jar%3Corg.springframework.security.authentication).[dao](eclipse-javadoc:%E2%98%82=0-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/4.2.3.RELEASE%5C/spring-security-core-4.2.3.RELEASE.jar%3Corg.springframework.security.authentication.dao).AbstractUserDetailsAuthenticationProvider里的方法



这个方法的实现





# Token实现类

接口[com](eclipse-javadoc:%E2%98%82=boot-security-jpa/src%5C/main%5C/java%3Ccom).[boot](eclipse-javadoc:%E2%98%82=boot-security-jpa/src%5C/main%5C/java%3Ccom.boot).[security](eclipse-javadoc:%E2%98%82=boot-security-jpa/src%5C/main%5C/java%3Ccom.boot.security).[server](eclipse-javadoc:%E2%98%82=boot-security-jpa/src%5C/main%5C/java%3Ccom.boot.security.server).[service](eclipse-javadoc:%E2%98%82=boot-security-jpa/src%5C/main%5C/java%3Ccom.boot.security.server.service).TokenService

## redis实现

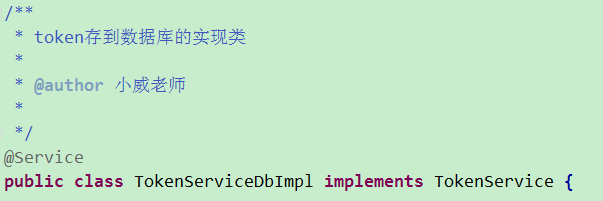
JWT



简单uuid，已作废



## 数据库实现



## 切换到数据库存储

根据上图发现，redis实现比数据库实现的service多了一个注解@Primary，那么只要将com.boot.security.server.service.impl.TokenServiceJWTImpl类上的注解@Primary挪到com.boot.security.server.service.impl.TokenServiceDbImpl诺过去即可。或者直接干掉TokenServiceJWTImpl类上的注释



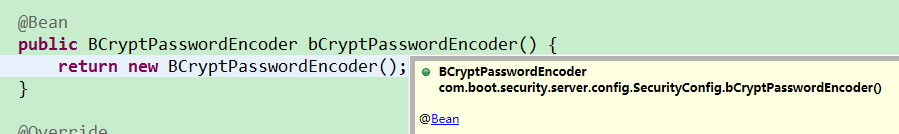
其实接口有多个实现类，并且都@Service了的话

@Autowired

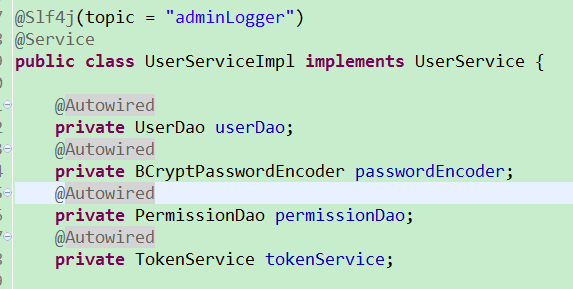
**private** TokenService tokenService;

**自动注入起作用的就是带@Primary的实现类。**

# 密码加密



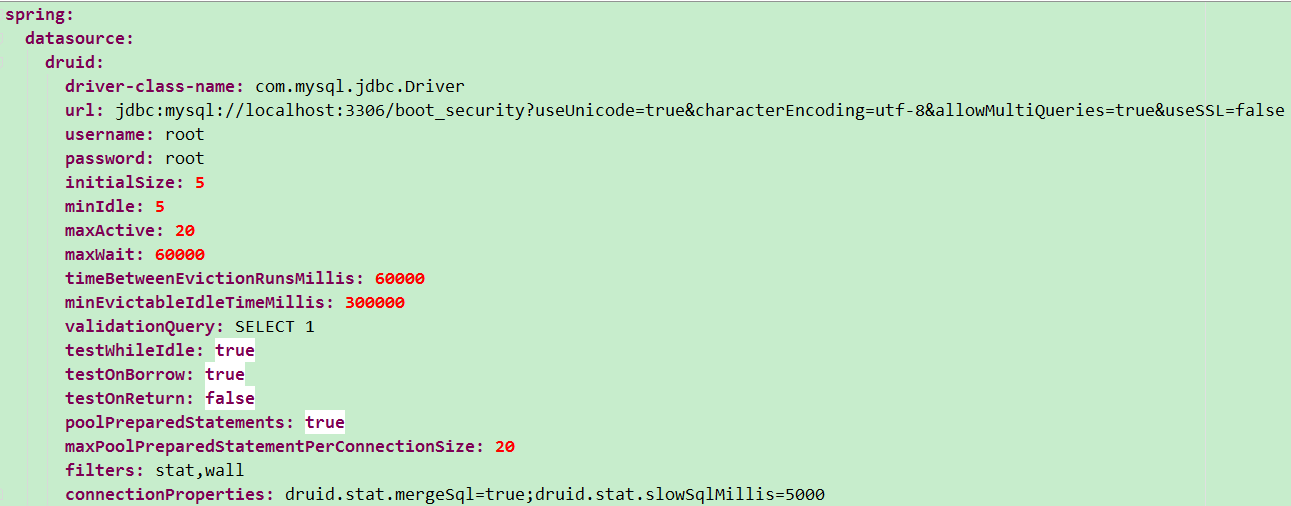
具体参考添加用户和修改密码的UserServiceImp.java



# 详细配置

## 数据源配置

在application.yml中用的是阿里巴巴的druid，主要会修改端口号，数据库名，用户名和密码。



## Mybatis配置

在application.yml中



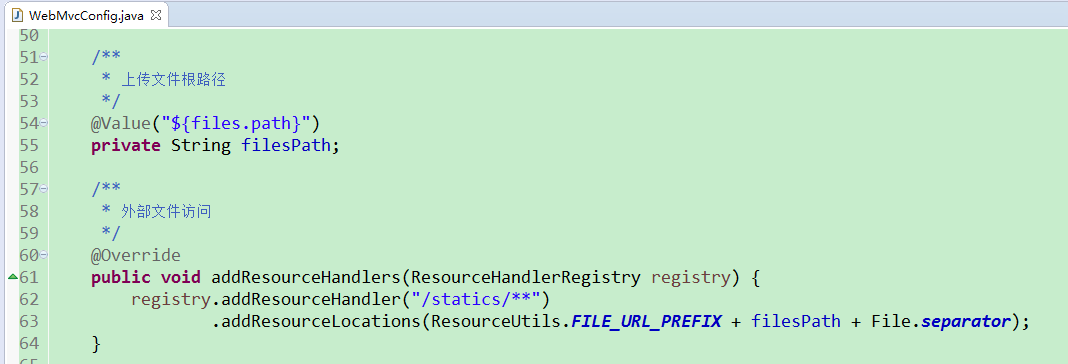
type-aliases-package，配置类别名的包，如有多个包，用逗号分开继续写，如com.boot.security.service.model, com.boot.security.service.test.model

mapper-locations，配置复杂sql的Mapper文件，简单sql直接用注解写在dao文件里面了。

## 上传文件



上传文件会存储在该路径下，对应



如d:/files/a.png,访问路径就是<http://localhost:8080/statics/a.png>



此处配置springmvc上传文件大小限制

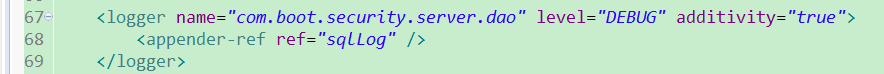
## 日志配置



日志文件配置位置、root日志级别，自定义业务日志级别，生成日志文件位置，日志文件分包大小

对应logback-admin.xml中的





## 端口号配置



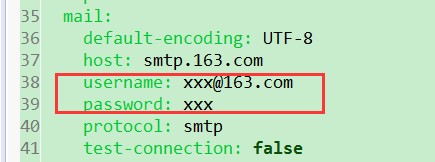
## Redis配置



## 异步、线程池配置



## 邮件配置

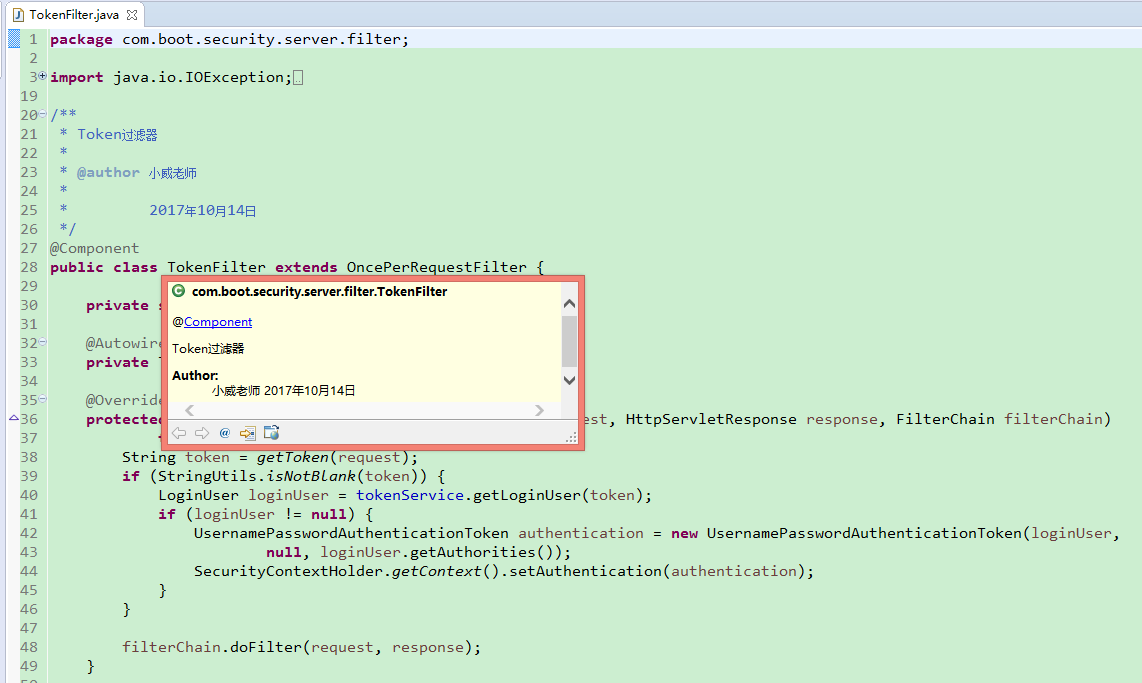


将用户名密码改掉

# 接口开发访问说明

如可以<http://localhost:8080/users/current?token=049ba87d-6d> 这样以参数的形式带上token，也可以把token放入header中。

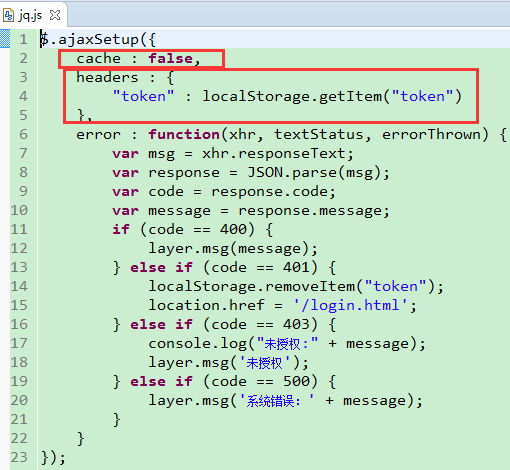
[com](eclipse-javadoc:%E2%98%82=boot-security/src%5C/main%5C/java%3Ccom).[boot](eclipse-javadoc:%E2%98%82=boot-security/src%5C/main%5C/java%3Ccom.boot).[security](eclipse-javadoc:%E2%98%82=boot-security/src%5C/main%5C/java%3Ccom.boot.security).[server](eclipse-javadoc:%E2%98%82=boot-security/src%5C/main%5C/java%3Ccom.boot.security.server).[filter](eclipse-javadoc:%E2%98%82=boot-security/src%5C/main%5C/java%3Ccom.boot.security.server.filter).TokenFilter



从请求参数或者header中解析token，

在页面开发中，只需要引入jq.js即可，具体根据你的页面文件位置进行引入

如<script type="text/javascript" src="../../js/jq.js"></script>



该jq.js文件，对ajax进行了全局处理，禁用缓存，将token放入了header

# 代码生成介绍

## 输入表名

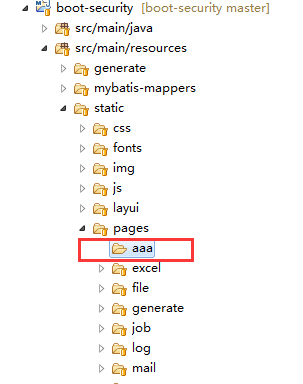


## 预览



保存，将根路径下java文件copy到项目里，

页面文件copy到static下的pages里，单独建一个文件夹放入页面文件如aaa



Mapper.xml放入mybatis-mappers文件夹下即可，重启项目，配置菜单和权限即可。

# 定时任务管理介绍

## 添加任务



仅支持无参数方法，选定springBeanName会自动联动方法名。

Job名请使用唯一值，cron表达式自己输入，可点击检查来校验。