CMS第二次设计报告

1. **小组成员**

前端：吴梦凝19301050

后端：吴梦凝19301050

设计：任城仪19301045

1. **概要设计**

本系统基于第一次作业进行迭代，在原有的基础上将前后端分离，并使用Restful API接口规范。

本系统采用springboot+restful api+Mybatis+Spring security+jwt+vue+junit

+Tomcat+Bucket4j的方式进行实现。

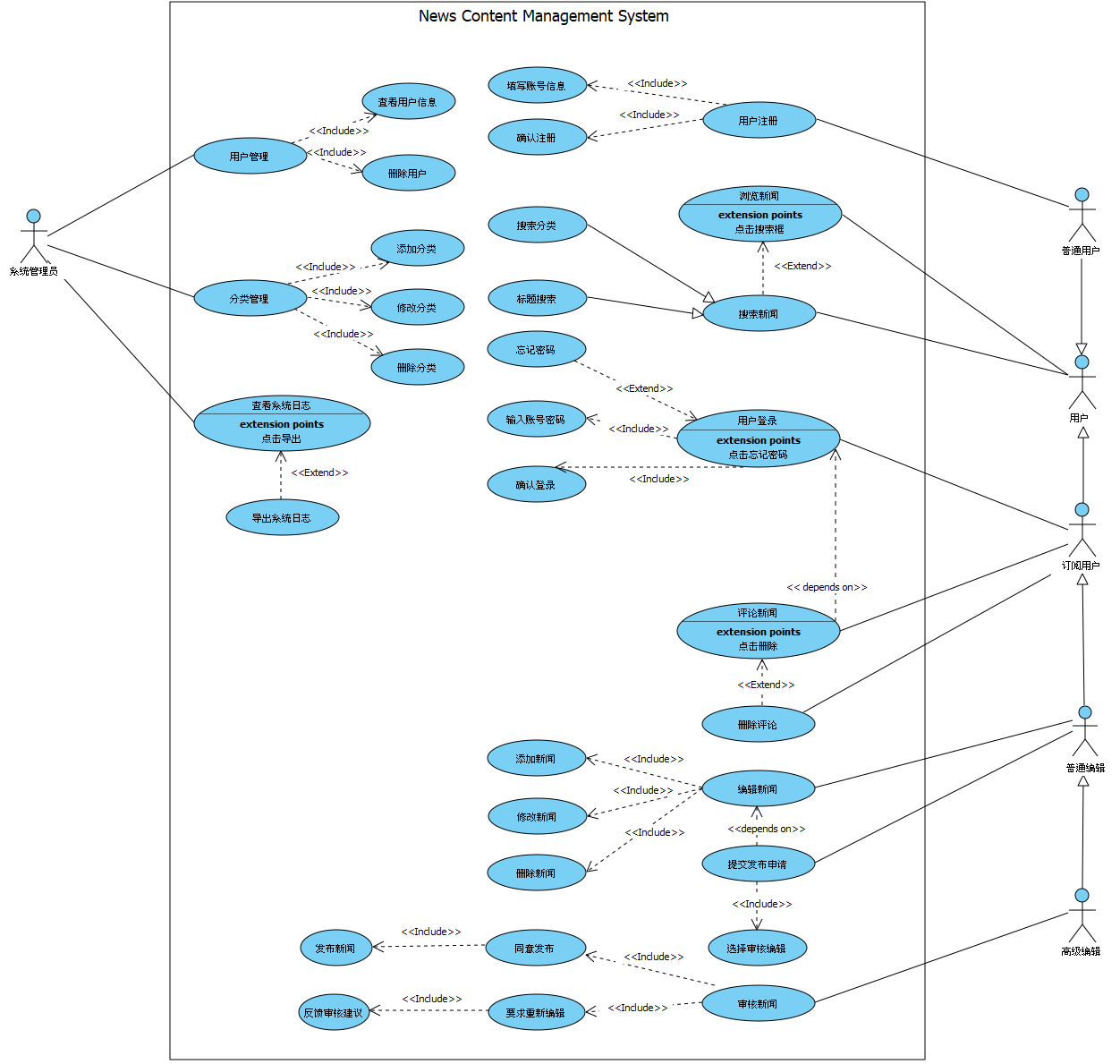
|  |  |
| --- | --- |
| Springboot +Mybatis+ Tomcat | 与第一次作业功能相同 |
| Restful api | 在原有springboot项目上前后端分离并规范其接口风格 |
| Spring security | 用于对系统安全进行防护，包括授权和登录 |
| JWT | 用于实现前端用户登录和授权； |
| Vue  （包含相关组件） | 用于实现前端界面，使用vue-router作为统一路由 |
| Ajax | 用于前后端之间的数据传输 |
| Bucket4j | 用于令牌桶模式的限流配置 |

1. **详细设计**
   1. **系统功能**

本新闻内容管理系统具有以下功能：

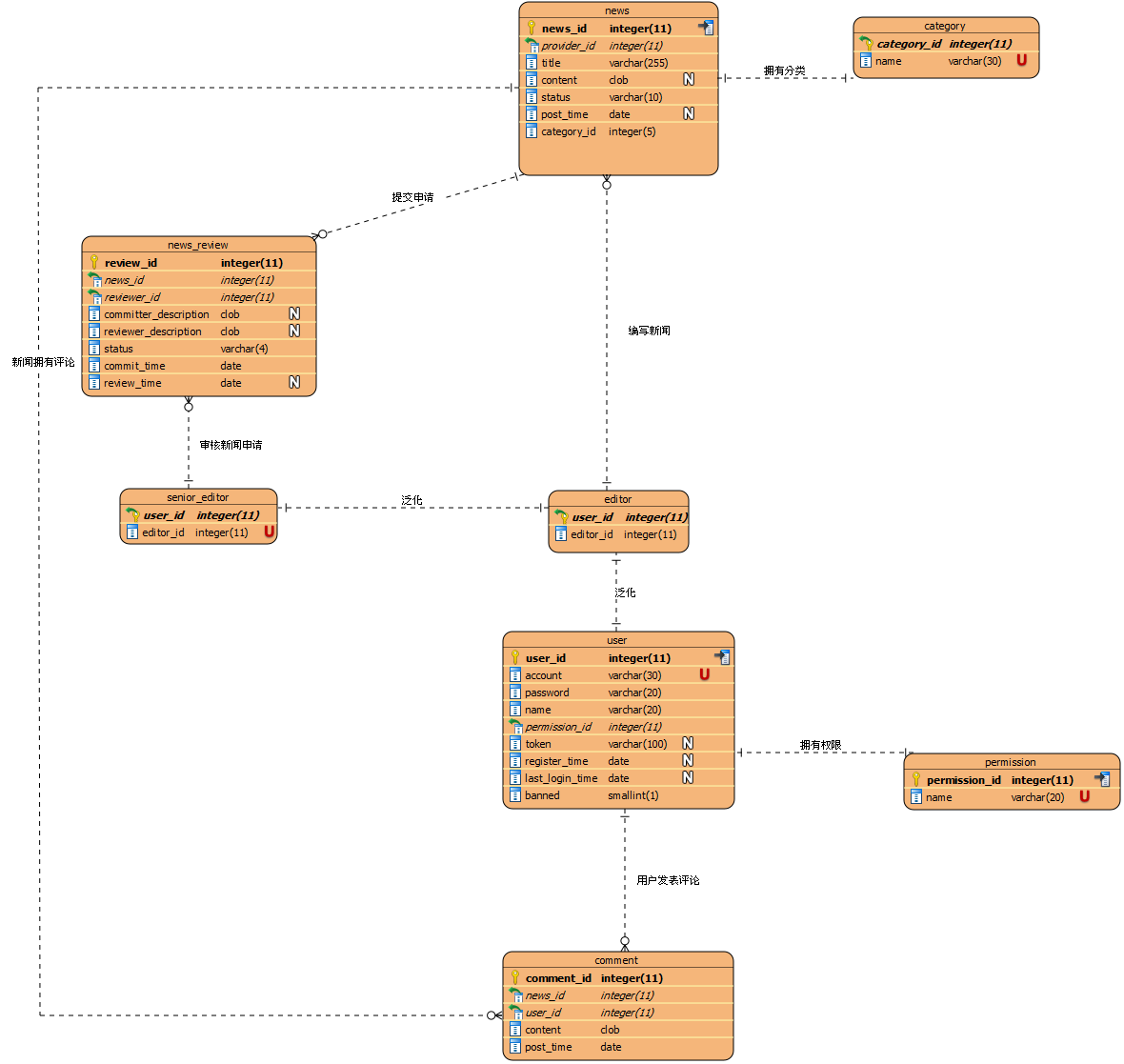
|  |  |
| --- | --- |
| 编号 | 功能描述 |
| 1 | 用户登录注册 |
| 2 | 用户阅览新闻 |
| 3 | 用户搜索新闻 |
| 4 | 用户添加评论信息 |
| 5 | 用户删除评论信息 |
| 6 | 编辑添加新闻 |
| 7 | 编辑提交发布申请 |
| 8 | 编辑管理自己发布的新闻 |
| 9 | 高级编辑审核新闻发布申请 |
| 10 | 高级编辑管理审核列表 |
| 11 | 管理员用户查看用户列表 |
| 12 | 管理员用户查看分类列表 |
| 13 | 管理员用户删除用户账号 |
| 14 | 管理员用户修改分类 |
| 15 | 管理员用户删除分类 |
| 16 | 管理员用户修改用户发言权限 |

具体用例图如下：



* 1. **数据库设计**

本系统主要包括用户，新闻，审核申请，权限和评论这几个实体，ER图如下：



* 1. **Restful API接口规范**

本系统在第一次作业的基础上，将前后端分离，不再使用thymeleaf而是采用restful api接口格式。我将系统中所有数据传输的接口返回格式都更换为统一的Response格式，方便不同前端的实现。

* 1. **JWT用户登录**

本系统通过JWT实现前端用户登录。当用户点击登录时，前端将用户的用户名和密码传输至spring boot后端，后端会通过authenticationManager对用户信息进行验证。验证成功后，后端将生成具有一定有效时长的token，并返回给前端用户。

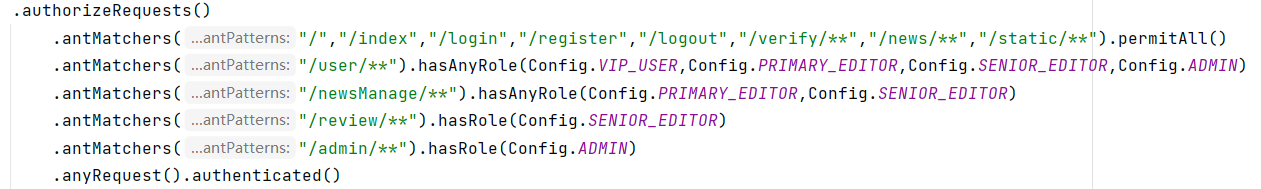


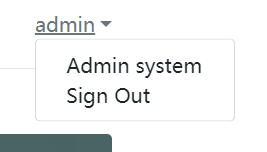
前端用户将token加入请求中，从而在每次请求时验证token是否过期。Token有效期内，用户不需要再次登录，可以直接访问用户权限内的功能。而当用户token过期时，系统会返回response提醒用户再次登录。

* 1. **Spring Security用户权限控制**

本系统包括5个不同的身份权限，普通用户（未登录），订阅用户（ROLE\_USER），普通编辑(ROLE\_EDITOR)，高级编辑(ROLE\_REVIEWER)和管理员用户(ROLE\_ADMIN)。这5个身份权限分别能够访问不同的功能，如订阅用户能够发布评论，编辑能够发布新闻，高级编辑能够审核新闻，管理员用户能够管理用户信息。

具体权限如下：



****

* 1. **Vue实现前端界面**

本系统在前后端分离后，选择vue作为前端实现，并整合了vue-route、bootstrap、vuex等技术。用户登录后，vuex将会把获得的token和用户数据（账号，用户名等）存入localStorage，当用户使用其他功能时，将从localStorage中获取用户数据。如下图就获取了用户名和权限。

****

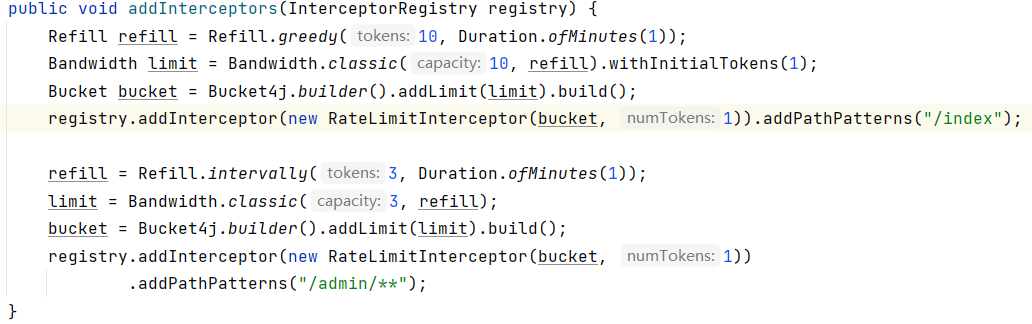
* 1. **Ajax数据传输**

在vue与springboot进行数据传输时，本系统采用ajax请求进行数据传输。此外，当用户登录后，ajax会将用户token直接写入请求的表头中，方便用户在每次请求时验证token的有效性。具体如下：



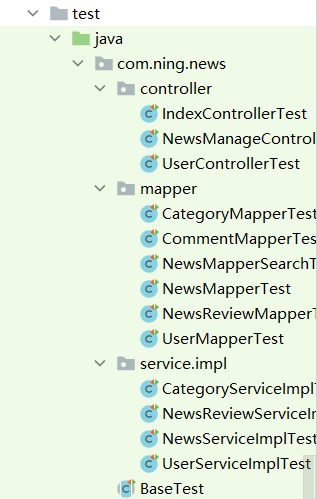
* 1. **Bucket4j实现限流配置**

本系统使用buucket4j对springboot服务器进行限流配置。具体如下：



* 1. **Junit4 实现单元测试**

本系统继续使用Junit4完成了对Mapper，Service和Controller层的单元测试。具体的测试文件结构如下：



1. **运行截图**

如下：

