**E-bookshop项目设计文档**

项目成员

童路勤 18301077

卢君亮 18301075

高梦楠 18301063

目录

[一、基本信息 3](#_Toc72527003)

[1.1开发人员 3](#_Toc72527004)

[1.2开发环境及工具 3](#_Toc72527005)

[二、总体设计 3](#_Toc72527006)

[2.1项目介绍及功能 3](#_Toc72527007)

[2.2界面流程描述 4](#_Toc72527008)

[2.3系统总体逻辑结构图: 5](#_Toc72527009)

[2.4数据库设计: 5](#_Toc72527010)

[2.5 API设计 6](#_Toc72527011)

[2.5.1前端路由设计 6](#_Toc72527012)

[2.5.2 Restful API设计 6](#_Toc72527013)

[三、详细设计 7](#_Toc72527014)

[3.1 Controller 7](#_Toc72527015)

[3.1.1 BookController 7](#_Toc72527016)

[3.1.2 UserController 7](#_Toc72527017)

[3.2 Service 8](#_Toc72527018)

[3.2.1 BookService 8](#_Toc72527019)

[3.2.2 UserService 8](#_Toc72527020)

[3.3 Entity 8](#_Toc72527021)

[3.4 Mapper 8](#_Toc72527022)

[3.5 Security 9](#_Toc72527023)

[3.5.1 Security Config 9](#_Toc72527024)

[3.5.2 JWT 9](#_Toc72527025)

[3.6 Rate Limit 9](#_Toc72527026)

[3.7 OpenAPI 10](#_Toc72527027)

# 一、基本信息

## 1.1开发人员

|  |  |
| --- | --- |
| 人员 | 职责 |
| 童路勤 | Vue前端 |
| 卢君亮 | Restful接口、JWT、流量控制等后端 |
| 高梦楠 | 测试 |

## 1.2开发环境及工具

操作系统：Windows 10

开发语言：Java、Html、CSS、JavaScript等

开发IDE：IntelliJ IDEA

编译环境：Java JDK 1.8

开发框架：Spring Boot、mybatis、Vue、JWT、OpenAPI、Bucket4J等

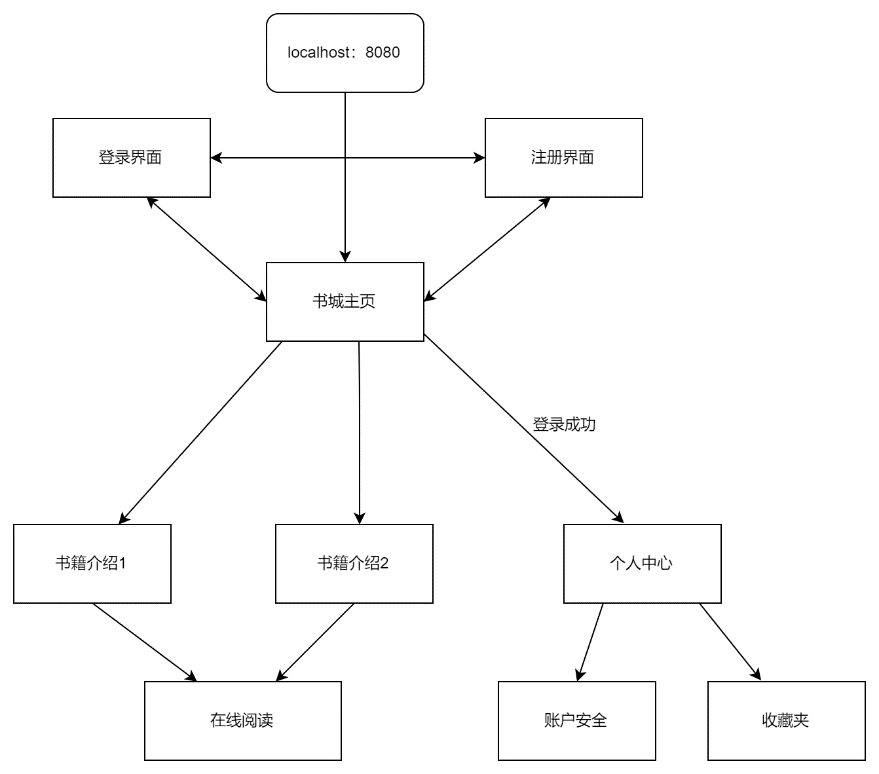
# 二、总体设计

## 2.1项目介绍及功能

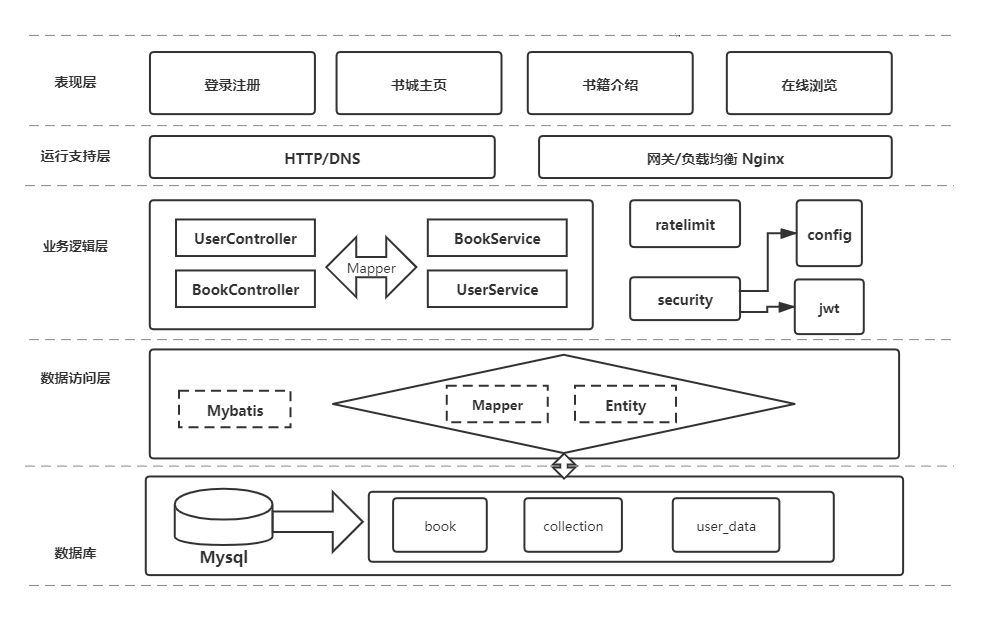
E-bookshop网站的开发主要包括后端数据库的建立，前端网页的设计，以及如何用mybatis进行前后端对接。在后端数据库设计中，我们确保数据一致和完整，而在前端网页中，我们力求做到页面简洁明了，方便操作。本次开发中，以Spring MVC框架+Mybatis+Vue实现功能，同时使用Spring Security实现角色权限管理与登录验证、JWT实现Token验证、OpenAPI生成文档、Bucket4J实现流量控制。

项目主要功能有：登录、注册、查看图书上架信息，在线阅读书籍，添加或删除书籍收藏，修改个人信息等。

## 2.2界面流程描述



## 2.3系统总体逻辑结构图:



## 2.4数据库设计:

|  |  |
| --- | --- |
| 表 | 描述 |
| Book | 电子书信息 |
| Collection | 用户收藏信息 |
| Role\_url | 用户角色信息 |
| User\_data | 用户信息 |

Book

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 描述 |
| Title | Varchar | 书籍名称 |
| Author | Varchar | 作者 |
| mAbstract | Text | 简介 |
| Category | Varchar | 书籍分类 |
| filePath | Varchar | pdf文件路径 |
| Phases | Text | 章节目录 |
| imagePath | Varchar | 封面图片路径 |

Collection

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 描述 |
| Username | Varchar | 用户名 |
| Title | Varchar | 书籍名称 |

User\_data

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 描述 |
| Username | Varchar | 用户名 |
| Password | Varchar | 密码 |
| Phone | Varchar | 手机 |
| Role | Varchar | 角色 |

role\_url

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 描述 |
| Url | Varchar | 允许的url |
| Role | Varchar | 角色 |

## 2.5 API设计

### 2.5.1前端路由设计

项目采用Vue-router hash模式构建的前端路由，路由前缀/#

|  |  |
| --- | --- |
| URL | 描述 |
| / | 主页 |
| /login | 登录注册页 |
| /abstract/{title} | 书籍介绍页 |
| /preview/{title} | 在线阅读页 |
| /personal | 个人信息页 |

### 2.5.2 Restful API设计

项目采用Spring Restful API设计

|  |  |  |
| --- | --- | --- |
| URL | HTTP方法 | 描述 |
| / | GET | 前端页面入口 |
| /login | POST | 登录 |
| /register | POST | 注册 |
| /books | GET | 获取书籍列表 |
| /books/{title} | GET | 获取某本书籍信息 |
| /user/{username} | GET | 获取用户信息 |
| /user/{username} | POST | 修改用户信息 |
| /user/{username}/collections | GET | 获取用户收藏列表 |
| /user/{username}/collections/{title} | GET | 获取用户对某本书的收藏信息 |
| /user/{username}/collections/{title} | DELETE | 删除用户对某本书的收藏信息 |

# 三、详细设计

## 3.1 Controller

### 3.1.1 BookController

public Map getBooks()

获取书籍列表

public Map getBookByTitle(String title)

获取指定书籍信息

public Map getCollectionByUsername(String username)

获取用户收藏信息

public Map getCollection(String username, String title)

获取用户收藏某本书的信息

public Map addCollection(String username, String title)

添加收藏

public Map deleteCollectionString username, String title)

删除收藏

### 3.1.2 UserController

public Map register(Map map)

注册

public Map getInfoByUsername(String username)

获取用户信息

public Map modifyUserPhone(String username, Map map)

修改用户手机信息

## 3.2 Service

### 3.2.1 BookService

调用getBookByTitle（String title）方法获得图书信息，利用findAllbooks返回所有图书信息，调用setCollection方法给用户添加图书到收藏夹中，利用getCollectionByusername方法通过用户名查询用户收藏夹信息，利用deleteCollection方法删除用户收藏信息。

### 3.2.2 UserService

loadUserByUsername方法通过传入用户名查询有无该用户，有则返回UserInfo对象，无则返回null。定义changePhone方法方便修改用户信息。

## 3.3 Entity

包括Book，Collection，UserInfo三个实体类。里面定义了各类名称和构造器。

## 3.4 Mapper

包括BookMapper、CollectionMapper、SecurityMapper、UserInfoMapper四个接口。分别负责向数据库添加，返回书信息，添加/删除图书到收藏夹，通过url获取能访问该url权限，通过用户名寻找密码和手机号，向数据库中添加/修改用户信息等。

## 3.5 Security

### 3.5.1 Security Config

SecurityConfigure:SpringSecurity相关配置文件，定义未经登录就可访问的界面、登录页面、登录成功界面以及登出界面，加入自己实现的过滤器过滤非法的Token，并通过自定义的successHandler与failureHandler实现登录成功和登陆失败的逻辑，若登录成功则生成一个新的token并返回。

### 3.5.2 JWT

先实现JwtUtils工具类，实现token的生成、检验和通过token获取用户名的方法，再通过继承OnceperRequestFilter实现自己的token过滤器，通过获取请求头中的Authorization字段的token，调用工具类，通过后生成一个新的UsernamePasswordAuthenticationToken对象并将其加入到SecurityContext中，即该用户通过验证。通过继承 AuthenticationEntryPoint实现token失败的逻辑，即返回错误信息。

## 3.6 Rate Limit

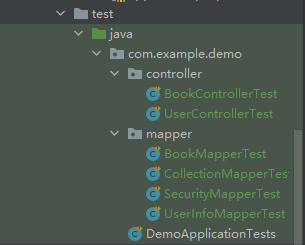
使用Bucket4J框架，继承WebMvcConfigurer和HandlerInterceptor对/user/\*\*路由实现流量控制。

## 3.7 OpenAPI

在每个Controller前添加了@Operation、@Tag等注解，得到OpenAPI文档。OpenAPI是一种基于Resful风格对API进行格式化描述的一种规范，包括：端点描述，操作的参数（输入/输出），认证信息，联系信息等。@RestController相当于@ResponseBody和@Controller的结合，通过将@ResponseBody注解在Controller类上，可以使类中所有的方法序列化。而@Controller用来表示表示Spring中的类是否可以接受Http请求，常与@ResponseBody绑定使用。@Operation是swagger项目中的注解，用过swagger-spring项目实现了与SpringMVC框架的无缝集成功能。@RequestMapping是用来标识http请求地址与Controller类方法之间的映射。

**3.8 Junit Test**

在/test/com.example.demo目录下，包含两个文件controller和mapper，分别对controller类和mapper类进行测试。具体结构如下：



**测试的部分截图如下：**

