**E-bookshop项目开发文档**

## 

项目成员：

王昱

胡锦浩

姚聪

目录

[1](#_Toc72513872)

[1. 开发规划 3](#_Toc72513873)

[1.1人员分工 3](#_Toc72513874)

[1.2开发环境及工具 3](#_Toc72513875)

[2.总体设计 3](#_Toc72513876)

[2.1项目介绍: 3](#_Toc72513877)

[2.2功能结构图 4](#_Toc72513878)

[2.3用例图 4](#_Toc72513879)

[2.4数据库设计: 5](#_Toc72513880)

[3.详细设计 6](#_Toc72513881)

[3.1 Controller设计 6](#_Toc72513882)

[3.1.1 BookController 6](#_Toc72513883)

[3.1.2 AuthoriseController 6](#_Toc72513884)

[3.2 entity设计 7](#_Toc72513885)

[3.2.1 Book 7](#_Toc72513886)

[3.2.2 UserInfo 7](#_Toc72513887)

[3.2.3 Role 7](#_Toc72513888)

[3.3 Repository 7](#_Toc72513889)

[3.3.1 BooksRepository 7](#_Toc72513890)

[3.3.2 UserInfoRepository 7](#_Toc72513891)

[3.3.3 RoleRepository 7](#_Toc72513892)

[3.4 service 8](#_Toc72513893)

[3.4.1 bookservice 8](#_Toc72513894)

[3.4.2 UserInfoService 8](#_Toc72513895)

[3.4.3 RoleService 8](#_Toc72513896)

[3.5 Config 8](#_Toc72513897)

[3.5.1 JWTConfig 8](#_Toc72513898)

[3.5.2 WebConfig 8](#_Toc72513899)

[3.5.3 JWTFilter 8](#_Toc72513900)

[3.6 Testing 9](#_Toc72513901)

[3.6.1 单元测试 9](#_Toc72513902)

[3.6.2 集成测试 9](#_Toc72513903)

[3.7前端 10](#_Toc72513904)

[3.7.1实现步骤 10](#_Toc72513905)

[3.7.2界面 10](#_Toc72513906)

[4.swagger接口文档 13](#_Toc72513907)

# 1. 开发规划

## 1.1人员分工

|  |  |  |
| --- | --- | --- |
| 角色 | 分工 | 人员 |
| 项目经理 | 项目主体设计、前端界面编写、文档撰写、swagger使用 | 王昱 |
| 开发人员 | 测试模块、数据库连接、文档撰写 | 胡锦浩 |
| 开发人员 | JWT登录验证、角色权限识别、文档撰写 | 姚聪 |

## 1.2开发环境及工具

开发语言：java

开发软件：idea，webstorm

开发框架：Spring Boot，Vue

# 2.总体设计

## 2.1项目介绍:

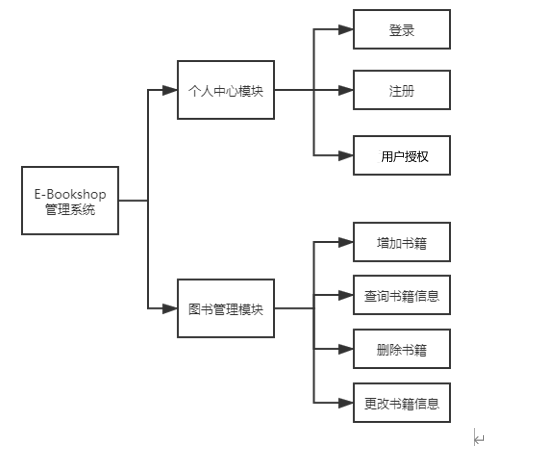
E-bookshop网站的开发，通过Spring Boot + Vue的方式前后端分离实现功能。主要功能有：登录注册，用户授权，图书信息的增删改查等功能，并使用Spring Security + JWT实现角色权限管理与登录验证，controller使用Restful风格。同时提供openAPI，可通过resful形式直接对后台数据进行操作。（Note：管理员授予权限功能未提供界面，只提供了API接口。）

前端接口：localhost:8080

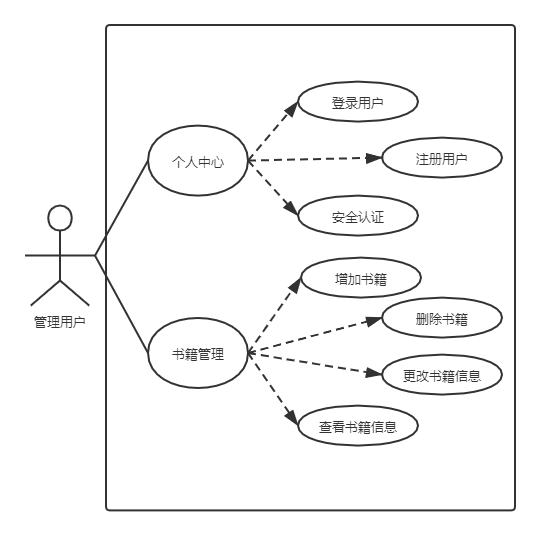
后端接口：localhost:8888

后端接口文档（swagger）： localhost:8888/doc.html，另附”接口文档.md”文件

## 2.2功能结构图

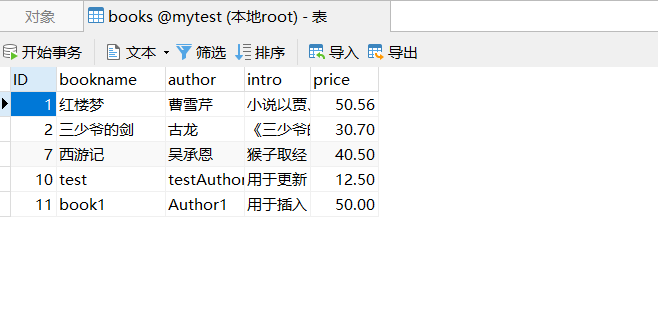


## 2.3用例图

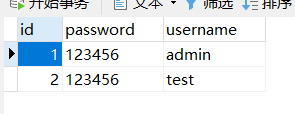


## 2.4数据库设计:

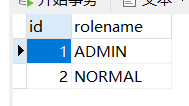
Book表



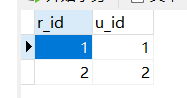
Users表



Role表



User\_role表



# 3.详细设计

## 3.1 Controller设计

## 3.1.1 BookController

使用RestFul风格重写上次作业的controller，实现书籍管理部分接口设计，包括查看书籍列表、对书籍进行增删改等（只有ADMIN用户可进行书籍管理操作）。

1. 返回书籍列表

@GetMapping(value = "/books")

public List<Book> getBooks()

1. 返回指定书籍

@GetMapping("/books/{id}")

public Book getBook(@PathVariable("id") Integer id)

1. 增加书籍

@PostMapping(value = "/books")

public void addBook(@RequestBody Book book)

1. 修改指定书籍

@PutMapping("/books")

public void modifyBook(@RequestBody Book book)

1. 删除指定书籍

@DeleteMapping("/books/{id}")

public void deleteBook(@PathVariable("id") Integer id)

1. 根据书名模糊查询

@GetMapping("/books/searchResult/{searchKey}")

public List<Book> searchBooks(@PathVariable("searchKey") String searchKey)

## 3.1.2 AuthoriseController

使用RestFul风格重写上次作业的controller，实现管理员（即拥有ADMIN角色授权）对注册用户的授权工作，包括获得所有用户授权信息和对指定用户授权。

1. 返回所有用户信息

@GetMapping("/users")

public List<Object> getUsers()

1. 对指定用户授权

@PutMapping("/users/assignment")

public List<Object> assignUsers(@RequestBody Map<String,Object> request)

## 3.2 entity设计

### 3.2.1 Book

Book类属性定义，包括（ID,bookname，author，address，price），同时实现定义类的构造器和get、set方法

### 3.2.2 UserInfo

UserInfo类属性定义，包括（id, username, password），同时实现定义类的get，set放方法。

同时Userinfo和Role通过JPA实现多对多联系，可通过user对象获得对应role角色列表。

### 3.2.3 Role

Role类属性定义，包括（ID,rolename），同时实现定义类的构造器和get、set方法

## 3.3 Repository

### 3.3.1 BooksRepository

继承了JpaRepository，自动实现了有关book表的相关Dao层操作。

### 3.3.2 UserInfoRepository

继承了JpaRepository，自动实现了有关userInfo表的相关Dao层操作。

### 3.3.3 RoleRepository

继承了JpaRepository，自动实现了有关role表的相关Dao层操作。

## 3.4 service

### 3.4.1 bookservice

设计IBookService接口，定义相应的数据库操作。BookServic类实现IBookServic接口，通过JPA实现对应的数据库操作。

### 3.4.2 UserInfoService

设计IUserInfo接口，实现了UserDetailsService的接口，定义实现了响应数据库操作，并自定义了一个findByUsername的方法，来支持查找相应用户名对应的UserInfo类。

### 3.4.3 RoleService

设计IRolerInfo接口，实现了RoleService的接口，定义实现了响应数据库操作。

## 3.5 Config

### 3.5.1 JWTConfig

JWTUtils 类：配置jwt的相关函数，包括：Jwt的生成，校验以及从Jwt中获取用户信息。

### 3.5.2 WebConfig

SecurityConfig 类：继承了WebSecurityCinfigerAdapter类，配置http的相关需求，以及跨域的相关配置。

### 3.5.3 JWTFilter

JWTAuthenticationEntryPoint 类： 实现了AuthenticationEntryPoint 接口，处理了在没有登陆时的相关信息。

JWTAuthenticationFilter 类：继承了 UsernamePasswordAuthenticationFilter 类，从输入流中获取登录信息并进行验证，成功时给返回的response添加JWT令牌，失败时给response添加失败的相关信息。

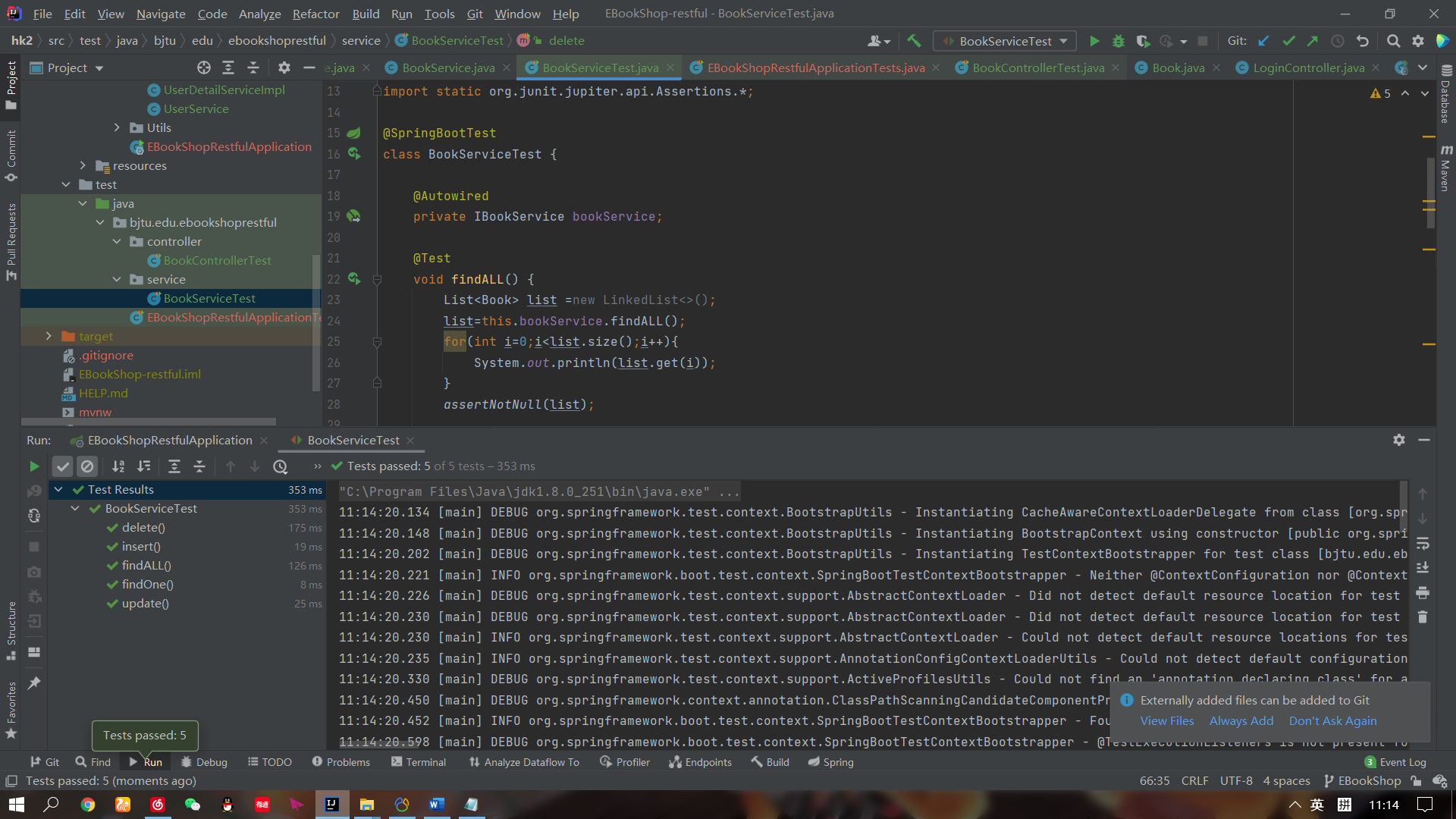
JWTAuthorizationFilter 类 ：继承了BasicAuthenticationFilter 类，设置拦截器，对前端吃的请求信息进行拦截，判断是否有Authorization信息，无则放行，有则获取用户信息。

## 3.6 Testing

       本项目基于Spring框架提供的测试模块spring-test，针对Service层进行单元测试，针对Controller层进行集成测试

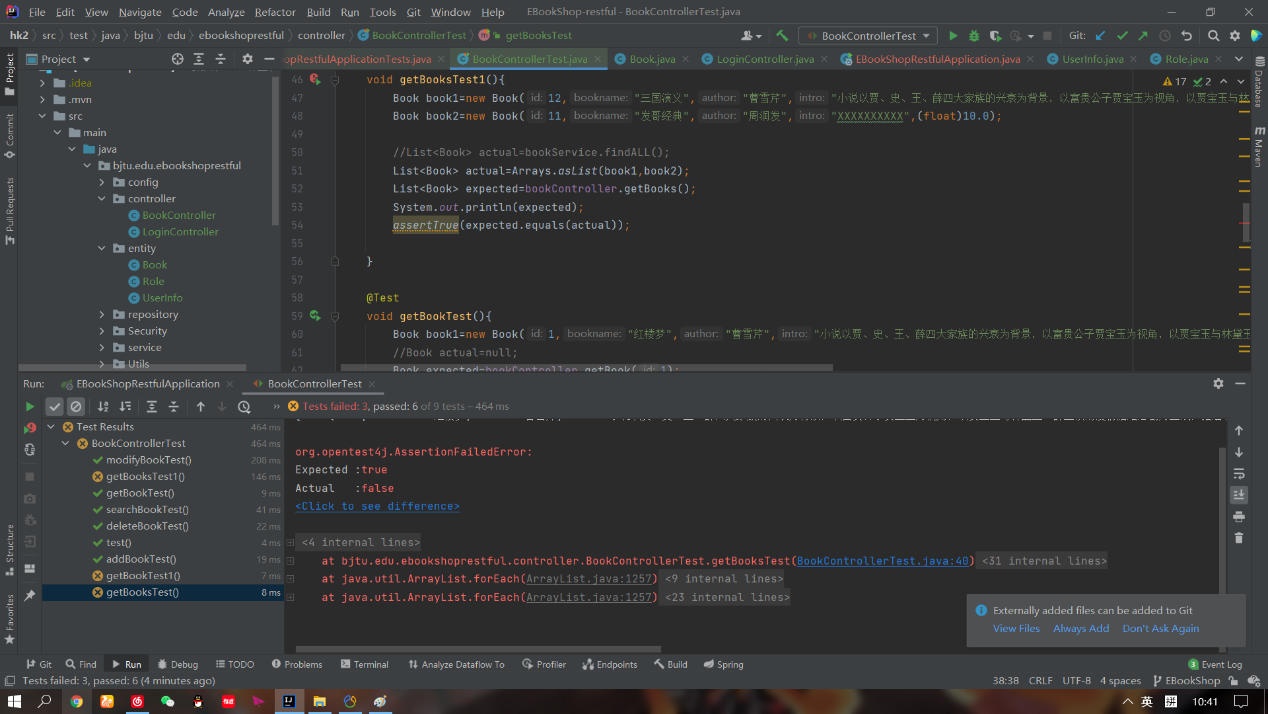
### 3.6.1 单元测试

       本项目针对Service层进行单元测试，实现了BookServiceTest类，分别初始化实例并实现了对数据库增删改查功能的测试，共进行了5个测试，通过5个，未通过0个，测试通过



### 3.6.2 集成测试

       本项目针对Controller层进行集成测试，分别实现了针对BooksController,的测试类，对项目的增删改查功能进行测试，通过设置不同的边界值，共进行了9个测试，通过6个，未通过3个，测试通过



## 3.7前端

### 3.7.1实现步骤

1.使用上次的HTML模板

2.通过vue-cli创建前端工程项目

3.改造模板为vue形式，同时提出共同部分和特殊部分形成组件

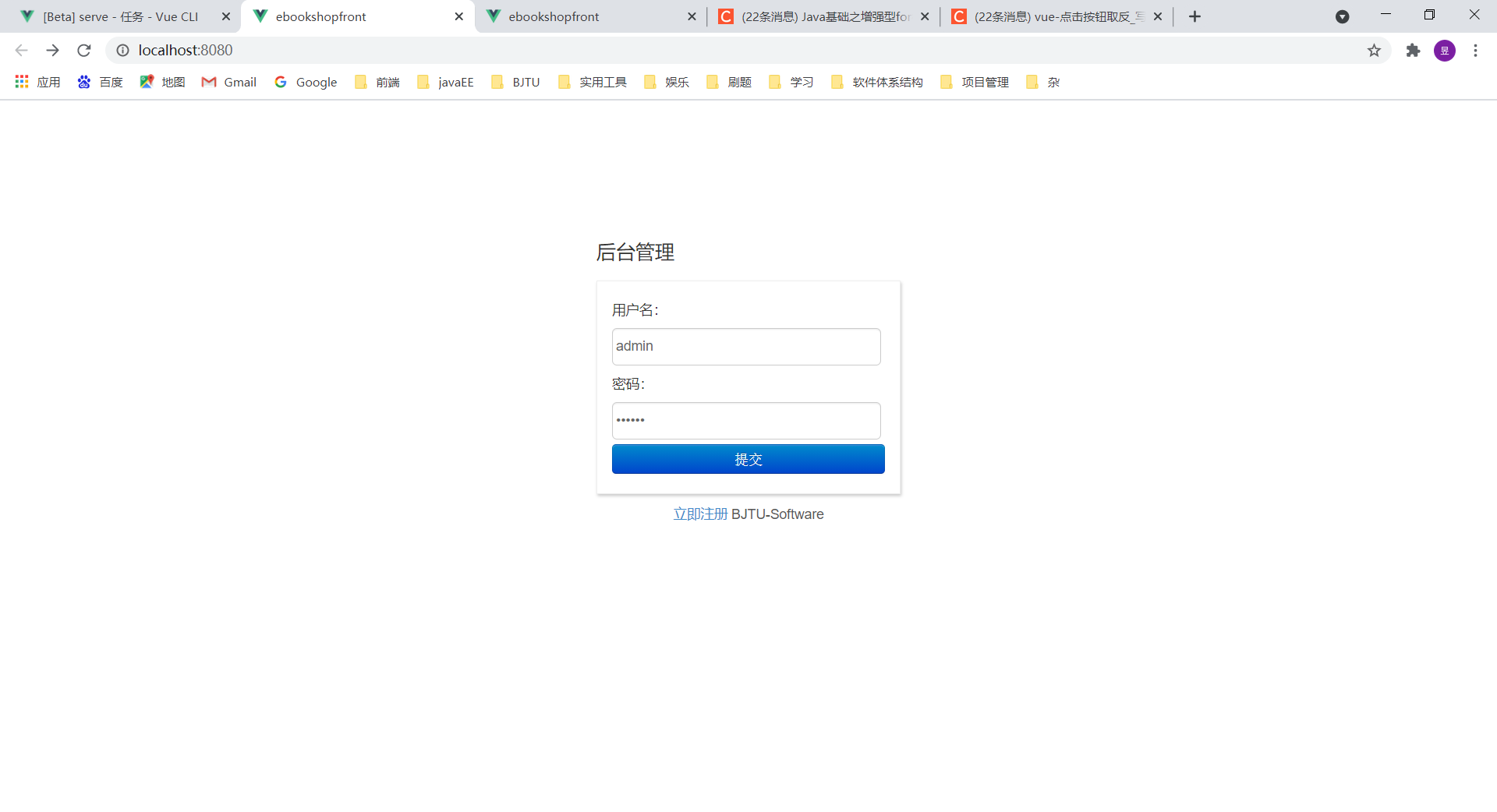
4.通过vue-router实现前端导航

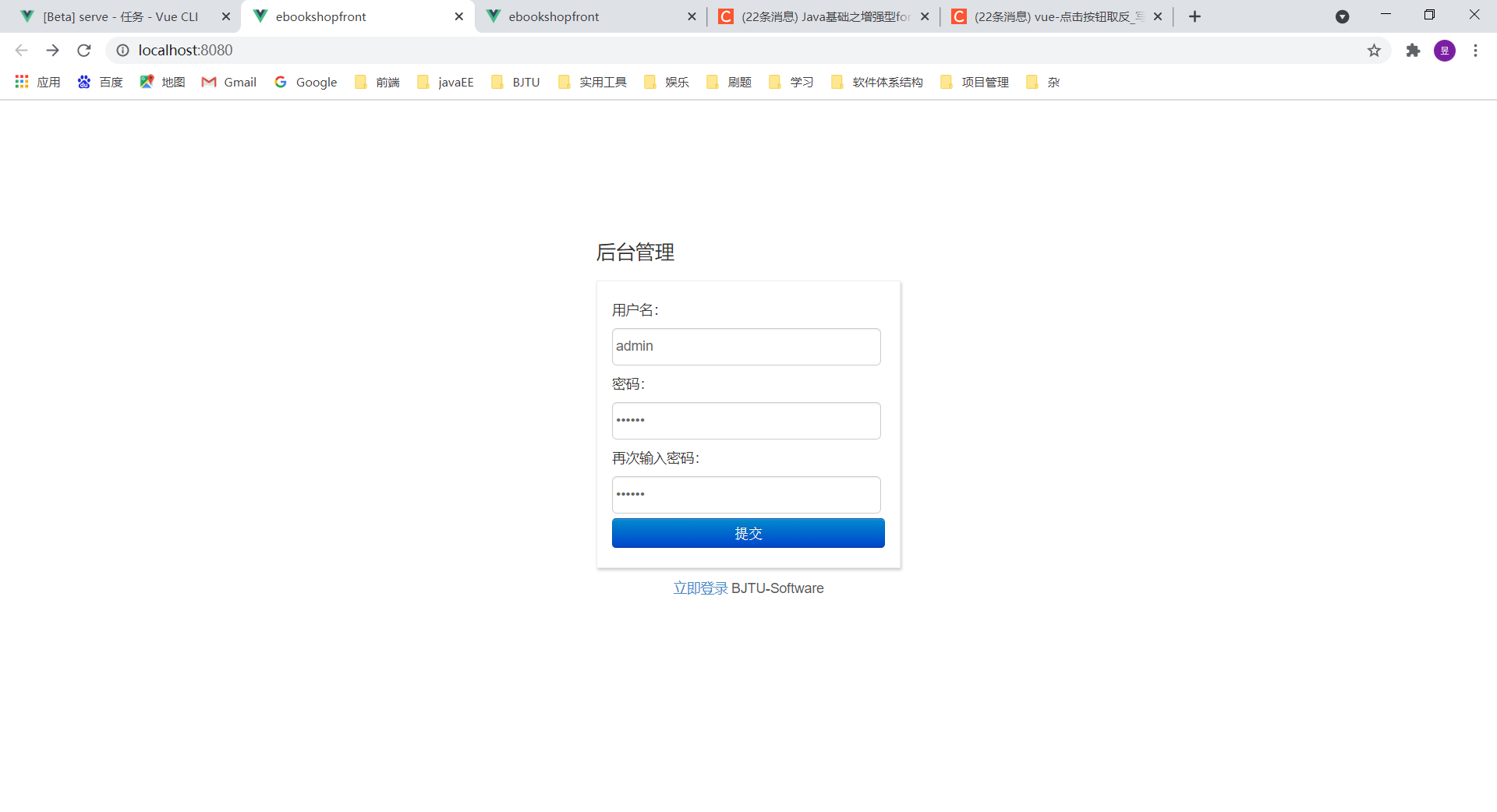
5.通过axios实现异步发送http请求

### 3.7.2界面

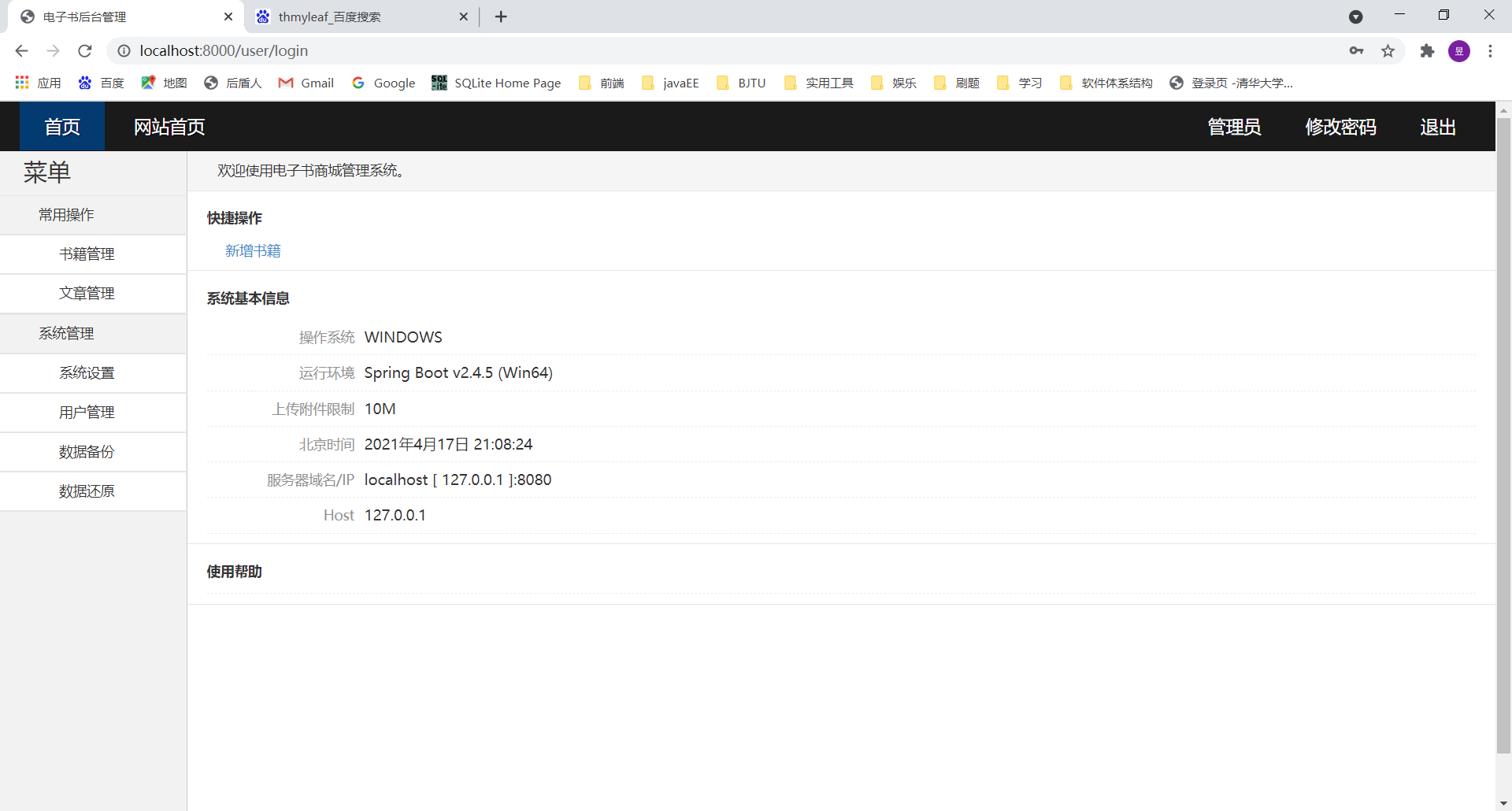
Note：管理员授予权限未提供界面，只提供了API接口。

1.登录注册界面（login.vue）

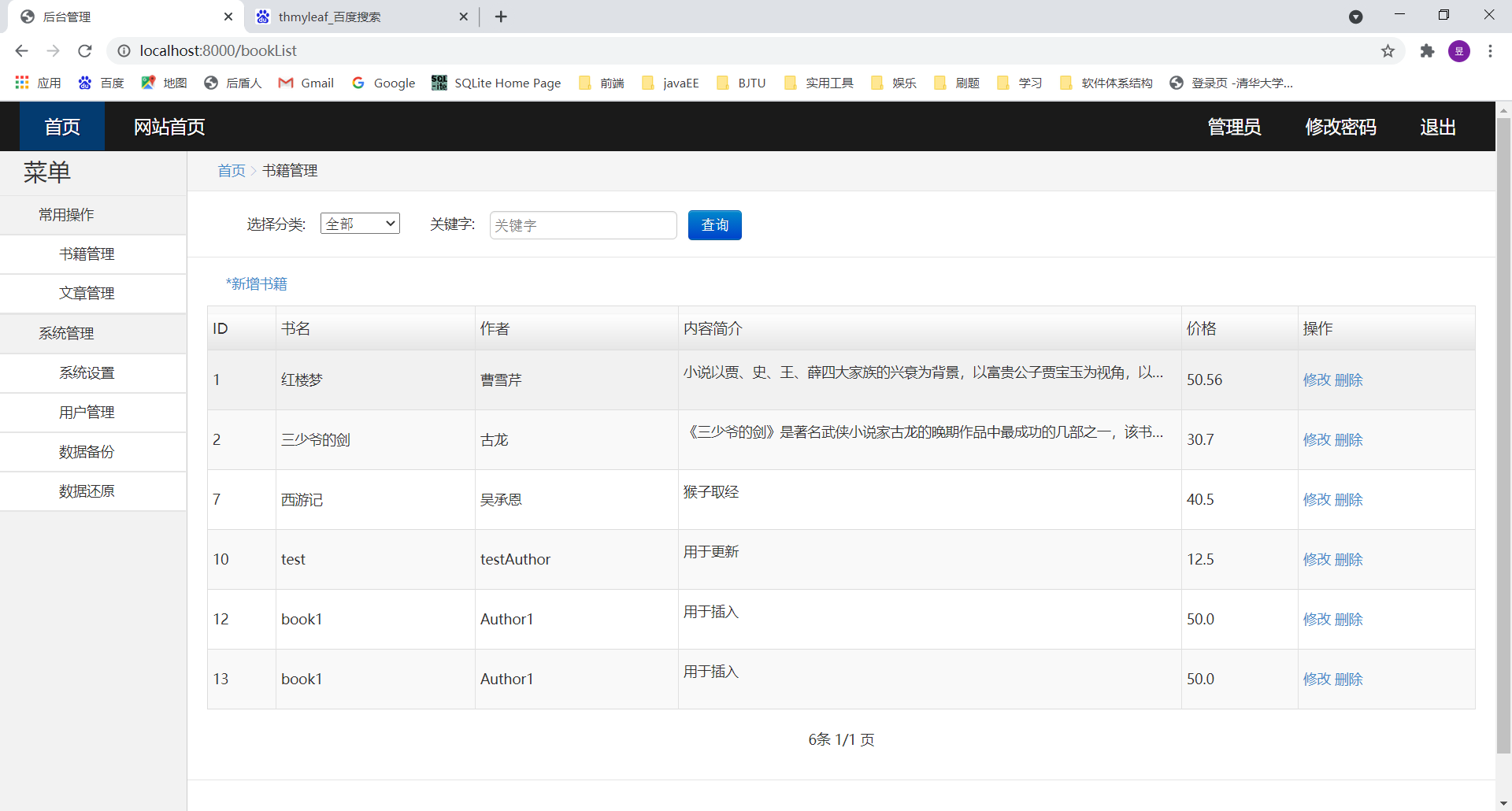




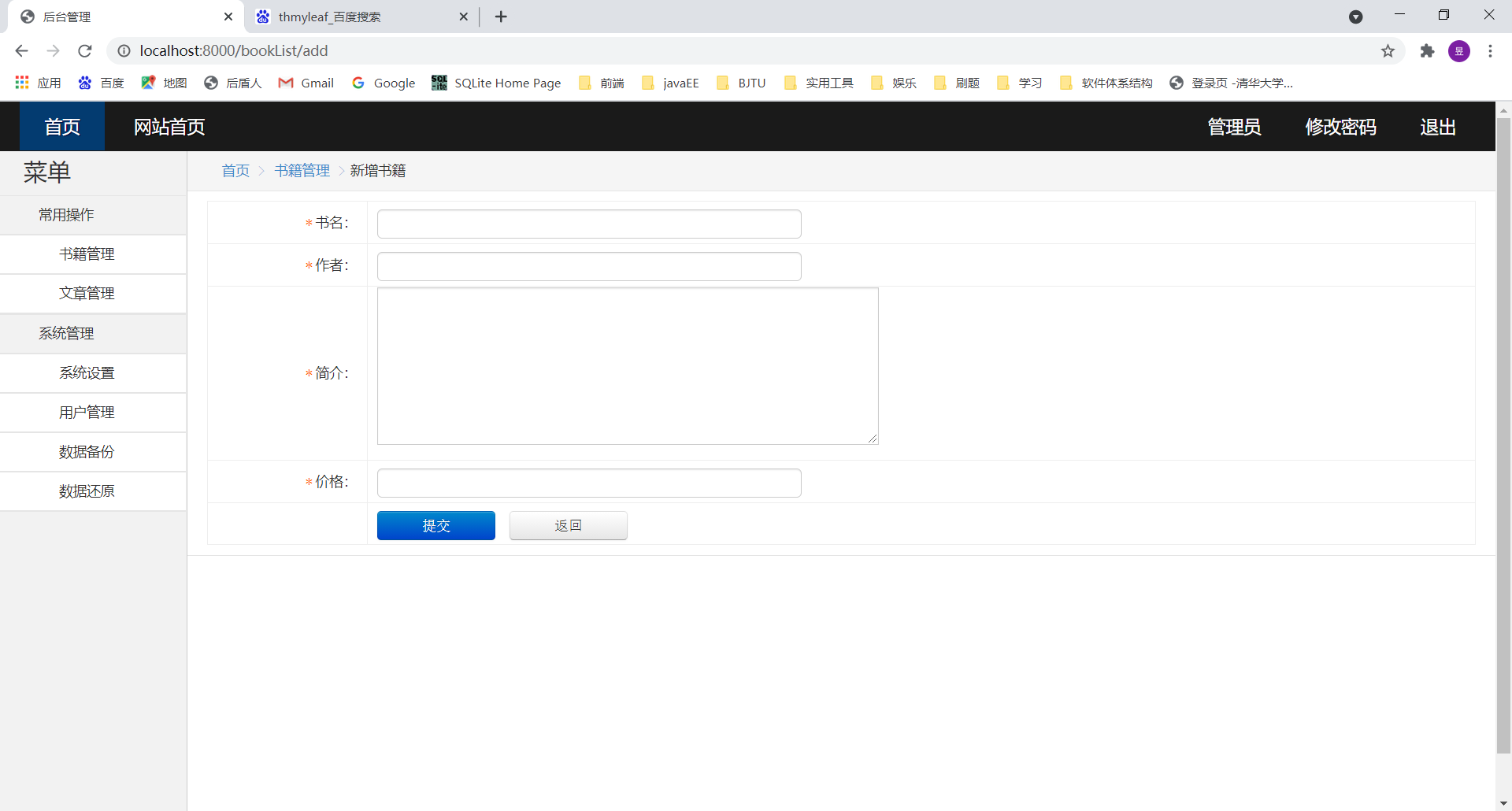
2.主界面(index.vue)



3.书籍列表界面(bookList.vue)



4.增加、修改书籍界面(insert.vue,modify.vue)



# 4.swagger接口文档

通过引入swagger依赖，实现接口文档输出。

文档地址：localhost:8888/doc.html

截图：

