**J I A N G S U U N I V E R S I T Y**



**本 科 毕 业 论 文**

在线文库管理系统的设计与实现

Design and Implementation of Online Library Management System

学院名称： 计算机科学与通信工程

专业班级： 计算机科学与技术1401

学生姓名： 陆健 健

指导教师姓名： 邢玉萍 萍

指导教师职称： 讲 师 师

2018年5月

在线文库管理系统的设计和实现

专业班级：计算机科学与技术1401 学生姓名：陆健

指导教师：邢玉萍 职称：讲师

摘要 现如今已经进入到了互联网高速发展的时期，自从互联网+的观点的提出，各类互联网平台更是层见叠出，比如互联网+购物诞生了淘宝、京东等一众电商平台，互联网＋饮食诞生了美团、饿了么等外卖平台，互联网+出行诞生了共享单车、滴滴打车等等。这些互联网平台不但利便了人们的衣食住行，提供了工作岗位，更是丰富了我们每一个人的业余生活。随着互联网思维的渗透人心，各行各业都将会通过互联网打造一个平台，无论是传统行业，还是新兴行业，而互联网也会更大的发挥它的作用。

在互联网的时代，毫无疑问网络已经成为我们获取信息的首要媒介之一，比如程序员会经常去浏览掘金、github这些技术社区；一般人也会去没事刷刷知乎、头条，从而了解时事，获得娱乐。本文要开发的系统，是一个基于web交互式的文档分享平台，在这里，用户可以在线查看和下载文档，不过有可能会需要扣除用户的积分。系统所积累的文档主要来自会员的上传，系统本身是不去修改用户上传的文档内容的，用户通过上传文档可以获取网站虚拟的积分，用于下载自己需要的文档，从而实现了资源的共享，可以避免没有权限的用户随意修改和复制源文档，于是保护了相关的版权。

关键词：文库管理 资源共享 Vue SpringBoot

Design and Implementation of online library management system

**Abstract** Now has entered the Internet high-speed development period, since the Internet + point of view, all kinds of Internet platform has appeared, such as the Internet + shopping taobao, jingdong was born the electric business platform, the Internet + diet was born beauty, hungry? Take-out platform, such as Internet + travel was born Shared cycling, drops a taxi, and so on. These Internet platforms not only make it easier for people to live, eat, live and work, but also enrich every one of us in our spare time. As Internet penetration of the heart of thinking, in all walks of life will be through the Internet to build a platform, whether it is a traditional industry, and emerging industries, the Internet will be more play its role.

In the age of the Internet, there is no doubt that the Internet has become one of the primary media for us to get information. Ordinary people will also go to nothing to brush the knowledge, headlines, so as to understand current affairs, entertainment. The system being developed in this article is a web-based interactive document sharing platform where users can view and download documents online, but may need to deduct user credits. System accumulated documents mainly comes from members of the upload, the system itself is not to modify the user upload document content, users, through the integration of the upload documents to get the web virtual document for download their needs, so as to realize the resource sharing, can avoid no permissions users to modify and copy the source document, so to protect the copyright.

**Key words** Document management Resource Sharing Vue SpringBoot

目 录

[第一章 绪论 1](#_Toc30382)

[1.1课题研究背景及意义 1](#_Toc30403)

[1.1.1研究背景 1](#_Toc23732)

[1.1.2研究意义 1](#_Toc6400)

[1.2国内外研究现状 2](#_Toc30641)

[1.3课题研究内容 3](#_Toc18193)

[1.4系统可行性分析 3](#_Toc25605)

[1.5论文组织框架 4](#_Toc17314)

[第二章 相关技术与理论介绍 6](#_Toc14925)

[2.1前后端分离技术与实现 6](#_Toc10994)

[2.1.1理解MVC模式与MVVM模式 6](#_Toc23427)

[2.1.2 了解Restful架构 7](#_Toc256)

[2.2基于token的身份验证 8](#_Toc28393)

[2.3了解WebSocket 10](#_Toc15935)

[2.4实验环境与开发工具介绍 10](#_Toc10360)

[2.5本章小结 11](#_Toc14032)

[第三章 系统总体设计 12](#_Toc25519)

[3.1在线文库管理系统体系结构设计 12](#_Toc22668)

[3.2系统功能模块设计 13](#_Toc15862)

[3.3 数据库设计 15](#_Toc24168)

[3.3.1 概念结构设计 15](#_Toc29934)

[3.3.2 逻辑结构设计 17](#_Toc19453)

[3.4本章小结 19](#_Toc12385)

[第四章 系统实现与测试 20](#_Toc23089)

[4.1运行环境与技术支持 20](#_Toc23740)

[4.1.1 系统运行环境 20](#_Toc1627)

[4.1.2 技术支持 20](#_Toc8749)

[4.2系统的实现 20](#_Toc16216)

[4.2.1使用WebSocket实现消息的推送 20](#_Toc9695)

[4.2.2用户上传模块的实现 22](#_Toc18534)

[4.2.3用户下载模块的实现 23](#_Toc28137)

[4.2.4用户评论模块的实现 24](#_Toc26566)

[4.2.5用户收藏模块的实现 25](#_Toc17902)

[4.2.6权限验证模块的实现 25](#_Toc26477)

[4.3系统的运行 26](#_Toc12669)

[4.3.1首页设计 26](#_Toc24659)

[4.3.2注册与登录界面设计 26](#_Toc8883)

[4.3.3 文档筛选页面 27](#_Toc19741)

[4.3.5用户中心 28](#_Toc7856)

[4.4后台功能实现 28](#_Toc1846)

[4.4.1 登录模块实现 28](#_Toc1748)

[4.4.2 后台管理模块 29](#_Toc10647)

[4.5本章小结 31](#_Toc13549)

[第五章 总结与展望 32](#_Toc28352)

[5.1总结 32](#_Toc18866)

[5.2展望 33](#_Toc3013)

[致 谢 34](#_Toc18502)

[参考文献 35](#_Toc22603)

# 第一章 绪论

## 1.1课题研究背景及意义

### 1.1.1研究背景

随着互联网的发展越来越普及，网络显然已经成为我们获取各种信息以及资讯的主要途径之一，而文档是这些资源的最广泛的表现形式。那么如何快速的获取到我们想要的资源呢？众包模式作为如今比较成熟的商业模式，特别是在互联网行业中，比如快手、知乎等等这些用户生成内容（UGC）网站，这些网站的核心价值几乎完全来自用户进行的价值创造，并且不产生任何直接成本。这种模式的成立是在人们是把这种创造当做一种文娱方式的基础上建立的，并享受因此带来的他人认同感，以此实现自我价值。比如大家在知乎上分享一些有趣的话题，邀请别人回答一些问题或者是写一些技术文章等。另外号称最大的同性交友网站GitHub也是众包模式的一个体现，作为第三方托管平台，资源全部来自于外部用户。

所以对于本系统，众包模式也会是一个非常好的方法，所有的文档资源都是来自于用户的上传，系统本身并不会去编辑修改它们，保证文件的纯洁性。系统主要的任务就是管理这些文档，提供用户分类浏览、快速搜索的功能。另外采用积分制的形式去鼓励用户上传文档，分享资源，活跃社区气氛。

### 1.1.2研究意义

无论是工作和学习中，还是在日常生活中，网络信息的获取对人们来说都是非常重要的，作用也日益明显。当我们需要在网上查找资料的时候，大多都还是需要自己整理，很难找到现成的，直接就可以用的。所以就需要一个文档分享的平台，用户将平常自己整理过的一些资料、遇到的一些问题的解决方法整理成文档，或者常见的工具包等资源分享至平台上，由平台统一管理这些资源。如果用户需要一些资料，只需搜索关键词，或者按类别寻找，就能够找到自己想要的资源，实现用户在短时间内，高效准确的获取想要的文档，解决自己的问题，避免了时间和精力的耗费。

当然，在对资源的管理过程中，还是有很多问题需要去解决的，比如文档的审查，涉及到色情、暴力的文档是不能通过，不可上传的。还要避免用户的恶意上传与下载，对于同一文档只能上传一次。很多法律上的问题也是需要去考虑的，但是实现这样一个平台的价值还是很大的。

## 1.2国内外研究现状

对于文档的管理，国外的技术比较成熟，理念也更加先进，不得不承认是领先于我国的。在国外的一些大公司中，早已采用分布式的管理模式，有着跨平台型、分布式存储和可扩展性的特点。当然最近几年我国的互联网也是发生了翻天覆地的变化，互联网巨头的科技水平也是可以跻身国际先进行列的。

在国内，大家会经常在百度文库、道客巴巴、豆丁网和CSDN上下载文档。百度文库、道客巴巴、豆丁网中资源的范围更大，包含各个行业，而且更注重的是可以看见的文字或图片；CSDN是中国最大的IT和社区服务平台，文档也大都是关于IT技术的，文档的类型更多，而且下载之前是看不到文档的内容的，但是可以根据文档的评分以及用户的评价来决定是否下载，如果发现文档不符合相应要求，还可以进行举报。

在国外，关于文档共享平台也有很多，比如谷歌Drive，它是由谷歌公司创建的来同步和存储文件的服务，允许用户将文件在云中存储、同步和共享文档；iCloud是苹果公司针对自家系统iOS推出的一个云存储和云计算服务，用户可以从iCloud中获取照片、音乐和视频以及购买的任何电影。iCloud允许用户在远程服务器上存储文档、照片和音乐等数据，这样他们就可以下载到他们的设备上，并与其他用户共享。Dropbox是Dropbox公司由美国公司运营的文件托管服务，它可以提供云存储、支持文件同步，有个人和客户端软件，可以存储照片、文档、视频等文件。

从文档分享系统的功能上来说，快捷有效的检索、实时高效的存储和便利的操作是文档分享系统的主要功能，系统架构也从最初的C/S架构升级到基于Web下的分布式架构[1]。

## 1.3课题研究内容

本课题的主要工作是设计和实现一个基于web的文库管理系统，是在线互动式分享文档的平台。在这里，用户可以在线浏览和下载文档，但是下载是需要扣除用户的积分的，网站所积累的文档来自管理员的添加和用户的上传，网站本身不编辑或修改用户上传的文档内容，但是如果发现用户上传的文档不符合要求，会及时删除[2]。用户通过上传文档来获取系统虚拟的积分，用于下载自己需要的文档，从而实现了资源的有机共享，能保证没有权限的用户不能修改和复制源文档，保障了文档的安全性和相关的版权，并有效地管理了文档信息。

本课题拟开发一个在线文库的管理系统，完成以下功能：

（1）分析使用系统的用户。实现用户注册、登录操作，用户只有在登录成功后才能进行相关的上传下载操作。

（2）文档类别管理：在用户上传文档时，会让用户给自己的文档分类，方便文档的分类浏览，分类存放。

（3）用户上传文档：会员将自己喜欢的文档上传到系统，经管理员审核通过可以增加相应的积分，如果上传的文档不符合要求，会被删除。

（4）用户下载文档：用户将自己喜欢的文档下载到到自己的计算机上，下载成功后会扣除相应的积分，并给提供文档的用户增加相应积分，如果用户的积分不足将不能下载文档。

（5）文档评价：用户对自己下载的文档可以进行评价与打分，不过不可评价自己的上传的文档，未下载也不可评论。

（6）文档收藏：用户可以随时收藏自己喜欢的文档，等到自己的积分够了或者有了下载的需求再进行下载。

## 1.4系统可行性分析

在线文库管理系统在开发过程中所用到的开发工具，如VSCode、MySQL和GitHub等工具都是免费的，IDEA用的也是社区版的（免费），虽然功能少了点，但是通过插件的方式也能满足要求。

技术可行性：经过大学四年计算机科学技术专业的学科培养，我深刻地体会了面向对象思想的种种优势，曾经修习并通过了面向对象程序设计、数据库原理和数据结构等专业课程，因此我对面向对象程序设计的架构思想、数据库的操作和SQL语言和数据结构及相关算法都有一些了解，同时大大增强了我的学习能力，使我接受一门新的编程语言或者技术不是太难。并且我曾经在一家公司有过暑期的实习经历，对前端的一些知识如HTML、JavaScript、CSS和JQuery有着一定程度的掌握，同时用Java语言独立开发过一个小型的项目，接受新的前端框架Vue2.0和基于Spring的Java后台框架SpringBoot也是可以的。因此，基于以上的学习和实习经历，实现在线文库管理系统在技术上是可行的。

法律可行性：此系统开发过程中所使用到的工具或框架，皆是Community版或者开源的，在不违背其开源协议的前提下，使用这些工具没有任何法律方面的问题。在线文库管理系统由本人独立完整开发，不会对任何人、公司、社区造成知识产权侵犯的问题。另外此系统不作任何商业用途，因此，在线文库管理系统的设计与实现，在法律上也是完全可行的。

操作可行性：在线文库管理系统基于B/S架构，用户只需要操作系统上一个浏览器软件即可访问和使用此系统，并且不用考虑系统版本更新的问题。此系统提供了界面简洁优雅的操作界面，功能高度集成，数据以列表方式格式化显示，合理地利用了屏幕的空间。通过使用Ajax技术和后台异步交互，所有的操作都可以在同一个页面完成，减少了页面的跳转所需要等待的时间，大大降低了对后台资源的消耗。同时，在线文库管理系统还提供了众多的快捷操作和相关提示，使用此系统的用户不需要掌握任何的计算机专业方面的知识，即可熟练使用此系统。对于管理员用户，系统还提供了简单方便的批量导出功能，方便对用户的上传下载记录进行收集、整理与统计，所以此系统使用简单，操作高效，在操作上是完全是可行的。

## 1.5论文组织框架

本文共包含了五个章节的内容，各章节概述内容如下所示。

第一章是绪论，首先介绍的是本课题背景和意义，而后介绍了文档管理系统的研究现状，之后阐述了本课题所研究的内容，最后概述论文的组织框架。

第二章介绍了与本系统相关的一些技术和理论知识，对上传和下载的主要实现方法进行解释，对我们实现本文系统的实验环境的介绍。

第三章是系统的设计，首先对系统流程做了大概的概述，然后介绍了系统的功能以及数据库的概念、逻辑结构设计。

第四章是关于系统的实现与测试，主要介绍了上传下载的实现，对系统功能模块进行测试与结果展示。

第五章是总结及展望，总结本系统的完成情况，提出一些存在的问题并对今后的研究提出展望。

# 第二章 相关技术与理论介绍

## 2.1前后端分离技术与实现

### 2.1.1理解MVC模式与MVVM模式

MVC是一种经典的设计模式，全名为Model-View-Controller，即模型-视图-控制器，在各类开发语言中都有自己的实现[3]。其中，模型是用于封装数据的载体，例如，在java中一般通过一个简单的POJO（Plain Ordinary Java Object）来表示，其本质是一个普通的Java Bean[4]。对于视图来说，它更关心的是页面看起来是什么样子的，不管是用JSP还是原生HTML页面，当然在现今公司的开发中，大多数都是采用后者，降低与后端的耦合性。控制器的作用是来整合模型和视图的，比如说用户发送了一个HTTP请求，首先该请求经由控制器（controller）到后台获取数据并将其封装为相应的数据模型返回至视图中进行一定的页面展示，如下图：

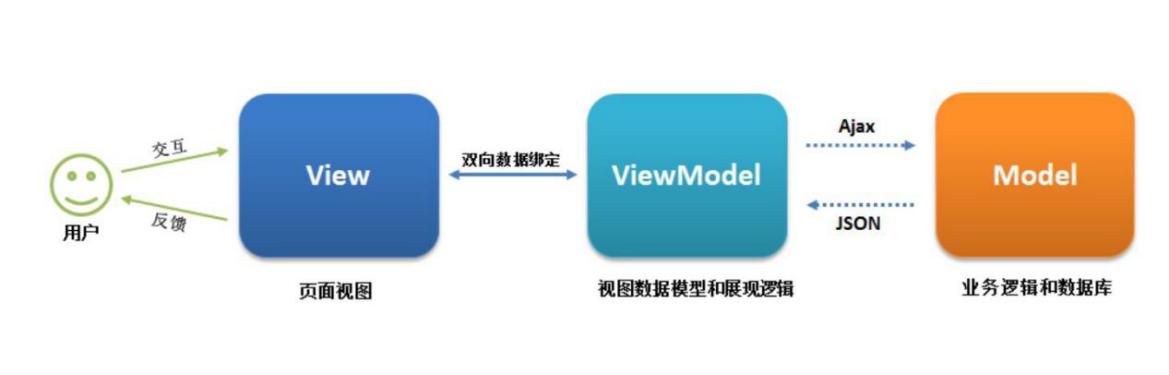


图2.1 MVC模型

MVVM（Model-View-ViewModel）是现今开发中比较流行的一种开发模式，它其实是采用了“数据模型数据双向绑定”的理念，而视图和模型之间是没有太大联系的，通过视图模型（ViewModel）进行交互，并且这种交互是双向的，也就是说视图中的数据的变化会同时修改数据源，而数据源的数据变化也会马上反应到视图上[5]。如下图：

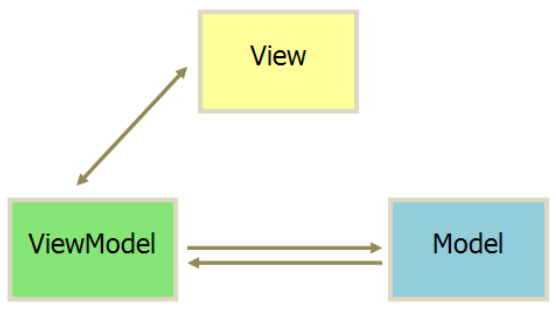


图2.2 MVVM模型

### 2.1.2 了解Restful架构

在web开发的早期，网页是前后端融在一起的，比如之前的PHP、JSP等。在之前的web1.0时代是没有什么大问题的，但是随着互联网的高速发展，手机、PC、Pad等各种客户端的出现使得前后端的分离似乎是势在必行了，因为这些客户端获取的数据基本都差不多，所以就需要一套统一的接口来提供数据，这样不同客户端都只从后端调用接口获取数据渲染页面，避免了代码重复。另外对于诸如Google platform，百度开放平台、微信开放平台等这样的开放平台来说，它们并不需要一套界面去展示，于是REST便派上了用场，只用它来提供一套服务接口即可。

REST（状态转移）是一种软件架构与设计风格，但并不是一种标准，它提供了一套设计原则和约束条件，用于客户端和服务端之间的交互[6]。在服务器端，应用程序的状态和功能可以被划分为资源（resource），比如应用对象、数据库记录和算法等都可以称为资源实体，每个资源通过URI（通用资源标识符）来标记自己唯一的地址，然后借由使用诸如GET、POST、PUT和DELETE等标准的HTTP请求方法去传输资源在服务器与客户机之间的状态[7]。通常RESTful形式的API是基于JSON数据传输的，效率更高，也更容易实现数据的缓存，软件的分层与组件的重用性也更好。

2.1.3 单页面应用程序

单页应用程序（SPA）是只有一个Web页面的应用程序，这种形式的Web应用程序在一开始就会加载单个HTML页面并在用户与应用程序交互时动态更新该页，单页Web应用程序在当今的前后端分离开发模式中应用的也特别广泛，许多传统网站已经或已经转化为单页Web应用程序。优点如下：

1. 操作体验流畅

有本地应用的感觉，应用程序切换过程中不会频繁有被“打断”的感觉。因为单页Web应用程序与服务端仅仅通过数据来交互，有了AJAX异步调用，使得界面框架的迁移非常便捷，可以付出很小的代价就迁移成其他的客户端产品。

1. 前端组件化

前端开发不再以页面为单位，更多地采用组件化的思想，代码结构和组织方式更加规范化，便于修改和调整。

1. API共享

如果提供的服务是多端的（浏览器端、Android端、IOS端、微信等），单页面应用的模式便于你在多个端共用API，可以显著减少服务端的工作量。易于更改的UI部分已被提前到多个端，而仅受业务数据模型影响的API更可能稳定并提供更好的服务。

1. 组件共享

在某些对性能体验要求不高的场景，或者产品处于试错阶段，借助于一些技术（Hybrid、React Native），可以在多种客户端之间实现组件的共享，就不必重复造轮子，浪费不必要的资源，这对于快速迭代来说有很大的意义。

## 2.2基于token的身份验证

HTTP是一种无状态的协议，我们并不知道是谁访问了应用。当客户端使用用户名和密码来进行身份验证并通过后便可以获取数据信息，但是下次再次发送请求时还得同样的操作去验证它，这样似乎不太符合常理。一般的解决方案是当用户登录时，如果验证通过会在服务器上生成一条记录说明是谁在登录，然后把该记录ID号发送给客户端，客户端将该ID号存储在cookie中，下次用户再发送请求到客户端时就携带cookie信息，如果能在服务端找到相应的记录表明用户通过身份验证，返回请求的数据至客户端。在这里，我们需要在服务器上存储为登录用户生成的session，可以存储在内存、磁盘或数据库中，但是需要在服务器上定期清理过期session，否则日积月累服务器会撑不住。

由于session是在服务器中存在的，并且需要依靠客户端的cookie机制，如果客户端禁用cookie，虽然也可以通过服务器URL重写发起请求，但仍然不符合后端只提供API的规范。在这一点上，基于令牌的认证方法可以派上用场，令牌（token）是由服务器生成的字符串，但并不需要存储在服务器端，客户端收到令牌后，再次请求服务端时就可以根据此令牌验证身份，如果令牌有效则验证通过，流程如下：

1. 客户端使用用户名和密码请求登录
2. 服务端收到请求，去验证用户名和密码
3. 验证成功后，服务端会签发一个token并发送给客户端。
4. 客户端收到token以后将它存储在本地中。
5. 以后客户端每次向服务端请求资源的时候需要带着服务端签发的token。
6. 服务端收到请求后去验证客户端携带的token，如果验证成功，则返回数据。

实施token的方法有很多，在本系统中，我采用的是标准方法JWT（JSON Web Tokens），如下图：

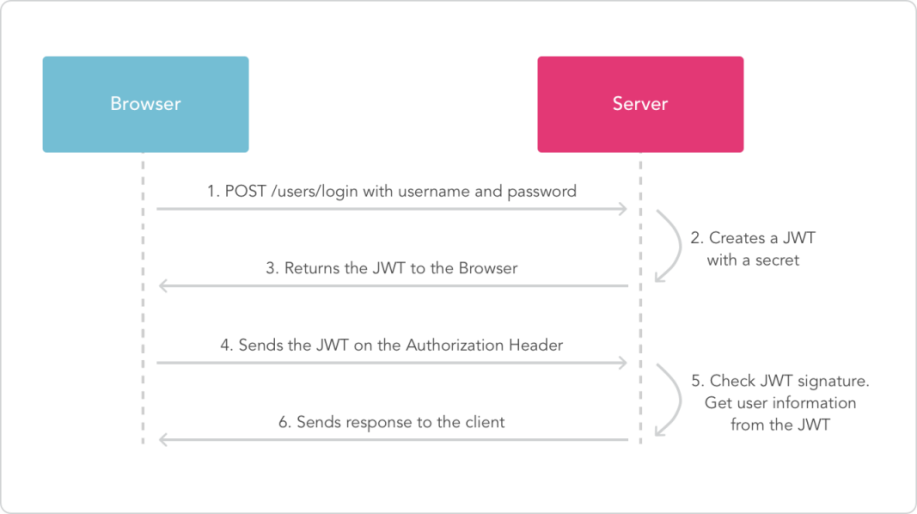


图2.3 token验证流程图

JWT是基于JSON风格的身份认证规范，对于无状态和分布式的Web应用来说，这种授权方式特别适用，它主要包含了三部分：Header、Payload和Signature。

1. Header

在header中通常包含的有两部分：token的类型和采用的加密算法。{“alg”: “HS256”, “type”: “JWT”}接下来对这部分内容使用Base64Url编码组成了JWT结构的第一部分。

1. Payload

token的第二部分是负载，它包含了claim，claim是一些实体（通常指用户）的状态和额外的元数据

1. Signature

创建signature需要使用头中指定的签名算法来签名编码的头部和有效载荷以及秘密密钥。

## 2.3了解WebSocket

传统的HTTP协议是无状态的，也就是说它不会记住特定的客户端。每个请求（request）都是由客户端发起的，接着服务器在处理后返回响应（response）结果，相反的服务端却很难去主动向客户端发送消息，很显然这不适合当今的大部分互联网应用。为了增强用户体验，加强与用户的交互，服务端会经常向客户端推送消息，而不可能让用户主动刷新获得数据。

近年来，随着WebSocket协议的提出，扩展了浏览器和服务器的通信功能，不再是传统的HTTP协议那样只能是由客户端去请求服务端，借由WebSocket协议可以建立浏览器与服务器的全双工通信，使得服务端也能够主动向客户端发送消息[8]。当客户端和服务端成功建立WebSocket连接后，它们之间便可以通过TCP连接进行通信，并且WebSocket连接本质上是TCP连接，所以不需要再每次传输数据时都携带重复的头数据，因此它也具有更加少量的数据传输。

## 2.4实验环境与开发工具介绍

2.4.1 实验环境

（1）Windows 10（旗舰版）64位操作系统

（2）CPU为 Intel(R) Core(TM) i5-7200U [CPU@2.50GHz](mailto:CPU@2.10GHz) 2.70GHz

（3）内存为8GB

（4）500G硬盘，其中128G固态硬盘（C盘）

2.4.2 开发工具

（1）数据库

采用的是MySQL数据库，由于MySQL数据库具有体积小、速度快、成本低的优势，更重要的是MySQL是开源的，可以免费用于商业活动，所以被广泛用在中小型网站中[9]。

1. 前端开发工具

前端代码采用VSCode编辑器，Visual Studio Code是由微软推出的一款轻量级代码编辑器，它是免费开源的，并且跨平台支持Win、Mac以及Linux，运行流畅[10]。支持几乎所有主流的开发语言，支持语法高亮、智能代码补全、自定义热键、代码托管工具，还有丰富的插件扩展。

1. 后端开发工具

后端开发环境采用的是IntelliJ IDEA 15（社区版），IEDA在业界是公认为最好的java开发工具之一，尤其是在智能代码提示、重构方面，支持各类版本工具（git、svn、github等）、JUnit、CVS。另外，IDEA也有丰富的插件支持，便捷的debug调试[11]。

1. 代码托管平台

本系统所有代码都放在github上，github是一个开源的分布式版本控制系统，随着越来越多的应用程序转移到了云上，GitHub已经成为了托管代码、发布代码的首选方法。尤其是在团队合作中，每个成员拉取分支，提交后再进行合并，大大提高开发效率。

## 2.5本章小结

本章首先简要介绍了前后端分离开发的模式，包括前端开发流程与后端RESTful 架构，然后介绍了基于token的身份验证，保证系统的安全性，接着了解了可以全双工通信的WebSocket协议，弥补了HTTP协议不能由服务端主动向客户端发起请求的缺点，最后介绍实验环境及开发工具。

# 第三章 系统总体设计

## 3.1在线文库管理系统体系结构设计

设计和实现本系统时采用了面向对象的核心思想，整体架构基于前后端分离的方式，后端负责设计接口与路由转发，前端调用接口获取数据进行页面渲染，将系统划分为：表现层、业务层和数据层[12]。如下图所示：

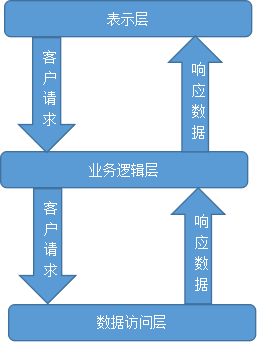


图3.1 三层体系结构图

（1）表现层（UI）：用户通过UI界面和系统产生交互，根据用户相应的请求返回数据。

（2）业务逻辑层（Business Logic）：系统各个模块的具体实现，负责控制系统运行的逻辑，并处理用户输入的数据，进而操作数据层。

（3）数据层（Data）：与数据库直接交互的操作层，具体为对数据库数据表的C(Create)R(Retrieve)U(Update)D(Delete)操作，数据层负责将逻辑层的数据持久化至数据库，并为逻辑层提供数据。

整个系统作为一个大的项目，按照上述的三层架构划分。对于表现层来说就是一个前端（UI）项目，它是可以独立开发的，不依赖于后端的开发进度，只要定义好接口的规范即可；对于业务逻辑层和数据层来说，它们统一包含在后端项目里，其中数据层也是和逻辑层没有耦合性的，它只是在数据的层面操作，然后以接口的形式供逻辑层调用，生成相应的服务。层与层之间可以做到相互独立，任何一层的改变都不会影响到其它层的功能，这样做可以尽量降低系统的耦合性，使得维护和管理系统更加方便，同时也让系统的扩展与升级更容易。

## 3.2系统功能模块设计

本系统是基于前后端分离的在线文库管理系统，主要分为前端和后端两个模块，后端采用Java的SpringBoot框架，生成各种API提供前端数据；前端都是基于Vue.js生成的单页面应用程序，分为用户前端界面和后台管理界面，其中用户前端展示的主要有登录注册、热门标签、资源分类、上传下载等功能模块，后台管理主要包括用户管理、文档管理、分类管理等功能模块，其结构如下图所示：



图3.2 在线文库管理系统功能模块图

1. 登录注册功能

登录注册功能是大部分系统都会有的常见功能，本系统也不例外，当用户注册登录成功后系统会生成一个token作为以后访问系统的令牌，在令牌有效的前提下，用户才可以继续访问，否则返回登录页面[13]。登录注册流程图如下：



图3.3 注册登录功能流程图

1. 分类展示功能

为了便于用户快速找到自己需要的文档，也为了方便文档的管理，所以对文档设置了分类。另外在展示文档列表上采取分页查看和滚动加载的模式，这样就不需要一次性查询所有文档数据，加载所有文档列表。

1. 收藏评价功能

这一部分的功能是为了增强与用户的交互性和用户的操作体验，当用户浏览到了喜欢的文档，但是可能因为某些原因暂时不能下载，就可以先收藏下来。当用户下载完后，可以对文档进行评价，打个评分，作为别人的参考。

1. 上传下载功能

上传下载是系统最核心的部分，用户需要注册登录成为会员，然后上传文档至管理员审核，审核通过后会获得相应的积分，然后可以去下载自己需要的文档，如果积分足够，便可以下载，下载完成后扣除相应积分，并给该文档的上传者增加相应积分[14]。该部分流程图如下：



图3.4 上传下载功能流程图

1. 消息推送功能

设计这一部分也是为了体验能够更加人性化，当用户上传了文档后，为了可以使管理员及时收到信息，就需要后台主动推送消息给前端，通知管理员有任务了，就不需要管理员自己去刷新页面了。

1. 文档管理功能

这部分属于后台管理模块，管理员可以通过界面实现对文档的管理，在本系统中只允许删除和查看操作。

## 3.3 数据库设计

根据系统的需求分析，对系统进行数据库的概念结构设计与逻辑结构设计，这一部分的设计对整个系统的设计有很大的影响。

### 3.3.1 概念结构设计

数据库概念模型：根据文档管理系统的需求分析，设计的各个实体如下所示:



图3.5 用户实体图



图3.6 资源类型实体图



图3.7 子分类实体图



图3.8 父分类实体图



图3.9 标签实体图



图3.10 文档实体图

数据库整体E-R图如下图所示：



图3.11 整体E-R图

### 3.3.2 逻辑结构设计

本系统MySQL数据库命名为resource\_management，主要建立的数据表有resource表、user表、download表等，具体展示部分如下所示。

表3-1 user表

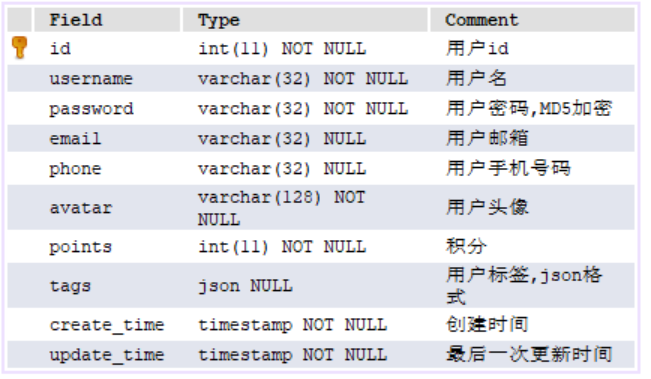


表3-2 resource表

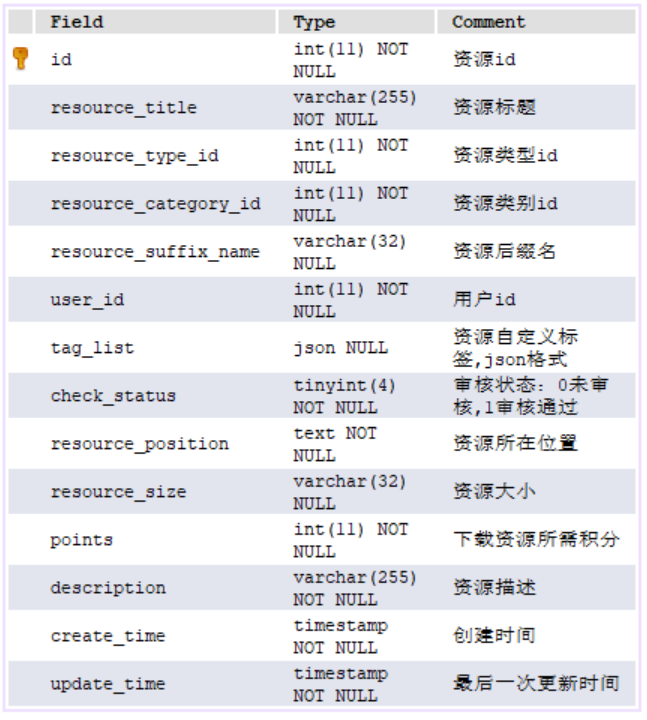


表3-3 resource\_type表

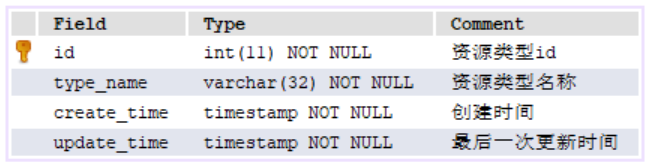


表3-4 parent\_category表

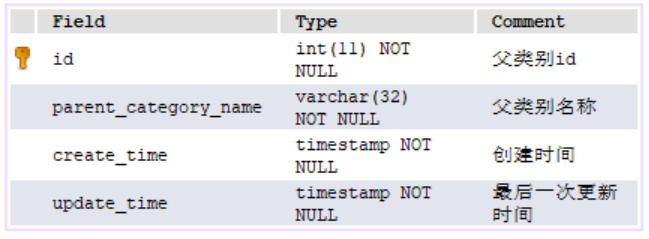


表3-5 child\_category表

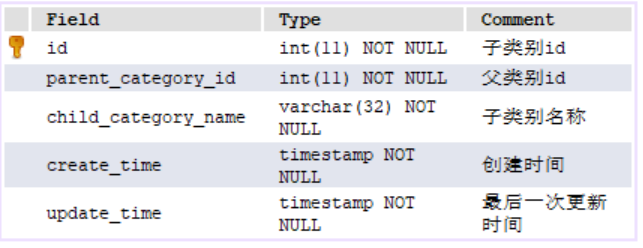


表3-6 collection表

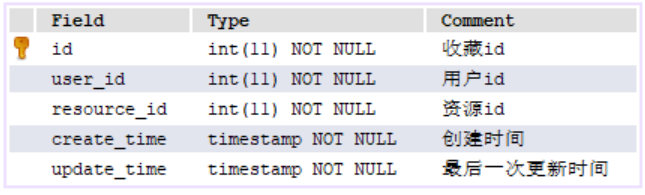


表3-7 download表

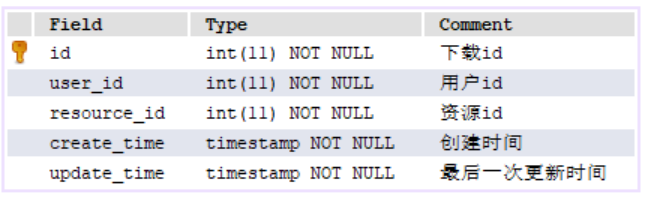


表3-8 remark表

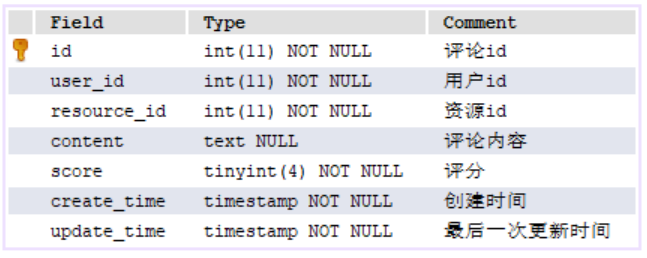
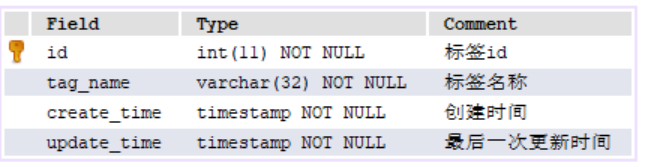


表3-9 tag表



## 3.4本章小结

本章主要介绍了系统的总体设计，首先是要进行充分的需求分析，了解系统到底需要哪些功能，明确功能模块，接下来是进行数据库的概念结构设计与逻辑设计结构，明确功能模块与表的关系，在对系统总体设计有了一定的了解之后，接下来就可以开始编写代码了。

# 第四章 系统实现与测试

## 4.1运行环境与技术支持

### 4.1.1 系统运行环境

实验的环境：Windows 10（旗舰版）64位操作系统，CPU为 Intel(R) Core(TM) i5-7200U [CPU@2.50GHz](mailto:CPU@2.10GHz) 2.70GHz，内存为8GB，另外500G硬盘，其中128G固态硬盘（C盘）。

### 4.1.2 技术支持

本系统使用的是Java的后端框架SpringBoot，IntelliJ IDEA 15作为后端开发工具，前端框架使用Vue2.0，Visual Studio Code作为前端编辑器，数据库使用的是Mysql5.7。

## 4.2系统的实现

### 4.2.1使用WebSocket实现消息的推送

由于SpringBoot已经整合了WebSocket，所以在SpringBoot中使用WebSocket也比较简单方便。首先后台开启WebSocket服务，配置WebSocket站点，前端连接该站点后，前后端就可以实现全双工的通信方式了，关键代码如下：

1. 后端代码

*//配置websocket站点*@ServerEndpoint(value = **"/websocket"**)  
@Component  
**public class** WebSocket {  
 **private static final** Logger ***logger*** = LoggerFactory.*getLogger*(WebSocket.**class**);  
 *// 在线人数* **public static int** *onlineNumber* = 0;  
 *//所有的对象* **public static** List<WebSocket> *webSocketList* = **new** CopyOnWriteArrayList<>();  
 *//会话* **private** Session **session**;  
 *//建立连接* @OnOpen  
 **public void** onOpen(Session session) {  
 *onlineNumber*++;  
 *webSocketList*.add(**this**);  
 **this**.**session** = session;  
 ***logger***.info(**"有新连接加入！当前在线人数："** + *onlineNumber*);  
 }  
 *// 连接关闭* @OnClose  
 **public void** opCloce() {  
 *onlineNumber*--;  
 *webSocketList*.remove(**this**);  
 ***logger***.info(**"有连接关闭！当前在线人数："** + *onlineNumber*);  
 }  
 @OnMessage  
 **public void** onMessage(String message, Session session) {  
 ***logger***.info(**"收到客户端发来的消息：{}"**, message);  
 sendMessage(**"欢迎连接！"**);  
 }  
 **public void** sendMessage(String message) {  
 **for** (WebSocket webSocket : *webSocketList*) {  
 ***logger***.info(**"websocket广播消息：{}"**, message);  
 **try** {  
 webSocket.**session**.getBasicRemote().sendText(message);  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}

1. 前端代码

websocketConnection() {

var websocket = null;

var \_this = this;

if('WebSocket' in window){

websocket = new WebSocket('ws://localhost:8082/websocket');

} else {

alert('该浏览器不支持websocket!')

}

websocket.onopen = function(event){

console.log('建立连接')

}

websocket.onclose = function(event){

console.log('连接关闭')

}

websocket.onmessage = function(event){

alert('有消息来了！')//弹窗提醒，

\_this.$refs.notice.play();

}

websocket.onerror = function(){

alert('websocket通信发生错误！')

}

window.onbeforeunload = function(){

websocket.close();

}

}

### 4.2.2用户上传模块的实现

用户选择文件上传，如果该文件不存在于服务器上，则允许上传，然后会经过后台管理员的审核，审核通过后则插入记录，并为该用户增加相应积分；若审核不通过则删除该文件，流程图如下：



图4.1 上传模块流程图

代码如下：

*/\*\*上传资源\*/***public static boolean** uploadResource(MultipartFile file, String filePath, String fileName) {  
 **boolean** isUpload = **false**;  
 File dest = **new** File(filePath + fileName);  
 *// 检测是否存在目录* **if** (!dest.getParentFile().exists()) {  
 dest.getParentFile().mkdirs();  
 }  
 **try** {  
 file.transferTo(dest);  
 isUpload = **true**;  
 } **catch** (IllegalStateException e) {  
 e.printStackTrace();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 **return** isUpload;  
}

### 4.2.3用户下载模块的实现

用户选择需要下载的文件后，首先判断该文件是否属于该用户，如果是则直接下载；如果不是先判断用户的积分是否足够，如果够则允许下载，下载成功后会删除用户相应积分并增加文件上传者相应积分。流程图如下：



图4.2 下载模块

代码如下：

**public static boolean** downloadResource(HttpServletRequest httpServletRequest,HttpServletResponse res, String fileName) {  
 res.setHeader(**"content-type"**, **"application/octet-stream;charset=UTF-8"**);  
 res.setContentType(**"application/octet-stream;charset:utf-8"**);  
 **try** {  
 res.setHeader(**"Content-Disposition"**, **"attachment;filename\*=UTF-8''"** + URLEncoder.*encode*(fileName,**"UTF-8"**));  
 } **catch** (UnsupportedEncodingException e) {  
 e.printStackTrace();  
 }  
 **byte**[] buff = **new byte**[1024];  
 BufferedInputStream bis = **null**;  
 OutputStream os = **null**;  
 **try** {  
 os = res.getOutputStream();  
 bis = **new** BufferedInputStream(**new** FileInputStream(**new** File(**"e://test//"** + fileName)));  
 **int** i = bis.read(buff);  
 **while** (i != -1) {  
 os.write(buff, 0, i);  
 *//os.flush();* i = bis.read(buff);  
 }  
 **return true**;  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 **return false**;  
 } **finally** {  
 **if** (bis != **null**) {  
 **try** {  
 bis.close();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}

### 4.2.4用户评论模块的实现

当用户下载了某资源，并且查看后，可以对该资源作出评价，打个评分，给后面的用户起到一个参考的作用，用户不能评价自己上传的资源，也只有下载了资源才能去评价，流程图如下：



图4.3 评论模块

### 4.2.5用户收藏模块的实现

用户浏览到自己喜欢的文档，但是积分暂时不够，或者觉得以后有用，可以先进行收藏，等到后面真的有需要了再去下载，这部分的逻辑比较简单，就是在数据库中增加一条记录。

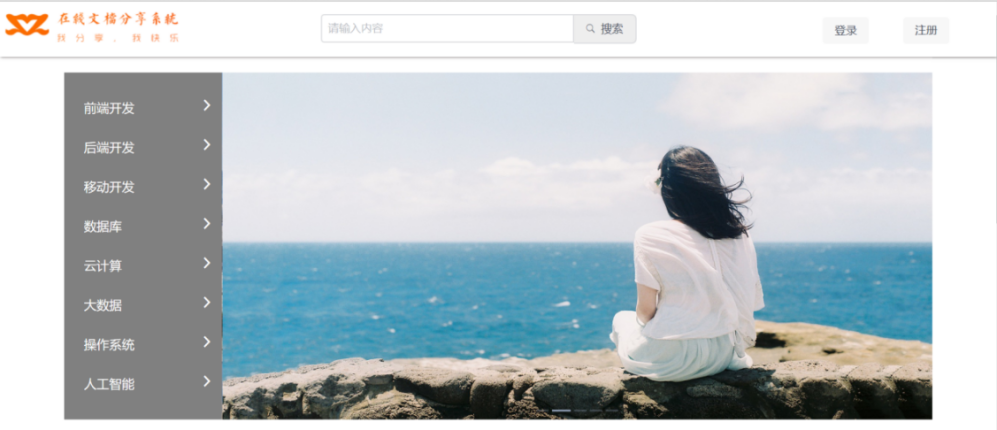
### 4.2.6权限验证模块的实现

利用Spring面向切面编程的特点（AOP），在用户请求后端接口时，首先进行拦截操作，如果请求的接口是公共的（如首页部分的请求），则不需要验证[15]；反之如果请求的接口是需要登录后才可以返回数据的，则提示用户登录，所以如何判断用户是否登录是关键因素，前面说过用token的机制。由于SpringBoot整合了JWT，所以代码的编写也很简单。当用户登录后，返回token给前端，以后前端再次发送请求时将该token携带进header里，后端拦截请求获取header里的token，判断该token是否有效。

## 4.3系统的运行

### 4.3.1首页设计

从下图中可以见到，首页分为文档类型模块、热门标签模块、文档列表模块3个主要的部分，对于文档列表添加了滚动加载的小功能。



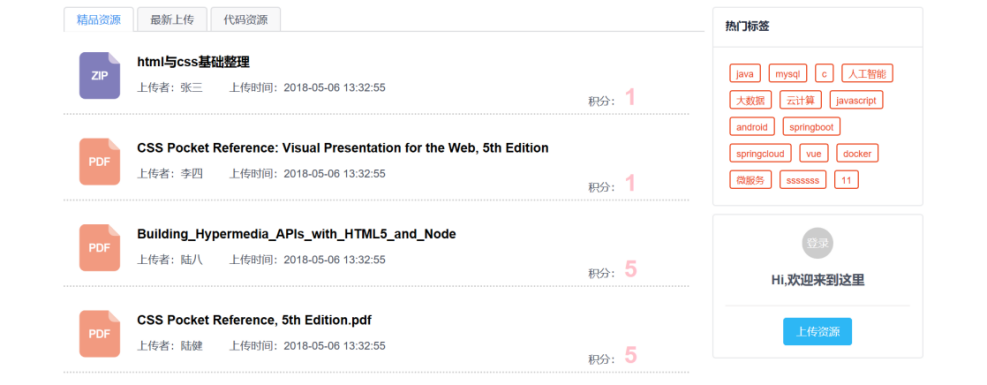


图4.4 主页界面

### 4.3.2注册与登录界面设计

当用户在登录该系统的时候，首先会在数据库中查找这个用户是否存在，若不存在则需要先注册然后才能登录。注册界面如下图所示：



图4.5 注册与登录界面

当用户注册成功后，才可以进行系统登录，登录界面如上图所示：

用户在登录时，输入用户名与密码，经过后台验证通过后，方可登录成功。

### 4.3.3 文档筛选页面

主要是提供用户根据关键字以及文档类别，快速找到自己需要的资源，分页展示搜索的结果，可以根据最新上传和最多下载的方式排序，文档筛选界面如下图所示：





图4.6 文档筛选界面

4.3.4文档详情界面

文档详情界面展示了文档的详细的信息和其他用户对此文档的评分与评价内容，除此之外还提供了下载和评论的功能，左边是这个文档的所有者的一些信息，如下图所示。

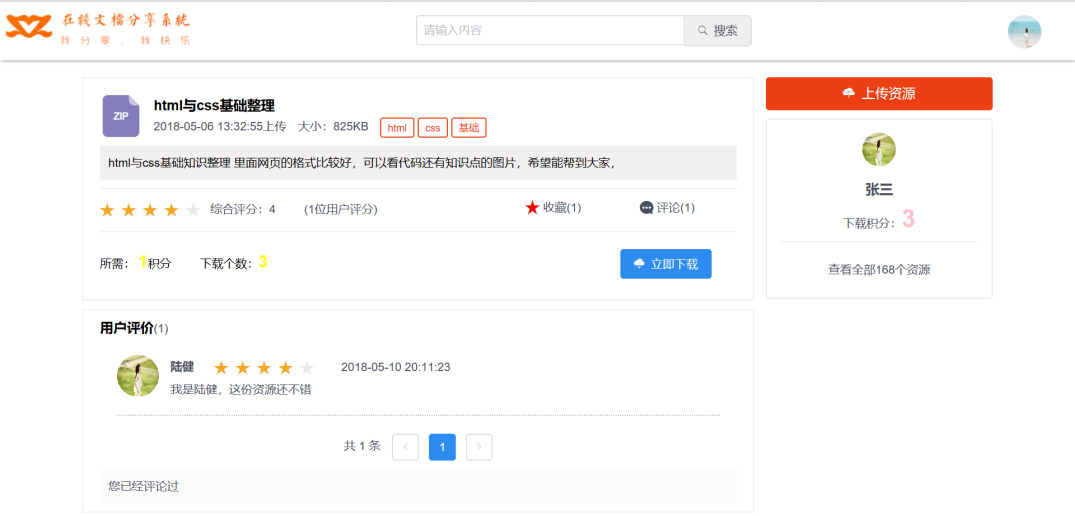


图4.7 文档详情界面

### 4.3.5用户中心

用户中心界面展示了用户的基本信息，有积分、文档数量、下载数量和收藏数量，可以查看自己的上传下载与收藏记录，如下图：



图4.8 用户中心界面

用户上传界面其实是一个表单，用户填写正确的字段，点击上传即可，如下图所示：



图4.9 用户上传界面

## 4.4后台功能实现

### 4.4.1 登录模块实现

后台登录只允许管理员有登录，非管理员是没有权限登录的，登录界面如下：

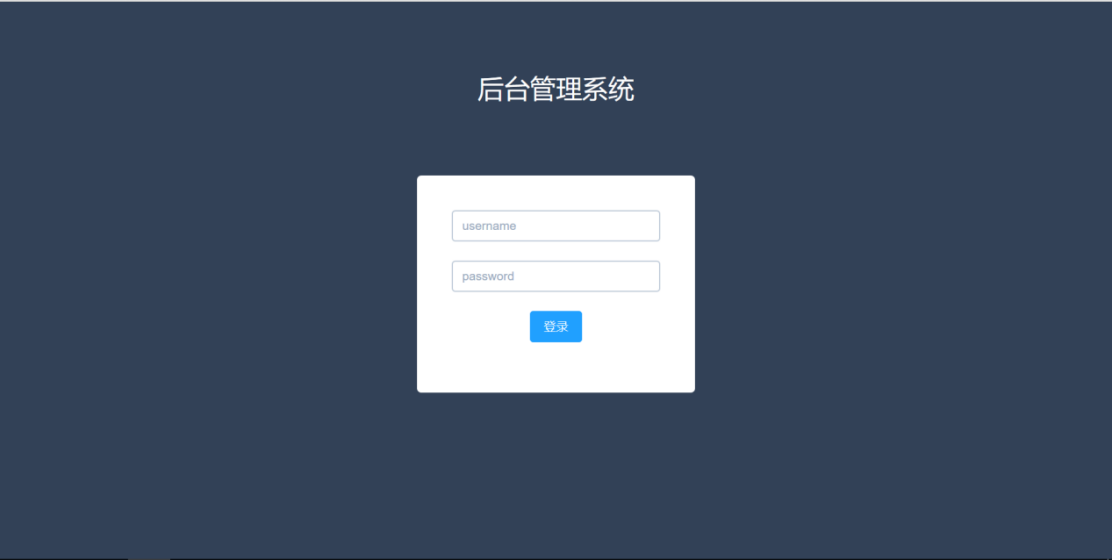


图4.10 后台登录界面

### 4.4.2 后台管理模块

后台管理主要包括用户管理、文档管理、分类管理等主要功能模块，管理员可以查看和删除用户信息，可以对文档进行审核，如果审核不通过，则予以删除，可以上传文档后缀图标，添加、删除和修改后缀相关信息。

用户管理界面分页展示了用户的基本信息，只提供了删除的操作，如果发现某用户有恶意行为，可以将其删除，如下图：



图4.11 用户管理界面

文档管理界面用一个Tab选项卡来展示的，分为全部、未审核和已审核，除了展示了文档的基本信息以外，还可以展开子项查看详细信息。另外提供了对新上传文档的审核操作，审核不通过则予以删除，但是对于审核通过的文档是不能随意删除的，如下图：

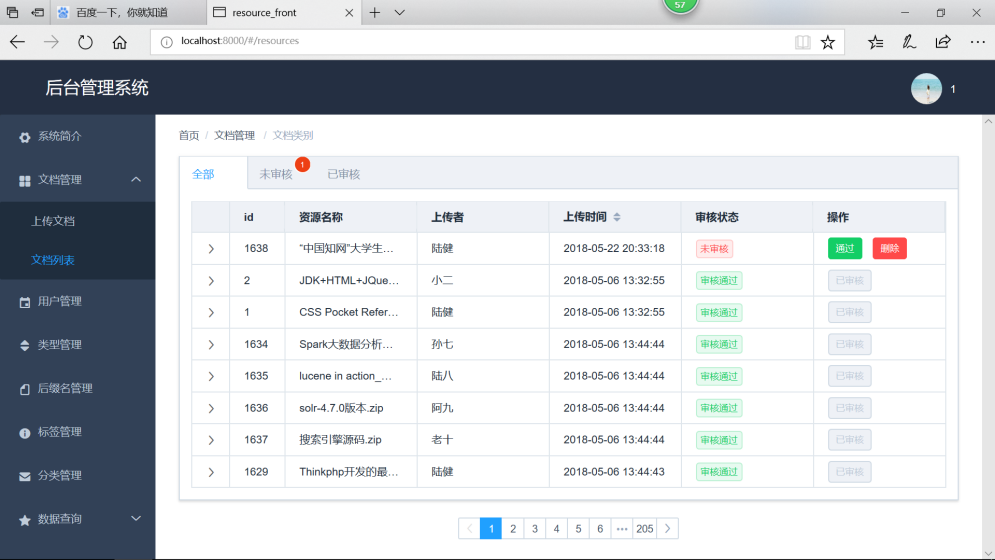


图4.12 文档管理界面

分类管理界面首先展示的是父级分类的信息，可以对其进行增加、修改、删除的操作，但是要注意删除父级分类的时候，其所有子级分类也会删除，如下图所示：

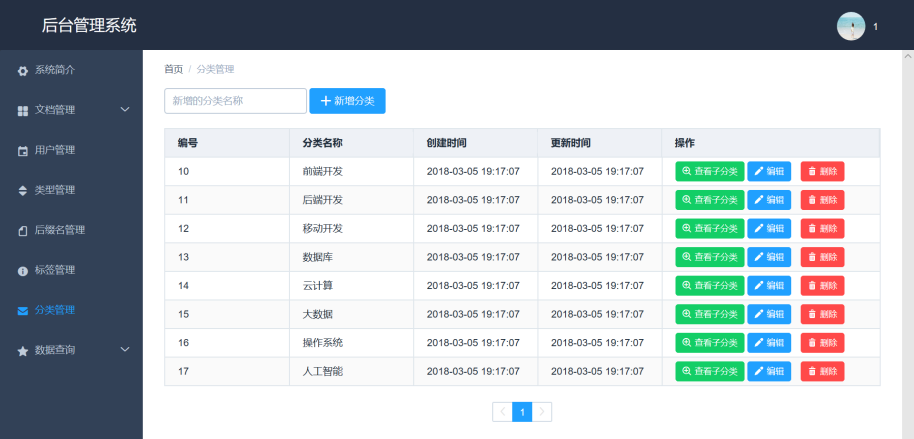


图4.13 分类管理界面

点击查看子分类按钮进入到子分类页面，该页面展示了特定父分类下面的所有子分类信息，对于每个子分类也是提供了增加、删除和修改的操作，如下图所示：

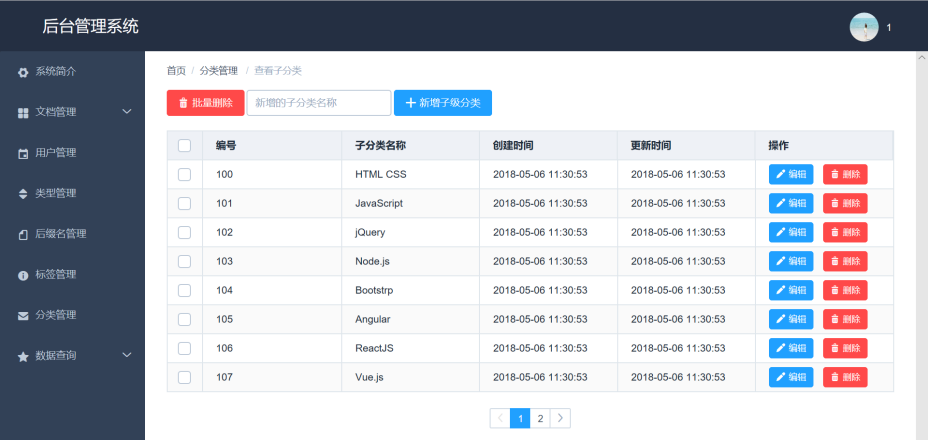


图4.14 子分类管理界面

资源类型管理界面展示了所有的类型信息，也是提供了增加、删除、修改的操作，如下图所示：



图4.15 子分类管理界面

标签管理界面展示的是标签列表，对应前台的热门标签，可以添加、修改和（批量）删除，如下图所示：

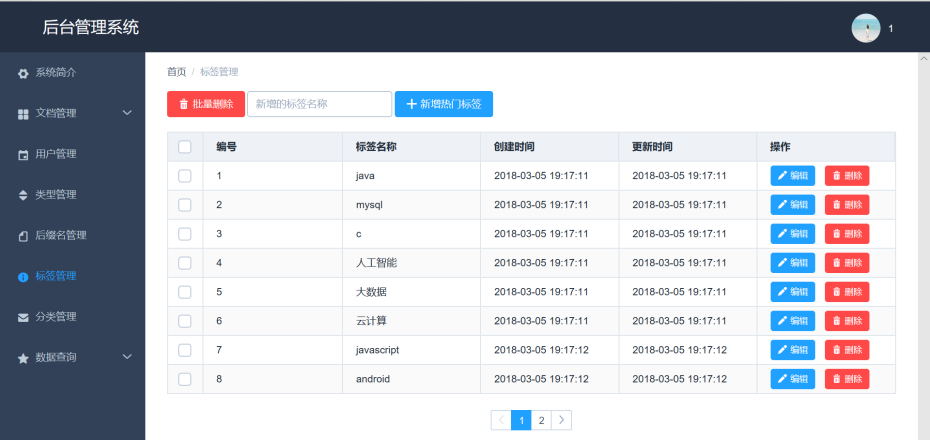


图4.16 标签管理界面

## 4.5本章小结

本章开始介绍了系统的运行环境和相关技术框架，系统是在Win64旗舰版操作的，使用的Java编写与SpringBoot框架和数据库层的框架MyBatis，然后介绍了系统的实现，并展示了系统的主要功能界面，包括系统的主页与后台管理，并对系统的几个模块界面做了简单的说明。

# 第五章 总结与展望

## 5.1总结

从寒假开始到4月份，差不多用了两个多月的时间来完成最后的毕业设计：在线文库管理系统。通过前期充分的需求分析，划分功能模块，再到后来的数据库设计，这些都花了不少的时间。由于上学期的课程设计，让我明白了前期的设计是多么的重要，避免了后期推翻重写，一开始就把功能划分好，设计的越详细，后面编写代码就会越顺利，当然不可能所有的都会考虑到，只要大的方向定好了，小修小补的问题都不是很大。后面到了编写代码的阶段了，顺序如下：

1. 先是设计了后台接口代码，从数据层开始，连接数据库，完成基本的增删改查，这部分不涉及到逻辑业务，就只是单纯的数据库计基本操作；接下来开始编写业务逻辑层的代码，写之前也是先画好流程图，仔细想好所有的点，避免写代码的时候有遗漏；最后就是设计API，这部分也不涉及到其他的，就只是简单的路由转发而已。上述每一层的代码开发完，都会进行相应的单元测试，避免这一层的代码是没问题的，也方便后面调试的时候容易定位错误。从整体设计上来说，这一部分利用分层设计的思想，结构上很清晰，代码也有相应的注释，做到易读。
2. 后端接口设计完成后，开始编写后台管理系统，这一部分借鉴了上学期的后台管理界面，因为管理系统大都差不多，也不要求界面有多华丽，只要实现相应功能就可以了，所以也是从网上找的后台网站模板，自己也做了修改，当然也是遇到了不少的坑的。总体上来说，这部分做的还是挺快的，调用后台接口获取数据稍加渲染即可。
3. 后台管理系统完成后，就到了重点：用户界面。这一部分只能自己硬着头皮上了，由于自己的页面设计能力不是太行，所以前期也是观察了好多网站的页面设计，最终确定去模仿CSDN去设计。前端的页面布局是我一直搞不太明白的痛点，下手之前我就知道会有更多的坑要去踩，但还是得去做。在写代码的过程中，遇到很多问题，刚开始还会去找最好的方法去解决，最后有点烦了，就不太注重代码的结构了，所以写的不是很优雅。总的来说，这部分是我写的最头疼的部分了，花的时间也是最多的，最后完成的差不多的时候，也是如释重负，送了一口气。

代码写的差不多了之后就该要测试了，主要就是模拟用户的行为，包括一些奇怪的操作，发现还是有很多没有考虑的地方，当然我知道不管怎么修改总会是有bug的，所以尽量做到常见的操作不会出现问题。

通过这次毕业设计，我真正意义上从零到一实现了一个自己的系统，对网站的整体规划、设计与实现有了更加深刻的体会，也领悟到了一些经验与教训，同时对数据库的设计和管理也理解的更加深刻了。当然本系统的设计还有很多不足之处，后期开发可以进一步去完善。

## 5.2展望

本文主要完成的是文档的管理与分享，基本完成了系统的设计与实现，当然改进的地方也有很多。

1. 对于文档的存放没有做到分类别存放，都放在了一个目录下，不便于以后查找资源，另外文档也都是放在本机，后续可以考虑搭建一套文件服务器，专门用来存放文件。
2. 如今机器学习与人工智能都非常火，以后可以做一个智能推荐，利用一些推荐算法为每个用户推荐他感兴趣的内容，让用户感觉到更加人性化，更愿意在这个平台上去分享自己的资源。
3. 可以在搜索方面做一些改进，目前系统采用的还是数据库的模糊查询，后续改进的话把查询的内容先分词，关联规则，也可以进行一些语义分析，做到搜索更加智能化。

随着互联网的高速发展，各种新技术层出不穷，要勇于创新，尝试在自己的系统中加入这些新技术，经常思考能否再进行一些优化，再大的系统都是一步一步成长过来的，希望后面自己可以一直维护着这个项目，一直做下去。

# 致 谢

在毕业设计的进行过程中，刚开始从零开始设计的时候还是比较麻烦的，不知道到底从哪入手，后来编写代码的过程中，也是遇到这样那样的坑，有很多问题自己也不能很好的解决。首先会先上网查找资料，找找看有没有相似的问题，是在解决不了的就会去求助我的老师，所以我要好好地谢谢我的指导老师--邢玉萍老师，正是在老师的悉心指导与耐心的督促下，我才能按时完成这次的毕业设计。邢老师特别的认真负责，从一开始就给我们一些资料，让我们寒假回去先了解了解如何去做这个系统。开学来写开题报告的时候，邢老师也是非常仔细的给我们很多修改意见。每次有不懂的地方去请教邢老师的时候，邢老师都特别负责的给我们意见，指导我们如何去解决问题。老师也会及时查看我们写的进度记录，给我们相应的意见与建议。

在一开始设计系统整体框架时，一点头绪也没有，也是遇到了很多困难，也意识到了后面写代码不可能一帆风顺的，提前给自己做好心理准备，一边做一边去学习。过程中也有想过放弃，不想做了，但也只是想想而已，调整一下，和同学聊聊天，缓解一下心情，又开始努力去做，所以在此我也还是要好好谢谢我的同学，在遇到困难的时候，我们互相鼓励，互相打气，互相加油，互相讨论，同时也学到了很多自己一个人学不到的东西。

除了感谢我的老师与同学，也要感谢一下我的家人，感谢他们在我最沮丧的时候给予我的鼓励，感谢他们一直对我的不抛弃不放弃。最后，我希望我的学校江苏大学可以越来越好，越来越繁荣，在即将要踏入社会工作的此刻，我衷心的祝愿我的学院越来越优秀，我的老师和同学事事如意，一切安好！

# 参考文献

1. 冯胜春.基于B/S架构的在线文库系统的研究与实现[D].中国海洋大学,2014.

[2] 陈则彬. 电子档案管理系统的研究与实现: [硕士学位论文]. 上海: 同济大学, 2008.

[3] 赵伟, 王志华, 周兵. 基于MVC的e-ERP系统的设计与实现[J]. 计算机应用与软件, 2013, 30(2) : 107-110.

[4] FLANAGAN.《Java技术手册》. 中国电力出版社,2002年6月:1-465

[5] 梁灏. Vue.js实战 [M]. 清华大学出版社, 2017.

[6] 王福强. SpringBoot揭秘：快速构建微服务体系 [M]. 机械工业出版社, 2016

[7] 王刚. 基于SOA的社区矫正数据集成子系统研究开发: [硕士学位论

文].西安: 西安理工大学, 2014.

[8] 宗原. 基于Web的结构安全监测平台设计与实现: [硕士学位论文]. 江苏: 江苏大学, 2015.

[9] 唐汉明. 深入浅出MySQL[M]. 人民邮电出版社, 2014.

[10] Elena Ferrari, Bhavani Thuraisingham. Web and Information Security. USA: IRM Press Publishing, 2007.20~33.

[11] Craig, Walls. Spring Boot实战 [M]. 人民邮电出版社, 2016.

[12] BruceEckel.《Java编程思想》. 机械工业出版社,2003年10月:1-378

[13] Cai Ning, Yeung R W. Secure Network coding on a wiretap network [J].

IEEE Trans on Information Theory, 2011, 57(1): 424-435.

[14] 王延刚, 何 斌, 宋 伟, 等. 面向工程机械的文档信息管理系统的设计

与实现［J. 计算机与现代化, 2013(1) : 185－188.

[15] Cay, S, Horstmann. Java核心技术 卷I[M]. 机械工业出版社, 2016.