**本科毕业设计（论文）**

**题目名称：** 基于微信的程序员微阅读小程序的设计与开发

**学 院：** 计算机科学技术学院

**专业年级：** 计算机科学与技术2019级

**学生姓名：**

**班级学号**

**指导教师：**

**二○二三 年 六 月**

# 摘 要

本论文介绍了一款面向程序员的在线书城微信小程序——“微书”的设计与实现。由于当前市场上缺乏专门为程序员和技术爱好者设计的便捷阅读平台，我们看到了设计和实现一个能够帮助他们获取和学习技术知识的平台的需求。因此，我们提出并实现了“微书”。

“微书”的开发涉及到了后端开发、前端开发以及数据库设计。在后端开发方面，我们使用了Java的Spring Boot框架，并设计了一系列的RESTful API以供微信小程序调用。在前端开发方面，我们采用了Uniapp框架，并包含了书架、书城、个人中心、H5阅读器、评论和热门推荐等模块，以提供丰富的阅读功能。在数据库设计上，我们选择了MongoDB作为存储书籍数据的数据库，对每本书籍都创建了一个文档并将书籍的元数据存储其中。另外，我们也使用EPUB框架提取本地电子书文件并进行处理，以提供高质量的阅读体验。

在实现过程中，我们进行了全面的测试，以确保系统的稳定性和可靠性。总的来说，这个项目旨在为程序员和技术爱好者提供一个便捷、舒适的阅读平台，不仅具有高质量的阅读体验，而且具有丰富的功能。我们相信，这个项目可以为程序员和技术爱好者提供更加便利和高效的阅读体验。

**关键字:** 微信小程序；EPUB框架；Spring Boot；MongoDB

# Abstract

This paper introduces the design and implementation of a WeChat Mini Program online bookstore targeting programmers, namely "WeBook". Currently, the market lacks a convenient reading platform specifically designed for programmers and tech enthusiasts. Hence, we identified the demand to design and implement a platform that helps them acquire and learn technical knowledge. Consequently, we proposed and realized " WeBook ".

The development of " WeBook " mainly involves backend development, frontend development, and database design. For backend development, we utilized Java's Spring Boot framework and designed a series of RESTful APIs for the WeChat Mini Program to call. For frontend development, we adopted the Uniapp framework, incorporating modules such as bookshelf, bookstore, personal center, H5 reader, comments, and popular recommendations to provide a wide range of reading functions. In terms of database design, we chose MongoDB as the database to store book data, created a document for each book, and stored the metadata of the books therein. Additionally, we used the EPUB framework to extract and process local e-book files to deliver a high-quality reading experience.

Throughout the implementation process, we conducted comprehensive tests to ensure system stability and reliability. In summary, this project aims to provide a convenient and comfortable reading platform for programmers and tech enthusiasts. It not only offers a high-quality reading experience but also a wide range of functions. We believe this project can provide a more convenient and efficient reading experience for programmers and tech enthusiasts.

**Keywords**: WeChat Mini Program; EPUB Framework; Spring Boot; MongoDB.

**目 录**

[摘 要 I](#_Toc136685125)

[Abstract II](#_Toc136685126)

[第1章 概述 1](#_Toc136685127)

[1.1 项目介绍 1](#_Toc136685128)

[1.2 国内外发展状况 1](#_Toc136685129)

[1.3 选题背景及意义 1](#_Toc136685130)

[第2章 方案论证 3](#_Toc136685131)

[2.1 可行性分析 3](#_Toc136685132)

[2.1.1 技术可行性 3](#_Toc136685133)

[2.1.2 经济可行性 3](#_Toc136685134)

[2.1.3 操作可行性 3](#_Toc136685135)

[2.1.4 法律可行性 4](#_Toc136685136)

[2.2 方案选择 4](#_Toc136685137)

[2.2.1 所使用的语言 4](#_Toc136685138)

[2.2.2 所涉及到的框架与主要技术 5](#_Toc136685139)

[2.2.3 所使用的主要开发工具 6](#_Toc136685140)

[2.2.4 所使用的数据库 6](#_Toc136685141)

[第3章 需求分析 7](#_Toc136685142)

[3.1 问题分析 7](#_Toc136685143)

[3.2 设计原理 7](#_Toc136685144)

[3.3 总体需求分析 7](#_Toc136685145)

[3.4 功能需求分析 8](#_Toc136685146)

[3.5 数据分析 9](#_Toc136685147)

[3.5.1 数据流分析 10](#_Toc136685148)

[3.5.2 E-R图 11](#_Toc136685149)

[3.5.3 数据字典 12](#_Toc136685150)

[3.6 性能分析 13](#_Toc136685151)

[第4章 设计论述 15](#_Toc136685152)

[4.1 概要设计 15](#_Toc136685153)

[4.1.1 总体功能模块设计 15](#_Toc136685154)

[4.1.2 系统用例设计 17](#_Toc136685155)

[4.1.3 系统流程图设计 17](#_Toc136685156)

[4.2 数据设计 18](#_Toc136685157)

[4.2.1 数据库逻辑结构设计 18](#_Toc136685158)

[4.2.2 数据表设计 19](#_Toc136685159)

[4.3 系统详细设计与实现 22](#_Toc136685160)

[4.3.1 系统登录模块 22](#_Toc136685161)

[4.3.2 我的书架模块 25](#_Toc136685162)

[4.3.3 H5阅读器模块 27](#_Toc136685163)

[4.3.4 书籍详情模块 32](#_Toc136685164)

[4.3.5 书城模块 37](#_Toc136685165)

[4.3.6 书籍分类模块 40](#_Toc136685166)

[4.3.7 个人中心模块 42](#_Toc136685167)

[4.3.8 提取书籍数据模块 44](#_Toc136685168)

[第5章 软件测试 46](#_Toc136685169)

[5.1 单元测试 46](#_Toc136685170)

[5.2 集成测试 48](#_Toc136685171)

[5.3 性能测试 49](#_Toc136685172)

[结 论 50](#_Toc136685173)

[致 谢 51](#_Toc136685174)

[参考文献 52](#_Toc136685175)

[附 录 53](#_Toc136685176)

# 第1章 概述

## 1.1 项目介绍

小程序，作为一种即开即用的应用，为用户提供了无需下载安装，随时随地便捷体验。用户只需通过扫描二维码或简单搜索即可访问。然而，现今的阅读环境，商业化和广告嵌入已成为主流，这无疑影响了读者的阅读体验。对此，我们希望打造微书——一款免费、舒适的阅读应用，让其成为用户（尤其是程序员）的移动图书馆，以深度和质量的内容对抗浮躁的阅读环境。基于此愿景，我选择微信小程序和Uniapp作为前端框架，并采用Java的Spring Boot框架以及MongoDB和MySQL数据库作为后端支持，精心开发了这款面向程序员的阅读应用——微书。

## 1.2 国内外发展状况

在线书城在全球范围内都得到了广泛的应用。例如，Amazon Kindle，Google Play Books，和Apple's iBooks等，都是全球知名的在线电子书服务[1]。这些服务平台通过提供在线阅读、购买、下载等一站式服务，大大便利了读者的阅读体验，微信小程序不仅在中国非常普及，但在海外市场也有其影响力。例如，像Facebook，Twitter，和Instagram等社交媒体平台也推出了类似的小程序或小应用功能。这些应用在全球范围内拥有大量的用户基础，也在不断推动着在线阅读平台的发展。然后，在技术层面，Java的Spring Boot框架、Uniapp框架和MongoDB数据库等，都是在全球范围内被广泛使用的技术。特别是在后端开发、前端开发和数据存储等领域，它们的应用已经十分成熟，有大量的国内外案例可以作为参考。[3]

在面向程序员和技术爱好者的在线书城方面，国外也有一些知名的平台，如O'Reilly Media，Packt等，它们提供大量的技术类电子书籍，并有自己的在线阅读平台。这些平台的成功经验和模式，对于我们本项目的发展也提供了一些参考和启示。总的来说，无论是在中国还是在全球范围内，在线书城和相关技术都在不断发展和进步。同时，对于特定群体，如程序员和技术爱好者的需求，也有越来越多的关注和应用。这为我们的项目提供了广阔的发展空间和巨大的市场潜力。

## 1.3 选题背景及意义

随着移动互联网和智能手机的普及，人们越来越习惯于在手机上阅读各种内容，包括新闻、文章、小说等。然而，目前市场上的移动阅读应用大多存在以下问题：内容质量参差不齐，广告过多干扰阅读，付费模式不合理限制用户选择，界面设计不美观不友好等，并且面向程序员的阅读应用也很少。为此，我采用了微信小程序作为前端基础，使用 Java 的 Spring Boot 框架和数据库 MongoDB、MySQL 作为后台支撑，开发了这个项目。微书旨在打造一个移动端的程序员的图书馆，提供免费的程序相关的电子书资源和纯净的阅读体验，让用户在浩瀚的书海中找到自己喜欢的书籍，享受纯净、便捷的阅读体验

# 第2章 方案论证

## 2.1 可行性分析

可行性研究的主要目标是对技术的创新性、经济效益和实施可能性等进行深入探讨和证实。我们需要对项目的优点和挑战有清晰的理解，同时预测项目的成功概率。在启动项目之前进行深入的可行性研究是前期调查的关键步骤。本章将从四个方面分析微书的可行性，包括技术可行性、经济可行性、操作可行性和操作可行性。

2.1.1 技术可行性

本项目采用Java的Spring Boot框架作为后端开发语言，Spring Boot是一种快速开发框架，它可以帮助我们快速构建可扩展的Web应用程序和RESTful服务。Spring Boot拥有丰富的生态系统，使得我们可以方便地使用各种插件和库，例如Spring Data MongoDB库，来实现快速的数据库开发。因此，Spring Boot是一个非常合适的选择，可以确保我们在开发过程中具有高效性和可扩展性。本项目采用MongoDB作为书城数据库存储书籍数据。MongoDB是一种非关系型数据库，它能够存储大量的文档型数据，因此适合存储图书数据。同时，我们为每本书籍创建了一个文档，并将书籍元数据存储在其中，这使得数据查询更加高效。我们选择了Uniapp框架进行前端开发。Uniapp是一个跨平台的开发框架，可以同时开发小程序、H5、iOS和Android应用。它具有丰富的组件库和工具，使得我们能够快速构建具有良好用户体验的前端界面。

综上所述，我们所选择的技术具有良好的可行性和适用性，并且能够帮助我们实现本项目的需求。

2.1.2 经济可行性

本项目是面向程序员和技术爱好者的在线书城微信小程序，旨在提供一个便捷、舒适的阅读平台，帮助用户获取和学习技术知识。根据市场调研，技术领域的读者群体数量庞大，而目前市面上类似的技术书籍在线阅读平台较少，因此本项目有良好的市场潜力。

2.1.3 操作可行性

技术上本项目选择了常用的Java的Spring Boot框架和MongoDB数据库来支持后端开发，这些技术在行业内得到广泛应用，并且有成熟的社区支持和丰富的开发文档。

团队配备：我们的团队成员具有丰富的技术开发经验和相关领域的知识，具备完成项目所需的技能和能力。我们的团队分工合理，各成员职责明确，可以有效协作完成项目。

项目进度和风险：我们采用敏捷开发方法，将项目分为多个阶段进行开发和测试，并不断进行迭代和优化。我们会密切关注项目进度和风险，及时解决问题并做出相应的调整。

综上所述，我们相信本项目具备良好的操作可行性，可以顺利地进行并得到有效的执行。

2.1.4 法律可行性

知识产权：在书城中展示的所有书籍都是经过版权方授权的，确保我们不会侵犯任何知识产权。我们将确保我们遵守相关的版权法律法规，并在与版权方达成授权合作之前进行充分的调查和确认。

用户数据和隐私：我们将遵守相关的数据隐私法律法规，确保用户数据不会被泄露、滥用、窃取或者非法出售。我们将采取安全措施来保护用户数据的安全，包括加密存储、访问控制、防止黑客攻击等措施。

广告合规：如果在微信小程序中展示广告，我们将遵守相关的广告法律法规，包括但不限于防止虚假广告、侵权广告和欺骗性广告等，确保广告的真实性和合规性。

综上所述，本项目符合相关的法律法规，并且采取了相应的措施来保证项目的合法性和合规性。我们将持续关注相关法律法规的变化，并做出相应的调整。

## 2.2 方案选择

2.2.1 所使用的语言

1. Java

Java是一种跨平台的编程语言，可以在不同操作系统和硬件平台上运行，这使得开发人员可以更加灵活地开发和部署应用程序。同时，Java拥有强大的生态系统和丰富的开发工具，使得开发人员可以更加高效地开发和维护Java应用程序。此外，Java具有面向对象的编程特点，可以提高代码的可重用性和可维护性，从而降低开发和维护成本。综合来看，选择Java作为后端开发语言是一个可靠的选择，可以提高系统的稳定性和可维护性[4]。

2. SQL

SQL是一种用于操作关系数据库的编程语言，其主要职责包括创建、更新以及查询数据库中的数据。相较于其他数据管理系统，SQL具有以下优点：

易于掌握：SQL的语法简单明了，易于理解和学习。

标准化程度高：SQL作为一种标准的编程语言，可跨不同的数据库管理系统使用。

高效率：SQL能有效处理大量数据，且在大型企业级应用中，SQL展现出了优秀的性能。

可扩展性：SQL能够很方便地扩展和维护，支持用户自定义存储过程、函数等。

安全性：SQL有完善的权限管理机制，可以对数据的访问和修改进行精细化控制。

综合来看，选择SQL作为数据库管理系统的语言是一种可靠的选择，可以提高系统的稳定性和可维护性。

2.2.2 所涉及到的框架与主要技术

1. SpringBoot

Spring Boot是基于Spring框架的一个轻量级应用开发框架，它为构建基于Spring的应用程序提供了许多方便快捷的工具和特性。Spring Boot能使应用程序的启动和运行过程迅速且高效，降低了开发者的等待时间。它通过自动化配置的方式，大大减少了开发者需要手动进行的配置工作，使开发者可以更专注于应用程序的业务逻辑部分。此外，Spring Boot还提供了一系列简洁强大的API和开发工具，帮助开发者更便捷地创建应用程序，并支持各种扩展，如新增库和框架、自定义配置和整合其他应用程序等。由于Spring Boot具备丰富的生态系统，可以轻松地与其他Spring框架和第三方库进行集成，因此选择Spring Boot作为应用开发框架是一种理想的选择，它可以提升开发效率并增强应用程序的稳定性。

2. Vue

Vue.js是一款流行的JavaScript框架，用于构建现代化的Web应用程序[5]。它具有轻量级、高效、灵活和易于学习的特点，可以快速地构建交互式用户界面。Vue.js采用组件化开发的方式，使得开发人员可以将应用程序拆分成多个小组件，便于管理和维护。此外，Vue.js还提供了丰富的指令和API，使得开发人员可以轻松地操作DOM和进行数据绑定，提高了开发效率。由于其轻量级和易于学习的特点，Vue.js在Web开发领域得到了广泛应用，并且拥有庞大的开发社区和丰富的第三方库支持，使得选择Vue.js作为应用程序的前端开发框架是一种可靠的选择，可以提高开发效率和应用程序的稳定性。

3. EpubLib

Epublib是一个Java开源库，用于处理EPUB电子书格式[6]。它提供了对EPUB文件格式的读取、编辑和创建功能，可以轻松处理EPUB电子书的元数据、章节结构和内容，方便地生成高质量的电子书文件。在本项目中，使用了Epublib框架提取本地电子书文件并进行处理，以提供高质量的阅读体验。

2.2.3 所使用的主要开发工具

1. IntelliJ IDEA

Idea是一种广泛使用的Java集成开发环境（IDE），它由JetBrains公司开发。Idea提供了强大的代码编辑、调试、版本控制和构建工具，以提高Java开发效率。Idea支持多种编程语言和框架，包括Java、Kotlin、Spring Boot等。在本项目中，使用Idea作为主要的开发工具，利用其丰富的插件和集成开发环境的优势来提高开发效率。

2. HbuilderX

HbuilderX是一个专业的跨平台前端开发集成开发环境，提供了代码编辑、调试、构建、打包等一系列开发工具。它支持多种前端框架，如Vue、React、Angular等，还提供了丰富的插件和模板，方便开发者快速搭建前端项目并进行开发。同时，HbuilderX也集成了多种移动端开发工具，如uni-app、cordova、hybrid等，支持开发多种移动端应用。

2.2.4 所使用的数据库

1. Mysql

MySQL是一种开源的关系型数据库管理系统，其具有高度的可靠性、可扩展性和可定制性。MySQL采用了标准SQL语言，具有较好的性能和安全性，广泛应用于各种规模的企业和网站。MySQL支持多种操作系统和开发语言，如Linux、Windows、Java和PHP等，可以方便地集成到现有的应用程序中。同时，MySQL拥有活跃的社区和丰富的文档，提供了大量的技术支持和资源，使得开发人员可以轻松地进行数据库开发和管理[7]。

2. MongoDB

MongoDB是一种基于文档的NoSQL数据库管理系统，以高性能、高可用性、易扩展性和灵活性而著称。它可以存储半结构化和非结构化数据，并支持丰富的查询和聚合操作。MongoDB使用BSON（二进制JSON）格式存储数据，可以轻松地扩展和水平扩展，使其成为大型和高可用性应用程序的理想选择。

# 第3章 需求分析

## 3.1 问题分析

随着移动互联网和智能手机的普及，人们越来越习惯于在手机上阅读各种内容，包括新闻、文章、小说等。然而，目前市场上的移动阅读应用大多存在以下问题：内容质量参差不齐，广告过多干扰阅读，付费模式不合理限制用户选择，界面设计不美观不友好等，并且面向程序员的阅读应用也很少。

## 3.2 设计原理

1. 简洁性

系统应该尽可能简洁明了，用户可以方便地找到所需的功能和信息，减少用户的学习成本和使用难度。

2. 高效性

系统应该尽可能地提高阅读效率和速度，快速地获取和处理书籍信息，并提供高质量的阅读体验。

3. 可扩展性

系统应该具备良好的可扩展性，方便后续添加新的功能和服务，并满足用户不断变化的需求。

4. 安全性

系统应该具备可靠的安全性保障，确保用户数据的安全性和隐私保护。

5. 用户体验优化

系统应该充分考虑用户体验，根据用户反馈和使用情况进行优化，提高用户满意度和留存率。

## 3.3 总体需求分析

随着互联网的普及，线上阅读已融入我们的日常生活。相比传统的纸质书籍，线上阅读为用户提供了更便捷的选择和获取书籍的途径，同时，丰富的图书资源可以满足用户的多元化需求。网络阅读器使得众多喜欢的书籍可以轻松容纳于手机中，并保存书签、阅读想法等数据，方便用户随时查阅。一款标准的线上书城项目应能帮助用户查找和挑选喜爱的书籍，提供书单列表、书籍更新状态、书签等书籍管理功能，以及在线阅读的H5阅读器。

微书项目包括前端和后端两部分，实现前后端分离。前端主要负责提供视觉舒适、易于使用的界面，并利用后端提供的接口显示数据。后端包含数据库设计、数据源和API接口三个部分。

微书应该具有如下几个重要的功能：

（一）用户能查阅书籍详情，并有选项将喜欢的书籍加入到个人书架。

（二）首页推荐：书城页面上会展示推荐专题、书籍分类和热门推荐等信息，帮助用户快速挑选喜欢的书籍。此外，搜索功能可以帮助用户通过书名直接找到书籍。

（三）用户能查阅其他用户对书籍的评价，并有权自己发表评论。

（四）个人书架页面方便用户管理自己的书籍，提供基本的搜索功能，便于用户快速找到想要阅读的书籍。

（五）微书应具备基本的用户管理功能，包括登录、查看个人信息等。

（六）阅读器：提供在线阅读功能，以方便用户在微信小程序中阅读自己喜欢的书籍。

（七）社交功能：用户可以与其他用户交流阅读体验、书籍推荐等，以增强用户的参与感和用户黏性。

## 3.4 功能需求分析

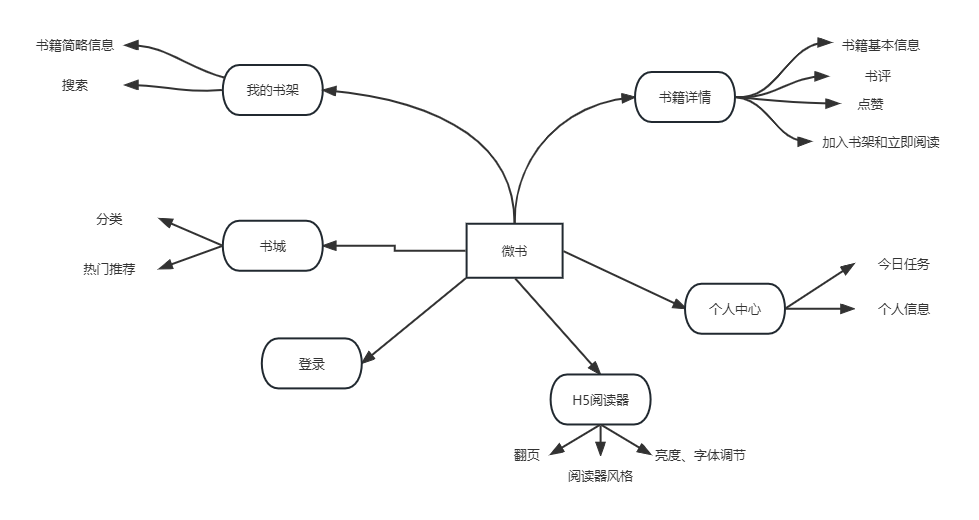
根据上面的分析，本系统分成六个主要的模块: 我的书架、书籍详情、H5阅读器、个人中心、书城和登录。

图3-1 功能模块图

如图3-1所示，是整个系统的功能模块图，每个模块说明如下：

1. 我的书架

该模块是微书的核心功能之一，旨在帮助用户更好地管理自己的阅读内容。在该模块中，用户可以展示已加入的书籍，支持用户进行筛选和搜索，便于用户快速找到自己想要阅读的书籍。

1. 书籍详情

在微书中，用户可以查看书籍的详细信息，包括书籍的名称、封面、作者、简介等内容。此外，该模块还提供了添加至书架和立即阅读按钮，让用户可以轻松添加自己感兴趣的书籍，并随时开始阅读。

1. H5阅读器

该模块是微书的在线阅读工具，提供左右滑动翻页、查看目录、切换风格、更改字体等多种功能，让用户可以根据自己的阅读喜好进行自定义设置。此外，H5阅读器还支持多种阅读模式，如夜间模式、护眼模式等，方便用户在不同的环境下进行阅读。

1. 个人中心

该模块是用户管理自己账户信息的主要入口，用户可以在该模块中查看自己的个人信息，包括已阅读时间、连续登录时间、总阅读书籍数量和每日任务等内容。

1. 登录

微书具备独立的用户管理系统。虽然在小程序中可以获取用户的一些基础信息，但这些信息不足以满足需求。因此，我们基于微信小程序注册用户为微书用户，并使用自有服务器单独管理用户的登录状态，例如登录时间和日期。

1. 书城

该模块允许用户浏览当前热门书籍和各种分类的书籍，使用户可以方便地选择自己感兴趣的书籍并加入到书架中。

## 3.5 数据分析

这个项目需要存储大量的文本信息和少量的图片。对于文本数据的处理，MongoDB提供了高效的速度。而且，MongoDB有着与MySQL类似的子文档构念，能够方便地实现不同文档之间的关系模型以及联合查询。因此，在开始微书应用的开发之前，选择使用MongoDB作为书籍信息存储的数据库，用于保存书籍的基础信息以及章节数据。另一方面，对于数据量较小的用户信息、评论数据以及排行榜信息，可以使用MySQL来存储这些信息。此外，Redis在实现排序方面非常方便，因此使用Redis来存储排行榜信息。

3.5.1 数据流分析

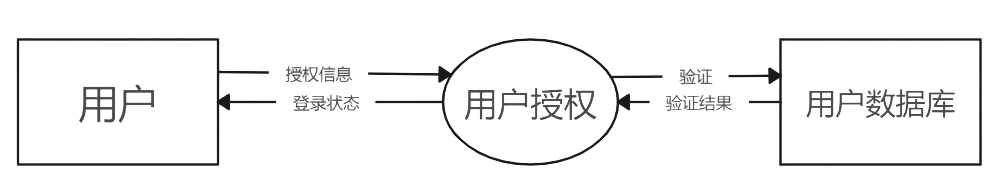
用户授权登录的数据流图，如图3-3所示：

图3-3 用户授权数据流图

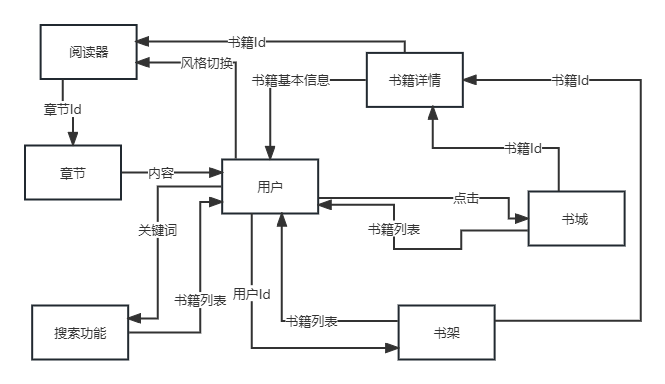
整个系统的数据流图，如图3-4所示：

图3-4 系统的数据流图

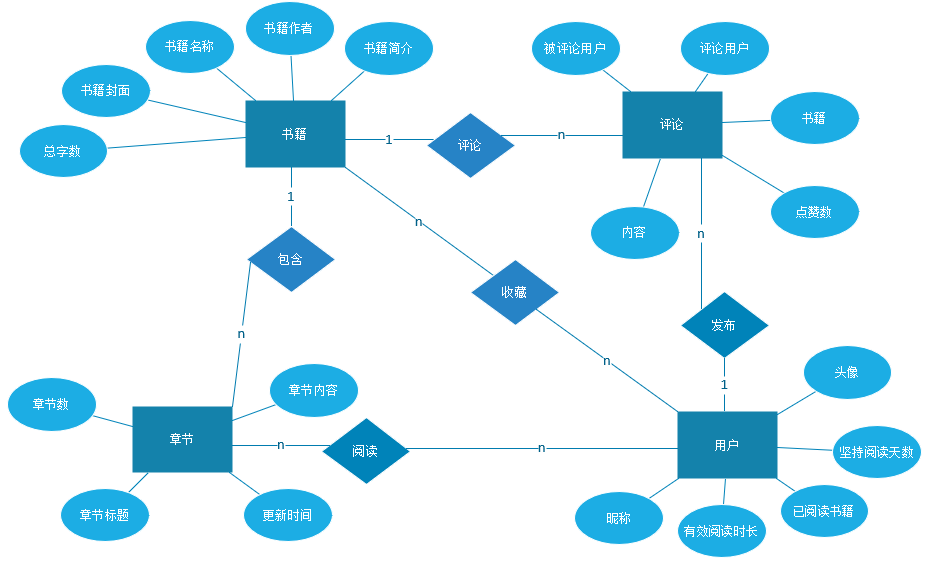
3.5.2 E-R图

图3-5 ER图

如图3-5所示，本实体-关系（ER）图中包含以下四个实体和相关的关系：

实体:

1. 书籍：这个实体存储一本书的基本信息，其属性包括书名、作者、书籍封面等。

2. 章节：书籍的组成部分，每个章节都有标题和内容等属性。每个章节都属于一本书籍。

3. 用户：使用该系统进行阅读和评论书籍的人。有头像、阅读时长、昵称等属性。

4. 评论：用户对书籍的评价和反馈。评论包含评论内容、评论日期和评分等属性。每条评论都属于一位用户和一本书籍。

关系:

1. 包含：每本“书籍”都包含着多个“章节”。这是一种一对多的关系，表明每个章节都只能属于一本书，而每本书可以包含多个章节。

2. 评论：每本“书籍”可以获得多条“评论”。这也是一种一对多的关系，表示每条评论只能针对一本书，而每本书可以有多条评论。

3. 发布：每名“用户”可以发表多条“评论”，这代表了一种一对多的关联，即每条评论都有其唯一的发布者，而每个用户则可以发布许多评论。

4. 收藏：每名“用户”可以收藏多本“书籍”。这展现出一种多对多的联系，因为一名用户有可能收藏多本书籍，同时一本书籍也能被多名用户收藏。

5. 阅读：每名“用户”可以阅读多个“章节”。这同样是一种多对多的关系，因为一名用户可以阅读多个章节，而一个章节也可以被多名用户阅读。

3.5.3 数据字典

表3-1 书籍实体的数据字典

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否可为空 | 描述 | 主键/外键 |
| 书籍Id | 字符串 | 64 | 否 | 书籍的唯一标识符 | 主键 |
| 书名 | 字符串 | 64 | 否 | 书籍的名称 |  |
| 作者 | 字符串 | 64 | 否 | 书籍的作者 |  |
| 书籍封面 | 字符串 | 64 | 否 | 书籍的封面图片 |  |
| 书籍简介 | 字符串 | 255 | 否 | 书籍的简介 |  |
| 总字数 | 整数 | 不适用 | 否 | 书籍的字数 |  |

表格3-1展示了书籍实体的相关信息。其中书籍Id是唯一标识符，用于标识每一本书籍。书名、作者、书籍封面和书籍简介使用字符串进行存储。总字数是一个整数类型，用于记录这本书的总的字数。这个数据字典的设计可以为用户提供书籍相关的基本信息，为程序提供更好的阅读体验。

表3-2 章节实体的数据字典

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否可为空 | 描述 | 主键/外键 |
| 章节Id | 字符串 | 64 | 否 | 章节的唯一标识符 | 主键 |
| 书籍Id | 字符串 | 64 | 否 | 章节对应的书籍的标识符 | 外键 |
| 内容 | 字符串 | 65535 | 否 | 章节的具体文字内容 |  |
| 标题 | 字符串 | 64 | 否 | 章节的标题 |  |
| 章节数 | 整数 | 不适用 | 否 | 标明当前是第几章 |  |
| 更新时间 | 日期 | 不适用 | 否 | 何时更新 |  |

表格3-2展示了实体"章节"的相关信息。其中，“章节ID”是章节的唯一标识符，为主键；“书籍ID”是外键，表示章节对应的书籍，与书籍表中的“书籍ID”相连接。“内容”和“标题”字段分别表示章节的具体文字内容和标题。“章节数”字段表示当前是第几章。“更新时间”表示章节的更新时间。

表3-3 评论实体的数据字典

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否可为空 | 描述 | 主键/外键 |
| 评论Id | 字符串 | 64 | 否 | 评论的唯一标识符 | 主键 |
| 书籍Id | 字符串 | 64 | 否 | 评论对应的书籍的标识符 | 外键 |
| 评论用户Id | 字符串 | 64 | 否 | 发布评论的用户的标识符 | 外键 |
| 被评论用户Id | 字符串 | 64 | 否 | 被评论的用户的标识符 | 外键 |
| 内容 | 字符串 | 65535 | 否 | 评论的具体文字内容 |  |
| 点赞数 | 整数 | 不适用 | 否 | 评论的点赞数 |  |

表格3-3展示了实体"评论"的相关信息。其中，“评论Id”是评论的唯一标识符，为主键。“书籍Id”是外键，表示评论所对应的书籍，与书籍表中的“书籍Id”关联。同时，“评论用户Id”和“被评论用户Id”分别表示发表评论的用户和被评论的用户，与用户表中的“用户Id”关联。“内容”字段表示评论的具体文字内容。“点赞数”表示评论获得的点赞数量。

表3-4 用户实体的数据字典

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否可为空 | 描述 | 主键/外键 |
| 用户Id | 字符串 | 64 | 否 | 用户的唯一标识符 | 主键 |
| 头像 | 字符串 | 64 | 否 | 用户的头像url |  |
| 昵称 | 字符串 | 64 | 否 | 用户的昵称 |  |
| 坚持阅读天数 | 整数 | 不适用 | 否 | 用户已经连续阅读的天数 |  |
| 已阅读书籍 | 整数 | 不适用 | 否 | 用户已经阅读过的书籍数量 |  |
| 有效阅读时长 | 整数 | 不适用 | 否 | 用户阅读的总时长 |  |

表格3-4展示了实体"用户"的相关信息。其中，“用户Id”是用户的唯一标识符，为主键。“头像”是用户的头像url，“昵称”则表示用户的昵称。“坚持阅读天数”表示用户已经连续阅读的天数，“已阅读书籍”反映的是用户所读过的书籍数，而"有效阅读时长"则代表了用户的阅读总时间。

## 3.6 性能分析

本项目的性能分析主要通过四个方面进行，分别是响应速度、负载能力、可扩展性和稳定性，通过这些维度的分析，可以得出微书的性能优点和不足，以及可能的改进方向[8]。

1. 响应速度

涉及到用户操作的反馈时间，如页面加载速度，书籍打开速度，搜索结果返回速度等。通过记录和分析这些操作的时间，可以对微书的响应速度进行评价。

1. 负载能力

通过模拟高并发场景，例如，同时有大量用户在线阅读、搜索、评论等，看系统的表现如何，是否出现延迟或崩溃等情况。

1. 可扩展性

随着书籍数量和用户数量的增加，系统是否能够顺利处理更大的数据量和更高的请求频率。这通常涉及到数据库的设计和选择，以及后端服务的架构设计。

1. 稳定性

在持续运行一段时间后，微书的稳定性如何？是否存在内存泄漏或者资源占用过高的问题？是否有频繁的服务中断或者异常现象？

# 第4章 设计论述

## 4.1 概要设计

概要设计是软件开发过程中的一个阶段，它的目标是通过对系统进行高层次的设计来确定系统的整体架构和组成部分，以满足系统需求。

1. 系统整体结构

本系统基于微信小程序开发，包括六个模块：书架、书城、个人中心、H5 阅读器、书籍详情和授权登录模块。整个系统采用前后端分离的架构，前端使用微信小程序技术，后端使用 Java 的 Spring Boot 框架。数据库采用 MongoDB 和MySQL存储。

1. 模块划分和接口设计

本系统包含六个模块，每个模块的功能如下：

书架：展示用户已经添加的书籍，支持阅读、搜索等功能。

书城：展示系统中所有的书籍，支持搜索、查看详情等功能。

个人中心：展示用户的个人信息，包括用户名、头像、消息等。

H5 阅读器：支持在线阅读电子书，包括翻页、目录、字体设置等功能。

书籍详情：展示一本书的详情信息。

授权登录：用户授权并选择头像和昵称

1. 数据库设计

该系统使用 MongoDB 和 MySQL 数据库进行数据存储，其包含以下的数据集合：

用户数据：涵盖用户的昵称、头像等元素。

书本数据：涵盖书名、作者、封面图、简介等元素。

用户图书数据：记录用户已将哪些书籍添加到书架上。

评论信息：保存用户的评论

1. 技术方案和开发环境

本系统采用微信小程序和 Java 的 Spring Boot 框架进行开发。前端使用微信小程序原生开发，后端使用 Spring Boot 提供的 Restful API 接口实现数据交互。开发工具包括微信开发者工具、IntelliJ IDEA 等。

4.1.1 总体功能模块设计

该系统被划分为以下六个模块：授权登录、书城、个人中心、H5阅读器、书籍详情和我的书架模块。系统总体结构图如下：

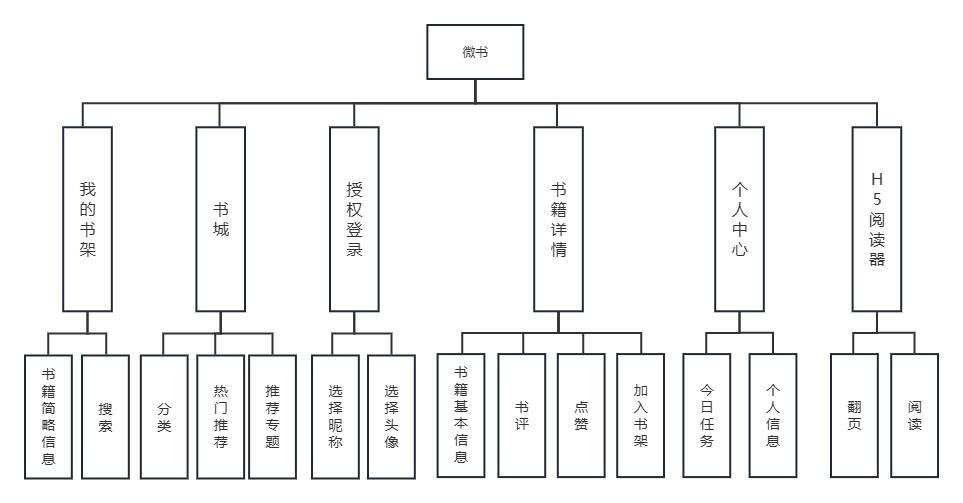


图4-1 总体结构图

如图4-1所示，是本系统的总体结构图，分为六个主要模块，各个模块的具体说明如下：

我的书架：这个模块是微书的核心功能之一，旨在帮助用户更好地管理自己的阅读内容。在后端，通过数据库管理系统（例如MySQL）来存储用户书架上的书籍。在前端，会使用Vue来构建用户界面，允许用户在书架上展示已加入的书籍，支持用户进行筛选和搜索。

书籍详情：在这个模块中，用户可以浏览书籍的全面信息，包括书名、封面、作家和概述等。同时，此模块内包含“收藏到书架”和“马上阅读”的操作，用户可以借此方便地将感兴趣的书籍收藏，并能随心所欲开始阅读。另外，设立了评论交流区，便于用户与其他阅读者进行互动。

H5阅读器：这个模块是微书的在线阅读工具。在前端，使用JavaScript和CSS来实现滑动翻页、查看目录、切换风格、更改字体等功能。同时，提供夜间模式、护眼模式等阅读模式，以适应用户在不同环境下的阅读需求。

个人中心：这个模块是用户管理自己账户信息的主要入口。该系统在后端使用数据库来存储和管理用户的个人信息，包括已阅读时间、连续登录时间、总阅读书籍数量和每日任务等信息。在前端，使用Vue来设计用户界面。

书城模块：这个模块是图书展示的主要区域。在后端，通过数据库存储图书信息，同时开发算法来实现热门推荐功能。在前端，设计分类功能，让用户可以根据自己的兴趣找到对应的书籍。同时，设置推荐专题功能，向用户推荐当前计算机的相关热点。

授权登录：这个模块主要负责用户的登录过程，通过微信的API获取用户信息并生成登录凭证。同时，也会生成一个唯一的token来保证用户登录状态的安全性，这个token会被绑定到用户的openid并存储到Redis数据库中。

4.1.2 系统用例设计

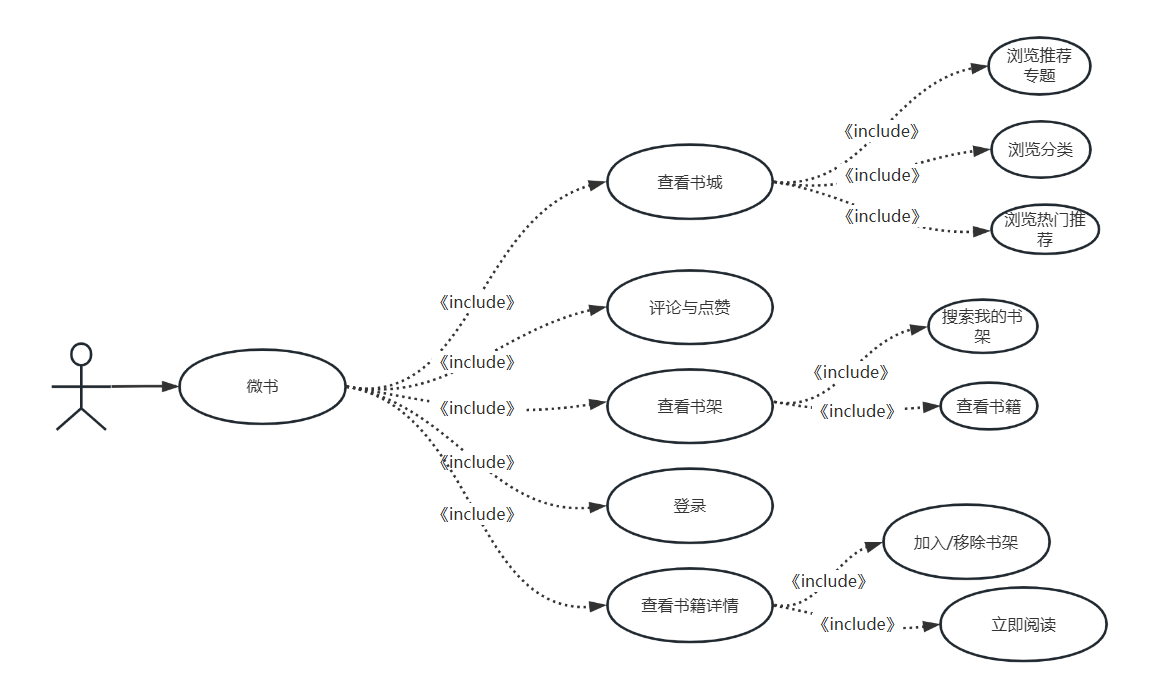


图4-2 系统用例图

如图4-2，本系统用例图展示了系统的一些交互关系。其中，用户作为系统的主要参与者，可以通过微信小程序登录系统，使用书架模块管理已添加的书籍，使用书城模块浏览所有书籍并进行搜索和筛选，使用个人中心模块查看个人信息和已购买的书籍，使用H5阅读器模块在线阅读电子书，以及进行评论书籍和点赞他人评论。本系统致力于为用户提供便捷、舒适的阅读体验，帮助用户轻松阅读自己想看的书籍，并支持书籍评论和热门推荐功能，帮助用户在海量书籍中找到自己喜欢的作品。

4.1.3 系统流程图设计

当用户扫描二维码访问微书时，系统首先确认登录状态。如果用户未登录，系统会提示用户授权并登录。用户选择昵称和头像后，微书获取用户的微信openid，从微信服务器获取并保存用户信息，用户则被引导至“我的书架”界面。

在书架界面，系统会检查用户是否已拥有书籍。已有书籍的用户可以通过点击查看详情，再点击即可使用H5阅读器阅读。否则，用户将被引导至书城以添加新的书籍。

在书籍详情页面，用户可以将书籍添加至自己的书架，或者从书架中移除。用户还可对书籍进行评论或者回应其他用户的评论。

阅读时，系统会记录用户的开始和结束阅读时间。阅读器中，用户可以更换阅读模式、调整亮度和字体，还能浏览书籍的所有章节。

在个人中心页面，用户的阅读时间、阅读天数和拥有的书籍等信息将展示出来。个人消息栏会展示其他用户对自己的回复，用户也可以直接回复他们。

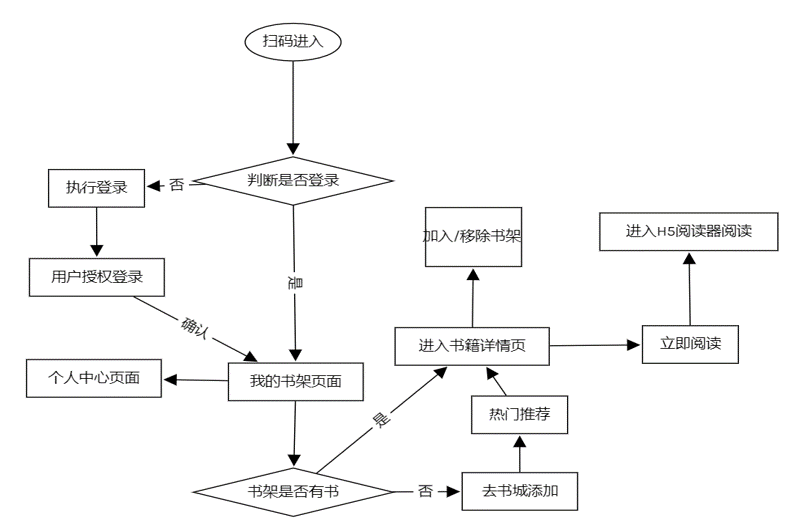
书城将呈现最新的专题推荐、不同的书籍类别以及热门书目推荐，以便于用户更有效地挑选出他们喜欢的阅读材料。下面是微书的操作流程图：

图4-3 系统流程图

如图4-3所示，用户在扫码进入系统后，首先进行登录状态的判断。若已登录，则直接进入“我的书架”页面；否则，用户需要进行授权登录后方可访问书架。在书架页面，系统会判断书架是否含有书籍。如书籍为空，系统会引导用户前往“书城”添加书籍。否则，用户可以选择点击书籍，进入书籍的详情页。在详情页，用户既可以选择发表评论，也可以选择使用H5阅读器开始阅读。

## 数据设计

4.2.1 数据库逻辑结构设计

在数据库设计中，应有八个主要的表：Books, Chapter, User, Bookofuser, Comments, Userlogin, userreading, 和 email。这些表之间的关系如下：

Books和Chapter：每本书（Books）可以有多个章节（Chapter），通过Books表的\_id字段和Chapter表的bookid字段相连接。

User和Bookofuser：每个用户（User）可以阅读多本书，这个关系通过User表的openid字段和Bookofuser表的userid字段相连接。同时，Bookofuser表的bookid字段也与Books表的\_id字段相连接，代表了用户正在阅读的书。

User和Comments：每个用户（User）可以发表多条评论，这个关系通过User表的openid字段和Comments表的userid字段相连接。同时，Comments表的bookid字段也与Books表的\_id字段相连接，代表了用户评论的书。

User和Userlogin：每个用户（User）可以有多次登录记录，这个关系通过User表的openid字段和Userlogin表的userid字段相连接。

User和userreading：每个用户（User）可以有多次阅读记录，这个关系通过User表的openid字段和userreading表的userid字段相连接。同时，userreading表的bookid字段也与Books表的\_id字段相连接，代表了用户正在阅读的书。

User和email：每个用户（User）可以收到多封电子邮件，这个关系通过User表的openid字段和email表的userid字段相连接。同时，email表的bookid和commentid字段也与Books表的\_id字段和Comments表的commentid字段相连接，代表了电子邮件相关的书和评论。

最后应使用Redis来存储用户对评论的点赞记录（Like）。对于至少有一个点赞的评论，使用一个Redis Set来存储所有点赞的用户ID，通过评论ID (commentid)来作为键（Key）。

4.2.2 数据表设计

表3-1 书籍

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 描述 |
| \_id | ObjectId | id，MongoDB自动生成 |
| factionName | String | 书籍名称 |
| des | String | 书籍简介 |
| headerImage | String | 书籍封面地址 |
| author | String | 书籍作者 |
| updateTime | Date | 更新时间 |
| newest | Int32 | 最新章节数 |

表格3-1展示了Book表。其中\_id是MongoDB自动生成的唯一标识符，用于标识每一本书籍。书籍名称、书籍简介、书籍封面地址、书籍作者和更新时间都是用字符串和日期格式进行存储。最新章节数是一个整数类型，用于记录书籍最新的章节数。这个表格的设计可以为用户提供书籍相关的基本信息，为程序提供更好的阅读体验。

表3-2 Chapter

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 描述 |
| \_id | ObjectId | id，MongoDB自动生成 |
| num | Int32 | 章节的序号 |
| name | String | 章节标题 |
| content | String | 章节内容 |

续表3-2 Chapter

|  |  |  |
| --- | --- | --- |
| bookid | String | 章节来源 |
| create\_time | Date | 创建时间 |

表格3-2的用途是存储一本书的不同章节内容，并记录它们的基本信息。具体来说，每个章节被分配了一个序号和标题，通过一个唯一的MongoDB自动生成的ID来标识，同时还包括它们的内容和所属书籍的ID。此外，表中还记录了每个章节的创建时间。这样，通过访问这个表，可以查询、更新和管理一本书的不同章节内容，包括它们的基本信息和创建时间。

表3-3 Bookofuser

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 描述 |
| userid | varchar | 用户id |
| bookid | varchar | 书籍id |
| hasRead | int | 正在阅读的章数 |

表格3-3的用途是记录用户与书籍之间的关系，包括用户正在阅读的书籍和章节数。具体来说，表中的每一行记录了一个用户正在阅读的书籍以及已经阅读的章节数，以及该书籍的ID和用户的ID。通过这个表，可以轻松地跟踪用户正在阅读的书籍，并记录他们的阅读进度。同时，这个表也可以作为一个关系表，将用户和书籍连接在一起，方便查询和管理。例如，可以使用这个表来查询某个用户正在阅读的所有书籍，并了解他们的阅读进度。

表3-4 Comments

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 描述 |
| commentid | varchar | 评论id |
| bookid | varchar | 书籍id |
| userid | varchar | 用户id |
| nickname | varchar | 昵称 |
| reply | varchar | 回复 |
| content | blob | 评论内容 |
| avatar | varchar | 头像图片地址 |
| time | datetime | 评论时间 |
| father | varchar | 父评论的id |
| likenum | int | 点赞数 |
| length | int | 评论长度 |
| isOpenMoreComment | tinyint | 能否评论 |
| isLike | int | 是否点赞 |

表格3-4的用途是记录书籍评论的相关信息，通过这个表，可以轻松地管理和查询书籍的评论信息，例如获取某个书籍的所有评论，查看某个评论的回复和点赞数，以及添加新的评论和点赞信息。同时，这个表也可以用于用户评论和互动的社交功能，例如在评论中添加回复和点赞等。

表3-5 User

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 描述 |
| openid | varchar | 用户id |
| avatar | varchar | 头像图片地址 |
| nickname | varchar | 昵称 |
| hasReadTime | int | 有效阅读时长 |
| continueReadDay | int | 坚持阅读天数 |
| hasBookNum | String | 已阅读的书籍 |

表格3-5的用途是记录用户的基本信息和阅读数据，通过这个表，可以轻松地管理和查询用户的基本信息和阅读数据，例如查询某个用户的头像和昵称，或者获取用户的有效阅读时长和阅读天数等信息。

表3-6 Userlogin

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 描述 |
| userid | varchar | 用户id |
| logindate | date | 登录日期 |
| logintime | datetime | 登录时间 |

表格3-6的用途是记录用户的登录信息，通过这个表，可以轻松地跟踪用户的登录行为，了解用户的活跃程度以及在不同时间段的登录情况。同时，这个表还可以用于用户行为分析，例如分析用户在不同时间段的活跃程度，以及预测用户未来的登录行为。

表3-7 userreading

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 描述 |
| id | varchar | 唯一标识 |
| userid | varchar | 用户id |
| bookid | varchar | 书籍id |
| start | timestamp | 开始时间 |
| end | timestamp | 结束时间 |

表格3-7的用途是记录用户的阅读信息，包括用户ID、书籍ID、阅读开始时间和结束时间。通过这个表，可以跟踪和管理用户的阅读行为，例如了解用户的阅读偏好、统计用户的阅读时长等等。

表3-8 email

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 描述 |
| id | varchar | 唯一标识 |
| userid | varchar | 用户id |
| bookid | varchar | 书籍id |
| commentid | varchar | 评论id |
| isRead | tinyint | 是否已读 |
| bgImgUrl | varchar | 背景图 |
| DigestAuthorName | varchar | 发送消息的作者 |
| DigestAuthorImg | varchar | 发送消息作者头像 |
| DigestDes | varchar | 内容 |
| title | varchar | 标题 |
| contentButtonText | varchar | 按钮显示文字 |
| contentButtonUrl | varchar | 点击后的URL |
| contentButtonEventHandler | varchar | 触发的方法 |

如表格3-8所示，通过这个表，系统可以实现向指定用户发送通知消息，通知消息可以包含关联的书籍ID或评论ID，并提供点击按钮以便用户进行相应的操作，一个用户评论另一个用户的通知也会发送到这里。

Like, 存储在Redis中，数据结构类型： Set ，键（Key）： 评论ID (commentid) ，值（Value）： 用户ID的集合 。这样的设计帮助追踪哪些用户已经对某个评论做出了点赞。对于至少有一个点赞的评论，都用一个Redis Set来存储所有点赞的用户ID。

## 4.3 系统详细设计与实现

4.3.1 系统登录模块

1. 设计描述

当用户通过扫码进入小程序时，会出现一个弹窗，让用户选择自己的昵称和头像。一旦用户完成选择并授权后，页面会自动跳转到主页面，也就是“我的书架”界面。如果用户未授权，则不会跳转到主页面。在用户未授权之前，其他界面会被置于灰色状态，无法进行点击操作。

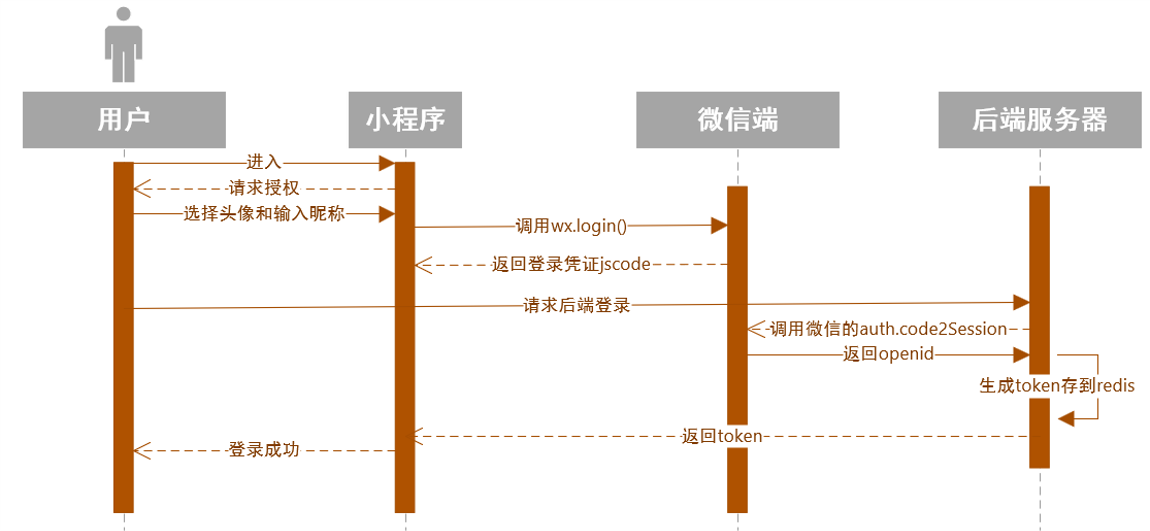
时序图如图4-4所示：

图4-4 登录时序图

2. 登录页面

登录授权页面如4-5和图4-6所示：

图4-5 登录授权页面

图4-6 授权后的登录页面

3. 登录逻辑实现

本项目采用微信提供的登录方式。考虑到小程序是基于微信的，因此这种方式的安全性更高。登录的流程如下：

第1步，用户点击微信提供的头像选择按钮和昵称输入框（<button open-type="chooseAvatar" @chooseavatar="chooseavatar"> 和 <input type="nickname" />）。微信小程序会向用户发送授权请求，要求获取用户微信账号的基本资料，如头像、昵称等。一旦用户同意授权，微信小程序便得到了相关权限，可以利用微信提供的API获取用户信息。

第2步，微信小程序的前端会调用微信提供的API wx.login() 来获取用户的微信登录凭证，即jscode。这个登录凭证是由微信服务器发出的，有效期很短，通常只有5分钟左右。因此，在获取到jscode后，需要尽快使用它来获取用户的openid和session\_key。

第3步，使用jscode调用微信的auth.code2Session接口可以获取用户的openid和session\_key。其中，openid是唯一的用户标识，可用于标识不同的用户，而session\_key是微信服务器用于加密用户数据的密钥。

第4步，为保证用户登录状态的安全性，需要在服务器端生成一个唯一的token，并将其与用户的openid绑定，然后将这个键值对存储到Redis数据库中。同时，需要设置token的过期时间，通常可以设置为几小时或几天。在本项目中，token过期时间设置为12小时，以防止token被长期滥用。当用户再次登录时，需要先检查token是否过期，如果过期了就需要重新生成一个token。

以上就是本项目登录的大致流程。通过这种方式，可以保证用户登录状态的安全性和可靠性，同时也提高了用户的使用体验和用户粘性。

4.3.2 我的书架模块

1. 设计描述

当用户打开我的书架页面时，他们可以使用两个主要的功能：展示收藏的书籍列表和搜索书籍。在展示书籍列表方面，我的书架页面提供了一个简洁易用的界面，让用户能够快速浏览他们的书籍收藏。每本书籍都会被列出来，并且包含书籍封面、书名、作者。用户可以通过单击书籍图像或标题来进入书籍详细信息页面，其中包括更多信息、书籍摘要和评论。还提供了一个搜索功能，让用户能够快速查找他们想要的书籍。搜索栏位于页面顶部，用户可以输入书名或者作者名进行搜索。在搜索结果页面中，用户可以看到所有符合搜索条件的书籍。

2. 我的书架页面

页面样式如图4-7所示：

图4-7 我的书架

3. 逻辑实现

首先，展示书籍列表的功能需要查询用户加入书架的书籍。为了实现这个功能，在 MySQL 数据库中创建了一个名为“bookofuser”的表，该表用于记录用户的书籍以及每本书的阅读章数。

当用户打开“我的书架”页面时，后端根据前端传来的userid在这个表中查询该用户所有的bookid。然后，根据bookid从 MongoDB 查询出书籍的具体信息，例如书籍的封面图片地址、书名以及最新章节等等。之后，结合之前查询出来的书籍id以及目前阅读章节等信息一起返回给前端。最后，前端利用 uniapp 的v-for循环将这些信息动态地展示在页面上，如图4-8所示，就是后端会返回的具体数据。

图4-8 具体的返回数据

书籍搜索的功能在前端利用正则表达式对用户输入的搜索词进行匹配，同时检查书籍标题和作者的信息。若存在符合的结果，相关书籍将被呈现；反之，若无符合的结果，这些书籍将被遮掩。为了方便用户查看搜索结果，在书籍搜索添加了高亮显示匹配项的功能。具体实现方式是在使用 JS 正则表达式的 exec() 方法时，获取每个匹配项的位置值，然后根据这个位置值在字符串的前后动态添加 <code> 和 </code> 标签来标记匹配项。同时，也需要在 CSS 中设置这种标签的样式。

以下是标记关键字并高亮的代码：

|  |
| --- |
| while (regExp.exec(tempStr) != null) {  console.log(++count);  lastIndex = regExp.lastIndex + 13 \* (count - 1);  //每次循环notChangeStr并非不变，而是多了<code></code>共计13个字符，所以为了保证后续循环中lastIndex的正确性应该将lastIndex自增13  leftStr = notChangeStr.substring(0, lastIndex - searchString.length);  rightStr = notChangeStr.substring(lastIndex);  notChangeStr = leftStr + '<code>' + searchString + '</code>' + rightStr;  } |

4.3.3 H5阅读器模块

1. 设计描述

H5阅读器模块是一种用于在线阅读的模块，它是在书籍详情页面中提供给用户的一个阅读界面，让用户可以方便地阅读书籍的内容。当用户在书籍详情页面点击“立即阅读”时，就会进入H5阅读器模块的页面。在H5阅读器模块中，用户可以选择向左或向右滑动屏幕来翻阅书籍的内容，这让用户可以方便地查看自己想要的章节和内容。此外，H5阅读器模块还提供了一些额外的功能，让用户可以更好地定制自己的阅读体验。用户可以点击屏幕中央的位置，出现阅读风格的调整选项。这些选项包括调整亮度、文字大小、文字风格等。通过这些选项，用户可以调整H5阅读器模块的外观，使其更符合自己的喜好和需要。另外，用户还可以选择想要阅读的章节。在H5阅读器模块中，用户可以轻松地查找到自己感兴趣的章节，并直接跳转到该章节的内容，方便快捷。

2. 页面

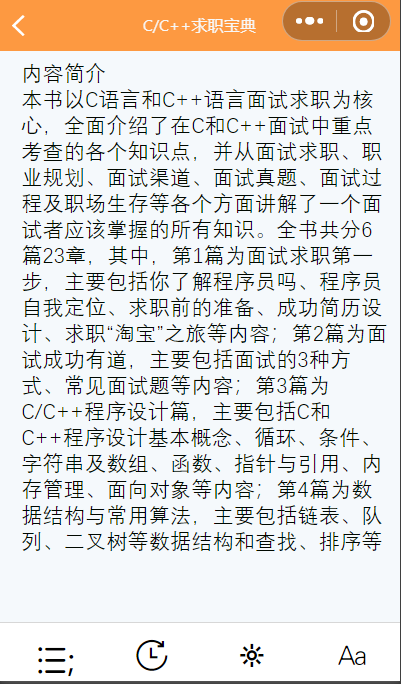
阅读页面和目录页面分别如图4-9和图4-10所示：

图4-9 阅读页面

图4-10 目录

3. 设计逻辑

（1）字符分页

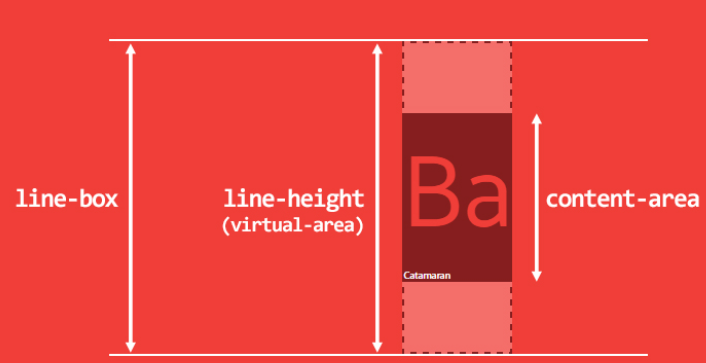
主要因素包括字体大小(font-size)、行高(line-height)以及字体类型(font-family)。字体大小决定了字体字符框的高度；而行高则决定了元素内基线间的最小间隔。字体类型的不同会对行高的默认值和字体大小产生影响。行高（line-heigth）对行框的布局产生影响。当用在一个块级元素上时，它定义了元素内部基线间的最小距离，而非最大距离。line-height 和 font-size 的计算值之间的差额均分为上下两部分，分别添加到文本行的上边和下边。这个最小的文本容器被称为行框（content-area）。当 line-height 设置为1或100%时，行框的高度就等同于内容区域的高度。如图4-11，展示了字体大小和行高的关系。

图4-11 行高和字体大小的关系

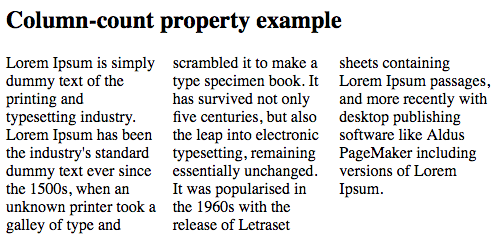
要实现精准的分页，需要先计算每页的字数，首先通过 JavaScript 获取章节内容的总字数，根据总字数和每页的字数计算最大分页数。接下来，通过 CSS3 中的 columns 属性，将文本内容分成多栏，每栏的高度就是一页的高度。设置 column-count 属性的值为最大分页数，这样就可以让文本内容自动分页了，如图4-12所示，是column-count的示例。需要注意的是，分页时还需要处理最后一页的内容，因为最后一页可能不足一整页。可以通过 JavaScript 计算最后一页的字数，然后根据实际情况调整布局，让最后一页的内容看起来更美观。

图4-12 column-count的示例

为了精确地对章节内容进行分页，需要遵循以下几个步骤：

1. 章节内容分割：首先利用正则表达式按照换行符"\n"将章节内容分割成多个文本段落。
2. 计算文本段落的最大行数：调用wx.getSystemInfo来获取设备的高度和宽度信息，并据此计算出每段文本能占据的最大行数，从而确定每段文本的实际高度。
3. 计算章节内容的总高度：把每段文本的实际高度累加，同时去掉空行的高度，这样就得到了章节内容的总高度。
4. 计算最大的分页数：把章节内容的总高度除以屏幕高度，向上取整并加一（如果章节内容的高度无法完整地填充整页，那么向上取整并加1可以保证最后一页由于内容不足而被漏掉），从而得到最大的分页数。

（2）检测用户手势

为了支持翻页，需要对用户手势进行检测。阅读器支持三种手势：1. 点击屏幕右部以前进一页，点击左部以后退一页；2. 向右滑动以前进一页，向左滑动以后退一页；3. 点击屏幕中心以激活阅读器控制栏。因此，在阅读组件上绑定了touchstart、touchmove和touchend事件。每次触摸事件会按序触发这些事件，并且每个事件都包含有touches数组，记录了触摸开始、结束和过程中的位置。

对于第一种翻页方式，如果touchmove没有触发，则视为用户通过点击屏幕进行翻页。在这种情况下，通过对比点击位置与屏幕两侧的距离，可以判断用户是向前翻页还是向后翻页。

对于第二种翻页方式，在touchmove事件触发的情况下，通过比较触摸开始时的x坐标和触摸过程中的当前x坐标，可以判断用户是向左滑动还是向右滑动。如果当前位置的x坐标大于开始时的x坐标，则视为向右滑动；反之，视为向左滑动。

第三种翻页方式，需要检测用户是否点击了屏幕中央。通过比较touchstart的触摸位置和touchend的触摸结束位置的坐标差，就可以判断触摸是否在屏幕中心点的容差范围内。如果在容差范围内，视为用户点击了屏幕中央。根据当前控制栏的显示状态，可以进行隐藏或显示控制栏的操作。

用户点击后，需要对当前显示的column进行滑动。为了实现滑动效果，需要将包含章节内容的HTML元素的定位设置为相对定位（relative）。这样，就可以通过改变元素的left属性来实现内容的左右滑动。然后，需要实现滑动的动画效果。首先，当手指触摸屏幕并左右滑动时，页面不会立即翻页，而是需要随手指滑动而移动。为了实现这一效果，可以在touchmove事件中比较手指滑动过程中的触摸点距离与touchstart起始位置的距离，并相应地增加或减少left值，使页面随手指移动。其次，当手指离开屏幕，即触发touchend事件时，就开始执行页面翻页。最后实现的翻页效果是一个简单的从左到右缓慢滑动到指定位置的动画。实现这一效果的原理很简单，就是使用JavaScript计时器，使left值从起始值缓慢变化到目标值，实现页面的平滑过渡。在计算目标值时，需要根据页面的宽度和滑动的方向进行计算。

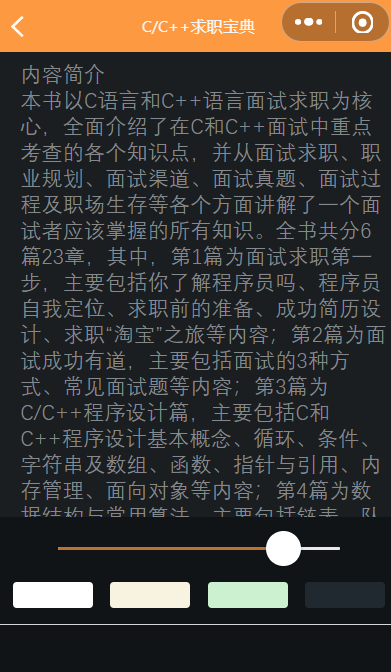
（3）阅读器风格切换、字体设置

图4-13 阅读器风格切换

如图4-13所示，阅读器的背景颜色从白色变成了黑色。

H5阅读器提供了四种独特的主题风格，分别为日间模式、夜间模式、护眼模式以及咖啡模式。用户可通过触碰屏幕中心来挑选自己偏好的阅读模式。每种主题都具备特定的属性值，例如页面背景、样式编号、工具栏底色和文字颜色等。这些属性值被储存在数据的某一属性中，并在用户挑选主题时被添加至对应的wxml元素上。字体切换的操作原理和主题切换一样，也是通过修改属性值来实现的，具体的属性值如下：

|  |
| --- |
| colorStyle: {content\_bg: '#f5f9fc', styleNum:1, slider\_bg: '#fd9941', slider\_none\_bg: '#dbdbdb', control\_bg: '#ffffff', control\_fontColor: '#fd9941'},  colorStyle: {content\_bg: '#f5f0da', styleNum:2, slider\_bg: '#a6832f', slider\_none\_bg: '#dbd6c3', control\_bg: '#f8f3e0', control\_fontColor: '#a6832f'},  colorStyle: {content\_bg: '#c0edc6', styleNum:3, slider\_bg: '#359112', slider\_none\_bg: '#a7ccab', control\_bg: '#ccf1d0', control\_fontColor: '#359112'},  colorStyle: {content\_bg: '#1a1e21', styleNum:4, slider\_bg: '#bb7333', slider\_none\_bg: '#212528', control\_bg: '#101417', control\_fontColor: '#bb7333'}} |

（4）目录实现

右上角的阅读时长是根据名为 "userreading" 的数据表实现的，通过累加每次阅读的时长来得到该书的总时长。以下是相应的 SQL 语句：SELECT SEC\_TO\_TIME(SUM(TIME\_TO\_SEC(TIMEDIFF(END, START)))) FROM userreading WHERE userid = #{userid} AND bookid = #{bookid}。

该书本的数据存储在 MongoDB 中，其中章节信息保存在名为 "chapter" 的数据表中。"chapter" 数据表包含了书本的 ID（bookid）和目录标题（name）。因此，可以轻松地从数据库中查询整个目录，代码如下：

|  |
| --- |
| 这是Java查询mongoDB代码：  Criteria c1 = Criteria.where("bookid").is(bookid);  Query query2 = new Query(c1);  List<Chapter> ts = mongoTemplate.find(query2, Chapter.class); |

然后点击其他目录即可直接跳转到该章的内容。

为防止一次查询出过多数据，我们需对章节执行分页操作。每次请求只返回当前章节的前后20章内容。当用户浏览至当前章节的后15章时，便会加载接下来的40章以确保流畅的翻页体验。分页功能的实现一般由后端接口完成。首先，后端会按照章节顺序从数据库中取出一本书的全部章节。然后，根据前端提供的当前章节，返回该章前后20章的内容。

4.3.4 书籍详情模块

1. 设计描述

书籍详情模块为读者提供了书籍的基本信息，包括书名、作者、书本封面、出版日期等等。这些信息为读者提供了一个初步了解书籍的概况，让他们可以快速判断书籍是否符合自己的需求。书籍详情模块还为读者提供了其他读者的书评和看法。这些书评是其他读者对书籍的评价和看法，让读者可以借鉴其他人的意见，帮助自己更好地了解这本书的内容和价值。同时，这也是一个互动平台，读者可以在书评下方留下自己的评论和意见，与其他读者交流和分享。书籍详情模块还提供了一个非常方便的功能：将书籍加入到自己的书架。这样，读者可以方便地管理自己的阅读列表，并随时随地查看自己的书籍收藏。同时，书籍详情模块还提供了立即进入阅读这本书的功能，让读者可以快速开始阅读。当用户点击“加入书架”和“从书架中移除”按钮时，将在数据库中进行相应的增删操作，并显示相应的添加或移除消息。在“精彩评论”栏的右侧，用户可以撰写关于该书的评论。每个评论右侧都有一个评论图标，用户可以点击该图标以对当前评论进行评论，如下图所示。此外，用户还可以通过点击点赞按钮来表示对评论的喜欢。如果已经点赞，则图标将变黑色，再次点击将取消点赞，图标将变回白色。

2. 页面

书籍详情页面和添加/移除书架和评论的页面如图4-14，图4-15，图4-16和图4-17所示：



图4-14 详情页面

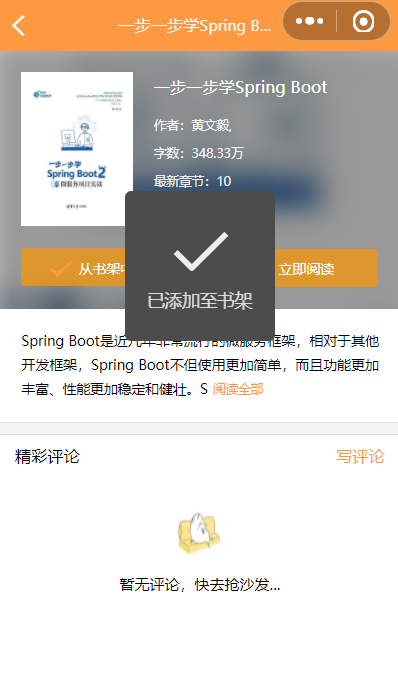
图4-15 添加到书架

图4-16 从书架移除

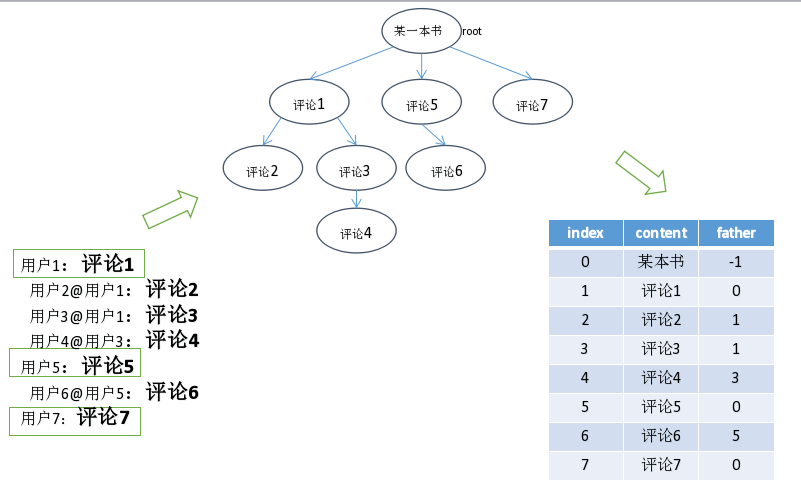
图4-17 评论界面

3. 设计逻辑

点赞功能使用了 Redis 的 Set 数据结构来实现，该数据结构自动去重，即一个用户不能点赞两次。评论的 ID 作为 Set 类型的键，值使用用户 ID，每次点赞都将一个用户的 ID 添加进该 Set 中。使用 SCARD 方法获取 Set 中的元素个数，即点赞数量。使用 Redis 提供的 ISMEMBER() 方法来判断用户是否在 Set 中。如果在 Set 中，点赞的按钮显示为已点赞，否则显示为未点赞。前端在用户点赞后将点赞的图标切换为已点赞状态，再次点击则会删除 Set 中的一个元素，图标变为未点赞状态。

实现书评的方法基于树状结构。首先，讨论如何储存一本书的全部评论。设想一本书的评论结构如下图，需要将它转化为树状结构。树的根节点代表书籍本身，根节点的直接子节点则代表直接针对该书的评论，而这些子节点下的其他节点则表示对某条评论的回复。在数据结构中，最普遍的树状存储方式是采用双亲表达方式，即给每个节点编号，并按编号顺序将其存储成数组。每个数组项目都有一个字段记录当前节点的父节点编号。为了存储这些评论，我们在评论的数据表中添加了父节点ID，如图4-18所示。在数组中，每条评论都记录了提交评论的用户ID、评论时间、用户昵称、头像图片链接、评论内容，以及其父节点编号（即回复谁的评论，如果是根节点，则为root）等信息。

图4-18 评论的存储

尽管评论已经存入数据库，我们还需将它们转换为前端能轻易处理的数据格式。在Uniapp的微信小程序里，列表循环通常由v-for指令和v-if指令配合完成。可以将书评划分为两个部分：直接评论某书的评论（如评论1、评论5和评论7）和回应其他评论的评论（如评论2和评论3）。在前端显示时，首先展示那些直接评论某书的评论，然后在这些评论下方添加一个“查看更多”的按钮，提示用户查看对该评论的子评论。因此，后端需要返回的数据格式应清晰明了：评论应按照发布时间和回复顺序排列，相互回复的评论应相邻，而无关的评论则按时间顺序排列。

那么如何从数据库中查询的评论列表中获取适合前端解析的数据呢？基本的策略是通过两次遍历实现。第一次遍历整个评论列表，找出所有直接子节点，即father字段为root的节点。同时，为处理子评论，需要在每个节点内加入一个child变量，用来存储其子评论列表。筛选出这些直接子节点后，得到一个新的子数组。然后，对每个直接子节点遍历子数组，找出所有father为这个直接子节点的子评论节点。通过这种递归查找，直到某个节点不再含有子评论节点为止。下面是相关的代码：

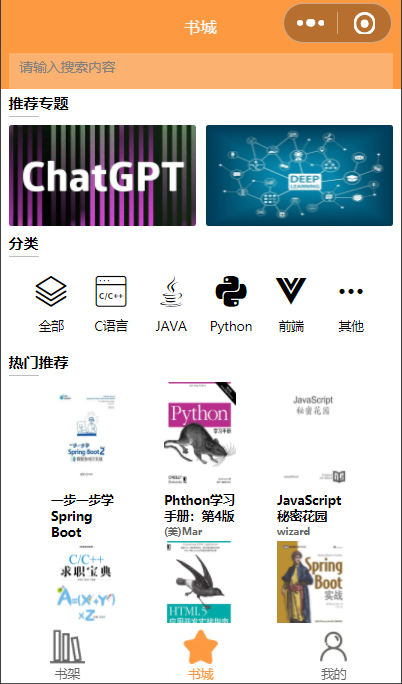
|  |
| --- |
| List<Comments> comments = commentsMapper.selectList(new QueryWrapper<Comments>().eq("bookid", bookid).eq("father", "root"));  List<CommentsVO> res = new ArrayList<>();  for (Comments comment : comments) {  String commentid = comment.getCommentid();  List<Comments> child = commentsMapper.selectList(new QueryWrapper<Comments>().eq("father", commentid));  CommentsVO commentsVO = new CommentsVO();  comment.setLikenum(redisTemplate.opsForSet().size(commentid).intValue());  comment.setIsLike(redisTemplate.opsForSet().isMember(commentid,userid)?1:0);  commentsVO.setChild(child);  commentsVO.setRootComment(comment);  res.add(commentsVO);  } |

在代码中，首先通过selectList方法查询所有father字段为root的评论，得到一个包含所有直接评论的列表。然后对于每个直接评论，使用selectList方法查询所有father字段为该评论的评论，得到一个子数组，表示该直接评论下的所有子评论。同时，使用Redis的相关方法来获取点赞数量和判断用户是否点赞。最后，将直接评论和子评论封装到一个CommentsVO对象中，并添加到结果列表res中。

4.3.5 书城模块

1. 设计描述

书城是一个让用户可以查看各种类别和热门的书籍的地方。用户可以选择特定的类别来查看该类别下的所有书籍，也可以直接查看所有的书籍。现有的分类包括“全部”、“C语言”、“JAVA”、“Python”、“前端”和“其他”。除此之外，微书在书城提供了推荐专题，这为用户提供了一个方便的途径来找到他们感兴趣的书籍。每个推荐专题都会与某个话题相关，例如计算机大事件，这些话题将定期更新以保持内容的新鲜度。这使得用户可以轻松地找到与他们感兴趣的主题相关的大量书籍。用户还可以在页面最上方的输入框中输入搜索关键词，系统会返回相关的书籍供用户选择。

2. 页面

书城的页面如图4-19所示。

图4-19 书城

3. 设计逻辑

（1）搜索

微书提供的“搜索”功能是当用户输入了搜索内容（每当任意输入一个字符时），就会发送请求到后端。能够做到每输入一个字符都请求，是因为使用了下面这个：

|  |
| --- |
| <input placeholder="请输入搜索内容" @focus="setIsSearching" @blur="judgeIsNull"  @input="finishedInput"  v-model="searchValue" /> |

每当用户输入一个字符时，就会执行@input指定的函数，那么这个函数就会发送搜索请求到后端，后端会用如下代码进行查询：

|  |
| --- |
| criteria.orOperator(Criteria.where("author").regex(pattern),  Criteria.where("factionName").regex(pattern));  query1.addCriteria(criteria);  List<Book> list = mongoTemplate.find(query1, Book.class); |

该方法创建一个查询对象和一个标准对象，并使用标准对象创建一个包含关键字的正则表达式的OR查询条件。这意味着，如果图书的作者或书名中包含关键字，就会被匹配到。最后，方法使用MongoTemplate对象执行查询，并返回结果列表。这个列表包含了与关键字匹配的所有图书对象。

（2）热门推荐

该功能可以展示数据库中最受欢迎的6本书，热度是通过点击量来计算的。为了实现这一功能，微书使用了Redis中的Zset类型。当用户点击书籍时，对应书籍ID的得分将增加1。Zset会自动维护得分的顺序，这使得Java端可以通过查询前6个书籍的ID来获取最热门的书籍列表。

当用户去访问一个书籍的详情页时，会在启动页面时发送 “/getBookDetail”请求到后端去获取这本书的所有基本信息，然后后端就会在此时执行这条语句： redisTemplate.opsForZSet().incrementScore("BookReadCount", bookid, 1);

这是一个使用RedisTemplate操作Redis数据库中的Zset数据类型的Java代码行。第一个参数是Zset的名称（即"BookReadCount"），第二个参数是书籍的ID（即bookid），第三个参数是增量值（即1）。该行代码的目的是将特定书籍的阅读计数器增加1。

当后面要取出热门推荐的书籍时，会执行下面这些代码：

|  |
| --- |
| Set<String> topBooks = redisTemplate.opsForZSet().reverseRange("BookReadCount", 0, 5);  Query query = Query.query(Criteria.where("\_id").in(topBooks));  List<Book> topBook = mongoTemplate.find(query, Book.class); |

在第一行中，redisTemplate是一个SpringBoot自动注入的RedisTemplate对象，它用于与Redis数据库进行交互。opsForZSet()方法返回一个ZSetOperations对象，它提供了Zset数据类型的一系列操作方法。reverseRange()方法用于按照分值从大到小的顺序返回指定Zset的前6个成员（即书籍ID）。在这个例子中，第一个参数是Zset的名称（即"BookReadCount"），第二个参数是起始索引（即0），第三个参数是终止索引（即5），这意味着只返回前6个成员。

在第二行中，使用in()方法和topBooks集合作为参数创建一个包含查询条件的Criteria对象，该对象将在MongoDB中查询符合条件的所有文档。在这个例子中，"\_id"是MongoDB文档中表示书籍ID的字段。所以Criteria.where("\_id").in(topBooks)表示查询\_id在topBooks集合中的所有书籍。

在第三行中，使用MongoTemplate对象执行查询，并将结果列表存储在topBook变量中。在这个例子中，查询的结果是与Redis中最受欢迎的6本书的ID匹配的所有书籍。最后，方法返回查询结果列表。

（3）分类：

分类功能是在前端展示6个不同的按钮，当用户点击时会去分类页面，携带着按钮的参数，分类页面就会知道去后端如何查询，自己的页面如何展示，按钮如下：

|  |
| --- |
| <navigator url="/pages/classify/classify?index=0">  <view>全部</view>  </navigator>  <navigator url="/pages/classify/classify?index=1">  <view>C语言</view>  </navigator> |

可以看到，去classify页面携带的index的值是不同的，然后分类页面加载时会执行下面代码：

|  |
| --- |
| if (options.index) {  this.currentIndex = options.index  self.getClassifyData(options.index, self.page);  } |

这里根据url中传过来的分类index，加载指定的分类数据，index默认值1。

4.3.6 书籍分类模块

1. 设计描述

该模块旨在根据不同的分类方式展示书籍。当用户在书城页面点击一个分类进入该页面时，该分类文字将高亮显示，而其他分类文字将变为灰色。然后，该页面将显示当前分类下的所有书籍供用户选择。

2. 分类页面

分类“全部”的页面和懒加载的页面如图4-20和图4-21所示。

图4-20 全部

图4-21 加载中

3. 设计逻辑

在将书籍数据写入MongoDB数据库时，会设置一个名为“index”的字段。该字段有五种选项：1表示“C语言”，2表示“JAVA”，3表示“Python”，4表示“前端”，5表示“其他”。当用户在书城中选择其中一种类型时，系统会携带该类型对应的“index”值去查询MongoDB,代码逻辑如下：

|  |
| --- |
| if (index.equals("0")) { //如果是0，说明是全部查询  list = mongoTemplate.find(query1, Book.class);  map.put("list", list);  map.put("total", mongoTemplate.findAll(Book.class).size());  } else {  query1.addCriteria(Criteria.where("index").is(index));  list = mongoTemplate.find(query1, Book.class);  map.put("list", list);  Query query2 = new Query();  query2.addCriteria(Criteria.where("index").is(index));  } |

代码中的第一行if语句判断给定的条件是否为0，如果是0，则查询所有符合条件的数据。如果条件不为0，则根据条件添加查询条件，并查询符合条件的数据。查询结果被封装到一个名为list的List中，并将list和查询到的总数封装到一个名为map的Map中。如果条件不为0，还会创建一个名为query2的查询对象，用于进一步查询。

在前端显示时，懒加载可以减少页面的初始加载时间和带宽占用。它只在用户需要访问某个资源时才进行加载，而不是在页面一开始就加载所有资源。这可以帮助提高页面的响应速度，因为在页面初始加载时只需加载必要的资源，而不是所有资源，从而加快页面的加载速度。此外懒加载还可以减少带宽占用，因为只有当用户需要访问资源时才进行加载，而不是在页面一开始就加载所有资源。这可以帮助降低服务器负载，从而提高网站的性能和可伸缩性。

当用户向下滑动页面时，会出现“加载中”的图标，此时前端会使用AJAX技术向Java端发送请求以请求下一页的书籍。该请求使用AJAX技术实现，可以异步地向服务器发送请求并更新页面，而不需要重新加载整个页面。在这个过程中，Java端会根据请求参数返回下一页的书籍数据，前端再将数据渲染到页面上。这种实现方式可以提高用户交互的流畅性和速度，并减少对服务器的负荷。

懒加载机制和AJAX技术的应用可以提高前端页面的性能和用户体验，并且减轻了服务器的负荷。这些技术在现代Web开发中已经广泛应用。

4.3.7 个人中心模块

1. 设计描述：

个人信息页面显示了个人头像、个人昵称、有效阅读时长、坚持阅读天数、已阅读书籍数量、用户的消息和用户今日的任务。今日任务主要展示今天需要阅读的书本。

2. 个人中心页面：

图4-22 个人中心

个人中心的页面如图4-22所示。

3. 设计逻辑：

（一）阅读天数设计实现：

实现方式为在MySQL数据库中创建一个名为userlogin的表。每当用户登录时，会写入一条记录。查询连续阅读天数的方法是：先查询该用户的所有数据，然后从第一条记录开始，逐条检查下一条记录和当前记录之间的时间差是否为一天。若是，则将总连续天数加1。完成计算后，将结果返回给前端。代码如下：

|  |
| --- |
| for (int i = 0; i < list.size() - 1; i++) {  Instant instant1 = list.get(i).getLogindate().toInstant();  Instant instant2 = list.get(i + 1).getLogindate().toInstant();  ZoneId zoneId = ZoneId.systemDefault();  LocalDate localDate1 = LocalDate.ofInstant(instant1, zoneId);  LocalDate localDate2 = LocalDate.ofInstant(instant2, zoneId);  long l = ChronoUnit.DAYS.between(localDate2, localDate1);  if (l == 1) {  res++;  } else {  break;  }  } |

代码中通过调用 list.get(i) 和 list.get(i + 1) 来获取相邻的两个日期时间对象，并将它们转换为 Instant 对象。然后代码使用 ZoneId.systemDefault() 获取当前时区，并将 Instant 对象转换为 LocalDate 对象，以便计算日期差。最后，代码调用 ChronoUnit.DAYS.between(localDate2, localDate1) 来计算相邻两个日期时间对象之间的日期差，并将其赋值给变量 l。如果日期差为 1 天，则将 res 的值加 1；否则，跳出循环。执行之后，返回符合连续登录天数。

（二）今日任务设计实现：

在MySQL的bookofuser数据表中，根据hasRead字段查询出阅读进度的章数最大的两本书，并在此页面上展示。如果用户是第一次登录，也就是说在bookofuser中没有该用户的记录，那么将使用热度最高的两本书代替。

（三）我的消息设计实现：

在这里展示各种消息，包括系统消息和其他用户对自己的评论。使用轮播图进行展示，并通过微信小程序提供的swiper元素实现，代码如下：

|  |
| --- |
| <swiper style="height:250rpx;" class="swipe" duration="600" @change="changeEmail" :current="currentEmail">  <swiper-item v-for="item in emails"></swiper-item>  </swiper> |

当其他用户评论自己时，后端会接收到相关信息，包括书本ID、评论者ID、评论内容以及父评论ID。如果这是一个一级评论，父评论ID将被标记为“root”。否则，父评论ID将是一个具体的数字。在插入评论表时，会根据这个父评论ID是否为“root”来判断这条评论是否为一级评论。如果不是一级评论，那么说明这是对其他用户评论的回复。此时，可以先去查询出父评论的发布者是谁，并根据这个发布者ID和当前用户ID将消息存储到消息表中。用户登录时，会查询与自己相关的消息记录。如果有相关记录，那么这些消息将会在前端的“所有消息”中显示出来。核心代码如下：

|  |
| --- |
| if(!comments.getFather().equals("root")&&!commentFather.getUserid().equals(comments.getUserid())) {  //父节点不为root,且不是自己评论自己的时候才发送邮件  emailServie.insertReplyEmail(commentFather, comments);  } |

4.3.8 提取书籍数据模块

微书使用本地的电子书文件作为数据源，利用Java编程语言中的EpubLib框架对这些文件进行解析和处理，以提取各种元数据和内容信息，并将它们以文本和其他元数据的形式存储到MongoDB数据库中。EpubLib是一个开源的Java库，可以方便地解析和操作epub文件，提供了丰富的API和工具，用于获取电子书的书名、作者、出版社、出版日期、ISBN号等元数据信息，以及正文文本和封面图片等内容。在实现过程中，微书会逐一遍历本地存储的epub文件目录，对每个文件进行解析和提取信息，最终将结果存储到MongoDB数据库中。这样，用户可以通过查询MongoDB数据库来获取图书信息，实现快速、高效的图书检索和阅读。总之，通过利用Java的EpubLib框架，微书平台可以轻松地处理epub文件，并提供更好的图书阅读和管理体验。代码的逻辑如下：

1.首先，使用EpubReader类来读取EPUB文件，代码如下：

|  |
| --- |
| EpubReader reader = new EpubReader();  Book book = reader.readEpub(new FileInputStream("path/to/epub/file")); |

2. 接下来，可以使用Book类的各种方法来获取EPUB文件中的元数据信息和内容信息，例如：

|  |
| --- |
| String title = book.getTitle();  List<Author> authors = book.getMetadata().getAuthors();  Resource coverImage = book.getCoverImage();  List<Resource> contents = book.getContents(); |

其中，getTitle()方法用于获取书名，getAuthors()方法用于获取作者列表，getCoverImage()方法用于获取封面图片资源，getContents()方法用于获取正文内容资源列表等。

3.在获取正文内容时，需要注意epub文件中的内容可能会分散在多个资源文件中，因此需要进行一定的处理和组装。可以通过以下代码将所有的正文内容组装成一个字符串：

|  |
| --- |
| StringBuilder sb = new StringBuilder();  for (Resource content : contents) {  InputStream is = content.getInputStream();  BufferedReader reader = new BufferedReader(new InputStreamReader(is, "UTF-8"));  String line;  while ((line = reader.readLine()) != null) {  sb.append(line);  sb.append("\n");  }  reader.close();  }  String text = sb.toString(); |

4. 此外，在使用EpubLib解析epub文件时，还需要注意一些特殊情况，例如某些epub文件可能没有封面图片或缺少某些元数据信息等。针对这些情况，需要进行一定的判断和异常处理，以确保程序的稳定性和健壮性。

# 第5章 软件测试

本次软件测试采用单元测试和集成测试两种测试方法，这是因为这两种测试方法在软件开发中都具有重要的作用和价值。

单元测试是一种针对软件中最小的可测试单元进行测试的方法，其主要目的是确保这些单元的功能实现是正确的，且没有潜在的缺陷。在单元测试中，通常采用各种测试框架和工具来对单元进行测试，以验证其功能是否符合预期。通过单元测试，可以更早地发现并解决软件开发过程中出现的问题，减少后期的修复成本，并提高软件的质量和可靠性。

集成测试是一种测试软件系统中多个模块之间互相协作的能力的方法，其主要目的是测试不同模块之间的接口是否正确，以及模块之间是否能够正确地协同工作。在集成测试中，通常采用黑盒测试和白盒测试相结合的方法来进行测试，以尽可能地覆盖软件系统的各个功能和场景。通过集成测试，可以发现和解决不同模块之间的接口问题和协作问题，提高软件系统的整体性能和可用性。

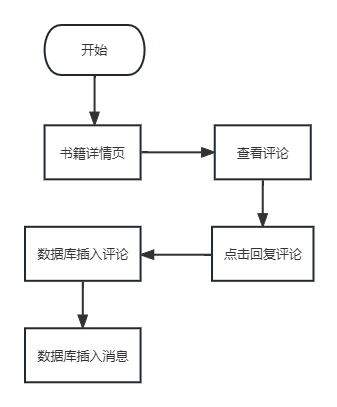
单元测试和集成测试是软件测试中最常用的两种测试方法，它们在软件开发过程中都扮演着重要的角色，能够帮助开发团队更快地发现和解决软件缺陷，提高软件的质量和可靠性，为用户提供更好的使用体验。

## 5.1 单元测试

单元测试是软件开发中一项重要的测试方法，旨在验证一个软件单元（通常是一个类或方法）是否符合预期的设计和功能要求。在单元测试中，开发人员编写测试用例对代码进行测试，以确保代码质量和正确性。单元测试的名称来源于其测试的单元，这些单元应该足够小，足够快，且不具有依赖性。一个单元测试应该只测试一个简单的业务逻辑。

Junit是一个广泛使用的Java单元测试框架，提供了一个简单而有效的方式来编写和运行测试用例，方便验证代码的正确性。使用Junit进行单元测试可以为后续的开发和维护工作提供支持。因此，使用Junit对每个代码单元进行测试。

如图5-1所示，这是一个用户评论其他用户的评论的流程：

图5-1 用户评论

其中进入详情页，查看评论，回复评论和插入消息这几个都是完全独立的单元，所以下面将对这几个单元进行单元测试。代码如下：

|  |
| --- |
| @SpringBootTest  public class Mytest {  @Resource  BookService bookService;  @Test  public void testGetBookDeatil(){  String bookid = "测试ID";  Book book = bookService.findOne(bookid);  System.out.println("测试的详情信息是： "+book);  }  } |

下面的表5-1显示了单元测试的过程以及结果：

表5-1 单元测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **测试单元** | **测试过程** | **预期结果** | **实际结果** | **结论** |
| 获取详情 | 使用一个书本ID：并调用查询方法 | 查询出书本的具体信息 | 书本ID对应的信息都查询出来了 | 测试通过 |
| 查看评论 | 使用一个书本ID：并调用评论的方法 | 查询出该书本的评论 | 查询出了对应书本的所有评论 | 测试通过 |
| 用户回复评论 | 编写一个AddCommentsVo对象来调用添加评论方法 | 方法调用成功，数据库增加了该评论 | 数据库成功增加了该评论 | 测试通过 |
| 数据库插入消息 | 编写一个子评论的AddCommentsVo并调用对应的插入方法 | 方法调用成功，数据库增加了消息 | 数据库成功增加了消息 | 测试通过 |

## 5.2 集成测试

集成测试是一个软件检验方法，主要检查系统内不同组件或模块之间的集成及配合是否顺畅。在这个过程中，测试人员将各个模块聚合，并测试它们的相互配合，以确认系统按预期运行。集成测试可分为两种方式：底层到高层的集成测试和顶层到底层的集成测试。底层到高层的集成测试从最基础的模块开始检验，逐步向上至整个系统。反之，顶层到底层的集成测试则从最高级别的模块启动检验，逐级向下至整个系统。集成测试通常是软件开发生命周期中的一个关键步骤，因为它可以帮助发现系统中的集成问题，从而避免在生产环境中出现意外错误。同时，集成测试还可以帮助确定系统是否符合需求和规范，并提高系统的质量和可靠性。这里使用自上而下的集成测试方法来测试本系统。

根据模块划分，本系统被拆分为六个模块，分别为书架、书城、个人中心、H5 阅读器和书籍详情模块和授权登录，由于采用自上而下的集成测试方法，所以对于每个模块，需要设计测试用例来验证其功能是否符合预期；对于模块间的交互，需要设计测试用例来验证它们之间的集成是否正常；对于前端和后端之间的交互，需要设计测试用例来验证它们之间的接口是否正常。

表5-2显示了部分模块集成测试的过程以及结果。

表5-2 集成测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试项目 | 测试过程 | 预期结果 | 实际结果 | 结论 |
| 在小程序里我的书架测试搜索、展示、阅读的功能是否正常 | 进行搜索查看搜索结果是否正确，查看书籍展示是否正常 | 搜索结果正常  页面显示正常 | 全部正常 | 测试通过 |
| 在小程序里书城模块测试搜索、分类、展示是否正确 | 进行搜索查看搜索结果是否正确，点击分类查看 | 搜索结果正常  分类信息书籍正常 | 全部正常 | 测试通过 |
| 在小程序里个人中心测试是否显示正常，按钮是否正常 | 查看个人中心的已阅读时间等，点击各种按钮 | 个人中心相关信息正确，按钮均点击正常跳转 | 全部正常 | 测试通过 |
| 在小程序里阅读器测试能否调整字体、滑动等 | 左滑或右滑，点击正中间调整阅读参数，点击其他章节 | 滑页正常，阅读风格调整正常，可以切换章节 | 全部正常 | 测试通过 |
| 在书籍详情测试能否正常获取信息 | 进入时是否显示信息 | 书籍基本信息显示正常 | 正常显示 | 测试通过 |
| 在小程序授权时测试能否正常获取信息 | 授权时点击名称和头像测试是否能选择 | 名称和头像可以选择自己微信的名称和头像 | 正常获取 | 测试通过 |

## 5.3 性能测试

在此次性能评估中，我们采用了基于Java的开源性能评估工具JMeter进行测试。JMeter能够模拟并发用户的活动，以检查Web应用和服务的性能及负载承载力，从而使我们对系统的性能状况有更确切的了解。在评估过程中，我们主要关注了系统的反应时间、吞吐量以及并发用户数量等性能参数。反应时间定义为从用户发出请求到系统完成响应的时间，吞吐量是系统在规定时间内能处理的请求数，而并发用户数量则是系统同时容纳的用户数。通过这次评估，我发现系统在正常负载下运行良好，能满足用户的基本需求。系统的响应时间在大部分测试场景下都保持在较低的水平，最长响应时间不超过10秒，平均响应时间在2-3秒之间。这说明系统对于普通用户的操作请求，都可以在短时间内完成响应。系统的吞吐量在高并发测试场景下也表现良好，最高可达到5000个请求/秒。这意味着系统在短时间内可以处理大量的请求，对于高流量的情况也能够应对自如。但是，当并发数达到500左右时，系统开始出现性能瓶颈，响应时间开始明显变长。但整体来看，系统的稳定性和可靠性都较高，对于大部分用户的使用需求都可以满足。

# 结 论

本论文详细介绍了针对程序员和技术爱好者开发的在线书城微信小程序“微书”的设计与实现。这个小程序后端开发采用了Java语言编写的Spring Boot框架，而对于书城的数据存储，选择了MongoDB作为数据库，并利用EPUB框架提取本地电子书文件进行处理，提供高质量的阅读体验。

该小程序的设计和实现旨在为用户提供一个方便、高效、可靠的阅读平台，满足用户对技术知识的学习和获取需求。为此，该小程序考虑用户需求，提供了书架、书城、个人中心、H5阅读器、评论和热门推荐等模块，为用户提供丰富的阅读功能。

在实现过程中，我面临了一些挑战和难点。例如，评论结构设计涉及到如何实现用户对书籍的评价和交流；阅读器开发需要考虑到用户体验和操作体验；后端性能提升需要通过优化算法和调整配置等方式实现。为了克服这些问题，我采用了很多方法，包括认真思考、精心设计、查阅资料、请教老师等方式。通过不断地优化和完善，最终实现了该设计的开发。

我还进行了单元测试、集成测试等，并让同学来使用测试，不断记录和修改完善，实现了高效稳定的系统。小程序的优点和创新性在于采用EPUBLib框架和提供个性化推荐等功能。EPUBLib框架提供了高质量的阅读体验，而个性化推荐功能则可以帮助用户更好地获取和学习技术知识。

最后，该小程序有望为程序员和技术爱好者提供更好的阅读体验，并具有广阔的应用前景。未来，我还可以进一步完善该小程序的功能和性能，以更好地满足用户需求，促进技术知识的学习和传播。

# 致 谢

在大学四年的学习和生活中，我遇到了很多挑战和困难，但同时也得到了很多帮助和支持。首先，我要感谢我的导师。他不仅仅是我的导师，更是我的良师益友。他在整个研究过程中一直给予我鼓励和指导，对我的论文提出了很多宝贵的建议和意见。他的严谨治学、认真负责的工作态度、广博的学识以及对我的深切关怀，都让我深受启发和感动。感谢他的教诲和支持，是他的悉心指导让我在研究过程中不断地成长和进步。

我也要感谢我的同学和朋友们。在大学四年的生活中，我们相互扶持、相互鼓励，共同进步。他们的陪伴和支持让我在学习和生活中都能够有所收获，同时也让我在困难时刻得到了帮助和支持。

此外，我还要感谢大学四年里所有的老师们，他们在教学和科研中不断追求进步，将自己的学识和经验传授给我们。他们的辛勤工作和悉心教导让我获得了丰富的知识和技能，同时也让我更好地了解自己的兴趣和潜力。

我还要感谢那些为本篇论文提供帮助和支持的人，无论是提供数据、文献、书籍，还是在调查问卷中参与答题，都是不可或缺的贡献者。感谢他们的支持和帮助，是他们的默默付出为我提供了极大的帮助和支持。大学四年的学习和生活是我人生中非常宝贵的时光，也是我成长和进步的重要阶段。感谢所有支持和帮助我的人，愿我们都能够在未来的日子里继续为自己的梦想而奋斗。

最后，我要感谢开源社区，感谢GitHub的存在，我们可以轻松地分享代码、协作开发、追踪问题和发布版本。在GitHub上，我可以找到各种各样的开源项目，涵盖了从基础库到复杂应用的各个领域。通过学习和借鉴这些项目，为我们提供了许多宝贵的资源和机会，提高自己的编程能力和代码质量。在以后，我也将积极参与到开源社区中，为其他开发者和用户做出自己的贡献，推动整个行业的发展和进步

所有这些人和组织的帮助和支持是我成功完成这个项目的重要保证。再次表达我的感激之情！

# 参考文献

1. 闵照金. 中小型书城盈利模式实践分析[J]. 云南开放大学学报，2015:82-84.
2. 吴昌政. 基于前后端分离技术的web开发框架设计[D]. 南京邮电大学，2020.
3. 费晓. 网上书城设计与实现[D]. 北京邮电大学，2010.
4. 龚高. JAVA网上书城的设计与实现[D]. 湖北大学，2010.
5. 徐鹏涛. 基于Vue的前端开发框架的设计与实现[D]. 山东大学，2020.
6. 张杰. Android平台EPUB电子书阅读器的设计与实现[D]. 南京大学，2013.
7. 陆向珍. Linux系统下Mongo数据库的快速部署[J]. 电脑编程技巧与维护，2015.
8. 齐鹏飞. 基于微信的智慧校园小程序设计[D]. 西安电子科技大学，2020.
9. 赵圆圆;伍岑. 前后端分离的系统解耦部署方案设计与实现[J]. 湛江科技学院，2022.11.
10. 马雪山;张辉军. 前后端分离的Web平台技术研究与实现[J]. 克拉玛依职业技术学院，2022.4.
11. 朱文杰; 姚雨晨. 基于Spring Boot读写分离架构的理财系统[J]. 湖北师范大学，2022.11.
12. 杨微. 新工科背景下微信小程序应用开发教学设计[J]. 广州软件学院，2023.5.
13. Song Junhui, Zhang Min, Xie Hua. "Design and Implementation of a Vue.js-Based College Teaching System"[J]. International Journal of Emerging Technologies in Learning (iJET), 2019​.
14. Gede, Surya Rahayuda, and Ni Putu Linda Santiari. "Penentuan Rute Evakuasi Bencana Kebakaran Menggunakan Algoritma Dijkstra berbasis Web Framework Vue.js."[J]. Jurnal Sistem dan Informatika (JSI), 2020​.
15. Roman Baida, Maksym Andriienko, and Małgorzata Plechawska-Wójcik. "Performance analysis of frameworks Angular and Vue.js."[J]. Journal of Computer Sciences Institute, 2020.
16. 吴保升. 基于JavaEE的网上书城系统设计[J]. 塔里木大学信息工程学院，2021.11.
17. 邓明杨. 基于JAVA Web技术的网上书城的设计与实现[J]. 广西城市职业大学，2020.4.

# 附 录

为前端提供API的类：

|  |
| --- |
| @RestController  @Slf4j  public class ApiController {  @GetMapping("/api/user/uploadAvatar")  public Result uploadAvatar(@RequestParam("userid")String userid,@RequestParam("tempurl")String tempurl) throws IOException {}  @GetMapping("/api/user/getAllEmails")  public Result getEmails()  @GetMapping("/api/user/getAllTasks")  public Result getAllTasks()  @GetMapping("/api/chapter/chapterByNumAndBookId")  public Result chapterByNumAndBookId(@RequestParam("bookid") String bookid  , @RequestParam("num") int num)  @GetMapping("/api/chapter/maxChapterNum")  public Result getmaxChapterNum(@RequestParam("bookid") String bookid)  @GetMapping("/api/user/recommendBooks")  public Result recommendBooks()  @PostMapping("/api/user/RemoveLikeNum")  public Result RemoveLikeNum(@RequestBody AddCommentsVo addCommentsVo)  @GetMapping("/api/user/hasBookNum")  public Result GethasBookNum()  @GetMapping("/api/book/getOneBookReadingTime")  public Result getOneBookReadingTime( @RequestParam("bookid") String bookid)  @GetMapping("/api/user/continueTime")  public Result GetcontinueTime()  @GetMapping("/api/user/endReading")  public Result endReading(@RequestParam("token") String token)  @GetMapping("/api/user/StartReading")  public Result StartReading( @RequestParam("bookid") String bookid)  @GetMapping("/api/book/getClassify")  public Result getClassify(@RequestParam("index") String index, @RequestParam("page") String page)  @GetMapping("/api/xs\_rank/getRank")  public Result getRank(@RequestParam("rankType") String rankType)  @GetMapping("/api/book")  public Result getAllBooks()  @GetMapping("/api/book/searchBook")  public Result searchBook(@RequestParam("keyword") String keyword)  @GetMapping("/api/chapter/getContentById")  public Result getContentById(@RequestParam("sectionId") String sectionId, @RequestParam("bookid") String bookid)  @GetMapping("/api/book/getMulu")  public Result getMulu(@RequestParam("bookId") String bookid, @RequestParam("sectionNum") Integer sectionNum)  @GetMapping("/api/chapter/getContentByHistory")  public Result getContentByHistory( @RequestParam("bookid") String bookid)  @GetMapping("/api/user/JudgeIsInList")  public Result GetJudgeIsInList( @RequestParam("bookid") String bookid)  @GetMapping("/api/user/getuserid")  public Result GetUserId()  @PostMapping("/api/user/addMyBooks")  public Result addMyBooks(@RequestBody deleteData deleteData)  @PostMapping("/api/user/deleteMyBooks")  public Result deleteMyBooks(@RequestBody deleteData deleteData)  @GetMapping("/api/user/signIn")  public Result EveryDaySignIn(@RequestParam("userid") String userid)  @GetMapping("/api/user/getMyBooks")  public Result getMyBooks()  @GetMapping("/api/user/getUserInfo")  public Result getUserInfo()  @PostMapping("/api/book/addLikeNum")  public Result addLikeNum(@RequestBody AddCommentsVo addCommentsVo) |

下面是Uniapp中的微信小程序配置

|  |
| --- |
| {  "browser": {  "vertx": false  },  "window": {  "navigationBarTextStyle": "white",  "navigationBarTitleText": "",  "navigationBarBackgroundColor": "#FD9941",  "backgroundColor": "#ffffff"  }  ,  "networkTimeout": {  "request": 10000,  "connectSocket": 10000,  "uploadFile": 10000,  "downloadFile": 10000  },  "debug": true,  "sitemapLocation": "sitemap.json",  "tabBar": {  "color": "#676767",  "selectedColor": "#fd9941",  "borderStyle": "white",  "backgroundColor": "#ffffff",  "list": [  {  "pagePath": "pages/booklist/booklist",  "iconPath": "static/booklist\_gray.png",  "selectedIconPath": "static/booklist.png",  "text": "书架"  },  {  "pagePath": "pages/shop/shop",  "iconPath": "static/tuijian\_gray.png",  "selectedIconPath": "static/tuijian.png",  "text": "书城"  },  {  "pagePath": "pages/user/user",  "iconPath": "static/user\_gray.png",  "selectedIconPath": "static/user.png",  "text": "我的"  }  ]  },  "pages": [ //pages数组中第一项表示应用启动页，参考：https://uniapp.dcloud.io/collocation/pages  {    ,{  "path" : "pages/component/toast/toast",  "style" :  {  "backgroundTextStyle":"light",  "backgroundColor":"#ffffff",  "navigationBarBackgroundColor": "#FD9941",  "navigationBarTitleText": "干杯",  "navigationBarTextStyle":"white"  }    }  ,{  "path" : "pages/classify/classify",  "style" :  {  "enablePullDownRefresh": false,  "backgroundTextStyle": "light",  "backgroundColor": "#ffffff",  "navigationBarBackgroundColor": "#FD9941",  "navigationBarTitleText": "分类",  "navigationBarTextStyle": "white"  }  }  ,{  "path" : "pages/rank/rank",  "style" :  {  "backgroundTextStyle":"light",  "backgroundColor":"#fff",  "navigationBarBackgroundColor": "#FD9941",  "navigationBarTitleText": "排行榜",  "navigationBarTextStyle":"white"  }    }    ],  "globalStyle": {  "navigationBarTextStyle": "black",  "navigationBarTitleText": "uni-app",  "navigationBarBackgroundColor": "#F8F8F8",  "backgroundColor": "#F8F8F8"  },  "uniIdRouter": {},  "networkTimeout": {  "request": 10000,  "connectSocket": 10000,  "uploadFile": 10000,  "downloadFile": 10000  },  "debug": true,  "sitemapLocation": "sitemap.json",  "condition": { //模式配置，仅开发期间生效  "current": 0, //当前激活的模式(list 的索引项)  "list": [{  "name": "test", //模式名称  "path": "pages/login/wxlogin/wxlogin", //启动页面，必选  "query": "" //启动参数，在页面的onLoad函数里面得到  },  {  "name": "shop", //模式名称  "path": "pages/shop/shop", //启动页面，必选  "query": "" //启动参数，在页面的onLoad函数里面得到  },  {  "name": "book\_detail", //模式名称  "path": "pages/book\_detail/book\_detail", //启动页面，必选  "query": "" //启动参数，在页面的onLoad函数里面得到  },  {  "name": "classify", //模式名称  "path": "pages/classify/classify", //启动页面，必选  "query": "" //启动参数，在页面的onLoad函数里面得到  }  ]  }  } |

"browser": 浏览器相关的设置。这里的 "vertx" 是 false，但这项配置没有明确的含义，可能是自定义的属性。

"window": 定义全局窗口样式。其中包括：

"navigationBarTextStyle": 导航栏文字颜色，这里设置为 "white"，即白色。

"navigationBarTitleText": 导航栏标题文字，这里设置为空字符串，即没有标题文字。

"navigationBarBackgroundColor" 和 "backgroundColor": 分别定义导航栏背景色和页面背景色。

"networkTimeout": 网络请求超时设置。其中包括请求超时（"request"）、连接超时（"connectSocket"）、上传文件超时（"uploadFile"）和下载文件超时（"downloadFile"）的超时时长，这里都设置为10000毫秒。

"debug": 调试模式开关。这里设置为 true，即开启调试模式。

"sitemapLocation": 网站地图文件位置，这里设置为 "sitemap.json"。

"tabBar": 定义底部标签栏样式和相关页面。例如，颜色、选中颜色、边框样式、背景色和标签列表。

"pages": 定义应用程序中的页面信息，包括路径、样式等。

"globalStyle": 定义全局样式，包括导航栏文本颜色、导航栏标题文本、导航栏背景色和背景色。

"uniIdRouter": uni-app 的页面路由配置，这里为空。

"condition": 模式配置，仅在开发期间生效。可以设置多种模式，并指定启动页面和启动参数。此处定义了4种模式：test、shop、book\_detail 和 classify。