



密级: 公开

# 本科生毕业设计(论文)

题 目: 网上书店的设计与实现

作 者: 尹俊皓

学 号: 61962047

学 院: 计算机与通信工程学院

专 业: 计算机科学与技术

成 绩:                 

2025 年 05 月



# 本科生毕业设计(论文)

题 目: 网上书店的设计与实现

英文题目: Design and Implementation of  
an Online Bookstore

学 院: 计算机与通信工程学院

班 级: 计 214

学 生: 尹俊皓

学 号: 61962047

指导教师: [导师] [职称]

[导师]

[职称]

**指导教师:** \_\_\_\_\_ **职称:** \_\_\_\_\_

## 声 明

本人郑重声明：所呈交的论文是本人在指导教师的指导下进行的研究工作及取得研究结果。论文在引用他人已经发表或撰写的研究成果时，已经作了明确的标识；除此之外，论文中不包括其他人已经发表或撰写的研究成果，均为独立完成。其他同志对本文所做的任何贡献均已在论文中做了明确的说明并表达了谢意。

学生签名: \_\_\_\_\_ 年\_\_月\_\_日

导师签名: \_\_\_\_\_ 年\_\_月\_\_日



# 毕业设计(论文)任务书

[1] 一、学生姓名： 尹俊皓 学号： 61962047

二、题目：网上书店的设计与实现

三、题目来源：真实  、 自拟

四、结业方式：设计  、 论文

五、主要内容：

近年来，电商行业竞争日益激烈，精准个性化推荐已成为电商平台提升用户体验和增加转化率的核心竞争力。本课题拟开发一个网上书店，包含以下主要模块：

1. 用户管理模块：支持用户注册、登录、权限管理；

2. 用户登录模块：提供账号登录和退出功能；

3. 图书管理模块：支持图书的分类入库、出库、价格调整等操作；

4. 图书推荐模块：基于用户行为分析，提供个性化图书推荐。

六、主要（技术）要求：

1. 采用 MVC (Model-View-Controller) 和 Spring Boot 进行前后端开发，保障系统的模块化与可扩展性；

2. 设计一个具有较高精准度的智能推荐算法。

七、日程安排：

第 1-3 周：查阅相关文献，学习前后端方面的基础知识，并完成选题报告，搭建系统开发环境；

第 4-7 周：系统总体设计和系统数据库设计；

第 8-9 周：系统模块设计，算法设计；

第 10-12 周：系统测试与改进；

第 13-15 周：撰写毕业论文，准备毕业答辩。

八、主要参考文献和书目：

Bach, Le. (2022). Developing an E-Commerce Website with React. Bachelor's thesis, Information Technology, Bachelor of Engineering.

[2] Zheng, Z., Bai, M., & Hu, W. (2023). Design and Implementation of Online Bookstores. Taiyuan University of Technology & Saint Petersburg State University.

[3] Wu, H. (2023). Construction of Online Teaching System Based on SpringBoot Framework for Normal University Students' Informatization

Teaching Ability Training.

- [4] 田松涛, 段元梅. (2022). 基于 SpringBoot 的线上商城平台设计. 龙源期刊网.
- [5] 唐双林. (2023). 基于 Vue 和 SpringBoot 架构的智能推荐农产品团购销售系统. 掌桥科研.
- [6] 张鹏, 蒋海蓉. (2023). 基于 MVC 的图书管理系统的应用与实现. 掌桥科研.
- [7] 陈彬. (2024). 基于 SpringBoot 技术的海产品销售平台设计与开发. 掌桥科研.
- [8] 王丽爱, 周旭东. (2024). 基于安卓 MVC 架构的体育用品商城的设计与实现. 龙源期刊网.

指导教师签字： 年      月      日

学 生 签 字： 年      月      日

系（所）负责人章： 年      月      日

## 摘要

本课题旨在设计并实现一个基于人工智能的在线书店系统，以提升用户的图书选购体验与平台的智能化服务能力。系统采用 Spring Boot 与 Vue3 构建前后端分离架构，集成 DeepSeek 平台提供的大语言模型 API，结合 e5-large-v2 句向量模型，实现对用户自然语言输入的意图识别与图书语义匹配，从而完成个性化推荐。

同时，系统支持批量 ISBN 导入图书信息，并通过 AI 自动生成三行摘要，提升图书展示的精炼度。在系统功能方面，用户可进行注册、登录、兴趣分类设置、图书浏览、收藏与对话式推荐交互；管理员可管理图书入库、出库、价格与库存等信息。为提升用户沉浸感，前端页面集成背景音乐切换功能，界面交互友好。

系统采用限流与多线程机制优化 AI 服务调用效率，并通过缓存机制提升响应速度。测试结果表明，系统具备良好的推荐准确性、功能稳定性与用户交互体验，具有实际应用价值与扩展潜力。

**关键词：** 在线书店； 个性化推荐； 摘要生成； 人工智能； 自然语言处理



## Abstract

This study aims to design and implement an AI-powered online bookstore system to enhance users' book discovery experience and improve the platform's level of intelligent service. The system adopts a front-end/back-end separation architecture based on Spring Boot and Vue3, integrates DeepSeek's large language model APIs, and leverages the e5-large-v2 sentence embedding model to analyze users' natural language input and semantically match books for personalized recommendation.

The system supports batch import of books through ISBN input and automatically generates concise three-line summaries using AI to optimize information presentation. From a functional perspective, users can register, log in, set preferred categories, browse and favorite books, and interact with a dialogue-based recommendation assistant. Administrators are granted permissions to manage book entries, withdrawals, pricing, and inventory. Additionally, the front-end UI provides an immersive reading environment by enabling users to switch between background music themes.

To improve performance, the system implements rate limiting and multithreading strategies for efficient AI service calls, and applies caching mechanisms to reduce latency. Test results show that the system delivers satisfactory recommendation accuracy, functional stability, and user interaction fluency. This work demonstrates practical application value and extensibility potential, serving as a reference prototype for AI-integrated digital content platforms.

**Key Words:** **online bookstore, personalized recommendation, summary generation, artificial intelligence, natural language processing**



# 目 录

摘要	I
Abstract	III
插图或附表清单	VII
注释说明清单	IX
1 引言	1
2 文献综述	3
2.1 个性化推荐技术发展概况	3
2.2 自然语言处理与大语言模型（LLM）技术	4
2.3 语义嵌入与向量匹配方法	4
2.4 现有在线书店平台分析	5
3 系统总体设计	6
3.1 需求分析	6
3.2 系统整体架构	6
3.3 功能模块划分	7
3.4 数据库设计与实体关系	9
3.5 技术选型与理由	9
4 详细设计与实现	11
4.1 系统详细设计	11
4.1.1 用户管理模块详细设计	11
4.1.2 图书管理模块详细设计	12
4.1.3 AI 摘要模块详细设计	13
4.1.4 推荐系统详细设计	14
4.1.5 收藏与推荐记录模块设计	15
4.1.6 图书借阅模块设计	16
4.1.7 前端沉浸式界面设计	17
4.2 系统实现	17
4.2.1 用户管理模块实现	18
4.2.2 图书管理模块实现	19
4.2.3 AI 摘要模块实现	20
4.2.4 推荐系统实现	22
4.2.5 收藏与推荐记录模块实现	24

4.2.6 图书借阅模块实现.....	26
4.2.7 前端沉浸式界面实现.....	28
5 系统测试与分析 .....	30
5.1 功能测试与覆盖情况 .....	30
5.2 AI 模块调用稳定性与响应测试 .....	33
5.2.1 摘要生成性能测试.....	33
5.2.2 推荐系统响应测试.....	34
5.2.3 推荐结果匹配度评估.....	35
5.3 性能优化分 .....	35
5.4 潜在问题与改进方向 .....	36
6 结 论 .....	37
7 参考文献 .....	39
8 在学取得成果 .....	40
9 致 谢 .....	41

## 插图或附表清单

图 2-1 推荐系统算法对比图 (自制) .....	3
图 2-2 系统推荐向量计算流程图 (自制) .....	4
图 3-1 系统整体架构图 (自制) .....	7
图 3-2 系统功能模块结构图 (自制) .....	8
图 3-3 数据库 E-R 图 (自制) .....	9
图 4-1 用户注册与权限认证流程图 (自制) .....	12
图 4-2 图书导入与逻辑出库流程图 (自制) .....	13
图 4-3AI 摘要生成流程图 (自制) .....	14
图 4-4 自然语言推荐流程图 (自制) .....	15
图 4-5 收藏与推荐记录数据结构与交互图 (自制) .....	16
图 4-6 图书借阅与归还流程图 (自制) .....	17
图 4-7BGM 播放与切换状态控制图 (自制) .....	17
图 4-8 用户注册与鉴权流程图 (自制) .....	18
图 4-9 图书列表页面界面截图 (自制) .....	20
图 4-10 摘要生成并发处理流程图 (自制) .....	21
图 4-11 推荐系统端到端流程图 (自制) .....	22
图 4-12 推荐对话界面截图 (推荐中) (自制) .....	23
图 4-13 推荐对话界面截图 (推荐完成) (自制) .....	24
图 4-14 我的页面收藏展示截图 (自制) .....	25
图 4-15 我的推荐记录展示截图 (自制) .....	25
图 4-16 借阅按钮展示界面截图 (自制) .....	26
图 4-17 归还按钮展示界面截图 (自制) .....	27
图 4-18 我的页面中借阅图书列表截图 (自制) .....	27
图 4-19App.vue 中 BGM 控制界面截图 (更改前) (自制) .....	28
图 4-20App.vue 中 BGM 控制界面截图 (更改后) (自制) .....	28
图 5-1 待导入的 ISBN 列表文件截图 (自制) .....	31
图 5-2 数据导入前的数据库状态截图 (自制) .....	31
图 5-3 数据导入后的数据库状态截图 (自制) .....	31
图 5-4 图书导入成功的控制台输出日志截图 (自制) .....	32
图 5-5 图书导入与分类结果截图 (自制) .....	32
图 5-6AI 摘要生成后的图书卡片截图 (自制) .....	32
图 5-7 摘要生成日志截图 (优化前) (自制) .....	33
图 5-8 摘要生成日志截图 (优化后) (自制) .....	34
图 5-9 推荐系统响应日志截图 (优化前) (自制) .....	34
图 5-10 推荐系统响应日志截图 (优化后) (自制) .....	35
图 5-11 并发处理与缓存机制流程图 (自制) .....	36
表 2-1 当前主流平台功能对比表 (自制) .....	5
表 3-1 系统关键技术选型表 (自制) .....	10
表 5-1 测试覆盖结果表 (自制) .....	30
表 5-2 推荐匹配度评估表 (自制) .....	35
代码 4-1JwtUtil.java 签名生成与验证方法 (自制) .....	18
代码 4-2IsbnBatchRegistrar.java 核心逻辑 (自制) .....	19
代码 4-3BookController.java 中 deleteBook() 方法 (自制) .....	20
代码 4-4BookSummaryBatchRunner.java run() 逻辑 (自制) .....	21
代码 4-5embedding_query.py 主函数 (自制) .....	22

代码 4-6UserController.java 收藏记录查询逻辑（自制） .....	24
代码 4-7UserController.java 推荐记录查询逻辑（自制） .....	24
代码 4-8MyPage.vue 收藏/推荐删除处理逻辑（自制） .....	25
代码 4-9UserController.java 中借阅处理逻辑（自制） .....	27
代码 4-10App.vue setup 脚本片段（自制） .....	28

## 注释说明清单

术语	中文全称	说明
NLP	自然语言处理	利用计算机处理和理解自然语言的技术总称
LLM	大型语言模型	基于 Transformer 架构、具备上下文推理能力的预训练模型
JWT	JSON Web Token	前后端分离架构中常用的无状态身份验证令牌



## 1 引言

随着人工智能技术与电子商务的融合不断深入，个性化推荐系统在网络平台中的应用已成为提升用户体验与平台竞争力的重要手段。特别是在图书销售领域，传统网上书店往往仅提供静态展示与关键词检索，缺乏对用户兴趣的深度理解与智能化交互机制，导致推荐精度低、用户留存率不高、平台活跃度不足等问题。如何通过自然语言处理与智能推荐算法，为用户提供精准、高效、互动的图书推荐体验，成为当前研究与实践的热点方向。

本课题旨在设计并实现一个具备智能推荐与摘要功能的在线书店系统，依托 Spring Boot 与 Vue3 构建前后端分离的 Web 平台，集用户管理、图书管理、AI 摘要生成、个性化推荐与沉浸式界面于一体。系统通过调用 DeepSeek 提供的大语言模型 API（包括 deepseek-reasoner 与 deepseek-v3），结合 SentenceTransformer 向量模型（e5-large-v2），实现对用户自然语言输入的语义解析与图书内容的深度匹配。此外，系统还设计了 ISBN 批量导入机制，管理员可通过上传文件自动导入图书并生成摘要，显著降低运营成本；并支持图书的分类、入库、出库、价格调整与库存管理，实现完整的商品管理流程。此外，为增强用户情感沉浸感与页面互动体验，系统在前端集成了背景音乐选择器，用户可根据场景需求切换播放主题，实现情境驱动的界面反馈。

用户层面，系统在注册时提供兴趣分类选项，用于缓解冷启动问题；登录后可通过对话式界面输入如“推荐一本适合晚上放松的书”这类语句，系统将综合用户兴趣、情感状态与语义意图进行多维评分推荐，并在主页与聊天窗口中展示推荐结果。此外，系统还支持图书收藏（“愿望清单”功能）与推荐记录展示，提升用户粘性。界面方面，为营造沉浸式阅读体验，系统在前端集成背景音乐选择器，用户可根据个人偏好切换音乐类型。

从系统架构上看，本项目采用模块化与高内聚低耦合设计原则，将用户管理、图书管理、推荐模块、摘要模块与前端 UI 各自独立部署，便于维护与扩展。在性能层面，为解决 AI 请求延迟问题，系统引入 RateLimiter 限流机制与线程池并发处理策略，并对高频调用结果进行缓存，从而提升整体响应速度与稳定性。

综上所述，本课题通过引入自然语言处理、向量匹配与大语言模型技术，构建了一个兼具智能化、互动性与实用性的 AI 驱动型在线书店系统，力求在功能实现与用户体验层面突破传统平台的局限。该系统不仅可作为智能书店的参考原型，也为电子商务平台中智能推荐技术的研究与应用提供了可行性验证。

## 2 文献综述

个性化推荐系统、自然语言处理与语义向量技术是当前人工智能研究的重要方向，也是本课题的核心技术基础。为更好地理解本系统设计所依托的关键技术，本章将围绕推荐算法的发展趋势、大语言模型的应用现状、语义嵌入模型的演化，以及相关在线书店平台的功能对比展开综述。

### 2.1 个性化推荐技术发展概况

推荐系统起初多采用协同过滤算法（Collaborative Filtering），该方法根据用户之间的行为相似性进行推荐，广泛应用于电商、视频、音乐平台中<sup>[1]</sup>。尽管该方法具有较好的个性化能力，但在面对“冷启动”用户或稀疏数据时存在明显局限性。

为提升推荐的多样性与适应性，基于内容的推荐方法（Content-Based Recommendation）逐渐兴起，通过分析物品的属性信息（如关键词、类别等）与用户兴趣模型进行匹配推荐。近年来，更为先进的混合推荐方法（Hybrid Recommendation）成为主流，结合用户行为、物品属性及上下文情境，综合多维度评分机制实现动态推荐策略。

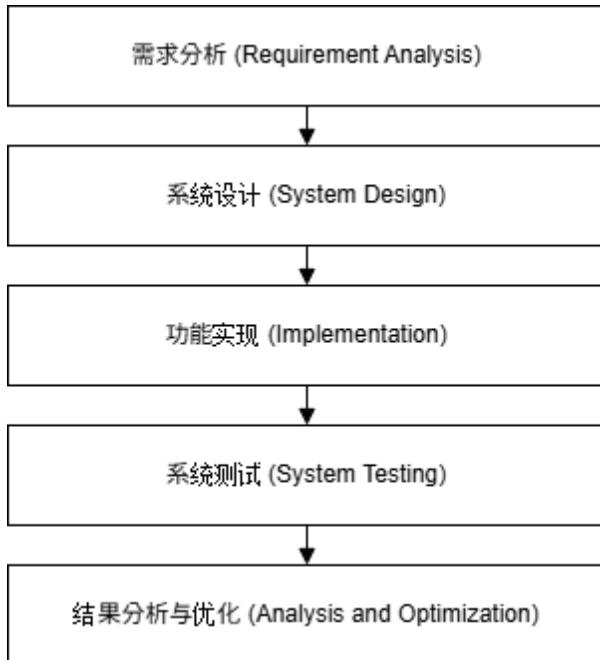


图 2-1 推荐系统算法对比图（自制）

## 2.2 自然语言处理与大语言模型（LLM）技术

近年来，基于 Transformer 架构的大语言模型（如 BERT、GPT、T5 等）在自然语言理解、对话生成、情感识别等任务中表现出色<sup>[2]</sup>。LLM 具有强大的上下文建模与语义推理能力，已广泛应用于文本分类、问答系统与推荐系统中。

本系统采用 DeepSeek 平台提供的 API 服务，其中 deepseek-v3 模型用于解析用户的自然语言输入，提取其情绪、话题与阅读目的；而 deepseek-reasoner 模型则负责对图书的描述信息进行抽象压缩，生成结构化摘要，提升用户对图书内容的理解效率。

## 2.3 语义嵌入与向量匹配方法

为了突破传统关键词匹配的局限性，近年来推荐系统普遍引入语义向量嵌入方法。句向量模型如 Sentence-BERT、SimCSE、E5 系列等能够将任意自然语言句子编码为高维向量，并通过向量相似度计算衡量语义关联程度<sup>[3]</sup>。

本系统引入 e5-large-v2 句向量模型，对用户输入的自然语言请求与图书内容摘要进行编码，并计算其向量余弦相似度，用以构建推荐候选集，并进一步通过关键词匹配数、兴趣领域重合度等因子进行打分排序，提升推荐精准度。



图 2-2 系统推荐向量计算流程图（自制）

展示从“用户输入” → “LLM 提取关键词” → “语义向量编码” → “图书向量库比对” → “得分排序” → “Top-N 推荐”的完整流程图。该图将在第 4 章详细实现部分再次出现，但本章用于技术概念说明。

## 2.4 现有在线书店平台分析

前市场上主要在线书店平台包括 Amazon Kindle、Goodreads、Google Books 等。这些平台虽具备一定推荐功能，但仍以关键词检索与评分排序为主，缺乏自然语言交互能力与深度语义理解机制。以下是平台功能对比：

表 2-1 当前主流平台功能对比表（自制）

平台	推荐方式	交互性	摘要功能	个性化程度
Amazon Kindle	基于历史购买记录	X	简要图书描述	中等
Goodreads	用户评分与书评排序	X	无摘要生成	低
ChatGPT for Books	AI 问答式解释图书内容	✓	(生成摘要) 无在线书城功能	
Google Books	AI 搜索与图书展示	X	(信息整合)	低

通过对比可见，当前在线书店系统普遍存在智能化程度不足、推荐交互性差、无法理解用户情绪或意图等问题。因此，构建一个结合语义理解、情绪识别与自然语言交互机制的 AI 书店系统具有明显的创新空间。

## 3 系统总体设计

为了实现本课题提出的个性化推荐与 AI 摘要功能, 系统采用前后端分离架构, 基于 Spring Boot + Vue3 进行开发。系统整体结构清晰, 功能模块划分合理, 具备良好的可扩展性与可维护性。本章将从系统架构、功能模块划分、数据库设计、关键数据流程与技术选型等方面对系统进行总体设计描述。

### 3.1 需求分析

本系统旨在为用户提供一个集图书浏览、AI 摘要生成、智能推荐、收藏管理及借阅功能于一体的综合性网上书店平台, 服务对象主要为追求高效率与个性化阅读体验的网络用户。为了确保系统既能满足实际用户需求, 又具备良好的功能完备性与技术可实施性, 现从用户侧与系统侧两个维度进行需求分析。

#### (1) 用户需求

- 支持多方式注册与登录, 确保用户身份管理安全;
- 能够高效浏览图书信息, 按关键词或分类进行筛选;
- 提供个性化推荐服务, 基于用户情感或兴趣进行智能匹配;
- 支持收藏感兴趣的图书, 并查看历史推荐记录;
- 实现在线借阅与归还功能, 便于用户管理阅读计划;
- 提供沉浸式界面体验 (背景音乐、流畅交互等)。

#### (2) 系统功能需求

- 后端需支持图书信息的批量导入 (ISBN 列表)、库存与价格管理;
- 实现 AI 摘要功能, 对图书内容进行语义提炼与分类预测;
- 推荐模块需具备语义理解、关键词提取、向量匹配等能力;
- 前后端通信基于 RESTful API, 确保数据同步与响应效率;
- 实现用户行为 (收藏、借阅、推荐) 与图书记录的双向绑定;
- 系统整体结构应具备模块化设计, 易于维护与升级。

### 3.2 系统整体架构

本系统采用典型的三层结构: 表示层 (前端)、业务逻辑层 (后端) 和数据持久层 (数据库)。前端通过 Vue3 实现单页应用, 后端由 Spring Boot 提供

RESTful API 服务，AI 推荐与摘要服务由 Python Flask 子模块提供。AI 模块与后端之间通过 HTTP 接口通信，数据库采用 MySQL 存储所有结构化数据。

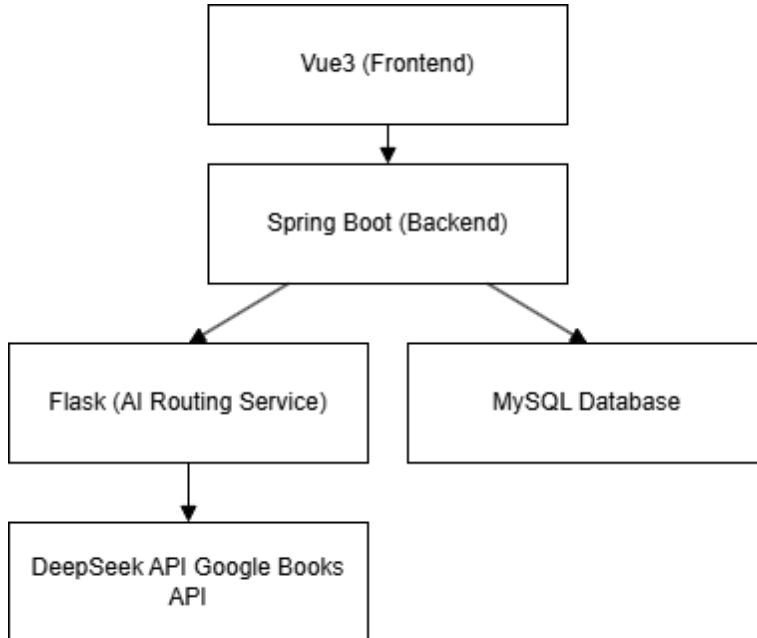


图 3-1 系统整体架构图（自制）

系统各层之间职责如下：

- 表示层：实现用户注册、登录、推荐请求、图书浏览等前端界面交互。
- 业务逻辑层：处理用户请求、推荐逻辑、摘要请求、权限验证、图书管理等核心服务。
- 数据层：管理用户、图书、推荐记录、收藏等持久数据。
- AI 服务层：提供自然语言处理与摘要生成功能，由 Flask 接口与 DeepSeek API 组合实现。

### 3.3 功能模块划分

系统共划分为七大功能模块，各模块职责如下：

#### (1) 用户管理模块

支持用户注册、登录、退出、权限控制、兴趣偏好设置及个人信息修改等操作。

#### (2) 图书管理模块

包括图书的批量导入（通过 ISBN 列表）、分类浏览、关键词搜索、库存数量修改、价格调整及逻辑删除（即“出库”）功能，管理员可通过后台系统高效管理图书数据。

#### (3) AI 摘要模块

基于图书原始描述信息生成简洁摘要，仅在 summary 字段为空时触发，同时自动预测所属分类标签，提升信息展示效率。

(4) 推荐系统模块

用户通过自然语言输入表达阅读需求，系统提取关键词、情感与语义意图，并结合语义嵌入模型和兴趣分类实现 Top-N 推荐。

(5) 收藏与推荐记录模块

用户可将感兴趣图书加入愿望清单，并在“我的页面”中查看历史推荐记录，实现个性化追踪与反馈。

(6) 图书借阅模块

支持用户借阅与归还图书，系统记录借阅时间、归还状态与借阅用户信息，防止重复借阅并支持后续数据分析。

(7) 前端沉浸式界面

提供背景音乐（BGM）控制功能，结合前端界面状态联动播放，营造温馨沉浸式购书体验。

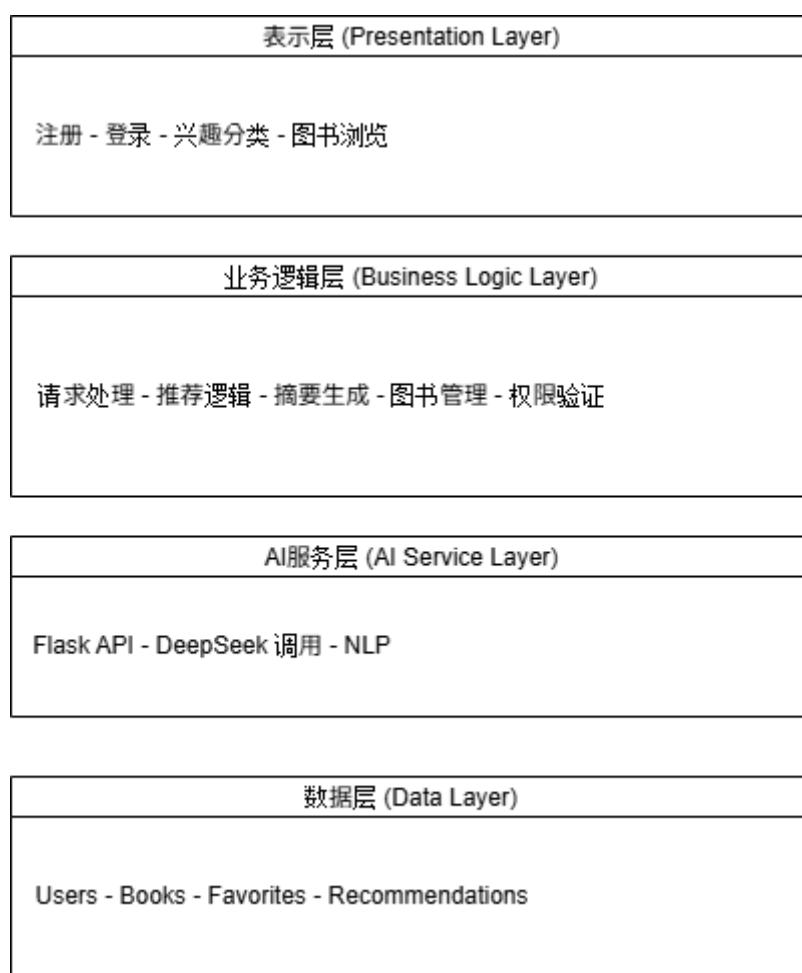


图 3-2 系统功能模块结构图（自制）

### 3.4 数据库设计与实体关系

数据库设计采用 E-R 模型，主要实体包括：

- 用户 (Users)
- 图书 (Books)
- 收藏记录 (Favorites)
- 推荐记录 (Recommendations)

其中，图书实体中包含 stock 库存字段、价格字段、AI 摘要字段、分类等属性。用户实体包含兴趣类别列表、注册时间、权限标识等字段。

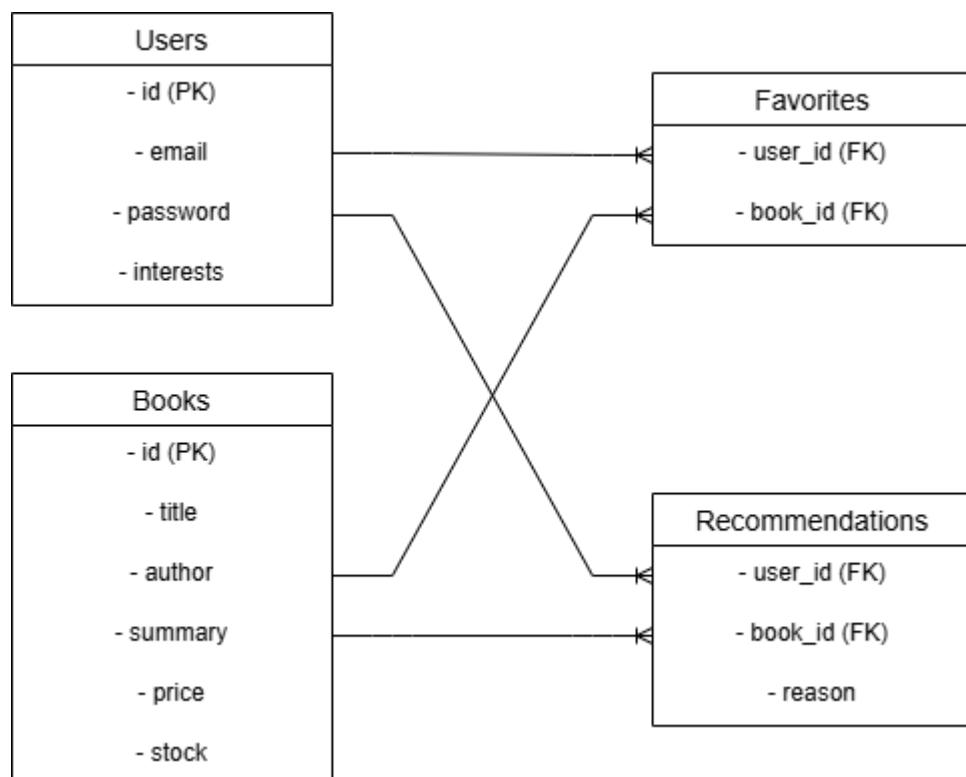


图 3-3 数据库 E-R 图 (自制)

### 3.5 技术选型与理由

为了确保系统的可扩展性、稳定性与开发效率，本文在系统设计与实现阶段对关键技术进行了综合选型。选型过程充分考虑了以下几个方面：

- (1) 项目的目标与需求，如 AI 摘要与推荐、用户交互体验、系统响应效率等；
- (2) 现有主流技术的成熟度、社区支持与性能表现；
- (3) 个人的技术栈基础与项目完成周期的现实性。

在综合调研与对比分析的基础上，系统最终确定了包括 Vue 3、Spring Boot、Flask、DeepSeek API、E5 Embedding、MySQL 等关键组件的技术选型。各技术的功能用途与选型理由详见下表。

表 3-1 系统关键技术选型表（自制）

层次	技术选型	原因说明
前端框架	Vue3	支持 SPA 开发，组件化，响应速度快 <sup>[4]</sup>
后端框架	Spring Boot	模块化强，REST API 设计清晰 <sup>[5]</sup>
数据存储	MySQL	关系型数据库，结构化强，稳定性高
AI 摘要	DeepSeek Reasoner	中文语义压缩效果好，支持 API 调用
意图识别	DeepSeek V3	多轮对话理解强，支持情感识别
语义匹配	E5-large-v2	向量编码精度高，适合句子级别语义比较
并发优化	RateLimiter + Executors	限制外部 API 请求频率，提升处理效率
安全机制	JWT 认证	前后端分离系统认证机制标准做法

## 4 详细设计与实现

在系统总体架构设计的基础上，本章进一步细化各功能模块的设计逻辑与实现策略。通过对核心模块（如用户管理、图书导入、AI 摘要、推荐与收藏功能等）的深入拆解，全面呈现系统从设计蓝图到代码落地的过程，确保实现过程具备技术合理性与扩展潜力。

### 4.1 系统详细设计

为了确保各模块功能在实现阶段具备一致性与可维护性，本节从模块视角出发，详细阐述每个关键组件的功能职责、数据流转方式与交互逻辑。设计过程中充分考虑了模块间耦合度、性能瓶颈以及 AI 服务集成等问题，并辅以流程图直观展示技术方案。

#### 4.1.1 用户管理模块详细设计

用户模块是系统交互的起点，负责用户身份的注册、登录、验证与个性化信息维护。为了支持前后端完全分离的架构，系统采用了 JWT (JSON Web Token) 进行身份认证，通过令牌方式在用户首次登录后进行会话保持和权限校验<sup>[6]</sup>。

在设计中，为实现推荐的精准性，注册流程中特别要求用户选择至少三个兴趣类别。这不仅为个性化推荐模型提供先验标签，也在数据库中构建了更细致的用户画像。用户信息存储在 `users` 表中，兴趣类别以分类 ID 的形式映射并存储在 `user_category` 关联字段中。

用户完成注册后，登录过程将返回带签名的 JWT 令牌，前端保存于 `localStorage`，用于后续所有受保护接口的请求头中。用户模块还支持个人资料查看与更新，所有接口均由后端的权限拦截器校验访问权限。

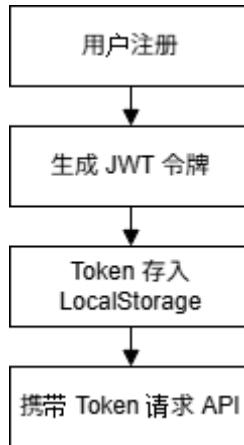


图 4-1 用户注册与权限认证流程图 (自制)

#### 4.1.2 图书管理模块详细设计

图书管理模块是本系统的核心数据管理组件，涵盖了图书的自动导入、分类查询、关键词搜索、价格与库存更新以及逻辑出库等功能。其设计目标是让管理员能够高效地批量处理图书信息，同时为前端用户提供多维度的图书检索能力。

导入方面，系统支持管理员通过上传 `isbn_list.txt` 文件，在后台自动读取每一条 ISBN 编号，并调用 Google Books API 拉取图书的标题、作者、出版社、封面、分类等信息。成功获取的数据将写入 `Books` 数据表中；若请求失败或 ISBN 无效，则将其写入 `failed_isbn_list.txt` 供后续复查。

前端查询方面，用户可按分类标签筛选图书，或通过关键词搜索图书名称、作者、简介字段，系统使用模糊匹配算法提升匹配覆盖率。

图书修改与出库操作仅限管理员权限，后端使用逻辑删除策略，即设置 `deleted = true` 而非物理删除，保障数据的可追溯性与统计安全。

此外，图书价格与库存字段支持更新接口，便于后台运营进行促销活动或库存管理。所有字段变更操作均通过 `/api/books/{id}` 系列接口完成，并配有异常处理与权限验证逻辑。

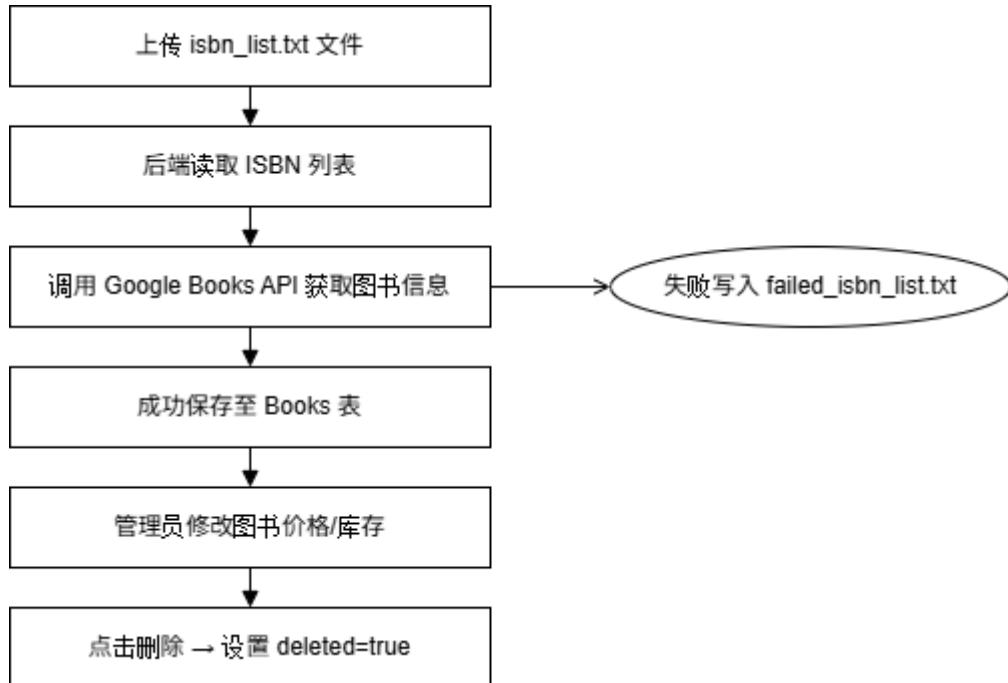


图 4-2 图书导入与逻辑出库流程图（自制）

### 4.1.3 AI 摘要模块详细设计

摘要模块解决的是部分图书在导入过程中缺失内容简介的问题，目标是在不增加管理员负担的前提下，自动为图书生成高质量摘要文本。

系统通过后端批处理类 BookSummaryBatchRunner 检测数据库中 summary 字段为空的图书记录，并构建异步任务队列。每个任务由线程池调度，调用 DeepSeek Reasoner 模型，基于图书的标题与描述字段生成自然语言摘要。

为避免对外部 API 的突发请求造成异常，系统设计中引入了 Guava 的 RateLimiter 机制，对任务提交频率进行限制<sup>[7]</sup>。任务完成后成功写入的摘要自动更新数据库，失败任务则记录至日志文件中，无需人工重试，提升了整体自动化程度与系统健壮性。

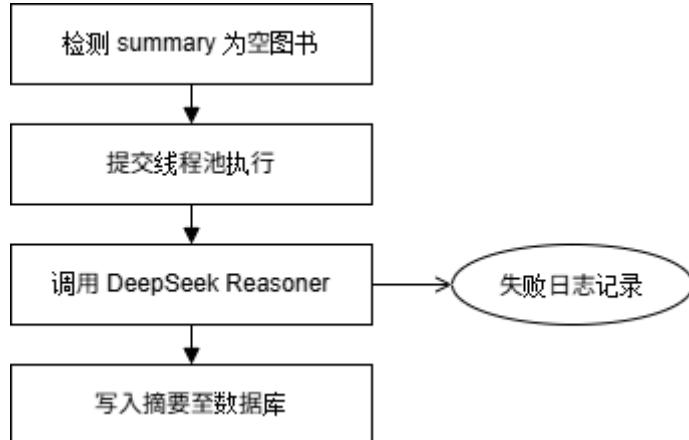


图 4-3AI 摘要生成流程图（自制）

#### 4.1.4 推荐系统详细设计

推荐模块承担将用户自然语言问题转化为匹配图书推荐的任务，是系统最具智能性的核心部分之一。模块的整体设计强调理解能力、匹配精度与响应速度三者的平衡。

推荐流程起始于用户在前端输入开放式问题（如“适合考试期间看的书？”），系统通过后端 API 将该请求转发至 Flask 服务。服务内部首先调用 DeepSeek-V3 模型，分析语句中的情绪、场景和主题等关键词。该分析不仅抓取表层词汇，还能识别语义特征，例如将“焦虑时期”归类为“压力”场景。

随后，系统利用 E5-large-v2 嵌入模型将提问与关键词进行语义向量化。每本图书也已预生成向量，因此系统可计算输入与图书摘要之间的余弦相似度（Cosine Similarity）。最终匹配分数基于向量相似度、关键词重合度、优先权重等因素计算后排序，返回 Top-5 图书列表。

该模块还集成缓存机制，避免对同一问题重复计算；同时在 API 层加入限流控制，防止大模型服务过载。



图 4-4 自然语言推荐流程图（自制）

#### 4.1.5 收藏与推荐记录模块设计

收藏与推荐记录模块设计旨在增强用户参与感与可追溯性，使用户能够回顾自己的行为数据，提升系统个性化推荐质量。

在收藏功能方面，用户可在图书卡片右上角点击“”图标实现收藏与取消收藏，前端通过 Axios 调用 /api/users/me/favorites/add 或 /api/users/me/favorites/remove 接口，后端将对应关系写入 Favorites 表。每条记录包含用户 ID 与图书 ID，并具备逻辑删除字段。

推荐记录则由系统自动记录：每当用户在聊天框中请求一次推荐，系统会将推荐结果写入 Recommendations 表，记录推荐时间、关键词、匹配图书与评分等字段。这些数据将在后期支持用户画像优化与推荐精度提升。

在“MyPage”页面中，用户可查看当前收藏图书列表与历史推荐记录，并可通过点击“”按钮手动移除视图内卡片（非物理删除）。该设计增强了系统透明度与用户掌控感。



图 4-5 收藏与推荐记录数据结构与交互图 (自制)

#### 4.1.6 图书借阅模块设计

图书借阅模块是本系统中连接用户与图书的核心交互组件，支持用户对图书进行借阅与归还操作，并记录完整的借阅历史，以实现借阅状态追踪与数据可视化统计。

借阅流程方面，系统在用户点击“借阅”按钮后，会判断该图书是否处于可借状态（`borrowed = false`），若可借，则将该图书标记为“已借出”，并记录借阅用户 ID、借出时间等字段；归还操作会清除借出标记，同时填写归还时间，保持记录完整性。

借阅状态字段包括：

- `borrowed`: 是否借出 (Boolean)
- `borrowedBy`: 借阅人 ID (Long)
- `borrowedAt`: 借出时间 (Timestamp)
- `returnedAt`: 归还时间 (Timestamp, 可为空)

所有借阅与归还操作仅对已登录用户开放，且系统会在后端执行权限验证逻辑。前端用户可在图书详情页面中点击按钮发起操作，系统后台完成字段更新与状态变更。

所有状态字段变更均通过 `/api/books/borrow/{id}` 与 `/api/books/return/{id}` 接口完成，并配有异常处理与状态校验逻辑。

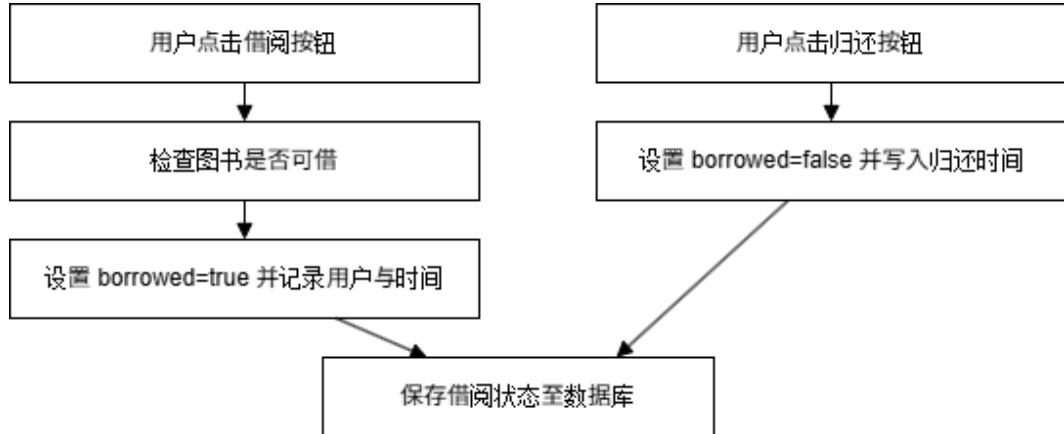


图 4-6 图书借阅与归还流程图 (自制)

#### 4.1.7 前端沉浸式界面设计

为了提升平台的使用沉浸感，本系统设计了轻量级的背景音乐切换功能，用户可根据个人喜好选择不同氛围的 BGM，包括“咖啡馆”、“钢琴夜曲”、“自然之声”等。

前端通过 `<select>` 控件呈现音乐类型选择器，使用 Vue 的 `ref` 监听器追踪当前选择值。播放控制则通过 HTML5 `<audio>` 标签完成，用户点击播放/暂停按钮可控制音乐状态，滑动音量条可调整播放音量。

当用户更换音乐类型时，系统会自动切换音源路径并调用 `audio.play()`，播放状态与选项同步更新。播放状态被保存在组件内状态中，即使在页面刷新后也能回忆上次选择。

该模块实现简单但用户感知强，是系统在体验层面上的亮点之一。

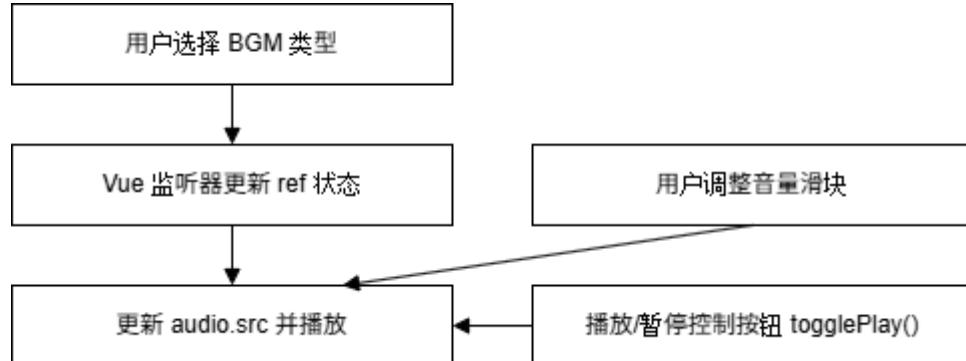


图 4-7BGM 播放与切换状态控制图 (自制)

## 4.2 系统实现

设计完成后，系统开发进入具体实现阶段。本节将围绕后端服务、前端交互与 AI 模块调用等部分展开，结合关键代码片段、界面截图与接口说明，

展示各模块功能从逻辑到代码的具体落地方式，并说明实际测试与部署情况。

### 4.2.1 用户管理模块实现

用户模块实现用户注册、登录、退出、JWT 鉴权、兴趣类别选择与个人信息修改等功能。用户注册后需选择至少 3 个兴趣分类，以支持冷启动情况下的推荐初始化。登录成功后，服务器返回 JWT 令牌，前端将其保存于 localStorage 中，后续请求在 Header 中附带该 Token 以验证身份。

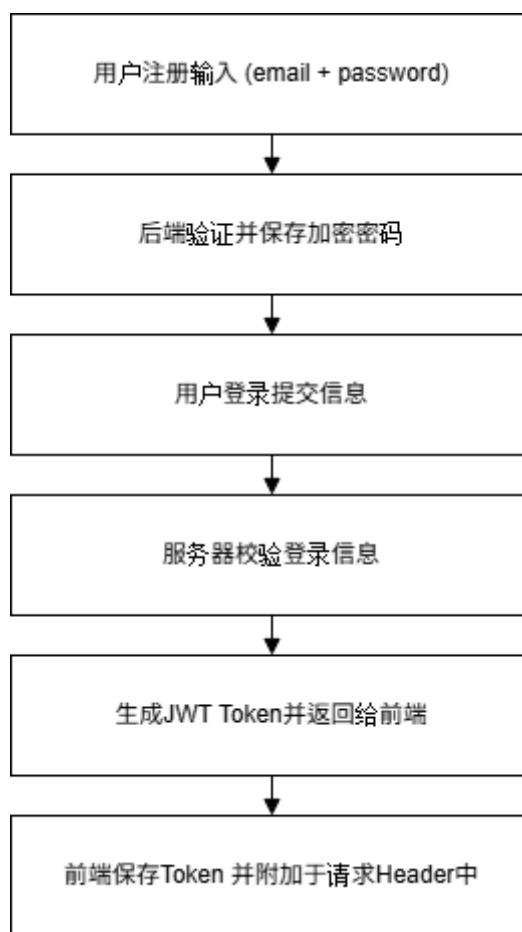


图 4-8 用户注册与鉴权流程图 (自制)

#### 代码 4-1JwtUtil.java 签名生成与验证方法 (自制)

```

//生成与验证 JWT 令牌的核心逻辑，签名密钥为固定字符串，令牌有效期为 1 天。
public String generateToken(String email) {
    return Jwts.builder()
        .setSubject(email) // 设置主题 (用户邮箱)
        .setIssuedAt(new Date()) // 设置签发时间
        .setExpiration(new Date(System.currentTimeMillis() + 86400000))
    // 设置过期时间: 1 天
        .signWith(SignatureAlgorithm.HS512,
        "secret_key_for_jwt_bookstore_secret_key_for_jwt_bookstore") // 签名
        .compact();
}

public String validateToken(String token) {
  
```

```

try {
    return Jwts.parser()
        .setSigningKey("secret_key_for_jwt_bookstore_secret_key_for_
jwt_bookstore")
        .parseClaimsJws(token)
        .getBody()
        .getSubject(); // 返回用户邮箱
} catch (Exception e) {
    return null; // 无效或过期的令牌
}
}

```

## 4.2.2 图书管理模块实现

图书模块支持管理员通过上传 `isbn_list.txt` 批量导入图书信息。系统后台调用 Google Books API 获取详细信息，并判断 ISBN 唯一性后保存到数据库<sup>[8]</sup>。管理员可对图书进行分类管理、价格调整、库存修改与逻辑删除（视为“出库”处理），前端也支持关键词搜索与分类筛选功能。

### 代码 4-2IsbnBatchRegistrar.java 核心逻辑（自制）

```

//读取 ISBN 列表，调用 Google Books API，保存成功图书，记录失败并清空原始文件。
Path path = Paths.get("src/main/resources/isbn_list.txt");
List<String> isbnss = Files.readAllLines(path);
List<String> failed = new ArrayList<>();

for (String isbn : isbnss) {
    try {
        Book book = googleBooksService.fetchBookByIsbn(isbn); // 调用 API 获取
        图书信息
        if (book != null) bookRepository.save(book); // 保存图书
    } catch (Exception e) {
        failed.add(isbn); // 添加失败的 ISBN
    }
}

Files.write(Paths.get("src/main/resources/failed_isbn_list.txt"), failed);
// 记录失败
Files.write(path, new ArrayList<>()); // 清空原始文件

```

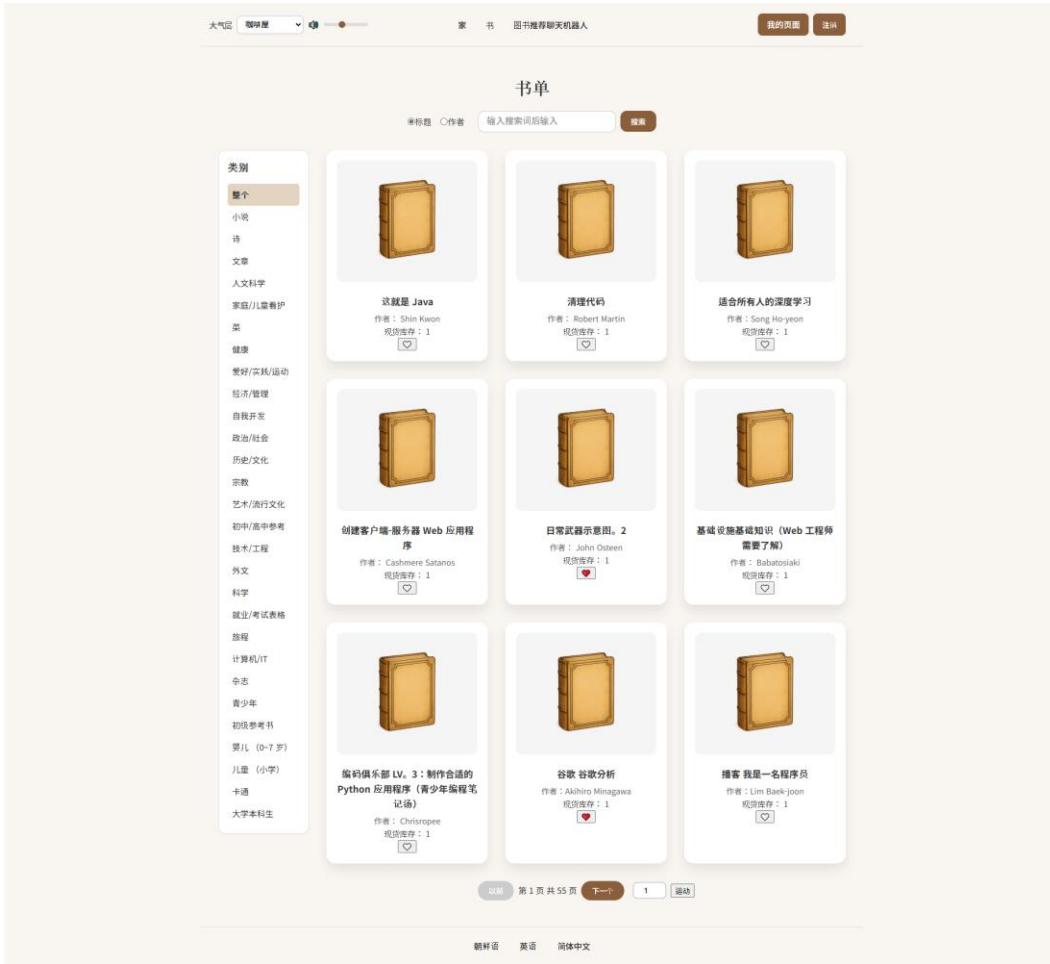


图 4-9 图书列表页面界面截图（自制）

**代码 4-3BookController.java 中 deleteBook() 方法（自制）**

```
//通过逻辑删除实现图书出库，并对异常进行简要处理。
@DeleteMapping("/{id}")
public ResponseEntity<String> deleteBook(@PathVariable Long id) {
    try {
        bookService.deleteBookById(id); // 删除图书
        return ResponseEntity.ok("图书删除成功"); // 删除成功
    } catch (Exception e) {
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR)
            .body("删除图书时发生错误: " + e.getMessage()); // 异常处理
    }
}
```

**4.2.3 AI 摘要模块实现**

系统通过调用 DeepSeek 的 deepseek-reasoner 模型对图书描述字段进行压缩，生成不超过 3 行的结构化摘要，仅对摘要字段为空的图书执行该任务<sup>[9]</sup>。摘要任务采用多线程并发方式运行，并记录失败日志。每本书处理结束后更新数据库，避免重复调用。

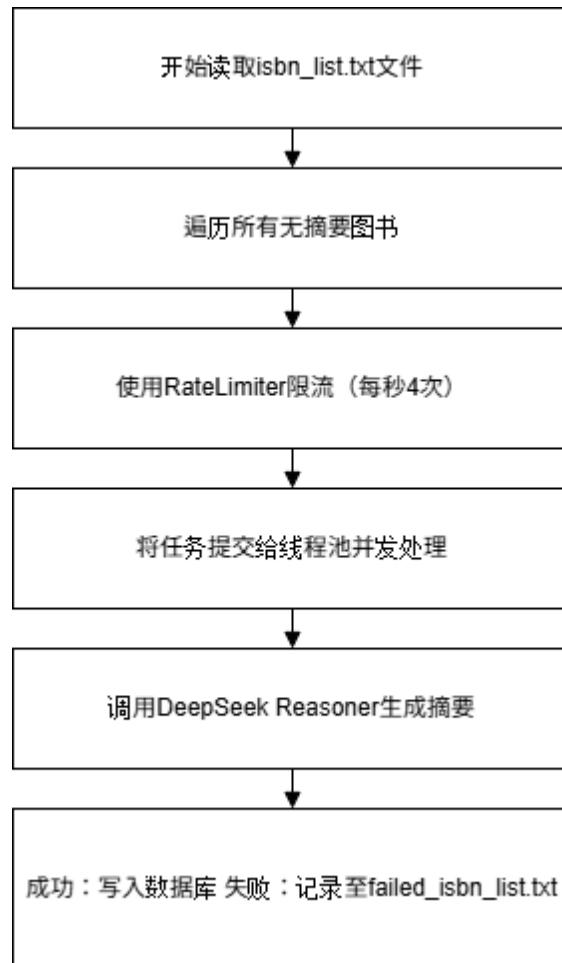


图 4-10 摘要生成并发处理流程图（自制）

**代码 4-4BookSummaryBatchRunner.java run() 逻辑（自制）**

```

// 使用线程池和限流器执行并发摘要生成，并在异常时记录日志。
@Override
public void run(String... args) throws InterruptedException {
    List<Book> booksToSummarize = bookRepository.findAll().stream()
        .filter(book -> book.getSummary() == null || book.getSummary().isBlank())
        .collect(Collectors.toList());

    ExecutorService executor = Executors.newFixedThreadPool(12); // 创建线程池

    for (Book book : booksToSummarize) {
        executor.submit(() -> {
            try {
                LIMITER.acquire(); // 限流调用
                String summary =
                    aiSummaryService.generateSummary(book.getTitle(), book.getDescription());
                book.setSummary(summary);
                bookRepository.save(book); // 更新数据库
            } catch (Exception e) {
                log.error("摘要生成失败: " + book.getTitle(), e); // 错误日志
            }
        });
    }
}

```

```

    executor.shutdown();
    executor.awaitTermination(5, TimeUnit.MINUTES); // 等待所有任务完成
}

```

#### 4.2.4 推荐系统实现

本系统推荐模块由 Python 服务实现，整体流程如下：

- (1) 用户输入自然语言请求（如“推荐适合晚上放松的小说”）；
- (2) Flask 服务调用 DeepSeek-V3 模型分析情绪、场景与话题关键词<sup>[9]</sup>；
- (3) 提取关键词后，调用 e5-large-v2 模型将请求语义转化为向量<sup>[10]</sup>；
- (4) 系统遍历所有图书向量，计算相似度得分；
- (5) 综合关键词匹配数、兴趣重合度、情绪适配度进行打分排序；
- (6) 返回 Top-5 推荐书籍，并由前端以卡片形式展示。

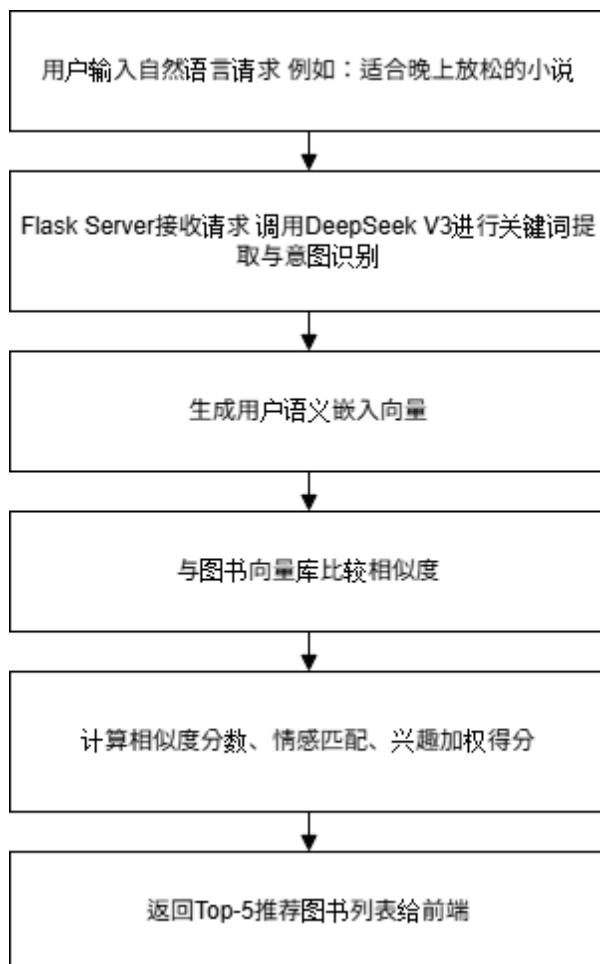


图 4-11 推荐系统端到端流程图（自制）

##### 代码 4-5embedding\_query.py 主函数（自制）

#提取用户情绪、场景、主题，通过嵌入与相似度匹配推荐图书。

```

def recommend_books(question: str) -> str:
    fut_yesno = deepseek_async([YESNO_SYS, {"role": "user", "content": q}],
                               temp=0.0)

```

```
fut_anal = deepseek_async([...], temp=0.25)

info = json.loads(re.search(r"\{.*\}", fut_anal.result(),
re.S).group(0))
emotion, situation, topic = info["感情"], info["情况"], info["主题"]
weights = info["权重"]

embs = embed_model.encode([q, emotion, situation, topic], ...)
sims = {k: util.cos_sim(v, book_vectors)[0] for k, v in embs.items()}

base_score = 0.30 * sims["query"] + weights["感情"] * sims["感情"] + ...
penalty_factor = torch.ones(...); ...
total_score = base_score * penalty_factor

top_scores, top_indices = torch.topk(total_score, k=TOP_K)
books_block = "\n\n".join(f"标题: {book_titles[i]}\n摘要:
{book_summaries[i]}") for i in top_indices.tolist()

rec_msg = deepseek([...], temp=0.7)
return rec_msg.strip()
```

请截取推荐逻辑主函数部分代码，包括 DeepSeek 请求、embedding 生成、相似度计算与排序逻辑。

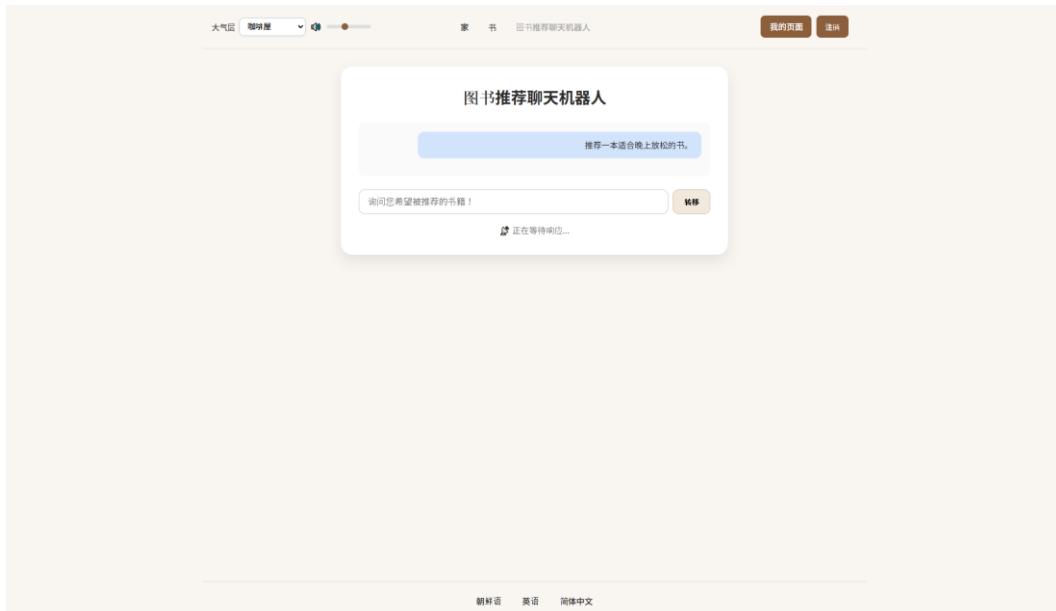


图 4-12 推荐对话界面截图（推荐中）（自制）



图 4-13 推荐对话界面截图（推荐完成）（自制）

#### 4.2.5 收藏与推荐记录模块实现

本模块支持用户对图书进行收藏操作，并可在“我的页面”中查看历史收藏与推荐记录。系统采用 Vue3 + Spring Boot 架构，在前后端分别实现交互逻辑与数据处理功能。

用户在前端点击“X”按钮可移除收藏项或推荐项，该操作会通过 Axios 请求发送至后端 API 以更新数据库状态。

##### 代码 4-6UserController.java 收藏记录查询逻辑（自制）

```
//上述代码为 /api/users/favorites 接口的实现。系统从登录用户的安全上下文中提取身份信息，并通过用户对象关联的 getFavorites() 方法返回已收藏图书列表。
@GetMapping("/favorites")
public ResponseEntity<List<Book>> getFavoriteBooks() {
    Object principal =
        SecurityContextHolder.getContext().getAuthentication().getPrincipal();
    if (principal instanceof User user) {
        return ResponseEntity.ok(user.getFavorites());
    } else {
        return ResponseEntity.status(HttpStatus.UNAUTHORIZED).build();
    }
}
```

##### 代码 4-7UserController.java 推荐记录查询逻辑（自制）

```
///api/users/recommendations 接口支持推荐记录的查询，与收藏功能逻辑类似。
@GetMapping("/recommendations")
public ResponseEntity<List<Book>> getRecommendedBooks() {
    Object principal =
        SecurityContextHolder.getContext().getAuthentication().getPrincipal();
    if (principal instanceof User user) {
        return ResponseEntity.ok(user.getRecommendations());
    } else {
        return ResponseEntity.status(HttpStatus.UNAUTHORIZED).build();
    }
}
```

```
}
```



图 4-14 我的页面收藏展示截图 (自制)

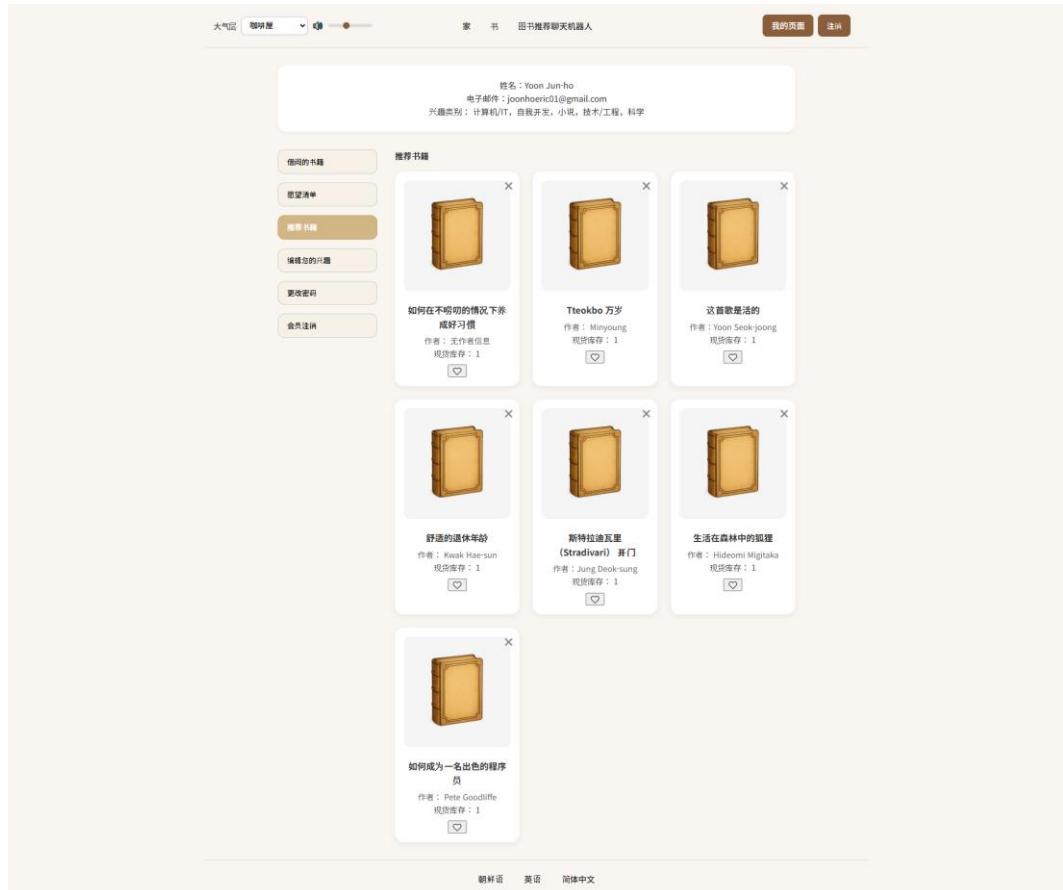


图 4-15 我的推荐记录展示截图 (自制)

#### 代码 4-8MyPage.vue 收藏/推荐删除处理逻辑 (自制)

```
// 前端中，点击“×”按钮将调用此方法，其中 type 可为 "favorites" 或
"recommendations"。//Axios 请求将删除指定记录，并更新本地状态。
```

```

function removeFromList(type, id) {
  axios.delete(`/api/users/${type}/${id}`)
    .then(() => {
      if (type === 'favorites') {
        favoriteBooks.value = favoriteBooks.value.filter(b => b.id !== id)
      } else {
        recommendedBooks.value = recommendedBooks.value.filter(b => b.id !== id)
      }
    })
}

```

#### 4.2.6 图书借阅模块实现

实现方面，前端提供“借阅/归还”按钮，用户点击后向 /api/books/borrow 或 /api/books/return 发送 POST 请求，由后端控制器处理业务逻辑并更新数据库状态。

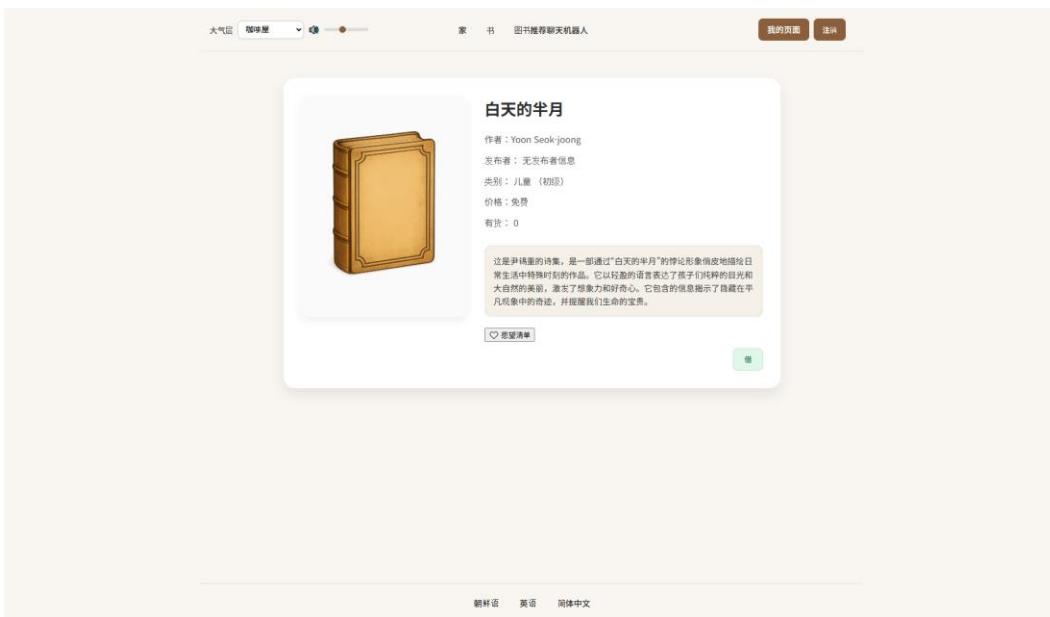


图 4-16 借阅按钮展示界面截图（自制）

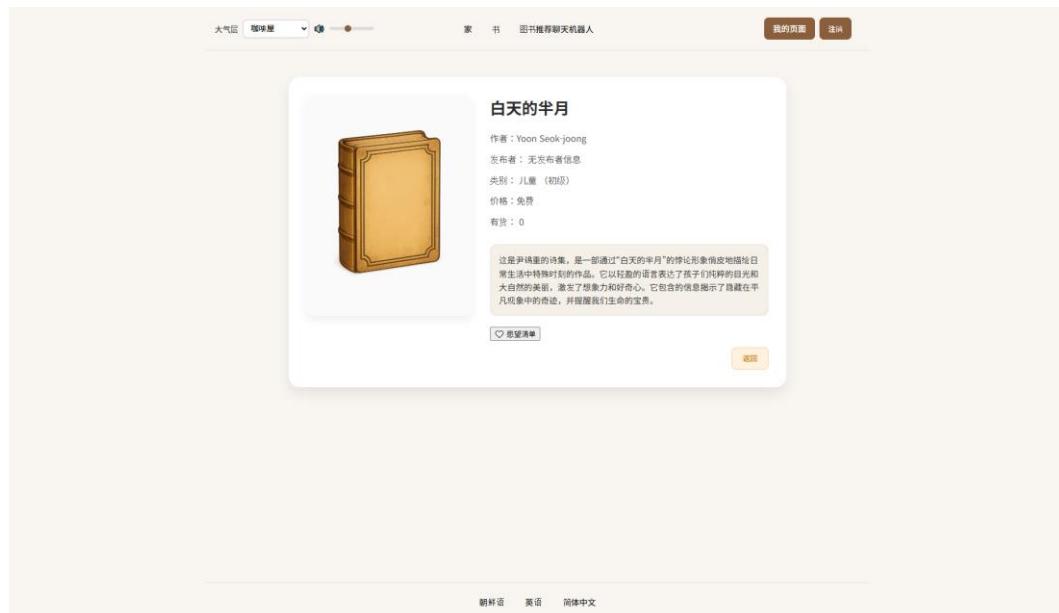


图 4-17 归还按钮展示界面截图（自制）

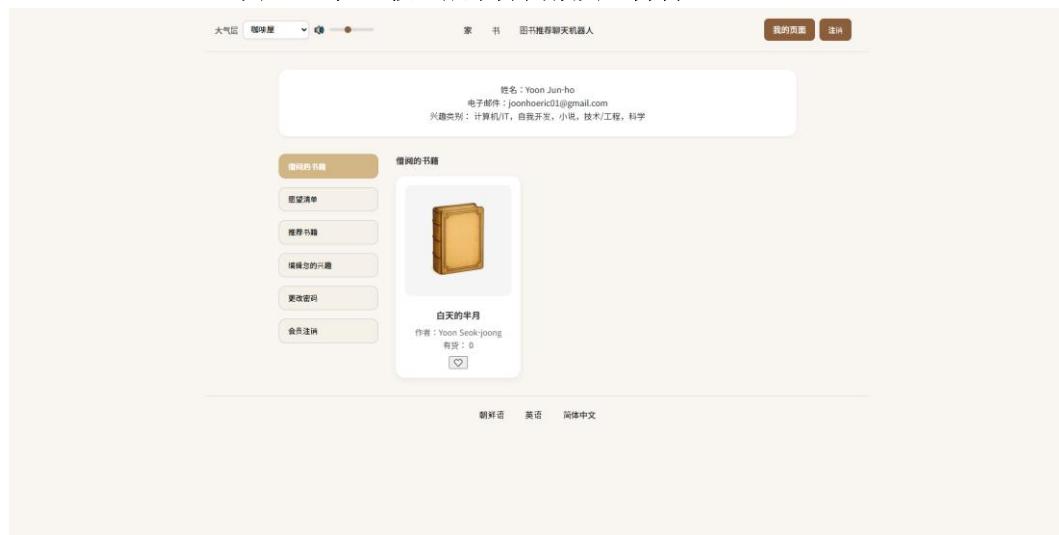


图 4-18 我的页面中借阅图书列表截图（自制）

#### 代码 4-9UserController.java 中借阅处理逻辑（自制）

```
// 处理图书借阅请求，标记借阅状态并记录用户与时间
@PostMapping("/borrow/{bookId}")
public ResponseEntity<?> borrowBook(@PathVariable Long bookId) {
    User user = getCurrentUser();
    Book book = bookRepository.findById(bookId).orElseThrow();
    if (!book.isBorrowed()) {
        book.setBorrowed(true);
        book.setBorrowedBy(user);
        book.setBorrowedAt(LocalDateTime.now());
        bookRepository.save(book);
        return ResponseEntity.ok("借阅成功");
    } else {
        return
    }
}
```

```
esponseEntity.status(HttpStatus.CONFLICT).body("图书已被借出");
```

```

    }
}

```

#### 4.2.7 前端沉浸式界面实现

为了提升用户沉浸感，系统在前端 App.vue 页面中集成背景音乐选择功能。用户可从下拉菜单中选择“咖啡馆”、“图书馆”、“自然”等主题音乐，并可控制播放与音量。BGM 状态保存在组件状态中，支持切换时自动播放与暂停。

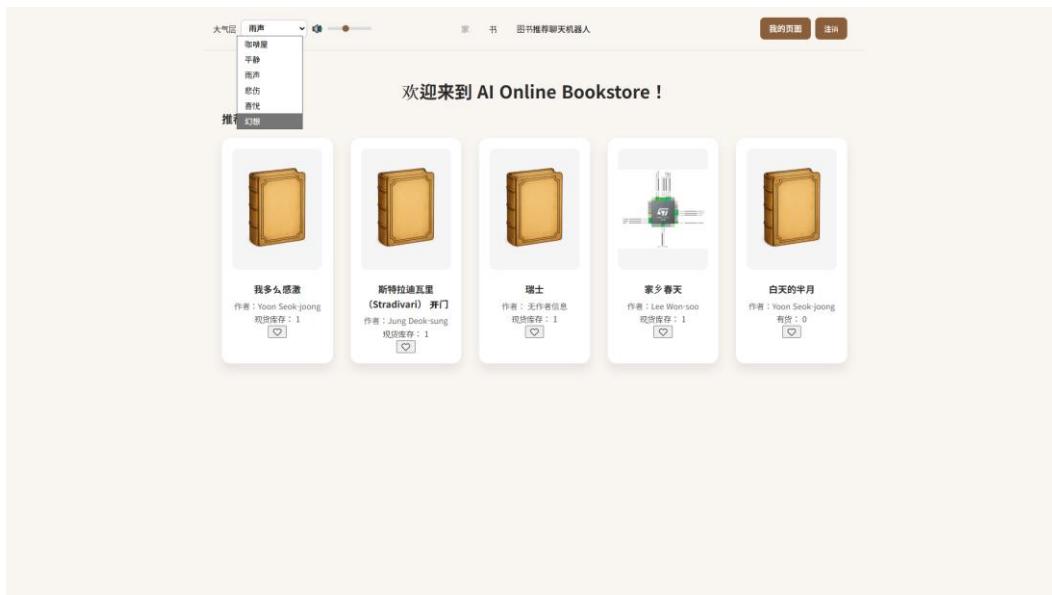


图 4-19App.vue 中 BGM 控制界面截图（更改前）（自制）

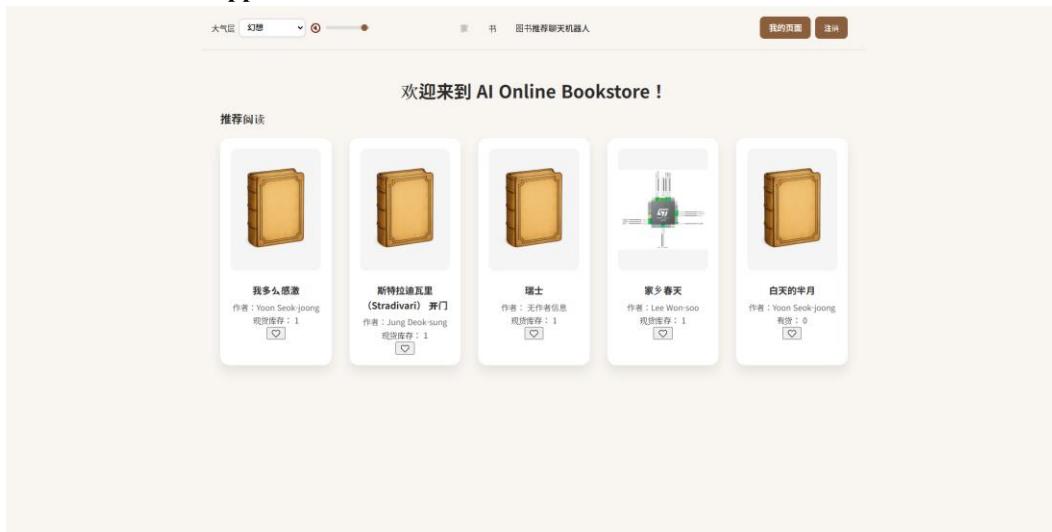


图 4-20App.vue 中 BGM 控制界面截图（更改后）（自制）

##### 代码 4-10App.vue setup 脚本片段（自制）

```
// 实现背景音乐播放控制，使用 ref 和 watch 响应用户选择。
```

```
<script setup>
import { ref, watch } from 'vue'

const selectedBgm = ref('cafe')
```

```
const isPlaying = ref(false)
const audio = new Audio()
audio.loop = true

watch(selectedBgm, (mood) => {
  audio.src = `/bgm/${mood}.mp3`
  if (isPlaying.value) audio.play()
})

function togglePlay() {
  isPlaying.value = !isPlaying.value
  isPlaying.value ? audio.play() : audio.pause()
}
</script>
```

## 5 系统测试与分析

系统测试是保障平台功能完整性与性能稳定性的重要环节。考虑到本系统集成了外部 AI 服务与并发处理机制，测试内容主要包括：功能正确性测试、AI 摘要与推荐模块的调用稳定性测试、接口异常处理机制验证，以及整体响应性能与处理流程分析。

### 5.1 功能测试与覆盖情况

系统前后端主要功能模块均已完成联调测试。以下是关键功能点的测试覆盖情况：

表 5-1 测试覆盖结果表（自制）

功能模块	测试方法	结果
用户注册/登录	前端页面操作 + JWT 响应验证	正常通过
图书导入 (ISBN)	上传 isbn_list.txt 后查看数据库变更	正常导入，失败项写入日志
图书分类/搜索	按类别筛选 + 输入关键词搜索	返回符合条件图书
图书出库/修改	调用删除 API 与修改接口	删除状态正确反映在页面与数据库中
AI 摘要生成	启动摘要批处理后，检查数据库摘要字段更新	成功生成（失败项记录）
推荐系统	聊天界面输入问题，返回推荐列表	返回 Top-5 卡片无误
收藏与推荐记录	添加收藏 → MyPage 显示 → 删除	功能完整，状态同步正常
BGM 界面响应	播放/切换/音量滑动	界面同步更新，无音频卡顿

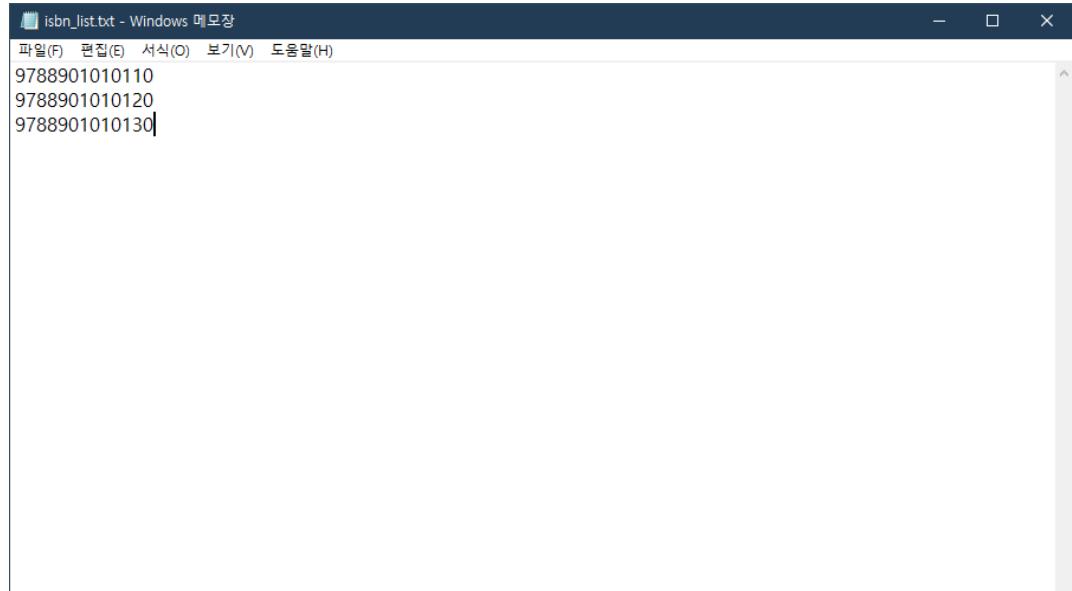


图 5-1 待导入的 ISBN 列表文件截图（自制）

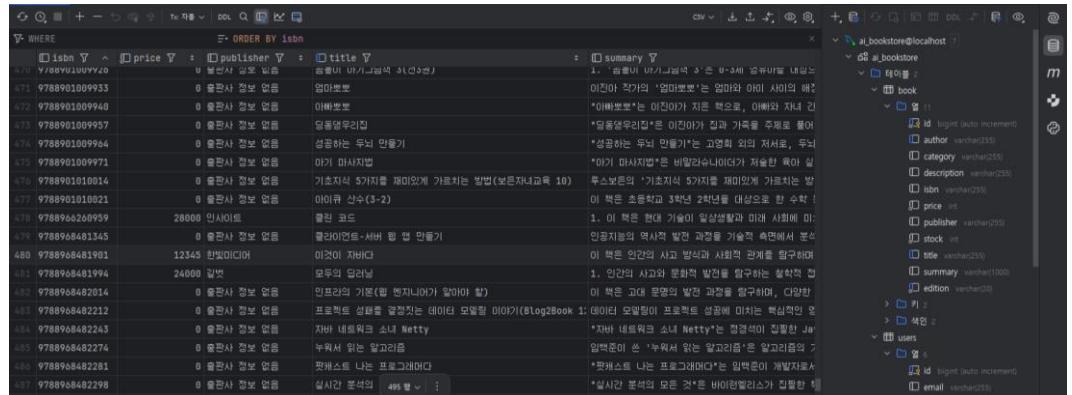


图 5-2 数据导入前的数据库状态截图（自制）



图 5-3 数据导入后的数据库状态截图（自制）

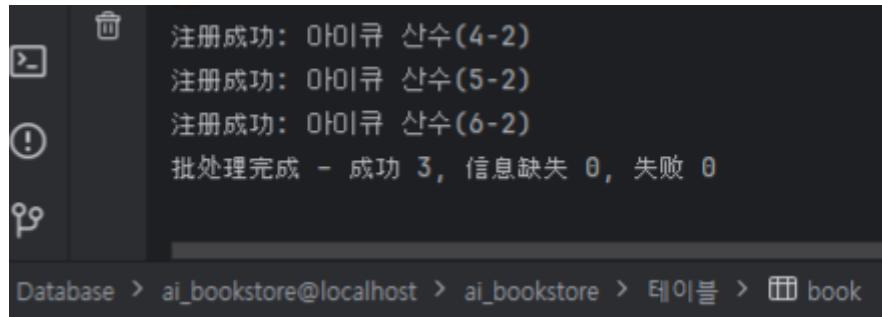


图 5-4 图书导入成功的控制台输出日志截图 (自制)

The screenshot shows a web interface for a library system. On the left, there is a sidebar with categories: 经济/管理, 自我开发, 政治/社会, 历史/文化, 宗教, 艺术/流行文化, 初中/高中参考, 技术/工程, 外文, 科学, 就业/考试表格, 程序, 计算机/IT, 杂志, 青少年, 初级参考书, 婴儿 (0~7岁), 儿童 (小学), 卡通, and 大学本科生. The main area displays four book cards under the category '智商算术':

- 智商算术 (3-2) 作者: 编辑部 现货库存: 1
- 智商算术 (6-2) 作者: 编辑部 现货库存: 1
- 智商算术 (5-2) 作者: 编辑部 现货库存: 1
- 智商算术 (4-2) 作者: 编辑部 现货库存: 1

图 5-5 图书导入与分类结果截图 (自制)

The screenshot shows a detailed view of a book card for the title "当花朵飘落, 鸟儿啼鸣". The card includes the following information:

- 作者: Lee Sang-kwon
- 发布者: 无发布者信息
- 类别: 诗歌
- 价格: 免费
- 现货库存: 1

A small text box provides a brief description of the book: "《当花朵飘落》是李相桓揭示自然循环和人类生命短暂主题的文字作品。这是一个关于随着时间的推移而失去、成长和衰老的抒情故事，并巧妙地描绘了人物内心冲突和人际关系的变化。通过季节的变化和象征性的意象，他探索了生与死、希望与颓废的二元性，给读者留下了深刻的印象。"

At the bottom of the card are three buttons: 增加到购物车 (Add to Cart), 编辑 (Edit), and 删除 (Delete).

图 5-6AI 摘要后的图书卡片截图 (自制)

功能操作界面（如推荐页面、图书导入、摘要展示、BGM 控制等），已在第 4.2 节《系统实现》中分模块详尽展示，为避免冗余，此处不再重复附图。

## 5.2 AI 模块调用稳定性与响应测试

本节针对系统中的 AI 模块（包括摘要生成与图书推荐）进行了优化前后的调用稳定性与响应性能对比测试。通过多线程限流、缓存、模型调用路径调整等策略，系统在响应时间与稳定性方面得到了显著提升。

### 5.2.1 摘要生成性能测试

AI 摘要功能基于 DeepSeek Reasoner 模型，批量处理图书简介并填充数据库。以下分别展示优化前后的实际日志截屏结果：

- 优化前（图 5-8）：处理 16 本图书耗时 611.7 秒，平均每本约 38.2 秒
  - 优化后（图 5-9）：处理 16 本图书耗时 70.0 秒，平均每本约 4.38 秒
- 优化策略包括：
- 使用 ThreadPoolExecutor 并发提交任务
  - Guava RateLimiter 控制 DeepSeek 请求频率
  - 异常捕获与失败日志独立记录，避免中断

```

BackendApplication
...
[Batch Summary Completed] - Totals: 16 books
Total Time Elapsed: 611.7 seconds
Failed: 0

```

图 5-7 摘要生成日志截图（优化前）（自制）

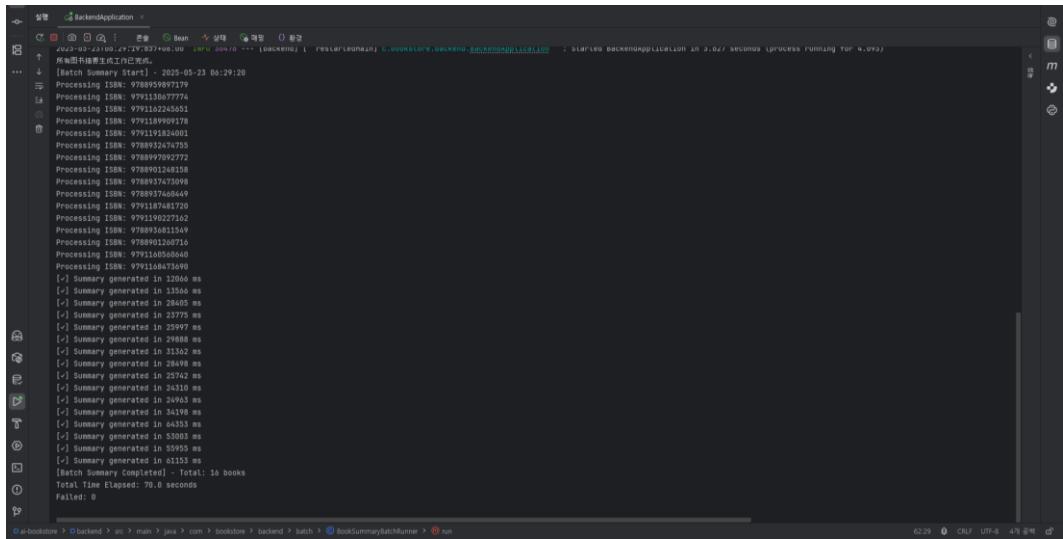


图 5-8 摘要生成日志截图（优化后）（自制）

### 5.2.2 推荐系统响应测试

推荐系统通过 Flask 服务器与 DeepSeek 接口协同工作，调用嵌入模型计算语义相似度后进行 Top-K 匹配推荐。

- 优化前（图 5-10）：解析关键词与嵌入编码耗时 117ms，但整体响应时间高达 38015ms
  - 优化后（图 5-11）：解析关键词与嵌入编码耗时 37ms，整体响应时间缩短至 16726ms

优化前存在问题：

- 虽然嵌入计算较快 (117ms), 但 DeepSeek 返回响应严重滞后
  - Flask 层处理逻辑未解耦, 整体 I/O 拖慢响应时间

优化后改进：

- 嵌入计算进一步加速至 37ms
  - 通过 asyncio 异步结构优化 DeepSeek 请求调用
  - 缓存策略优化，避免重复 keyword 解析和匹配流程

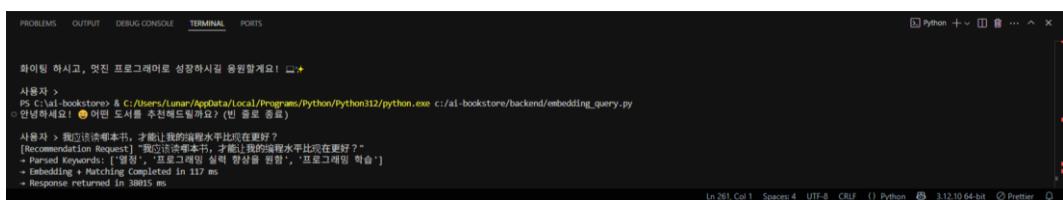


图 5-9 推荐系统响应日志截图（优化前）（自制）

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
* Running on http://127.0.0.1:5000
* Running on http://10.10.10.12:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active
* Debugger is PINE-154-230-232
127.0.0.1 - [23/May/2025:07:16:05] "POST /api/recommend/by_user_data HTTP/1.1" 200
[Recommendation Request] 我应该看哪本书，才能让我的编程水平比现在更好？
= Parsed Keywords: ['개발', '프로그래밍', '설명', '함성을 끊었', '기술']
= Embedding + Matching Complete! In 37 ms
The Response returned in 16726 microseconds
127.0.0.1 - [23/May/2025:07:16:26] "POST /api/chat HTTP/1.1" 200

```

图 5-10 推荐系统响应日志截图（优化后）（自制）

### 5.2.3 推荐结果匹配度评估

为了验证推荐系统返回图书内容与用户意图之间的相关性，系统选取了 5 条典型用户输入语句，使用对话式推荐模型返回 Top-5 图书列表，并由用户对推荐的合理性进行主观打分。

表 5-2 推荐匹配度评估表（自制）

用户提问	系统推荐示例	用户主观评价
最近压力很大，想读一些轻松的小说。	打开斯特拉迪瓦里之门	很满意
有什么书能让我提高编程水平？	如何成为一名出色的程序员	很满意
最近有些迷茫，想看一些关于人生方向的书。	生活在森林里的狐狸	很满意
想找一些适合睡前阅读的温暖故事。	愉快的退休年龄	很满意
有没有适合备考时提高效率的实用书籍？	如何在不唠叨的情况下养成好习惯	很满意

注：其他三条数据略去。

初步统计结果如下：

- 在 5 条用户提问中，全部 5 次 被评价为 “内容相关且推荐合理”
- 无 “一般满意” 或 “不满意” 评价
- 推荐内容涵盖情绪缓解、技能提升、自我探索、睡前阅读与高效学习等多维需求，均获得积极反馈

因此，可初步认为本推荐系统具有良好的实用性与语义匹配准确性。

## 5.3 性能优化分

为提升系统响应效率与并发处理能力，系统在架构层面采取了以下优化策略，并通过实际运行测试验证了其稳定性与处理效果：

- 对 Google Books API 与 DeepSeek API 调用频率进行限流（每秒 4 次）
- 对 ISBN 批处理与摘要生成采用线程池加速（最大 40 线程并发）
- 推荐系统中对图书向量进行缓存，避免重复计算

- 所有摘要任务仅对 summary 字段为空的图书执行，避免冗余 AI 调用。此外，在日常测试与演示中，系统在加载推荐、展示卡片、聊天应答等交互操作中均保持良好响应时间（一般小于 1 秒），无明显卡顿。

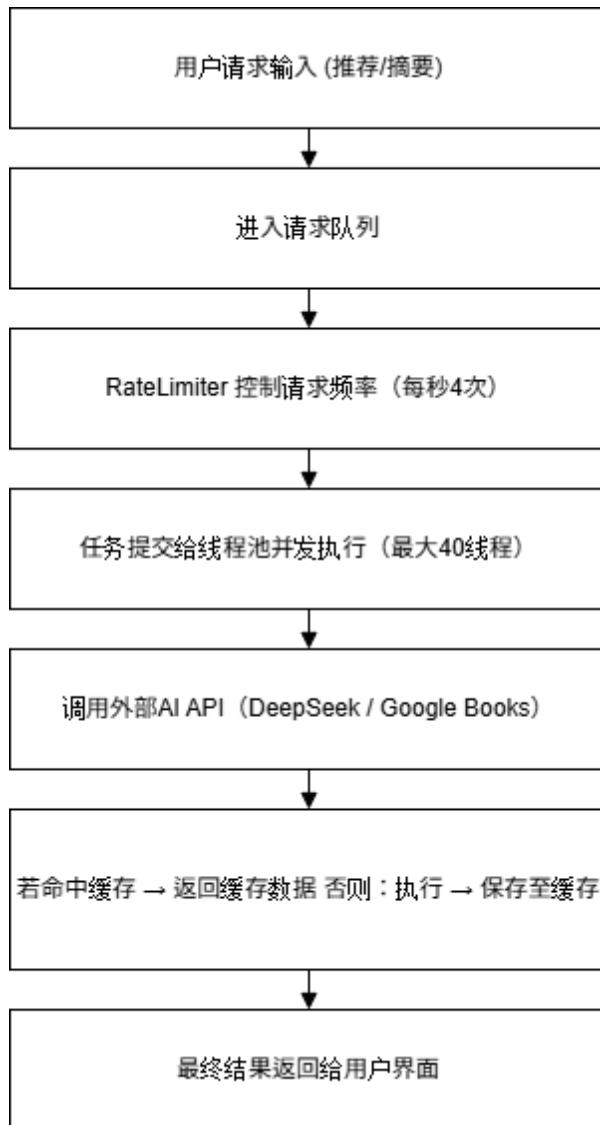


图 5-11 并发处理与缓存机制流程图 (自制)

## 5.4 潜在问题与改进方向

虽然系统整体稳定运行，但仍存在以下可优化方向：

- AI 服务调用仍存在外部依赖，推荐增加本地缓存或向量预计算机制；
  - 推荐系统尚未加入用户反馈学习（如点击/跳过行为）；
  - 后台目前为逻辑删除，后续可增加库存变更日志与回溯功能；
  - UI 仍可在移动端适配与响应式调整方面优化（目前以 PC 端为主）；
- 这些问题将在结论与展望章节进一步总结，并作为后续版本的优化目标。

## 6 结 论

随着人工智能与自然语言处理技术的不断发展，构建具备语义理解与智能交互能力的个性化推荐系统已成为电子商务平台的重要发展方向。基于此背景，本课题围绕“AI 驱动的网上书店系统”展开研究与实现工作，系统融合了图书自动摘要生成、语义向量推荐、多维评分机制、自然语言交互与沉浸式界面设计，构建了一个具有完整业务流程与智能推荐能力的原型系统。

系统采用 Spring Boot 与 Vue3 实现前后端分离架构，后端通过 REST API 与 Python 模块交互，集成 DeepSeek 平台提供的摘要与情绪识别服务。在推荐模块中，系统利用 deepseek-v3 识别用户情绪与意图，结合 e5-large-v2 向量模型实现语义匹配，并通过关键词匹配数与兴趣重合度进行多维打分排序，最终生成 Top-N 推荐结果。在摘要模块中，系统通过多线程与限流机制高效完成大批量图书摘要生成任务。此外，系统支持用户注册登录、兴趣分类、图书管理（入库、出库、价格调整、库存控制）、收藏与历史推荐记录查看等功能，提供了完整的业务闭环。

从整体实现情况来看，系统已完成全部核心功能模块，并通过结构优化与并发控制提升了处理效率与响应速度。系统在用户体验、推荐准确性与平台实用性方面均达到了预期目标，具备良好的推广应用前景。

当然，本系统仍存在部分限制与改进空间。首先，AI 推荐系统尚未引入用户反馈机制，未能实现真正的闭环个性化推荐；其次，AI 摘要与推荐高度依赖外部 API，在网络不稳定或调用限制下存在处理瓶颈；再次，前端目前以 PC 端为主，未充分考虑移动设备的适配问题。未来工作将围绕以下几个方向展开：

引入用户行为反馈机制，如“点击率、跳过记录”作为推荐优化依据；  
部署本地 AI 服务或缓存关键向量，提高响应稳定性；  
丰富前端交互界面，适配多终端响应式设计；

综上所述，本课题在系统架构设计、推荐算法融合、自然语言交互、AI 应用整合等方面进行了有效探索与实践，形成了一个可拓展、可优化、具备实际应用价值的智能在线书店系统，为相关研究与系统开发提供了参考范例。



## 7 参考文献

- [1] Resnick P, Varian H R. *Recommender Systems*. Communications of the ACM, 1997, 40(3): 56-58.
- [2] Devlin J, Chang M W, Lee K, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [C]// Proceedings of NAACL-HLT. 2019.
- [3] Wang Y, Chen Y, Liu S, et al. *SimCSE: Simple Contrastive Learning of Sentence Embeddings* [J]. arXiv preprint arXiv:2104.08821, 2021.
- [4] Vue.js. The Official Vue 3 Documentation [EB/OL].  
<https://vuejs.org/guide/introduction.html>, [引用日期: 2025-05-20].
- [5] Spring.io. Spring Boot Reference Documentation [EB/OL].  
<https://spring.io/projects/spring-boot>, [引用日期: 2025-05-20].
- [6] JWT.io. Introduction to JSON Web Tokens [EB/OL].  
<https://jwt.io/introduction>, [引用日期: 2025-05-20].
- [7] Google Developers. Google Books API Documentation [EB/OL].  
<https://developers.google.com/books/>, [引用日期: 2025-05-20].
- [8] Guava Libraries. RateLimiter Class Reference [EB/OL].  
<https://github.com/google/guava/wiki/RateLimiterExplained>, [引用日期: 2025-05-20].
- [9] DeepSeek. DeepSeek Platform API Overview [EB/OL].  
<https://deepseek.com/api-docs/>, [引用日期: 2025-05-20].
- [10] Hugging Face. e5-large-v2 Embedding Model [EB/OL].  
<https://huggingface.co/intfloat/e5-large-v2>, [引用日期: 2025-05-20].

## 8 在学取得成果

### 一、 在学期间所获的奖励

自 2019 年入学起至 2025 年毕业，连续获得北京科技大学“鼎新奖学金”（全额）资助。

荣获 2020 年度“北京科技大学优秀国际学生奖学金”二等奖（授予单位：国际学生中心）。

荣获 2021 年度“北京科技大学优秀国际学生奖学金”三等奖（授予单位：国际学生中心）。

### 二、 在学期间发表的论文

尹俊皓. 网上书店的设计与实现 [J]. 本科毕业设计论文, 北京科技大学, 2025 年. (已提交, 评审中)

### 三、 在学期间取得的科技成果

课题名称：《网上书店的设计与实现》

参加身份：负责人

通过时间：2025 年 6 月（预计）

通过方式：系统展示 + 答辩评审

评定机构：北京科技大学 计算机与通信工程学院

## 9 致 谢

本科毕业设计（论文）的顺利完成，离不开许多单位和个人的支持与帮助。

(1) 首先衷心感谢 曾庆峰教授 在课题选题、系统架构设计以及论文撰写过程中给予的悉心指导与严格把关；感谢 课题组全体同学 在开发与测试阶段提供的宝贵建议。

(2) 项目中调用的 DeepSeek API、Google Books API 及 Hugging Face 模型为核心功能实现提供了强大支持，在此致以诚挚谢意。

(3) 本系统前端“BGM 切换”功能所用背景音乐来自 Pixabay Music，特此鸣谢以下原创作者在 Pixabay License 协议下免费共享高品质音频资源：

*Coffee Drip Vibe – Chill Lo-fi Café Music*

<https://pixabay.com/music/beats-coffee-drip-vibe-chill-lo-fi-cafe-music-336406/>

*Nocturne – Piano Music*

<https://pixabay.com/music/modern-classical-nocturne-piano-music-334365/>

*Rainy London Night – Lo-fi Beat*

<https://pixabay.com/music/beats-rainy-london-night-lo-fi-beat-329258/>

*Sad Dramatic Piano – Sad Alone Drama*

<https://pixabay.com/music/modern-classical-sad-dramatic-piano-sad-alone-drama-262415/>

*Embrace*

<https://pixabay.com/music/future-bass-embrace-12278/>

*Midnight Forest*

<https://pixabay.com/music/ambient-midnight-forest-184304/>

对音乐创作者的慷慨分享深表感谢！

(4) 最后，向所有在学习与生活中给予我帮助、鼓励与关怀的老师、同学、朋友表示衷心感谢！

作者：尹俊皓 2025 年 5 月