logo：

徽标, 公司名称

描述已自动生成

### 详

### 细

### 设

### 计

**G02小组**

**二手书交易系统**

目录

[**1. 概述 3**](#_Toc23572)

[1.1 编写目的 3](#_Toc25957)

[1.2 项目背景 3](#_Toc31834)

[1.3 系统需求 4](#_Toc8048)

[1.4 参考资料 4](#_Toc15419)

[**2. 系统架构设计 4**](#_Toc23015)

[2.1 系统架构 4](#_Toc21038)

[**采用前后端分离的三层架构： 4**](#_Toc11561)

[**1. 表现层：Vue.js，用户交互界面。 4**](#_Toc27059)

[**2.业务逻辑层：Spring Boot RESTful 服务。 4**](#_Toc16595)

[**3.数据访问层：MyBatis 与 MySQL 交互。2.2 运行环境 4**](#_Toc30124)

[操作系统：Windows 10/MacOS 4](#_Toc18216)

[**3. HIPO图设计展示 5**](#_Toc26252)

[**4. 用户管理模块 6**](#_Toc24841)

[4.1 功能描述 6](#_Toc27397)

[4.2 界面设计 6](#_Toc5739)

[4.3 数据库设计 7](#_Toc22296)

[4.4关键算法设计 7](#_Toc12197)

[4.5 PDL详细设计 7](#_Toc27828)

[**5. 书籍管理模块 8**](#_Toc24081)

[5.1 功能描述 8](#_Toc23116)

[5.2 界面设计 9](#_Toc22228)

[5.3 数据库设计 9](#_Toc9988)

[5.4 关键算法设计 9](#_Toc22730)

[5.5 PDL详细设计 9](#_Toc16744)

[**6. 交易管理模块 10**](#_Toc7911)

[6.1 功能描述 10](#_Toc19666)

[6.2 界面设计 10](#_Toc2701)

[6.3 数据库设计 10](#_Toc27355)

[6.4 关键算法设计 11](#_Toc28070)

[6.5 PDL详细设计 11](#_Toc27028)

[**7. 评论与反馈模块 12**](#_Toc23177)

[7.1 功能描述 12](#_Toc17504)

[7.2界面设计 12](#_Toc20455)

[7.3 数据库设计 12](#_Toc5274)

[7.4 关键算法设计 13](#_Toc2397)

[7.5 PDL详细设计 13](#_Toc14446)

[**If valid then 13**](#_Toc16735)

[**8. 通知管理模块 14**](#_Toc22411)

[8.1 功能描述 14](#_Toc2767)

[8.2界面设计 14](#_Toc9911)

[8.3 数据库设计 14](#_Toc27221)

[8.4 关键算法设计 15](#_Toc28714)

[8.5 PDL详细设计 15](#_Toc26766)

[**9. 系统管理模块 16**](#_Toc11746)

[9.1功能描述 16](#_Toc11734)

[9.2界面设计 16](#_Toc27612)

[9.3数据库设计 16](#_Toc12055)

[9.4关键算法设计 17](#_Toc2223)

[9.5PDL详细设计 17](#_Toc126)

[**10. 搜索与展示模块 18**](#_Toc0)

[10.1功能描述 18](#_Toc11930)

[10.2界面设计 18](#_Toc13722)

[10.3数据库设计 18](#_Toc22944)

[10.4关键算法设计 18](#_Toc11817)

[10.5 PDL详细设计 19](#_Toc27568)

[**11. 总结 20**](#_Toc19176)

#### 1. 概述

##### 1.1 编写目的

* 本文档旨在为二手书交易平台的开发提供详细的指导方案，包括系统的需求、架构设计、功能模块划分、接口设计、数据结构设计、运行设计、出错处理设计、安全保密设计及维护设计等内容。本文档适用于项目管理者、系统设计师、开发人员及测试人员。

##### 1.2 项目背景

**项目的发起单位**：浙大城市学院软件工程G02小组

**项目名称**：二手书交易平台

**项目目的**：创建一个方便学生买卖二手书籍的在线平台，促进资源的循环利用，减轻学生经济负担。

##### 1.3 系统需求

##### **功能需求：**用户管理、书籍管理、交易管理、评论与反馈、通知管理、系统管理、搜索与展示。

##### **非功能需求：**性能优越、操作简单、安全性高。

##### **运行环境：**

##### **前端：**PC和移动端浏览器（HTML5 + Vue.js）。

##### **后端：**Spring Boot 3.3.5，MyBatis。

##### **数据库：MySQL。**

##### 1.4 参考资料

GB/T 11457-2006 软件工程术语

GB/T 8566-2007 计算机软件开发规范

GB/T 16260-2006 软件工程 产品质量

#### 2. 系统架构设计

##### 2.1 系统架构

##### 采用前后端分离的三层架构：

##### 1. 表现层：Vue.js，用户交互界面。

##### 2.业务逻辑层：Spring Boot RESTful 服务。

##### 3.数据访问层：MyBatis 与 MySQL 交互。2.2 运行环境

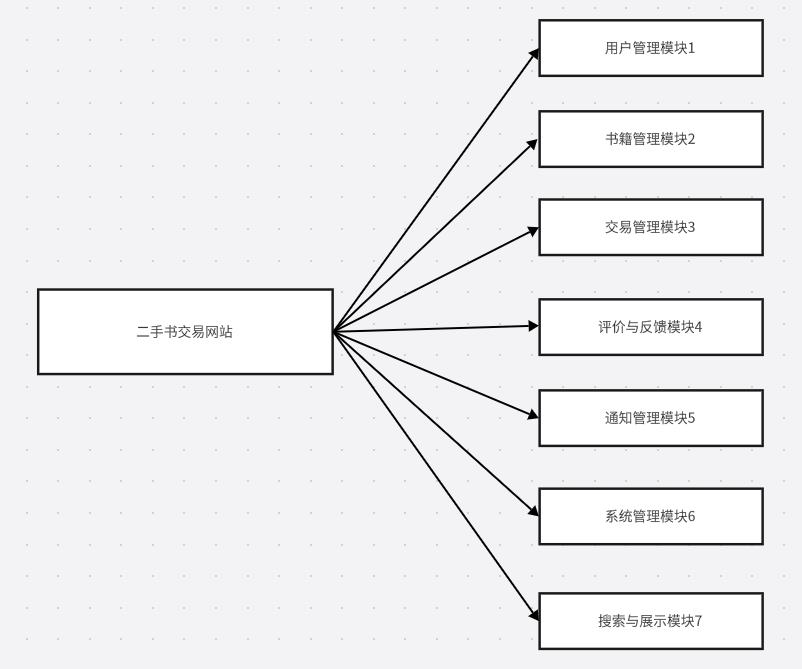
操作系统：Windows 10/MacOS

数据库：MySQL 5.7

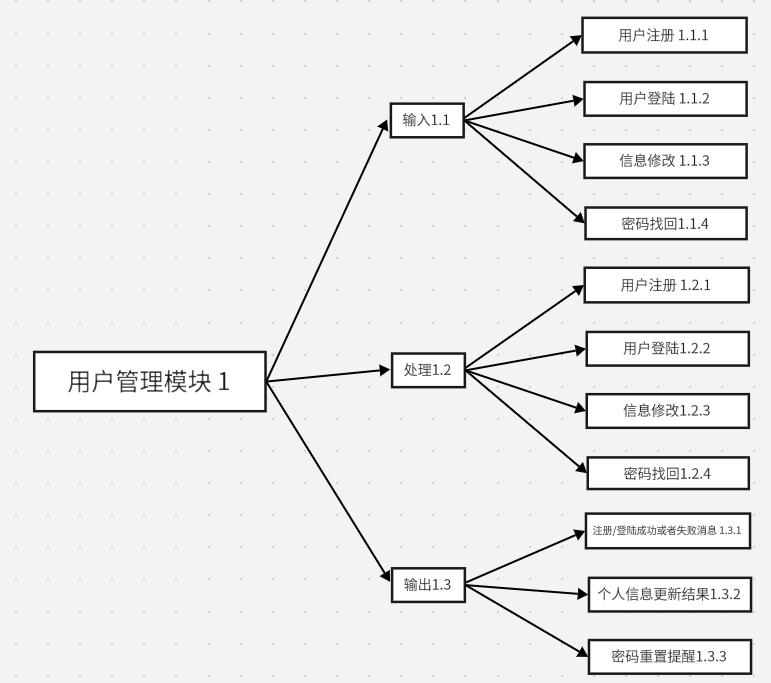
前端技术栈：HTML5/CSS3/JavaScript (React/Angular)

后端技术栈：Node.js/Express.js

#### 3. HIPO图设计展示



#### 4. 用户管理模块



##### 4.1 功能描述

* - 用户注册：新用户填写用户名、密码和邮箱/手机号完成注册。
* - 用户登录：通过用户名和密码验证登录，管理会话。
* - 信息修改：用户可以修改昵称、密码或其他信息。
* - 密码找回：通过邮箱或手机号重置密码。

##### 4.2 界面设计

* - 注册页面：包含用户名、密码、邮箱输入框，带注册按钮。
* - 登录页面：包含用户名、密码输入框，带登录和“忘记密码”按钮。
* - 个人信息修改页面：显示当前用户信息，提供修改选项。

##### 4.3 数据库设计

##### 图形用户界面 描述已自动生成4.4关键算法设计

* 检查用户唯一性
* ```java
* public boolean checkUsername(String username) {
* return userRepository.findByUsername(username) == null;
* }
* ```

##### 4.5 PDL详细设计

- 用户注册流程

BEGIN

INPUT 用户名、密码、邮箱

VALIDATE 用户名唯一性

IF 不唯一 THEN

OUTPUT "用户名已存在"

EXIT

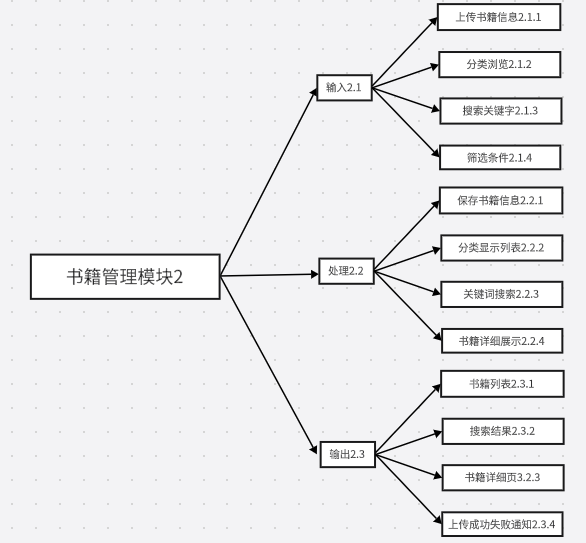
ENDIF

INSERT 用户信息 INTO 数据库

OUTPUT "注册成功"

END

#### 5. 书籍管理模块



##### 5.1 功能描述

* - 上传书籍：用户上传书籍基本信息和封面图片。
* - 书籍浏览：按分类展示书籍，提供分页功能。
* - 关键词搜索：用户输入关键词，检索书籍标题和作者。
* - 书籍筛选：支持按价格、类别等筛选书籍。

##### 5.2 界面设计

* - 书籍上传页面：包含标题、作者、价格、类别输入框，带上传图片按钮。
* - 书籍列表页面：以卡片形式显示书籍封面、标题、价格、购买链接。
* - 书籍详情页面：展示书籍详细信息、卖家联系方式。

##### 5.3 数据库设计

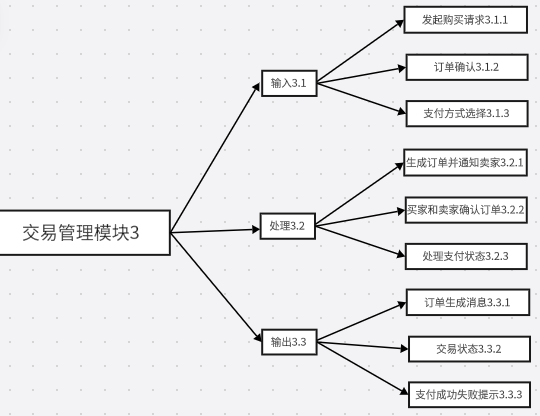
##### 表格 描述已自动生成5.4 关键算法设计

* \*\*关键词搜索算法\*\*
* ```java
* public List<Book> searchBooks(String keyword) {
* return bookRepository.findByTitleContaining(keyword);
* }
* ```

##### 5.5 PDL详细设计

* 关键词搜索流程
* BEGIN
* INPUT 搜索关键词
* QUERY 数据库 WHERE 书名包含关键词
* OUTPUT 搜索结果
* END

#### 6. 交易管理模块



##### 6.1 功能描述

* + - **发起购买请求**：买家点击购买按钮后，自动生成订单。
  + - **订单确认**：卖家确认订单后，买家可选择支付方式完成交易。
  + - **订单状态管理**：支持交易取消、退款等状态更新。

##### 6.2 界面设计

* + **订单页面**：显示订单详情（书籍信息、卖家信息、交易状态）。
  + **支付页面**：买家选择支付方式（微信支付/支付宝）并完成付款。

##### 6.3 数据库设计

##### 图形用户界面 中度可信度描述已自动生成

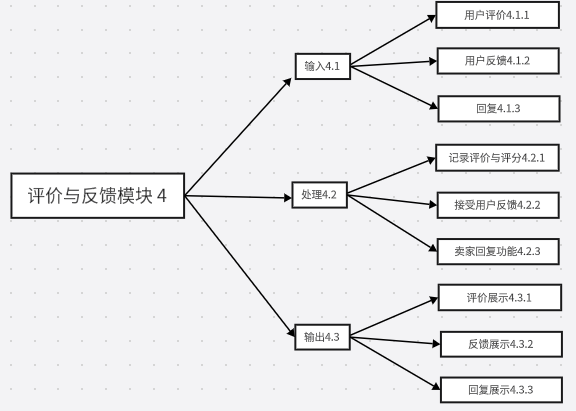
##### 6.4 关键算法设计

* **订单生成逻辑**
* ```java
* public Order createOrder(int bookId, int buyerId) {
* Book book = bookRepository.findById(bookId);
* return orderRepository.save(new Order(bookId, buyerId, book.getSellerId()));
* }
* ```

##### 6.5 PDL详细设计

* + **订单生成流程**
* BEGIN
* INPUT 书籍ID、买家ID
* QUERY 数据库查找书籍信息
* INSERT 新订单信息 INTO 数据库
* OUTPUT "订单生成成功"
* END

#### 7. 评论与反馈模块



##### 7.1 功能描述

* + - **评价功能**：交易完成后，买家可对卖家和书籍进行评分和评价。
  + - **反馈功能**：用户可向平台提交问题反馈。
  + - **回复功能**：卖家或管理员可对用户反馈进行回复。

##### 7.2界面设计

* + - **评价页面**：包含评分星级（1-5）、评价内容输入框，带提交按钮。
  + - **反馈页面**：用户填写问题描述，附加截图（可选）。
  + - **回复页面**：显示用户评价或反馈，并提供回复框。

##### 7.3 数据库设计

表格

描述已自动生成

##### 7.4 关键算法设计

存储评价：

public String storeComment(int userId, int bookId, int transactionId, int score, String commentText) {

if (isTransactionValid(transactionId)) {

String sql = "INSERT INTO CommentTable (userId, bookId, transactionId, score, commentText, timestamp) VALUES (?, ?, ?, ?, ?, ?)";

jdbcTemplate.update(sql, userId, bookId, transactionId, score, commentText, LocalDateTime.now());

return "Comment stored successfully";

} else {

return "Invalid transaction";

}

}

##### 7.5 PDL详细设计

存储评价

Procedure StoreComment(userID, bookID, transactionID, score, commentText)

Begin

Check if transactionID is valid and completed

If valid then

Insert (userID, bookID, transactionID, score, commentText, currentTime) into CommentTable

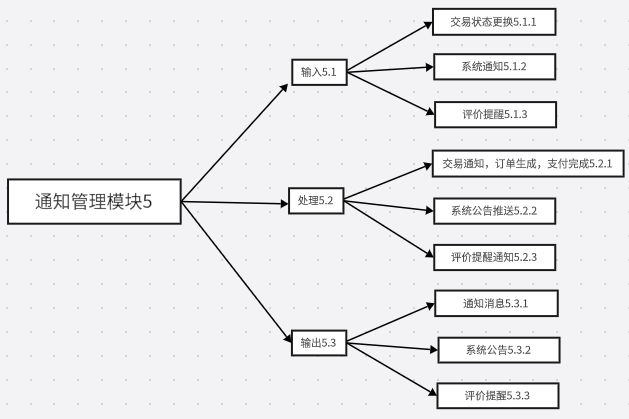
Return "Comment stored successfully"

Else

Return "Invalid transaction"

End

#### 8. 通知管理模块



##### 8.1 功能描述

* + - **交易通知**：用户收到订单生成、支付完成等系统消息。
  + - **系统公告**：平台发送重要通知（如维护时间、规则变更）。
  + - **评价提醒**：交易完成后，系统提醒买家提交评价。

##### 8.2界面设计

* + **通知列表页面**：以时间轴形式展示通知内容，支持已读标记。
  + **公告页面：**展示平台公告，按发布日期排序。

##### 8.3 数据库设计

##### 表格 描述已自动生成

##### 8.4 关键算法设计

实时通知推送：

public void pushNotification(int userId, String messageType, String messageContent) {

Notification notification = new Notification(userId, messageType, messageContent, LocalDateTime.now());

try {

webSocketService.sendNotification(notification);

logNotificationStatus(notification, "Delivered");

} catch (Exception e) {

logNotificationStatus(notification, "Pending");

}

}

private void logNotificationStatus(Notification notification, String status) {

String sql = "INSERT INTO NotificationLog (userId, messageType, messageContent, status, timestamp) VALUES (?, ?, ?, ?, ?)";

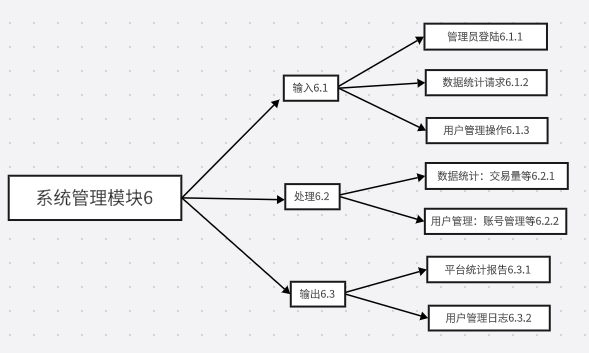
jdbcTemplate.update(sql, notification.getUserId(), notification.getMessageType(), notification.getMessageContent(), status, LocalDateTime.now());

}

##### 8.5 PDL详细设计

* 实时通知推送
* Procedure PushNotification(userID, messageType, messageContent)
* Begin
* Message = {userID, messageType, messageContent, timestamp}
* Push to WebSocket Queue
* If delivery successful then
* Log Notification as "Delivered"
* Else
* Log Notification as "Pending"
* End

#### 9. 系统管理模块



##### 9.1功能描述

* + - 管理员登录：管理员验证后进入管理后台。
  + - 数据统计：查看书籍发布数量、订单交易量、活跃用户等统计信息。
  + - 用户管理：封禁违规用户，删除不合法书籍信息。

##### 9.2界面设计

* + - 管理后台首页：包含数据统计图表（折线图、饼图）。
  + - 用户管理页面：展示用户列表，提供封禁操作按钮。
  + - 书籍管理页面：支持删除书籍信息的批量操作。

##### 9.3数据库设计

* 表格

  描述已自动生成

##### 9.4关键算法设计

统计交易量

public TransactionSummary getTransactionSummary(LocalDate startDate, LocalDate endDate) {

String sql = "SELECT COUNT(\*) as totalOrders, SUM(amount) as totalRevenue FROM TransactionTable WHERE transactionDate BETWEEN ? AND ?";

return jdbcTemplate.queryForObject(sql, new BeanPropertyRowMapper<>(TransactionSummary.class), startDate, endDate);

}

##### 9.5PDL详细设计

统计交易量

Procedure GetTransactionSummary(startDate, endDate)

Begin

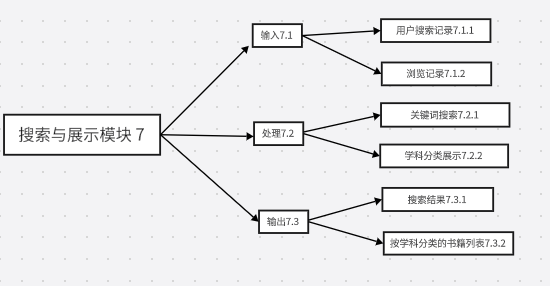
Query = "SELECT COUNT(\*) as TotalOrders, SUM(amount) as TotalRevenue FROM TransactionTable WHERE transactionDate BETWEEN ? AND ?"

Results = ExecuteQuery(Query, startDate, endDate)

Return Results

End

#### 10. 搜索与展示模块



##### 10.1功能描述

* + - 关键词搜索：用户输入关键词，检索书籍标题和作者。
  + - 学科分类展示：用户按学科浏览书籍列表。

##### 10.2界面设计

* + - 搜索结果页面：展示与关键词匹配的书籍列表。
  + - 分类展示页面：显示按学科分组的书籍，支持二级筛选。

##### 10.3数据库设计

##### 10.4关键算法设计

关键词搜索：

public List<Book> searchBooks(String keyword, int page, int pageSize) {

String tokens = Arrays.stream(keyword.split(" ")).map(String::trim).collect(Collectors.joining(" | "));

int offset = (page - 1) \* pageSize;

String sql = "SELECT \* FROM BookTable WHERE MATCH(title, author) AGAINST (? IN BOOLEAN MODE) ORDER BY relevance DESC LIMIT ? OFFSET ?";

return jdbcTemplate.query(sql, new BeanPropertyRowMapper<>(Book.class), tokens, pageSize, offset);

}

分类展示：

public List<Book> displayBooksByCategory(int categoryId, int page, int pageSize) {

int offset = (page - 1) \* pageSize;

String sql = "SELECT \* FROM BookTable WHERE categoryId = ? ORDER BY popularity DESC LIMIT ? OFFSET ?";

return jdbcTemplate.query(sql, new BeanPropertyRowMapper<>(Book.class), categoryId, pageSize, offset);

}

##### 10.5 PDL详细设计

关键词搜索：

Procedure SearchBooks(keyword, page, pageSize)

Begin

Tokens = Tokenize(keyword)

Query = "SELECT \* FROM BookTable WHERE MATCH(title, author) AGAINST (?) ORDER BY relevance LIMIT ?, ?"

Offset = (page - 1) \* pageSize

Results = ExecuteQuery(Query, Tokens, Offset, pageSize)

Return Results

End

分类展示：

Procedure DisplayBooksByCategory(categoryID, page, pageSize)

Begin

Offset = (page - 1) \* pageSize

Query = "SELECT \* FROM BookTable WHERE categoryID = ? ORDER BY popularity DESC LIMIT ?, ?"

Results = ExecuteQuery(Query, categoryID, Offset, pageSize)

Return Results

End

**求购模块**

**11. 功能描述**

**11.1 发布求购需求**

* 用户可详细填写求购图书的名称、作者、出版社、成色要求、接受价位、详情等信息，发布求购请求，系统自动记录发布时间及用户账号信息。

**11.2 求购需求管理**

* 用户能查看自己发布的所有求购记录，支持对求购信息进行修改（如调整价位、补充详情）、删除操作，方便用户根据实际情况灵活变更求购意向。

**11.3 求购回复查看与处理**

* 当卖家针对用户的求购需求给出回复后，用户可在求购模块及时查看回复内容、回复时间以及卖家信息；用户还可对回复进行标记，例如标记为 “已读”“重点关注”，便于后续跟进。

**11.2. 界面设计**

**11.2.1 求购发布页面**

界面布局清晰，采用表单形式引导用户依次输入图书名称、作者、出版社、成色要求、接受价位、详情等必填信息；提供友好的提示信息，如字数限制、格式要求等；设置 “发布” 按钮，方便用户一键提交求购需求，按钮在用户输入完整有效信息后才变为可点击状态。

**11.2.2 求购列表页面**

展示用户已发布的所有求购记录，每条记录以卡片形式呈现，包含图书关键信息（名称、作者）、发布时间、求购状态（是否已收到回复）；提供操作按钮，如 “修改”“删除”，点击 “修改” 可跳转到求购信息编辑页面，点击 “删除” 则弹出确认对话框，防止误操作。

* + 1. **求购回复页面**
    2. 当有求购回复时，以列表形式展示回复内容；每条回复包含卖家账号、姓名、回复时间、回复内容；设置 “已读” 标记按钮，用户点击后该条回复变色或添加特殊标识，方便区分已处理和未处理的回复；提供筛选功能，用户可按回复时间、卖家等条件筛选查看回复。

**11.3. 数据库设计**

**11.3.1 求购图书表（qiugoutushu）**

图形用户界面

描述已自动生成

图形用户界面, 应用程序

描述已自动生成

**11.4. 关键算法设计**

**11.4.1 发布求购信息**

收起

java

public boolean postQiugouInfo(QiugouBook qiugouBook, String userAccount) {

String sql = "INSERT INTO qiugoutushu (tushumingcheng, zuozhe, chubanshe, chengseyaoqiu, jieshoujiawei, xiangqing, yonghuzhanghao) VALUES (?,?,?,?,?,?,?)";

return jdbcTemplate.update(sql, qiugouBook.getTushumingcheng(), qiugouBook.getZuozhe(), qiugouBook.getChubanshe(),

qiugouBook.getChengseyaoqiu(), qiugouBook.getJieshoujiawei(), qiugouBook.getXiangqing(), userAccount) > 0;

}

**11.4.2 获取用户求购列表**

收起

java

public List<QiugouBook> getUserQiugouList(String userAccount, int page, int pageSize) {

int offset = (page - 1) \* pageSize;

String sql = "SELECT \* FROM qiugoutushu WHERE yonghuzhanghao =? ORDER BY fabushijian DESC LIMIT? OFFSET?";

return jdbcTemplate.query(sql, new BeanPropertyRowMapper<>(QiugouBook.class), userAccount, pageSize, offset);

}

**11.4.3 获取求购回复列表**

收起

java

public List<QiugouReply> getQiugouReplies(String userAccount, int page, int pageSize) {

int offset = (page - 1) \* pageSize;

String sql = "SELECT \* FROM qiugouhuifu WHERE yonghuzhanghao =? ORDER BY huifushijian DESC LIMIT? OFFSET?";

return jdbcTemplate.query(sql, new BeanPropertyRowMapper<>(QiugouReply.class), userAccount, pageSize, offset);

}

**11.5. PDL 详细设计**

**11.5.1 发布求购信息**

收起

plaintext

Procedure PostQiugouInfo(qiugouBook, userAccount)

Begin

Query = "INSERT INTO qiugoutushu (tushumingcheng, zuozhe, chubanshe, chengseyaoqiu, jieshoujiawei, xiangqing, yonghuzhanghao) VALUES (?,?,?,?,?,?,?)"

Result = ExecuteUpdate(Query, qiugouBook.tushumingcheng, qiugouBook.zuozhe, qiugouBook.chubanshe,

qiugouBook.chengseyaoqiu, qiugouBook.jieshoujiawei, qiugouBook.xiangqing, userAccount)

If Result > 0 Then

Return True

Else

Return False

End If

End

**11.5.2 获取用户求购列表**

收起

plaintext

Procedure GetUserQiugouList(userAccount, page, pageSize)

Begin

Offset = (page - 1) \* pageSize

Query = "SELECT \* FROM qiugoutushu WHERE yonghuzhanghao =? ORDER BY fabushijian DESC LIMIT?,?"

Results = ExecuteQuery(Query, userAccount, Offset, pageSize)

Return Results

End

**11.5.3 获取求购回复列表**

收起

plaintext

Procedure GetQiugouReplies(userAccount, page, pageSize)

Begin

Offset = (page - 1) \* pageSize

Query = "SELECT \* FROM qiugouhuifu WHERE yonghuzhanghao =? ORDER BY huifushijian DESC LIMIT?,?"

Results = ExecuteQuery(Query, userAccount, Offset, pageSize)

Return Results

End

上述求购模块涵盖了从用户需求出发的功能设计、直观便捷的界面规划、支撑数据存储的数据库架构、核心业务逻辑的算法实现以及流程梳理的 PDL 设计，旨在为二手图书交易平台的求购场景提供完善的解决方案，方便用户高效发布求购、管理求购信息及跟进求购回复。

#### 12. 总结

* 通过详细设计，系统满足用户需求，确保可操作性和扩展性。