

**团队项目实训报告**

|  |  |
| --- | --- |
| **实训名称：** | **微众银行区块链项目实训** |
| **组号：** | **第八组** |
| **项目名：** | **图书管理系统** |
| **组长姓名：** | **罗飞龙** |
| **学生专业：** | **软件工程** |
| **开课学期：** | **2020-2021-02** |

**华南理工大学软件学院**

**2021年06月**

**团队项目实训报告各个子报告目录**

1. [小组基本情况介绍 3](#_Toc44497440)
2. [图书管理系统项目需求规格说明书 5](#_Toc44497441)
3. [图书管理系统概要设计说明书 1](#_Toc44497442)8
4. [图书管理系统详细设计说明书 26](#_Toc44497443)
5. [图书管理系统测试报告 5](#_Toc44497444)3
6. [图书管理系统项目总结 5](#_Toc44497445)7

# **小组基本情况介绍**

|  |  |  |  |
| --- | --- | --- | --- |
| 组号： | **第八组** | | |
| 项目名： | **图书管理系统** | | |
| 项目地址： | **华南理工大学大学城校区图书馆** | | |
| 组长  姓名： | 罗飞龙 | 学号： | 201830670399 |
| 班级： | 18级卓越一班 | 专业： | 软件工程（卓越班） |
| 主要职责（分小点写清楚项目中承担的角色和承担的主要工作任务）： | | | |
| 1. 负责前端管理员页面的开发 2. 负责部分需求分析 3. 负责部分概要分析和部分详细分析 4. 负责部分软件测试 5. 负责部分文档的编写 | | | |
| 组员1  姓名： | 刘帆 | 学号： | 201830670337 |
| 班级： | 18级卓越二班 | 专业： | 软件工程（卓越班） |
| 主要职责（分小点写清楚项目中承担的角色和承担的主要工作任务）： | | | |
| 1. 负责前端用户页面的开发  2. 负责部分需求分析  3. 负责部分概要分析和部分详细分析  4. 负责部分软件测试  5. 负责部分文档的编写 | | | |
| 组员2  姓名： | 覃宏铭 | 学号： | 201830671303 |
| 班级： | 18级卓越一班 | 专业： | 软件工程（卓越班） |
| 主要职责（分小点写清楚项目中承担的角色和承担的主要工作任务）： | | | |
| 1. 负责web后台的开发  2. 负责部分需求分析  3. 负责部分概要分析和部分详细分析  4. 负责部分软件测试  5. 负责部分文档的编写 | | | |
| 组员3  姓名： | 陆梓文 | 学号： | 201830670382 |
| 班级： | 18级卓越二班 | 专业： | 软件工程（卓越班） |
| 主要职责（分小点写清楚项目中承担的角色和承担的主要工作任务）： | | | |
| 1. 负责智能合约的编写和后端的开发  2. 负责部分需求分析  3. 负责部分概要分析和部分详细分析  4. 负责部分软件测试  5. 负责部分文档的编写 | | | |
| 组员4  姓名： | 黄耀枫 | 学号： | 201730685387 |
| 班级： | 18级卓越一班 | 专业： | 软件工程（卓越班） |
| 主要职责（分小点写清楚项目中承担的角色和承担的主要工作任务）： | | | |
| 无 | | | |

# **二手物品交易市场系统项目需求规格说明书**

**目录**

**[1. 概述 6](#_Toc44497804)**

**[1.1. 编写目的 6](#_Toc44497805)**

**[1.2. 项目背景 6](#_Toc44497806)**

**[2. 系统功能需求 6](#_Toc44497807)**

**[3. 系统非功能需求 1](#_Toc44497819)6**

**[4. 系统接口：外部系统接口 1](#_Toc44497820)7**

1. **概述**
   1. **编写目的**

本设计说明书简单阐明了图书管理系统的基本设计思想、基本功能、模块划分以及模块间接口，便于各模块开发人员能更好地了解该系统的基本情况及各模块详细功能。

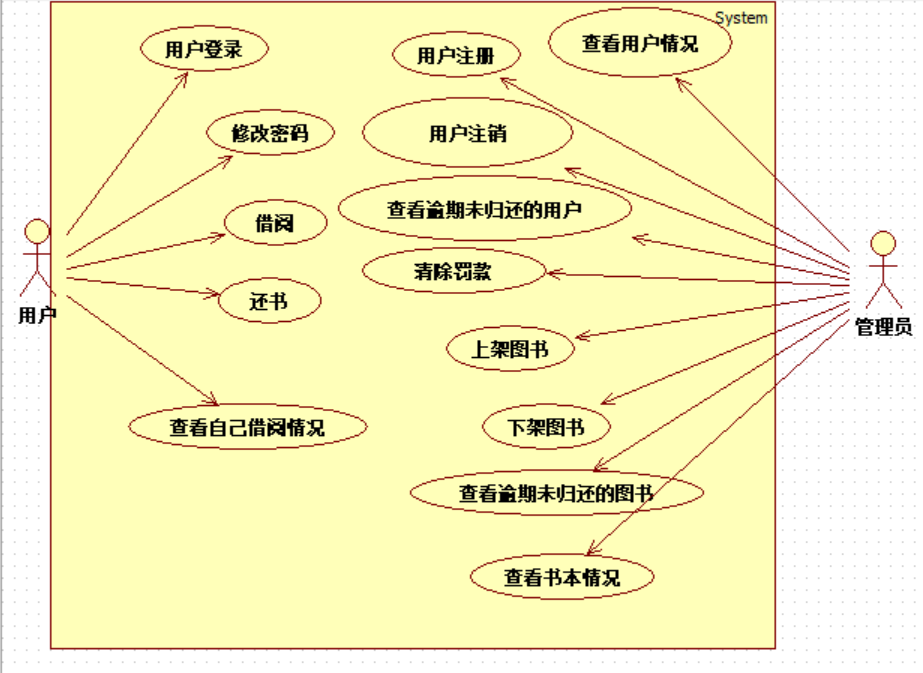
* 1. **项目背景**

随着经济的高速发展和人们生活水平的提高，人们开始重视精神文化消费，其主要载体就是图书。一个人阅读完一本图书后，这本图书对他就失去了之前的价值，所以出现了借读，归还的模式。这种模式的发展就是图书馆。

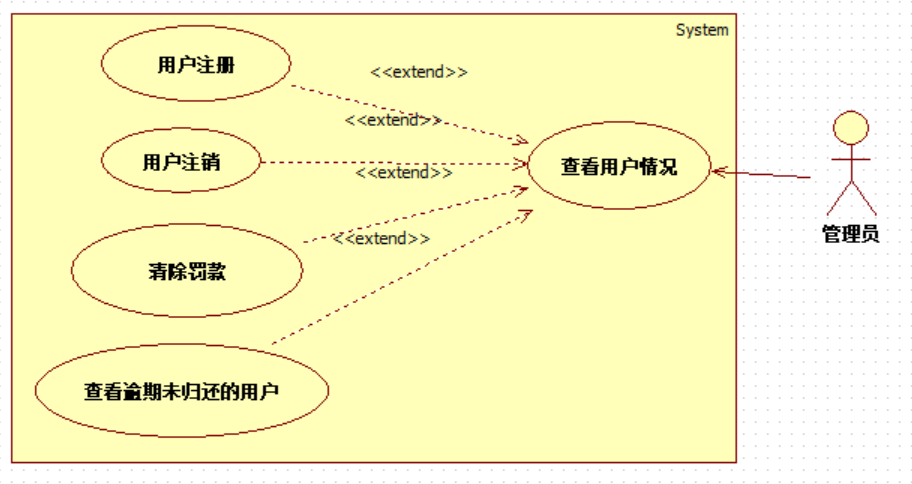
在此背景下，图书馆所藏的书本数量在不断提高，甚至对图书管理员有较高的要求(大学出现图书管理的专业)，所消耗的人力财力在增加。图书管理系统是利用计算机对书本，借读人员等进行快速，准确地整理，方便图书管理员管理，也简化了借读，归还流程，让读者更加便利。

本图书管理系统可以提供一个便于管理员管理图书与借读人员的平台。用户在系统中可以进行注册用户，借书，还书，查看借读情况等操作。本系统后台是基于FISCO-BCOS的区块链系统，在保证稳定性和高效性的同时能够提供足够的安全性，可以保护用户的隐私和交易的安全性。

1. **系统功能需求**
   1. **系统功能架构：**



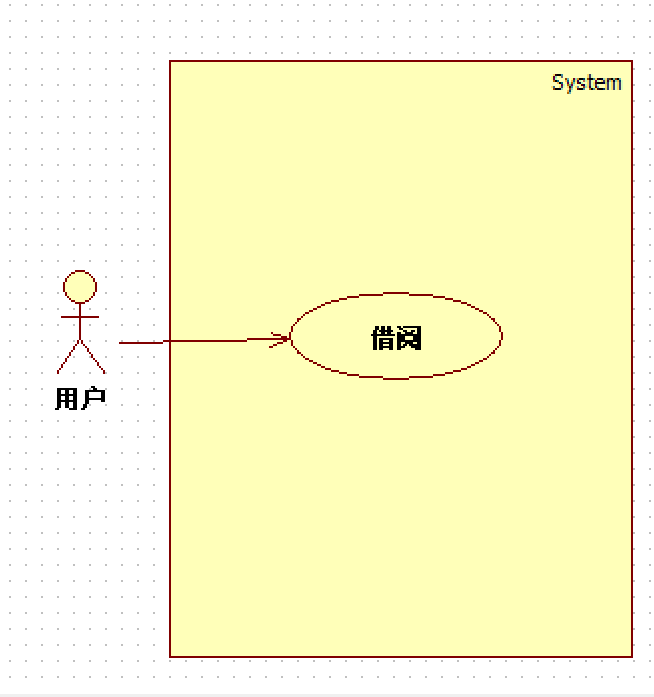
* 1. **查看用户**
     1. **用例图**



* + 1. **用例描述**

|  |  |
| --- | --- |
| 用例简要描述 | 管理员可以查看用户情况。管理员可以查看逾期未归还的用户。管理员来统一注册账户，初始密码默认设置。管理员可以注销用户。 |
| 用例角色 | 管理员 |
| 用例前置条件 | 管理员使用图书管理系统。 |
| 用例后置条件 | 用户信息被更新，或管理员查看用户信息 |
| 基本事件流 | 管理员打开图书管理系统  管理员点击用户管理按钮  系统显示所有的用户情况。 |
| 备选事件流1 | 管理员打开图书管理系统  管理员点击用户管理按钮  管理员点击注册用户按钮  系统显示注册界面，并要求输入账户名和id  管理员输入账户名和id  系统查重账户名，允许用户注册  系统刷新，显示所有的用户信息 |
| 备选事件流2 | 管理员打开图书管理系统  管理员点击用户管理按钮  管理员选中用户，点击注销按钮  系统显示确认信息  管理员点击确认  系统查找账户id，注销用户  系统刷新，显示剩下的用户信息 |
| 备选事件流3 | 管理员打开图书管理系统  管理员点击用户管理按钮  管理员点击查看逾期未归还的用户按钮  系统刷新，显示所有逾期未归还的用户信息 |
| 备选事件流4 | 管理员打开图书管理系统  管理员点击用户管理按钮  管理员选中用户，点击清除罚款按钮  系统显示确认信息  管理员点击确认  系统查找账户id，清除用户罚款  系统刷新，显示更新后的用户信息 |
| 备选事件流5 | 用户打开图书管理系统  系统显示注册界面，并要求输入账户名和密码  用户输入账户名和密码  系统查重账户名，用户名存在  提示“用户名已存在”，本次注册无效 |
| 备选事件流6 | 管理员打开图书管理系统  管理员点击用户管理按钮  管理员选中用户，点击注销按钮  系统显示确认信息  管理员点击取消，退出该用例  系统弹出取消提示 |
| 备选事件流7 | 管理员打开图书管理系统  管理员点击用户管理按钮  管理员点击查看逾期未归还的用户按钮  管理员再次点击查看逾期未归还的用户按钮，退出该用例  系统刷新，显示所有的用户信息 |
| 备选事件流8 | 管理员打开图书管理系统  管理员点击用户管理按钮  管理员选中用户，点击清除罚款按钮  系统显示确认信息  管理员点击取消，退出该用例  系统弹出取消信息，显示所有的用户信息 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒 |
| 用例相关业务数据 | 用户名、密码，用户id，用户罚款 |
| 用例相关人（参与讨论的人员） | 刘帆，罗飞龙，覃宏铭，陆梓文 |

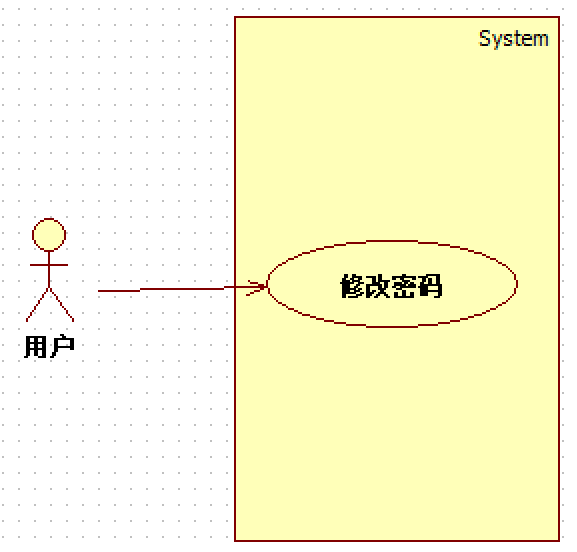
* 1. **用户借阅**
     1. **用例图**



* + 1. **用例描述**

|  |  |
| --- | --- |
| 用例简要描述 | 用户借阅图书 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户登录 |
| 用例后置条件 | 书本被借出，数据库更新 |
| 基本事件流 | 用户输入图书名  用户点击借阅  图书成功借阅，更新图书信息 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒 |
| 用例相关业务数据 | 图书数据 |
| 用例相关人（参与讨论的人员） | 刘帆，覃宏铭，陆梓文，罗飞龙 |

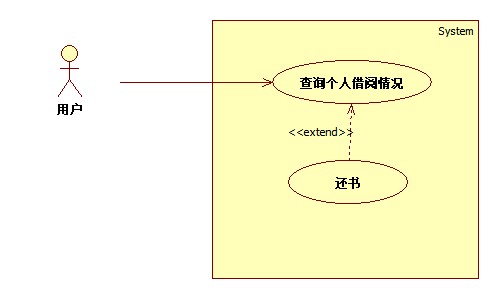
* 1. **修改密码**
     1. **用例图**



* + 1. **用例描述**

|  |  |
| --- | --- |
| 用例简要描述 | 用户修改自己的密码 |
| 用例角色 | 用户 |
| 用例前置条件 | 无 |
| 用例后置条件 | 用户修改密码成功 |
| 基本事件流 | 用户放上借阅卡(这里用输入账户名来代替)  用户点击“修改密码”  用户输入新的密码  用户点击“确认” |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒 |
| 用例相关业务数据 | 用户数据，账户，密码 |
| 用例相关人（参与讨论的人员） | 刘帆，陆梓文，覃宏铭，罗飞龙 |

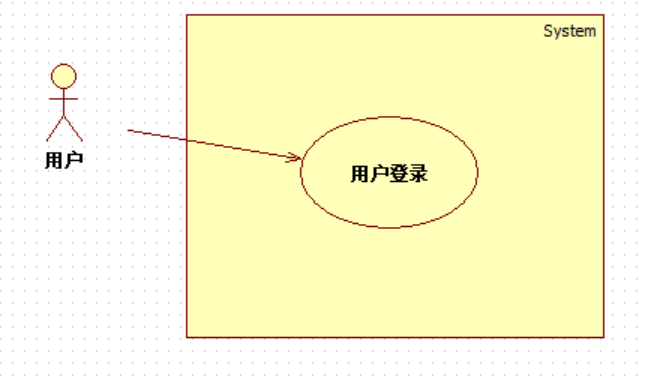
* 1. **查询个人借阅情况**
     1. **用例图**



* + 1. **用例描述**

|  |  |
| --- | --- |
| 用例简要描述 | 用户可以查看自己的借阅信息 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户登录 |
| 用例后置条件 | 用户查看到借阅信息列表 |
| 基本事件流 | 用户进入个人借阅记录页面 |
| 备选事件流1 | 用户进入个人借阅记录页面  用户筛选借阅记录 |
| 备选事件流2 | 用户进入个人借阅记录页面  用户选中一项借阅记录  用户点击还书  用户点击确定  系统提示还书成功，系统记录更新 |
| 备选事件流2.1 | 用户进入个人借阅记录页面  用户选中一项借阅记录  用户点击还书  用户点击取消  用户可以继续查看借阅记录、重新选择还书、或结束用例 |
| 备选事件流2.2 | 用户进入个人借阅记录页面  用户选中一项借阅记录  用户点击还书  用户点击确定  系统提示过时归还，系统记录更新 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒 |
| 用例相关业务数据 | 书籍信息（书籍ID、书籍名称等）  开始借阅时间、最迟归还期限、归还时间  状态（是否归还、是否过期、是否遗失赔偿等） |
| 用例相关人（参与讨论的人员） | 覃宏铭，刘帆，陆梓文，罗飞龙 |

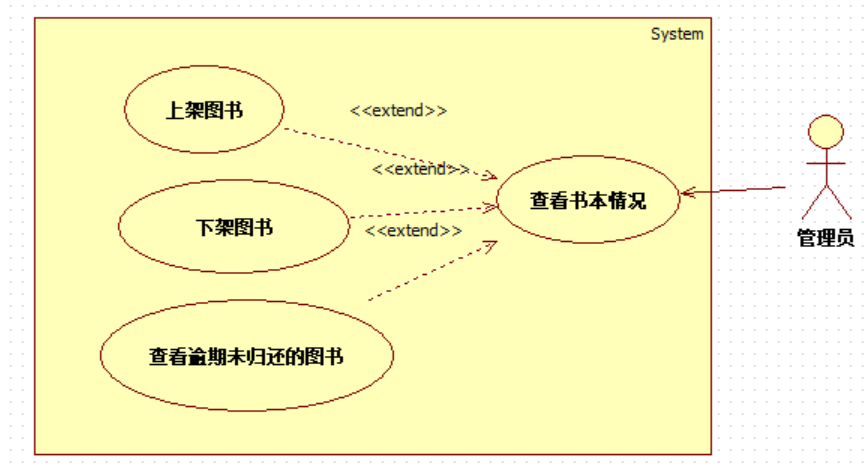
* 1. **用户登录**
     1. **用例图**



* + 1. **用例描述**

|  |  |
| --- | --- |
| 用例简要描述 | 用户可以登录自己的图书管理系统账号。 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户未登陆图书管理系统且系统存在用户账号 |
| 用例后置条件 | 用户成功登陆系统 |
| 基本事件流 | 用户打开图书管理系统  系统显示登陆界面，并要求输入账户名和密码  用户输入账户名和密码  系统验证登录信息，允许用户登录 |
| 备选事件流1 | 用户打开图书管理系统  系统显示登陆界面，并要求输入账户名和密码  用户输入账户名和密码  系统显示用户名错误或密码错误的提示信息  用户可以重新输入登录信息，也可以选择结束该用例 |
| 备选事件流2 | 用户打开图书管理系统  系统显示登陆界面，并要求输入账户名和密码  用户关闭系统  用户结束该用例 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 用户名、密码、账号身份 |
| 用例相关人（参与讨论的人员） | 覃宏铭，陆梓文，刘帆，罗飞龙 |

* 1. **查看书本情况**
     1. **用例图**



* + 1. **用例描述**

|  |  |
| --- | --- |
| 用例简要描述 | 管理员查看所有书本的情况。管理员可以查看逾期未归还的图书。管理员可以上架和下架图书。 |
| 用例角色 | 管理员 |
| 用例前置条件 | 管理员使用图书管理系统。 |
| 用例后置条件 | 书本情况被更新，或者管理员查看图书情况。 |
| 基本事件流 | 管理员点击查看书本情况按钮  系统显示所有的图书情况。 |
| 备选事件流1 | 管理员点击查看书本情况按钮  系统显示所有的图书情况。  管理员点击查看逾期未归还的图书按钮  系统显示所有的逾期未归还的图书情况。 |
| 备选事件流2 | 管理员点击查看书本情况按钮  系统显示所有的图书情况。  管理员点击上架图书按钮  管理员填写上架图书的相关信息  管理员点击确认  系统更新，显示所有的图书情况 |
| 备选事件流3 | 管理员点击查看书本情况按钮  系统显示所有的图书情况。  管理员选择一条图书记录。  管理员点击下架图书按钮  管理员点击确认  系统更新，显示剩余的图书情况 |
| 备选事件流4 | 管理员点击查看书本情况按钮  系统显示所有的图书情况。  管理员点击查看逾期未归还的图书按钮  管理员取消查看逾期未归还的图书按钮，退出用例  系统显示所有的图书情况 |
| 备选事件流5 | 管理员点击查看书本情况按钮  系统显示所有的图书情况。  管理员点击上架图书按钮  管理员填写上架图书的相关信息  管理员点击取消，退出该用例  系统弹出取消提示，显示所有的图书情况 |
| 备选事件流6 | 管理员点击查看书本情况按钮  系统显示所有的图书情况。  管理员选择一条图书记录。  管理员点击下架图书按钮  管理员点击取消，退出该用例  系统弹出取消提示，显示所有的图书情况 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求：  数据库进行加密 |
| 用例相关业务数据 | 图书描述信息，图书名称，图书借阅状态。 |
| 用例相关人（参与讨论的人员） | 罗飞龙，刘帆，覃宏铭，陆梓文 |

1. **系统非功能需求**

|  |  |
| --- | --- |
| 性能需求 | 系统能够适应400个用户，平均每个会话持续8分钟；  生成的web页面，通过速率100KBps在10秒内可以全部下载 |
| 可用性需求 | 正常运行时间:99.9%。 |
| 并发需求 | 能支持至少百个并发请求。 |
| 安全性需求 | 所有涉及信息的操作必须登录后或者在所允许的ip地址上才能完成；  系统只允许用户查看自己的信息，而不能查看其他用户的资料；  数据库进行加密。 |

1. **系统接口：外部系统接口**

|  |  |
| --- | --- |
| 用户界面 | 界面风格：本系统采用的是图形用户界面，界面简洁且搭配合理。  界面布局：系统页面较为合理，给人一种成熟稳重的感觉。  界面操作：页面上的按钮和文本框合理设计，让用户的使用更加方便快捷。  界面内容：界面设置导航，消息显示为弹出式。 |
| 硬件接口 | 软件之间交流的数据：用户账户上的信息，如已完成交易等。  通信协议：HTTP协议 |
| 软件接口 | 数据存储：采用FISCO BCOS 区块链  操作系统：开发系统为Ubuntu 20.04及Windows 10  工具：IDEA等  数据访问：Java SDK |
| 通信接口 | 通过浏览器访问 |
| 故障处理 | 登录失败或连接失败：尝试检查本地网络连接，并重启浏览器。 |

# **图书管理系统概要设计说明书**

目录

**[1. 概述 1](#_Toc44884412)9**

**[1.1. 系统简述 1](#_Toc44884413)9**

**[1.2. 软件设计目标 1](#_Toc44884414)9**

**[1.3. 参考资料 1](#_Toc44884415)9**

**[1.4. 修订版本记录 1](#_Toc44884416)9**

**[2. 术语表 2](#_Toc44884417)0**

**[3. 设计概述 2](#_Toc44884418)0**

**[3.1. 系统结构设计 2](#_Toc44884419)0**

**[3.1.1. 系统物理架构 2](#_Toc44884420)1**

**[3.1.2. 系统逻辑架构 2](#_Toc44884421)2**

**[3.1.3. 系统数据模型 2](#_Toc44884422)2**

**[3.2. 系统接口设计 2](#_Toc44884423)3**

**[3.3. 系统数据字典](#_Toc44884424) 24**

**[3.4. 系统智能合约设计 2](#_Toc44884425)4**

**[3.4.1. 主要solidity函数及功能 2](#_Toc44884426)4**

1. **概述**
   1. **系统简述**

随着经济的高速发展和人们生活水平的提高，人们身边物品的更新换代频率也在增高，更新后留下的旧产品需要处理；同时在各类互联网电商的营销刺激下，消费者容易形成“冲动消费”，因此身边也有越来越多的闲置物品需要处理；随着人们经济意识的增强，消费观念也在逐步发生改变，人们在某些情况下会更多选择性价比较高的二手商品。

在此背景下，二手物品交易市场的需求正在提高。二手商品交易市场，是基于一种全新消费文化和消费理念，伴随着互联网的高速发展，快速、便利地为消费者提供高性价比的二手物品的平台，既能满足人们的生活需要，又可以减少资源浪费。

本二手交易市场系统可以提供一个供用户自由发布和购置二手商品的平台。用户在系统中可以进行注册用户，发布商品，完成交易，查看交易等操作。本系统后台是基于FISCO-BCOS的区块链系统，在保证稳定性和高效性的同时能够提供足够的安全性，可以保护用户的隐私和交易的安全性。

* 1. **软件设计目标**

a) 节省开发费用和降低资源消耗。该项目的主要资源成本便是数据库的容量。因此数据库设计应该合理规范，尽可能的减少数据的冗余以降低储存患者的病历信息所需要的空间；

b) 较高的可靠性。该系统是应用于交易市场方面，要有极高的可靠性，以避免系统出现故障导致交易失败，钱款发生错误，漏帐，多帐等现象；

c) 良好的可维护性。若系统出现故障应该能够快速地被修复。此外该系统也能较容易地添加新的功能来帮助市场的运营。

* 1. **参考资料**

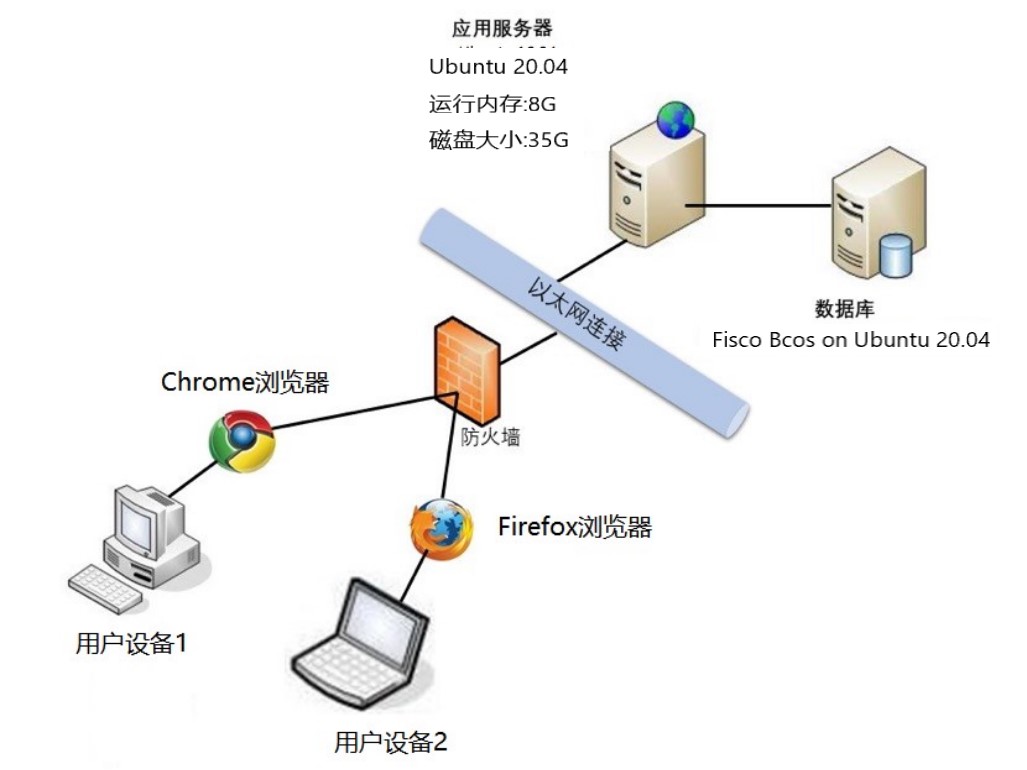
1. 《IT项目管理》
2. 《数据库系统概念》
3. 《软件工程》
4. 百度百科
   1. **修订版本记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2021.7.5 | v1.0 | 完成部分概要设计和详细设计，用例设计 | 刘帆 |
| 2021.7.5 | v1.1 | 完成部分概要设计和详细设计，用例设计 | 罗飞龙 |
| 2021.7.5 | v1.2 | 完成部分概要设计和详细设计，用例设计 | 陆梓文 |
| 2021.7.5 | v1.3 | 完成部分概要设计和详细设计，用例设计 | 覃宏铭 |
| 2021.7.11 | v1.4 | 完成部分测试，前端结果展示 | 罗飞龙 |
| 2021.7.11 | v1.5 | 完成部分测试，前端结果展示 | 罗飞龙 |
| 2021.7.11 | v1.6 | 完成部分测试，前端结果展示 | 刘帆 |
| 2021.7.11 | v1.7 | 完成部分测试，区块链相关代码，架构等相关部分完成 | 陆梓文 |
| 2021.7.11 | v1.8 | 完成部分测试，web后台相关部分完成 | 覃宏铭 |

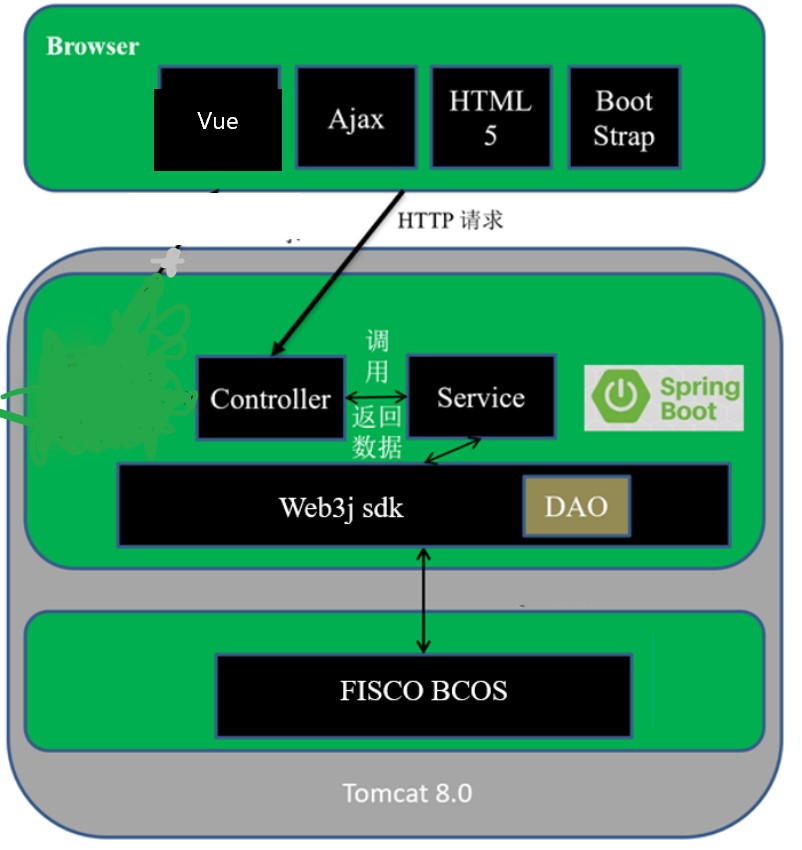
1. **术语表**

|  |  |  |
| --- | --- | --- |
| 术语 | 解释 | 备注 |
| 项目 | 软件工程中的一个项目，由一个项目经理和若干个普通成员组成。 | 项目是本项目的基本单位 |
| 任务 | 项目中包含的若干任务，任务由项目经理布置和指派人员。任务分为4层，最后一层无法再包含子任务。 |  |
| 进度请求 | 一个项目的推进由成员发起进度请求，项目经理批准后才生效。项目经理亦可直接更改任务的完成状态，不经过进度请求步骤。 |  |

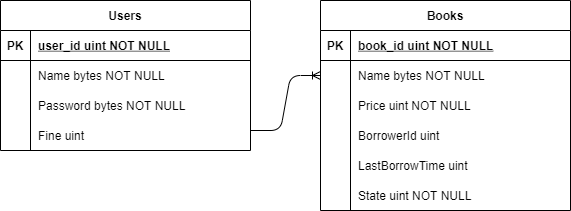
1. **设计概述**
   1. **系统结构设计**
      1. **系统物理架构**



* + 1. **系统逻辑架构**



* + 1. **系统数据模型**
       1. **E-R图**



* + - 1. **数据库表**

表1 用户信息表 (Books）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 能否为空 | 描述 | 备注 |
| Name | bytes | 否 | 用户名 |  |
| Password | bytes | 否 | 用户密码 | 初始化时分配默认密码 |
| Fine | uint | 是 | 罚款金额 | 罚款未缴清的用户不允许借书 |
| isUsed | bool | 否 | 用户是否已注册 |  |

表2 书本信息表 (Books)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 能否为空 | 描述 | 备注 |
| Name | bytes | 否 | 书本名 |  |
| Price | uint | 否 | 书本价格 |  |
| BorrowerId | uint | 是 | 借阅人Id | 外键 |
| LastBorrowTime | uint | 是 | 接触时间戳 | 单位毫秒 |
| State | uint | 否 | 书籍状态 | 0：在库1：借出 |
| isUsed | bool | 否 | 书籍是否已上架 |  |

* 1. **系统接口设计**

|  |  |  |
| --- | --- | --- |
| 所属 | 接口 | 功能 |
| 注册模块 | 用户注册 | 前端发送用户信息，后端执行注册 |
| 书本上架 | 前端发送书本信息，后端执行上架 |
| 登录模块 | 用户登录 | 前端发送用户名和密码，后端执行登陆 |
| 删除模块 | 下架书本 | 前端发送书本Id，后端执行删除 |
| 注销用户 | 前端发送用户Id，后端执行删除 |
| 借还书模块 | 借书 | 前端发送用户id和书本id，后端执行相应数据操作 |
| 还书 | 前端发送用户id和书本id，后端执行相应数据操作 |
| 管理模块 | 查看全部书本信息 | 前端发送请求，后端返回全部书本的名字、价格、借书人等功能性信息 |
| 查看被借出的书本信息 | 前端发送请求，后端筛选并返回全部被借出的书本的名字、价格、借书人等功能性信息 |
| 查看某个用户借的书本信息 | 前端发送用户id，后端筛选并返回全部改用户借的书本的名字、价格、借书人等功能性信息 |
| 查看所有用户信息 | 前端发送请求，后端返回全部用户的id、名字、未缴罚款等功能性信息 |
| 查看未缴清罚款的用户信息 | 前端发送请求，后端返回全部未缴清罚款的用户的id、名字、未缴罚款等功能性信息 |
| 用户修改密码 | 前端发送用户的id、旧密码和新密码，后端执行相应的操作 |

* 1. **系统数据字典**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 数据名称 | 数据业务需求描述 | 业务数据类型 | 所属表格 | 对应字段 | 字段类型 |
| 用户名 | 不超过15字符 | 字符串 | User | Name | bytes |
| 密码 | 6-16位字符 | 字符串 | User | Password | bytes |
| 罚款金额 | 用户的罚款金额 | int | User | Fine | uint |
| 书本价格 | 书本的标价 | int | Book | Price | uint |
| 书本名字 | 不超过30字符 | 字符串 | Book | Name | bytes |
| 书本状态 | 分为在库和借出两种状态 | int | Book | State | uint |

* 1. **系统智能合约设计**
     1. **主要solidity函数及功能**

|  |  |
| --- | --- |
| **函数名称** | **功能** |
| insertBook | 书本上架 |
| deleteBook | 书本下架 |
| insertUser | 用户注册 |
| deleteUser | 用户注销 |
| getBooks | 查看某个用户借的书本信息 |
| getAllBooks | 查看所有的书本信息 |
| getBorrowedBooks | 查看被借出的书本信息 |
| getAllUsers | 查看所有的用户信息 |
| getFinedUsers | 查看罚款未缴清的用户信息 |
| clearFine | 用户线下缴清罚款后管理员将罚款清零 |
| borrowBook | 借书 |
| returnBook | 还书 |
| userLogin | 用户登录 |
| changePassword | 用户修改密码 |

# **图书管理系统详细设计说明书**

**目录**

**[1. 需求概述 2](#_Toc44884518)7**

**[2. 需求分析 2](#_Toc44884519)7**

**[2.1. 宏观分析 2](#_Toc44884520)7**

**[2.2. 国内平台分析](#_Toc44884521) 28**

**[2.3. 用户分析](#_Toc44884522) 28**

**[2.4. 竞争态势分析](#_Toc44884523) 31**

**[2.5. 市场分析 3](#_Toc44884524)1**

**[3. 现状总结 3](#_Toc44884525)2**

**[3.1. 行业可行性 3](#_Toc44884526)2**

**[3.2. 未来发展方向 3](#_Toc44884527)2**

**[4. 用户角色分析 3](#_Toc44884528)2**

**[4.1. 角色 3](#_Toc44884529)2**

**[4.2. 参与业务 3](#_Toc44884530)3**

**[5. 功能性需求 3](#_Toc44884531)4**

**[6. 非功能性需求 3](#_Toc44884532)5**

**[7. 系统设计 3](#_Toc44884533)5**

**[7.1. 基本架构设计 3](#_Toc44884535)5**

**[7.2. 功能实现函数设计 4](#_Toc44884536)3**

**[8. 后台设计 4](#_Toc44884537)8**

**[9. 界面设计 4](#_Toc44884540)9**

1. **需求概述**

随着高校图书馆的普及，对于部分价格过高的图书和工具书，学生们越来越倾向于向图书馆借书。然而随着借阅人数的不断飙升，图书的保管问题又逐渐暴露了出来，破损，丢失图书的现象开始逐渐多了起来。

在此背景下，针对图书借阅进行溯源的需求正在提高。区块链的蓬勃发展与不可更改的特性与图书溯源需求一碰即燃，成功结合了起来。图书溯源，即对每本书的借阅人都留存记录，当发生破损和丢失现象时可以一层层往上追溯到责任人，减少找不到当事人的情况的发生。

图书馆图书借阅系统，正是基于当下的借阅溯源需求和借阅盛状而产生的一个系统。在本系统中，管理员可以查看每本书的借阅情况和每个用户的借阅情况。用户也可以进行借书。本系统后台是基于FISCO-BCOS的区块链系统，在保证稳定性和高效性的同时能够提供足够的安全性，防止信息被篡改。



图表 1图书馆图书存在类型

1. **需求分析**
   1. **宏观分析**
      1. **政策利好推动图书馆发展**

《图书馆政策法规和建设标准》

为促进公共图书馆事业的发展，加强和规范公共图书馆基础设施建设，提高建设项目决策水平，加速公共图书馆建设的标准化，规范化和现代化进程，以实现和保障人民群众利用图书管的权力，满足人民群众基本的知识，信息和文化需求，依据法律，法规指定本标准

《“十三五”时期全国公共图书馆事业发展规划》

全面贯彻党的十八大和十八届三中、四中、五中、六中全会精神，深入贯彻落实[习近平总书记系列重要讲话精神](https://baike.baidu.com/item/%E4%B9%A0%E8%BF%91%E5%B9%B3%E6%80%BB%E4%B9%A6%E8%AE%B0%E7%B3%BB%E5%88%97%E9%87%8D%E8%A6%81%E8%AE%B2%E8%AF%9D%E7%B2%BE%E7%A5%9E/15938320" \t "https://baike.baidu.com/item/%E2%80%9C%E5%8D%81%E4%B8%89%E4%BA%94%E2%80%9D%E6%97%B6%E6%9C%9F%E5%85%A8%E5%9B%BD%E5%85%AC%E5%85%B1%E5%9B%BE%E4%B9%A6%E9%A6%86%E4%BA%8B%E4%B8%9A%E5%8F%91%E5%B1%95%E8%A7%84%E5%88%92/_blank)和治国理政新理念新思想新战略，围绕中央关于加快构建现代公共文化服务体系的决策部署，按照公益性、基本性、均等性和便利性要求，以完善设施网络为基础，以丰富服务内容、强化资源整合、提高服务效能为重点，以完善体制机制为保障，努力构建覆盖城乡、服务高效、惠及全民的公共图书馆服务网络，进一步推进全民阅读，坚定文化自信，提高全民族科学文化素质和社会文明程度，增强人民群众对公共文化服务的获得感。

基于以上相关政策可知，我国图书馆标准和管理制度正在逐步完善，国家政策鼓励中国人们积极参与社会主义精神建设，广读书，多学习。整体而言，中国的政治环境上有利于图书馆的发展。

* + 1. **学生的阅读量正在增加，同时图书和读者繁多，为图书管理系统的出现奠定基础**

近年来，中国学生的入馆人数总体上升，这表明学生的学习欲望正在增强，同时图书馆作为公益性公共设施，给阅读提供很多便利条件，使个体读书，选书的成本降低。这其中不免会有很多难题，比如对于这么多图书，每天到来的大量读者，要如何进行管理？这给图书管理系统的发展提供机会。



图表 2015-2020年学生日均入图书馆人数

* 1. **国内平台分析**

图书管理系统在中国乃至世界都是长期存在的。伴随现代化图书馆的建设，每一个大型图书馆，几乎都有自己的图书管理系统，可以说图书管理系统是大型图书馆节省人力物力的最佳手段。

* 1. **用户分析**

**2.3.1读者分析**

随着当前经济社会的高速发展，人们的物质水平不断提高，其对于精神方面的追求不断提升。公共图书馆是人们汲取知识的重要场所，能为人们提供重要的精神文化体验。作为公共图书馆的相关工作者来说，应做好读者活动工作，以期吸引更多的读者进行阅读，不断提升自身的精神文化水平。

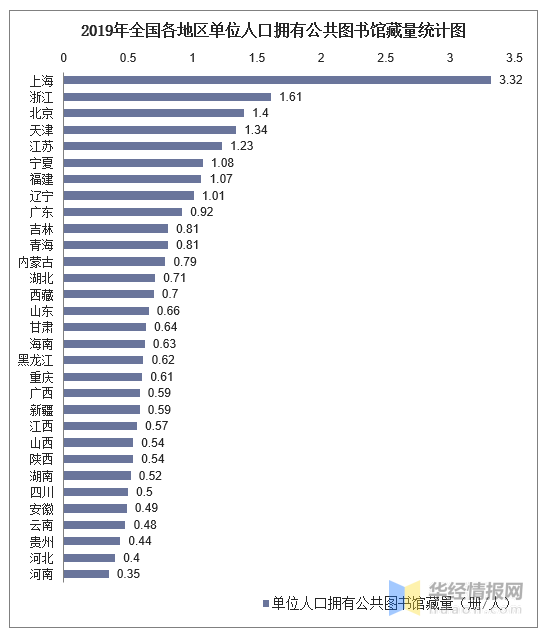


图表 2015-2020年学生日均入图书馆人数

近年来，中国学生的入馆人数总体上升，这表明学生的学习欲望正在增强，同时图书馆作为公益性公共设施，给阅读提供很多便利条件，使个体读书，选书的成本降低。但是借书的手续依旧比较繁琐。查看归还日期，查看自己借阅的书等操作还必须在图书馆进行，读者更加希望简化该流程。

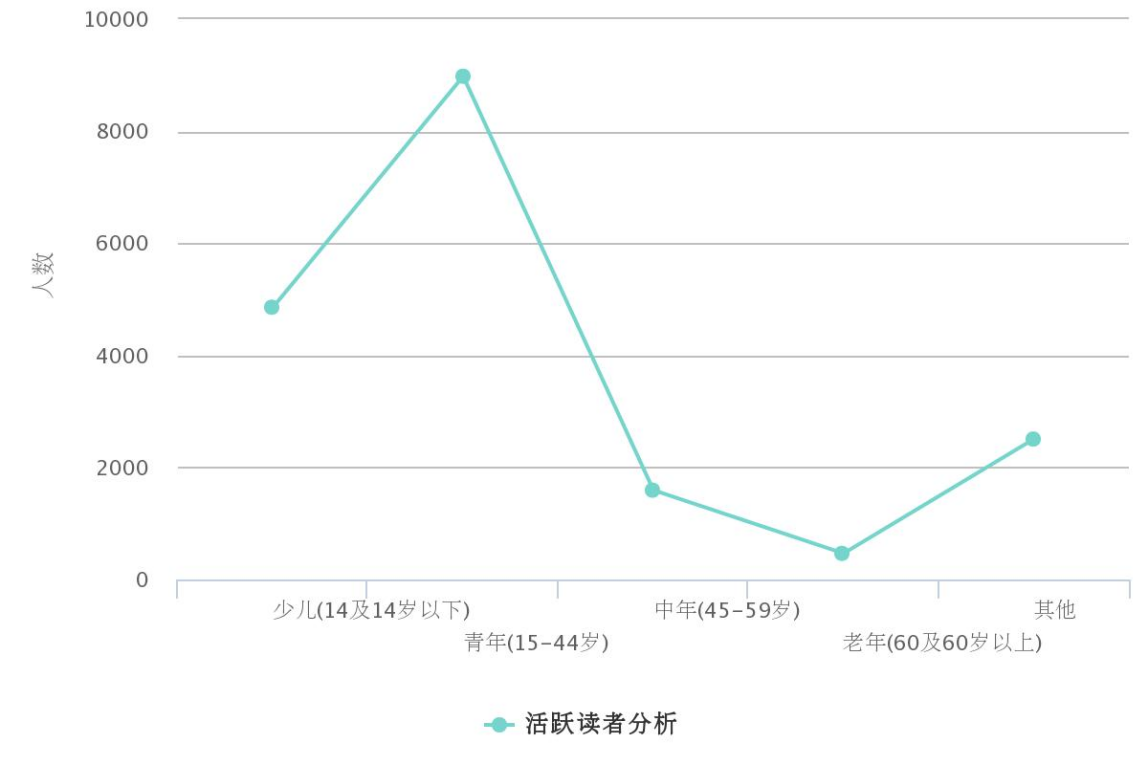
**2.3.4 管理员分析**

在此背景下，图书馆所藏的书本数量在不断提高，甚至对图书管理员有较高的要求(大学出现图书管理的专业)，所消耗的人力财力在增加。



图表 3 2019年中国各地区单位人口拥有公共图书馆藏量统计图

上图给出了2019年中国各地区单位人口拥有公共图书馆藏量统计图。随着社会主义精神建设的推进，还有大家价值意识的转变，在年轻群体的带动下，知识的价值被越来越多的人意识到。公共图书馆作为宝贵的资源，未来需求量肯定越来越大。



图表 4 [2017年度秦皇岛图书馆阅读报告](http://www.qhdtsg.net/Home/Article/detail/id/1633.html" \t "https://image.baidu.com/search/_blank" \o "2017年度秦皇岛图书馆阅读报告(一))

* 1. **竞争态势分析**

1. 供应商：一手图书的卖家。
2. 使用者：节俭的学生，倾向于借书而不是买书
3. 潜在进入者：部分教师，
4. 替代品：网上读书软件
5. 同行竞争：万方图书管理系统，目前占据了大部分高校的图书管理系统。
   1. **市场分析**

市场阶段一般分为导入期、成熟和衰落期。但考虑到图书馆的特殊性，在已有可用的平台的条件下一般不会轻易变动，所以可以认为国内图书借阅系统正在处于成熟期。

由于图书借阅的特殊性，一本图书的完整与否取决于用户的自觉性，难以通过规则去直接解决，因此唯有采用溯源罚款的方式才能间接建立起规则，而这个规则又依赖于市场长期的养成，难以一朝一夕成立。所以用户与图书馆直接还是存在一定的信任问题。

虽然难以得到全部数据，但是我们可以观察南京大学发出的消息。2020年南京大学的总借阅量达到了惊人的150132次，而这个仅仅是一个高校的数据，全部高校的数据将更加引人注目。

据《2019年广东公共图书馆事业发展报告》显示，2019年中国广东图书馆借阅人数达到了惊人的12210.6万人次，比起去年来稳步增长。

图书管理系统的市场到底有多大？《2020年广东公共图书馆事业发展报告》显示参与图书馆线下和线上活动的人次达到了3844.08万人次，比2019年高了25.1%。这证明了人们对于图书馆的热情高涨且正在不断提升。

总体上来讲，图书管理系统的市场尚处于成熟期，但是由于其特殊性可以期待来一场大洗牌，到时将会给许多的图书管理平台商提供机会，而我们的系统届时也可占据一时之地。

1. **现状总结**
   1. **行业可行性**

根据前面的分析，图书管理平台的发展暂时处于成熟期，而当前环境对其发展较为有利；在国家政策支持下，图书馆馆藏数量与地方不断扩大，当扩大到一定程度传统的图书管理平台将无法承受负荷，这时基于区块链的图书管理平台将显示其优势，再加上互联网技术普及以及新技术成熟应用。因此基于区块链的图书馆管理平台仍然又很大机会进入市场。

* 1. **未来发展方向**

在很多人的印象中，图书馆借阅平台是难以盈利的，因为它有稳定性，但是在稳定的背后也是一次变动全体变动的机遇，而图书馆继续扩大必然对数据库有着更高要求，此时基于区块链的图书馆图书管理平台将抓住机会，一发逆转。而区块链的特点又可以给图书管理带来新的可能性。

* + 1. **方向一：基于区块链的电子书借阅**

电子书近来也成为了图书馆借阅的热宠，不同于普通的保管信息，电子书所占空间要大得多，这时数据的报存难免会出错，而将电子书保存在区块链再加以共享可以很好的解决这个问题。

* + 1. **方向二：与在线阅读软件合作**

如今微信读书，网易蜗牛阅读等在线阅读软件开始崛起，而传统图书馆的图书管理平台又在相关领域已经占据了一席之地，这时可以适当的联动，先依附于在线阅读软件的热度，借着热度逐渐转型，成为桥接传统图书借阅模式和新型图书借阅模式的桥梁，然后在这个混合领域担当领头羊，占据新领域的一席之地。

1. **用户角色分析**
   1. **角色**

用户角色从整体来看，分为2种类型，读者用户，管理员。

* + 1. **读者用户**

读者用户的主要需求在于对自己借阅情况的查看，以及借书还书的便捷性。读者用户能够在我们的平台上查看到自己的借阅情况，比如已借书名、归还日期等，能够进行借书、还书操作。

* + 1. **管理员**

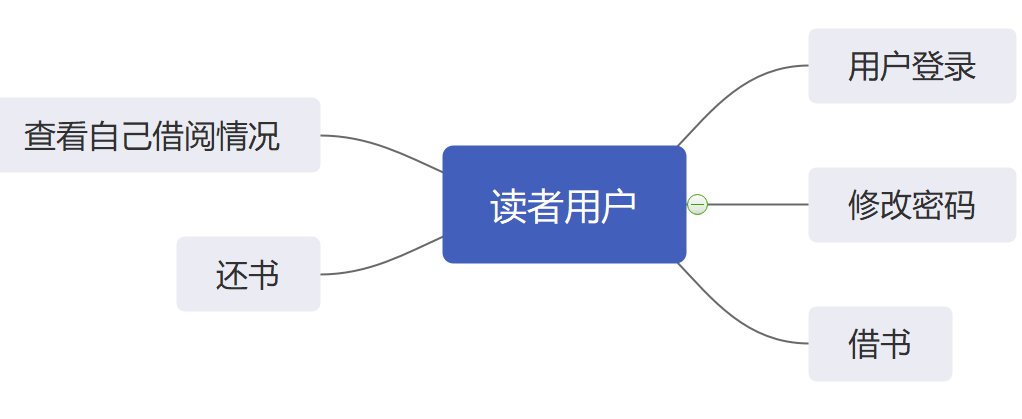
管理员主要负责用户和书籍的管理，管理员可以注册和注销用户，并查阅某些用户信息，同时还可以清除用户罚款。管理员同时也可以上架和下架图书，并查阅某些书本的信息。

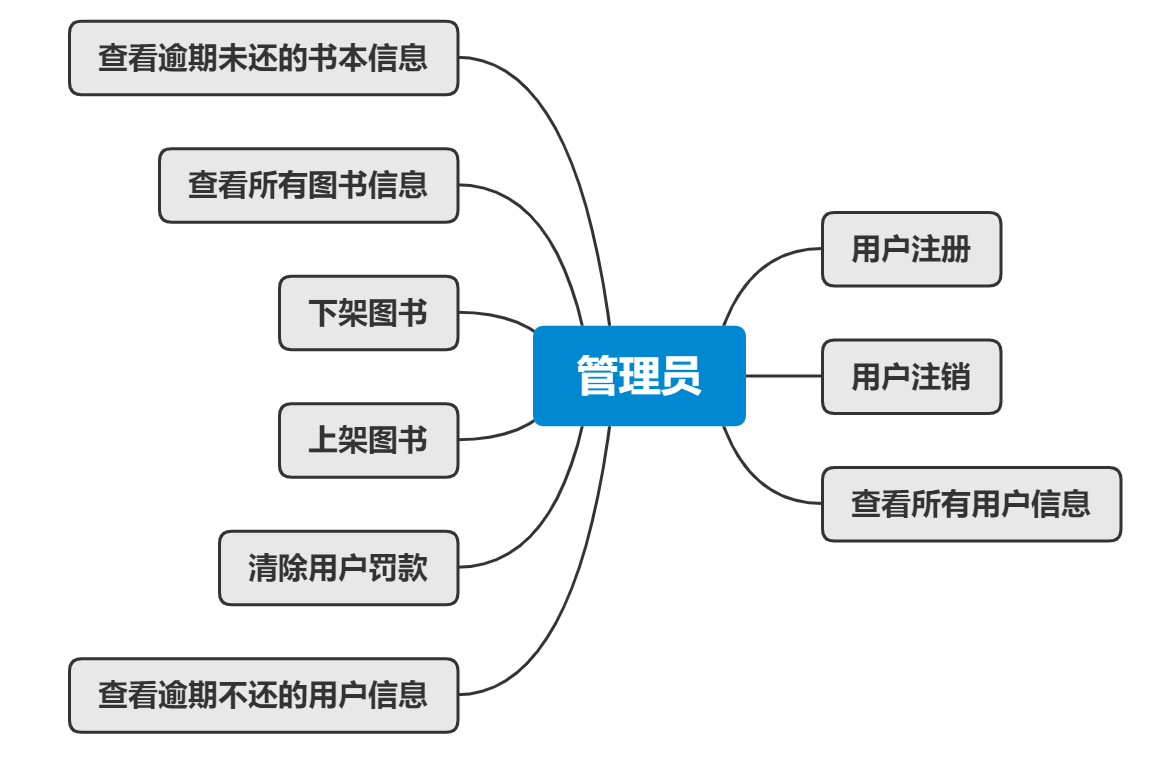
* 1. **参与业务**

读者用户：用户登录，修改密码，查看自己的借阅情况，借书，还书；

管理员：用户注册，用户注销，查看所有用户信息，查看逾期不还的用户信息，清除用户罚款，上架图书，下架图书，查看所有图书信息，查看逾期未还的书本信息；

在系统中，用户设计为两种类型：读者用户、管理员。以下为各种类型的用户对应的功能，其中，读者用户进行用户登录，修改密码，查看自己的借阅情况，借书，还书等操作。而管理员可以进行用户注册，用户注销，查看所有用户信息，查看逾期不还的用户信息，清除用户罚款，上架图书，下架图书，查看所有图书信息，查看逾期未还的书本信息等操作。





1. **功能性需求**

|  |  |  |  |
| --- | --- | --- | --- |
| 功能名称 | 用户角色 | 描述 | 备注 |
| 账号注册 | 管理员 | 管理员输入用户名，id进行账户注册 |  |
| 查看所有用户 | 管理员 | 管理员进入用户管理界面可以看到所有用户信息 |  |
| 查看逾期未归还的用户 | 管理员 | 管理员点击查看逾期未归还的用户按钮可以看到逾期未归还的用户信息 |  |
| 清除罚款 | 管理员 | 管理员选中用户，点击清除罚款按钮，清掉用户身上的罚款 |  |
| 用户注销 | 管理员 | 管理员选中用户，点击注销按钮，注销掉用户账号 |  |
| 上架图书 | 管理员 | 管理员输入书名，id，价格进行图书上架 |  |
| 查看所有图书 | 管理员 | 管理员进入图书管理界面可以看到所有图书信息 |  |
| 查看逾期未归还的书 | 管理员 | 管理员点击查看逾期未归还的书按钮可以看到逾期未归还的书本信息 |  |
| 下架图书 | 管理员 | 管理员选中图书，点击下架按钮，从数据库删除该书本 |  |
| 用户登录 | 读者用户 | 读者输入自己的账号和密码，进入读者操作界面 |  |
| 修改密码 | 读者用户 | 登录后，读者输入自己的旧密码和新密码，可成功修改密码 |  |
| 借书 | 读者用户 | 登录后，读者输入相应书索号，可借阅 |  |
| 还书 | 读者用户 | 登录后，读者输入相应书索号，可归还 |  |
| 查看借阅情况 | 读者用户 | 登录后，读者可在界面中间看到自己的借阅情况 |  |

1. **非功能性需求**

|  |  |
| --- | --- |
| 性能需求 | 1.生成的web页面，通过速率100KBps在10秒内可以全部下载 |
| 可用性需求 | 正常运行时间:99.9%。 |
| 安全性需求 | 1.所有涉及信息的操作必须登录后或者使用指定的设备才能完成；  2.系统只允许用户查看自己的信息，而不能查看其他用户的资料；  3.数据库进行加密 |

1. **系统设计**
   1. **基本架构设计**
      1. **基本组件设计**

参考用户角色分析，用户只需要存储普通用户。普通用户由管理员统一注册并分配默认密码123456，需要存储的功能性数据包括id、name、password和fine（罚款金额）。

由于FISCO BCOS提供的CRUD接口无法满足我们所需的全部功能，故采用mapping+结构体存储用户信息并与id进行映射，同时由于mapping无法进行遍历操作，我们另外使用一个数组Users\_index存储id。同时由于solidity没有null关键字无法判断结构体是否为空，我们在结构体中加入isUsed关键字，数据类型为bool，默认为false，可以用于判断是否发生id冲突。

表1 用户信息表 (Books）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 能否为空 | 描述 | 备注 |
| Name | bytes | 否 | 用户名 |  |
| Password | bytes | 否 | 用户密码 | 初始化时分配默认密码 |
| Fine | uint | 是 | 罚款金额 | 罚款未缴清的用户不允许借书 |
| isUsed | bool | 否 | 用户是否已注册 |  |

用户可以在终端机上完成对书本的借还同时管理员可以在管理平台上完成对书本的管理，因此需要基本的书本信息表记录书本的信息。Name、Price等属性在管理员调用函数时输入，BorrowerId、LastBorrowTime、State属性在用户进行借还书操作时输入或更改。

与用户表同理采用mapping+结构体存储信息并与id进行映射，并使用数组Books\_index存储id并加入isUsed关键字。

表2 书本信息表 (Books)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 能否为空 | 描述 | 备注 |
| Name | bytes | 否 | 书本名 |  |
| Price | uint | 否 | 书本价格 |  |
| BorrowerId | uint | 是 | 借阅人Id | 外键 |
| LastBorrowTime | uint | 是 | 接触时间戳 | 单位毫秒 |
| State | uint | 否 | 书籍状态 | 0：在库1：借出 |
| isUsed | bool | 否 | 书籍是否已上架 |  |

* + 1. **基本功能设计**

管理员根据相关名单，输入用户id和用户名，系统自动将密码初始化为123456

1. */\**
2. 添加用户
3. 返回值：
4. 1：成功
5. -1：id已被占用
6. \*/
7. function insertUser(uint user\_id, bytes name) public returns(int8)
8. {
9. User memory new\_user;
11. new\_user.Name = name;
12. new\_user.Password = "123456";
13. new\_user.isUsed = true;
15. if(Users[user\_id].isUsed || user\_id == 0)
16. {
17. return -1;
18. }
20. Users[user\_id] = new\_user;
21. Users\_index.push(user\_id);
23. emit insertEvent(msg.sender, "User", user\_id);
24. return 1;
25. }

管理员根据名单将用户注销

1. */\**
2. 删除书本
3. 返回值：
4. 1：成功
5. -1：书本不存在
6. \*/
7. function deleteBook(uint book\_id) public returns(int8)
8. {
9. if(!Books[book\_id].isUsed)
10. {
11. return -1;
12. }
14. delete Books[book\_id];
15. for(uint i; i< Books\_index.length;i++)
16. {
17. if(Books\_index[i] == book\_id)
18. {
19. delete Books\_index[i];
20. return(1);
21. }
22. }
23. }

管理员根据书册将书本上架

1. */\**
2. 删除书本
3. 返回值：
4. 1：成功
5. -1：书本不存在
6. \*/
7. function deleteBook(uint book\_id) public returns(int8)
8. {
9. if(!Books[book\_id].isUsed)
10. {
11. return -1;
12. }
14. delete Books[book\_id];
15. for(uint i; i< Books\_index.length;i++)
16. {
17. if(Books\_index[i] == book\_id)
18. {
19. delete Books\_index[i];
20. return(1);
21. }
22. }
23. }

管理员将丢失或损坏的书本下架

1. */\**
2. 删除书本
3. 返回值：
4. 1：成功
5. -1：书本不存在
6. \*/
7. function deleteBook(uint book\_id) public returns(int8)
8. {
9. if(!Books[book\_id].isUsed)
10. {
11. return -1;
12. }
14. delete Books[book\_id];
15. for(uint i; i< Books\_index.length;i++)
16. {
17. if(Books\_index[i] == book\_id)
18. {
19. delete Books\_index[i];
20. return(1);
21. }
22. }
23. }

管理员查看书本信息的三个方法

1. */\**
2. 获取某个用户借的书
3. \*/
4. function getBooks(uint user\_id) public returns(uint[], bytes[], uint[], uint[], uint[], uint[])
5. {
6. uint[] book\_ids;
7. bytes[] Names;
8. uint[] Prices;
9. uint[] BorrowerIds;
10. uint[] LastBorrowTimes;
11. uint[] States;
12. book\_ids.length = 0;
13. Names.length = 0;
14. Prices.length = 0;
15. BorrowerIds.length = 0;
16. LastBorrowTimes.length = 0;
17. States.length = 0;
18. for(uint i; i < Books\_index.length; i++)
19. {
20. if(Books\_index[i] != 0)
21. {
22. if(Books[Books\_index[i]].BorrowerId == user\_id)
23. {
24. book\_ids.push(Books\_index[i]);
25. Names.push(Books[Books\_index[i]].Name);
26. Prices.push(Books[Books\_index[i]].Price);
27. BorrowerIds.push(Books[Books\_index[i]].BorrowerId);
28. LastBorrowTimes.push(Books[Books\_index[i]].LastBorrowTime);
29. States.push(Books[Books\_index[i]].State);
30. }
31. }
32. }
34. return(book\_ids,Names,Prices,BorrowerIds,LastBorrowTimes,States);
35. }

38. */\**
39. 获取全部书
40. \*/
41. function getAllBooks() public returns(uint[], bytes[], uint[], uint[], uint[], uint[])
42. {
43. uint[] book\_ids;
44. bytes[] Names;
45. uint[] Prices;
46. uint[] BorrowerIds;
47. uint[] LastBorrowTimes;
48. uint[] States;
49. book\_ids.length = 0;
50. Names.length = 0;
51. Prices.length = 0;
52. BorrowerIds.length = 0;
53. LastBorrowTimes.length = 0;
54. States.length = 0;
55. for(uint i=0;i < Books\_index.length;i++)
56. {
57. if(Books\_index[i] != 0)
58. {
59. book\_ids.push(Books\_index[i]);
60. Names.push(Books[Books\_index[i]].Name);
61. Prices.push(Books[Books\_index[i]].Price);
62. BorrowerIds.push(Books[Books\_index[i]].BorrowerId);
63. LastBorrowTimes.push(Books[Books\_index[i]].LastBorrowTime);
64. States.push(Books[Books\_index[i]].State);
65. }
66. }
67. return(book\_ids,Names,Prices,BorrowerIds,LastBorrowTimes,States);
68. }
69. */\**
70. 获取借出的书
71. \*/
72. function getBorrowedBooks() public returns(uint[], bytes[], uint[], uint[], uint[], uint[])
73. {
74. uint[] book\_ids;
75. bytes[] Names;
76. uint[] Prices;
77. uint[] BorrowerIds;
78. uint[] LastBorrowTimes;
79. uint[] States;
80. book\_ids.length = 0;
81. Names.length = 0;
82. Prices.length = 0;
83. BorrowerIds.length = 0;
84. LastBorrowTimes.length = 0;
85. States.length = 0;
86. for(uint i; i < Books\_index.length; i++)
87. {
88. if(Books\_index[i] != 0 && Books[Books\_index[i]].State == 1)
89. {
90. book\_ids.push(Books\_index[i]);
91. Names.push(Books[Books\_index[i]].Name);
92. Prices.push(Books[Books\_index[i]].Price);
93. BorrowerIds.push(Books[Books\_index[i]].BorrowerId);
94. LastBorrowTimes.push(Books[Books\_index[i]].LastBorrowTime);
95. States.push(Books[Books\_index[i]].State);
96. }
97. }
99. return(book\_ids,Names,Prices,BorrowerIds,LastBorrowTimes,States);
100. }

管理员将用户罚款清零

1. */\**
2. 管理员将用户罚款清零
3. \*/
4. function clearFine(uint user\_id) public returns(int8)
5. {
6. return updateUserFine(user\_id,0);
7. }

管理员查看用户的两个方法

1. */\**
2. 获取全部用户
3. \*/
4. function getAllUsers() public returns(uint[], bytes[], uint[])
5. {
6. uint[] user\_ids;
7. bytes[] Names;
8. uint[] Fines;
9. user\_ids.length = 0;
10. Names.length = 0;
11. Fines.length = 0;
12. for(uint i;i < Users\_index.length;i++)
13. {
14. if(Users\_index[i] != 0)
15. {
16. user\_ids.push(Users\_index[i]);
17. Names.push(Users[Users\_index[i]].Name);
18. Fines.push(Users[Users\_index[i]].Fine);
19. }
20. }
22. return(user\_ids,Names,Fines);
23. }

26. */\**
27. 查看有罚款在身的用户
28. \*/
29. function getFinedUsers() public returns(uint[], bytes[], uint[])
30. {
31. uint[] user\_ids;
32. bytes[] Names;
33. uint[] Fines;
34. user\_ids.length = 0;
35. Names.length = 0;
36. Fines.length = 0;
37. for(uint i; i < Users\_index.length; i++)
38. {
39. if(Users\_index[i] != 0 && Users[Users\_index[i]].Fine > 0)
40. {
41. user\_ids.push(Users\_index[i]);
42. Names.push(Users[Users\_index[i]].Name);
43. Fines.push(Users[Users\_index[i]].Fine);
44. }
45. }
47. return(user\_ids,Names,Fines);
48. }
    1. **功能实现函数设计**
       1. **主要功能函数设计**

borrowBook和returnBook实现了借还书功能，系统将更改书本的State字段，当用户还书超时时，将计算用户的罚款并更改用户的Fine字段

1. */\**
2. 借书
3. 返回值：
4. 1：成功
5. 0：罚款未缴清
6. -1：查无此人
7. -2：查无此书
8. -3：书已借出
9. -4：未知错误
10. \*/
11. function borrowBook(uint borrower\_id, uint book\_id) public returns(int8)
12. {
13. User user = Users[borrower\_id];
14. Book book = Books[book\_id];
15. if(!user.isUsed)
16. {
17. return -1;*//查无此人*
18. }
20. if(user.Fine > 0)
21. {
22. return 0;*//罚款未缴清*
23. }
25. if(!book.isUsed)
26. {
27. return -2;*//查无此书*
28. }
30. if(book.State == 1)
31. {
32. return -3 ;*//书已借出*
33. }
35. if(updateBook(book\_id, borrower\_id, 1) == 1)
36. {
37. emit borrowEvent(borrower\_id, book\_id);
38. return 1;
39. }
40. return -4;
41. }

44. */\**
45. 还书
46. 返回值：
47. 1：成功
48. 0：超时
49. -1：查无此人
50. -2：查无此书
51. -3：书已在库
52. -4：未知错误
53. \*/
54. function returnBook(uint borrower\_id, uint book\_id) public returns(int8)
55. {
56. User user = Users[borrower\_id];
57. Book book = Books[book\_id];
59. if(!user.isUsed)
60. {
61. return -1;*//查无此人*
62. }
64. if(!book.isUsed)
65. {
66. return -2;*//查无此书*
67. }
69. if(book.State == 0)
70. {
71. return -3 ;*//书已在库*
72. }
74. int8 temp = updateBook(book\_id,borrower\_id,0);
75. if(temp == 0 || temp == 1)
76. {
77. return temp;
78. }
79. return -4;
80. }

userLogin对应登录功能，当用户进行登录时，系统将验证密码。

1. */\**
2. 用户登陆
3. 返回值：
4. 1：成功
5. 0：查无此人
6. -1：密码错误
7. \*/
8. function userLogin(uint user\_id, string password) public returns(int8)
9. {
10. bytes32 \_password = stringToBytes32(password);
12. int8 verify = verifyPassword(user\_id, \_password);
13. return verify;
14. }

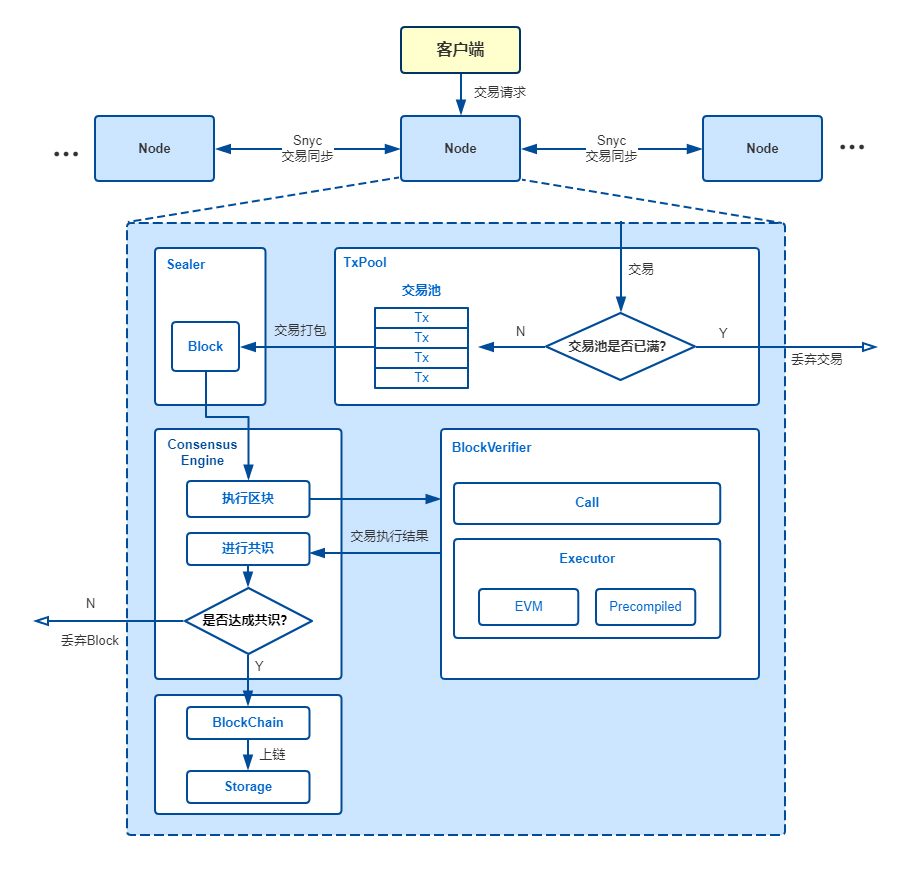
changePassword对应更改密码功能，当用户更改密码时必须验证旧密码

1. */\**
2. 用户修改密码
3. 返回值：
4. 1：成功
5. 0：查无此人
6. -1：密码错误
7. -2：新旧密码一样
8. \*/
9. function changePassword(uint user\_id, string old\_password, string new\_password) public returns(int8)
10. {
11. bytes32 \_old\_password = stringToBytes32(old\_password);
12. bytes32 \_new\_password = stringToBytes32(new\_password);
14. if(\_old\_password == \_new\_password)
15. {
16. return -2;
17. }

20. int8 verify = verifyPassword(user\_id, \_old\_password);
22. if(verify == 1)
23. {
24. Users[user\_id].Password = \_new\_password;
25. return 1;
26. }
27. else
28. {
29. return verify;
30. }
31. }
    1. **区块链设计**
       1. **区块链结构**

本系统采用单链四节点单群组的结构。

用户向节点发起RPC请求以发起交易，节点收到交易后将交易附加到交易池中，打包器不断从交易池中取出交易并通过一定条件触发将取出交易打包为区块。生成区块后，由共识引擎进行验证及共识，验证区块无误且节点间达成共识后，将区块上链。当节点通过同步模块从其他节点处下载缺失的区块时，会同样对区块进行执行及验证。



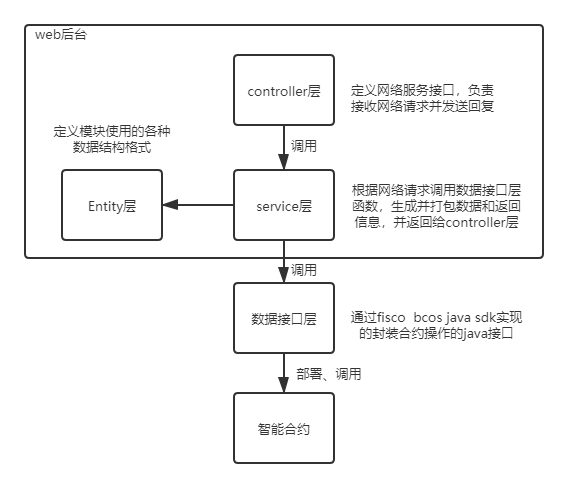
1. **后台设计**
   1. **环境设置**

Maven：4.0.0

Jdk：1.8.0\_211

IDE : IntelliJ IDEA

* 1. **后端架构**



后端的整体架构大约分为4层：数据接口层、Entity层、Service和Controller层，结构如上图所示。

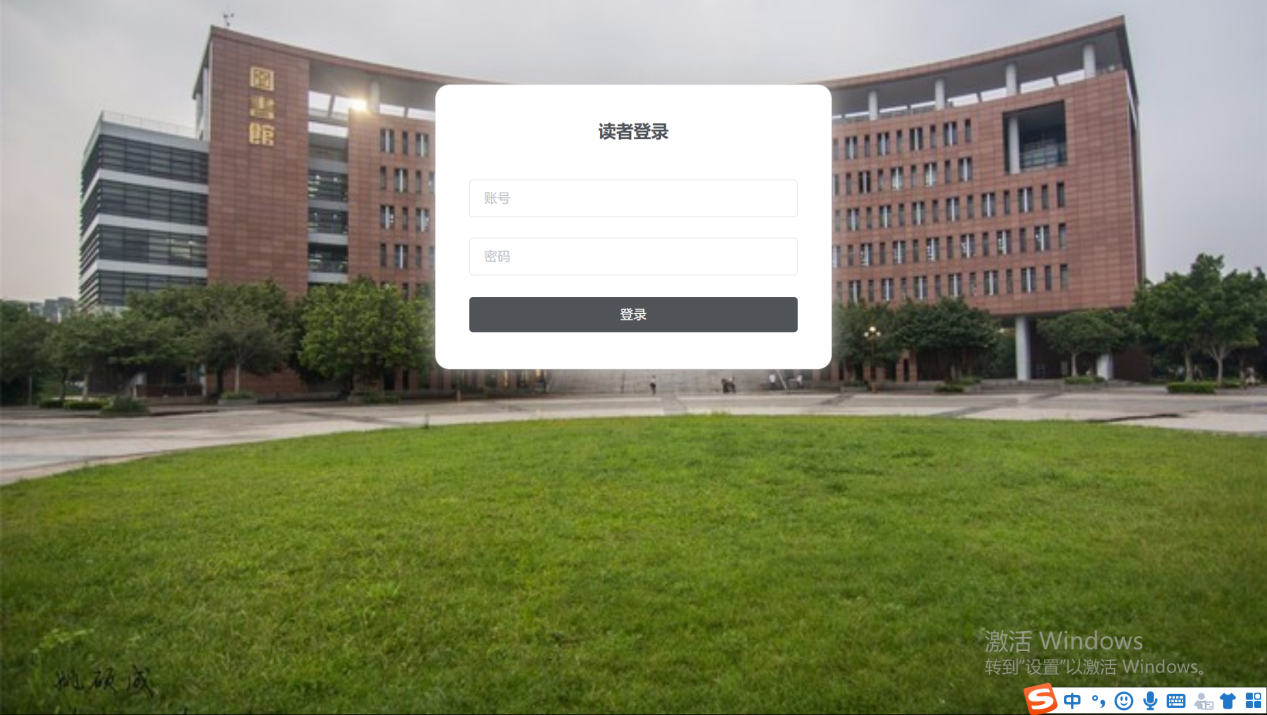
1. 数据接口：在智能合约转化成java文件以后，对其暴露的函数接口进行封装，使之能更好地被web后台的Service层调用
2. Service层：根据网络请求调用数据接口层函数，生成并打包数据和返回信息，将打包好的数据返回Controller层
3. Controller层：定义网络服务接口，响应网络请求并根据请求调用Service层，并发送Service层生成的回复
4. Entity层：定义web后台模块使用的数据结构格式
5. **界面设计**

****

****

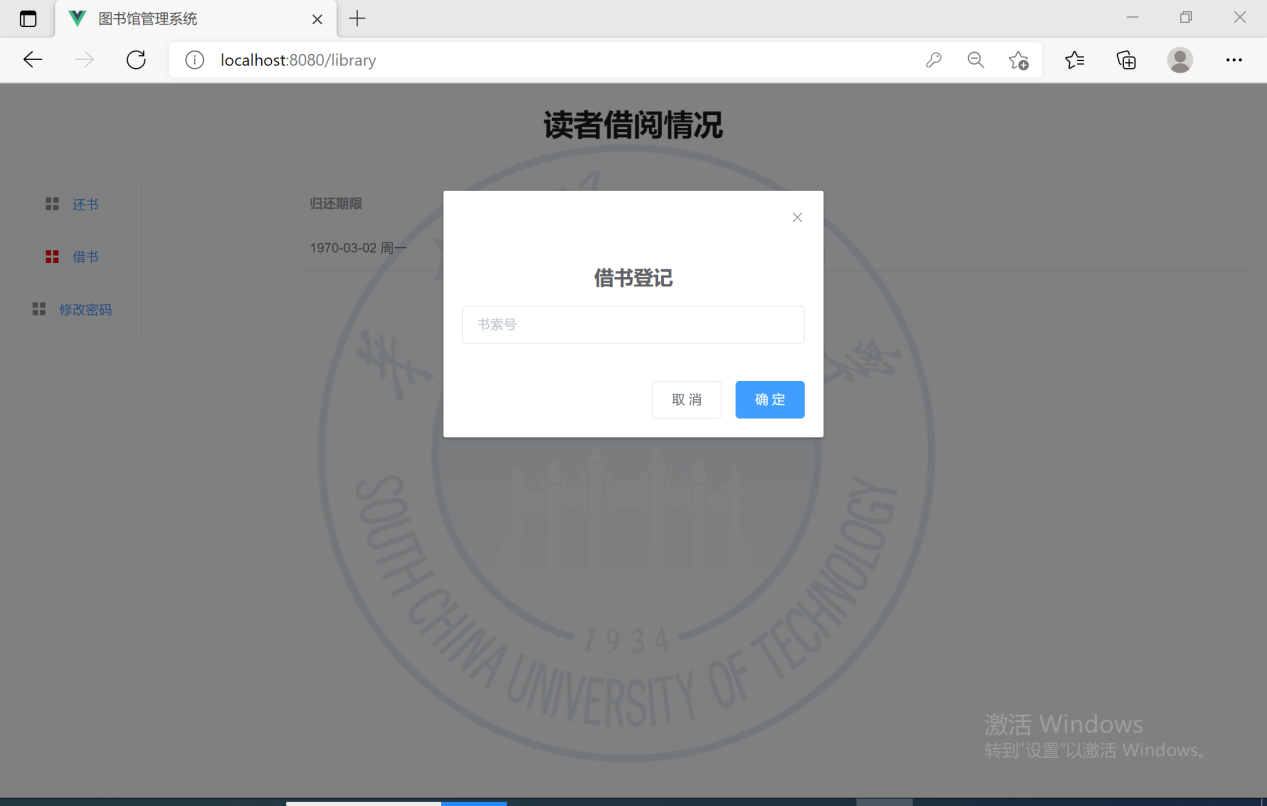
****

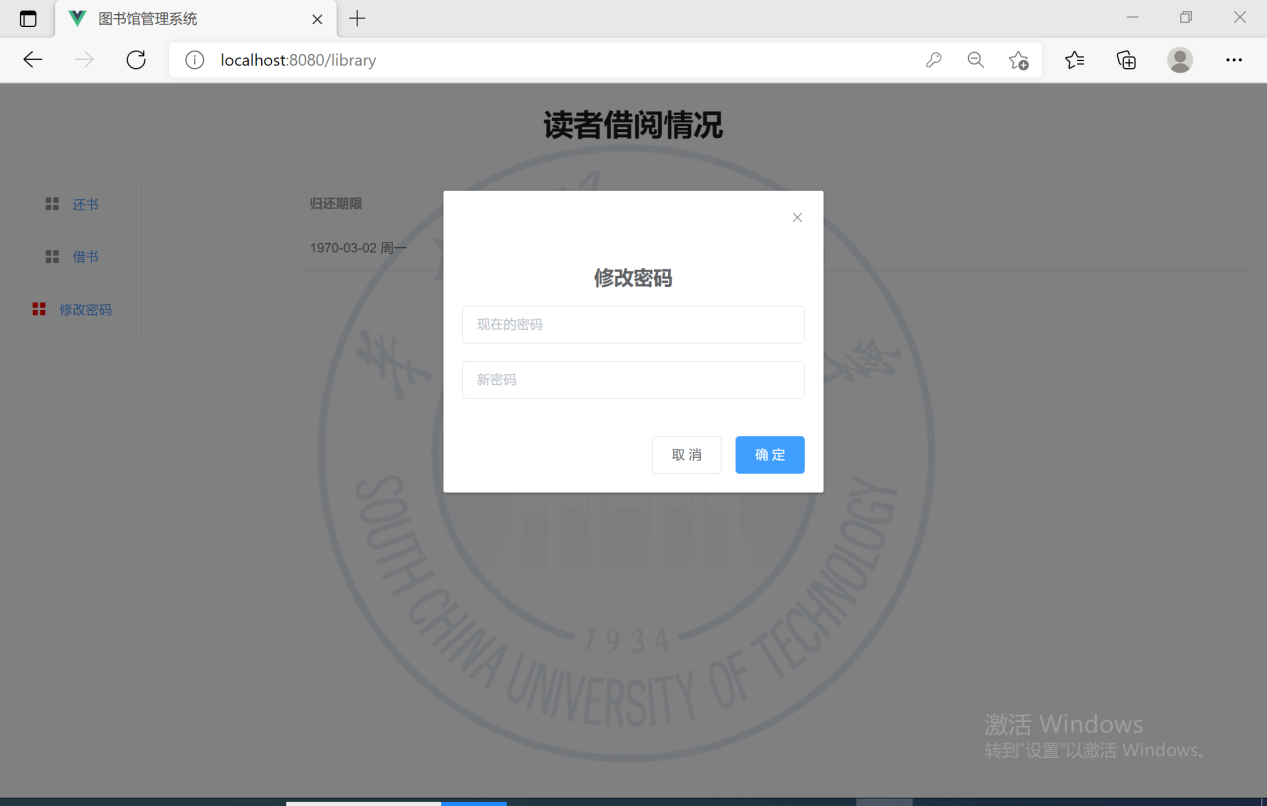
****

****

****

****

****

****

# **图书管理系统测试报告**

**目录**

**[1. 引言 5](#_Toc44498494)4**

**[1.1. 编写目的 5](#_Toc44498495)4**

**[1.2. 项目背景与系统简介 5](#_Toc44498496)4**

**[1.3. 术语 5](#_Toc44498497)4**

**[2. 参考资料 5](#_Toc44498498)4**

**[3. 测试概要 5](#_Toc44498499)4**

**[3.1. 测试的概要介绍 5](#_Toc44498500)4**

**[3.2. 用例设计方法 5](#_Toc44498501)4**

**[3.3. 测试环境与配置 5](#_Toc44498502)4**

**[4. 测试结果与缺陷分析 5](#_Toc44498503)5**

**[5. 测试建议与结论 5](#_Toc44498510)6**

**[5.1. 建议 5](#_Toc44498511)6**

**[5.2. 测试结论 5](#_Toc44498512)6**

1. **引言**
   1. **编写目的**

该报告用于对图书管理系统进行功能测试。

* 1. **项目背景与系统简介**

本次测试实验的对象是图书馆图书管理系统，包含基本的用户注册、用户登录、用户注销、上架图书等功能。

* 1. **术语**

无

1. **参考资料**

《软件测试》

1. **测试概要**
   1. **测试的概要介绍**

测试范围为图书馆图书管理系统部分功能的功能测试。

* 1. **用例设计方法**

黑盒测试用例法

* 1. **测试环境与配置**

a) 测试机1台 :安装Windows 10 系统， Edge浏览器.

b) 被测服务器1台：

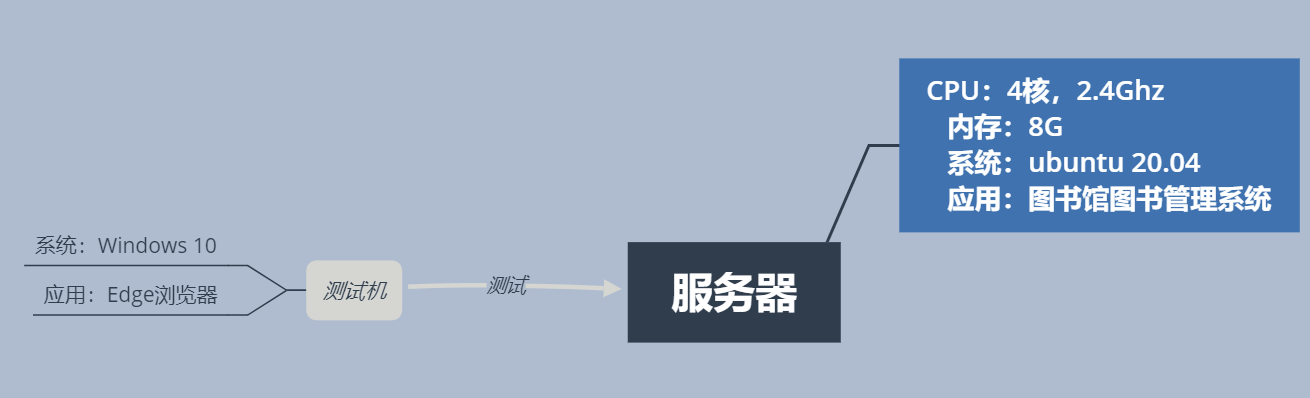
CPU：4核，2.4Ghz

内存：8G

系统：ubuntu 20.04

应用：图书馆图书管理系统

物理架构图：



1. **测试结果与缺陷分析**
   1. **测试组织**

主要测试人员：罗飞龙、刘帆、覃宏铭、陆梓文

* 1. **测试时间**

2021.7.11 测试项目开始

2021.7.11 执行测试，编写测试报告

* 1. **测试版本**

最终版

* 1. **覆盖分析** 
     1. **功能测试设计**

|  |  |  |
| --- | --- | --- |
|  | 有效等价类 | 无效等价类 |
| 用户注册 | （1）在账户栏输入数字、字母、下划线等 | （19）不输入任何内容 |
| （2）在Id栏输入数字 | （20）非法字符  （21）输入空格  （26）id重复 |
| 用户登陆 | （4）在账户栏输入数字、字母、下划线等 | （22）不输入任何内容 |
| （5）在密码栏输入数字、字母、下划线等 | （23）非法字符  （24）输入空格 |
| 上架图书 | （7）在书名栏输入汉字、数字、字母、下划线、空格等 | （25）不输入任何内容 |
| （8）在id栏内输入数字 | （26） 非法字符 |
| （8）在价格栏内输入数字 | （27） id重复 |
| 下架图书 | （11）选择图书，进行下架 | （29）图书id不存在 |
| 用户注销 | （14）选择用户，进行注销 |  |
| 清除罚款 | （16）选择用户，进行清除罚款 | （32）用户id不存在 |
| 查看逾期未还书 | （18）在书本管理界面点击查看逾期未还书按钮 |  |
| 查看逾期未还用户 | （18）在用户管理界面点击查看逾期未还的用户按钮 |  |

* 1. **残留缺陷和未解决的问题**

暂无

1. **测试建议与结论**
   1. **建议**

暂无

* 1. **测试结论**

测试完成目标，可以结束。

# **图书馆图书管理系统项目总结**

在本次区块链项目实训中，我们组决定的课题为图书馆图书管理系统。这个项目与我们之前做过项目的一个很大的不同和难点就是后端需要将数据发送到FISCO-BCOS节点处理与储存。因为大家都是初次接触区块链相关知识和了解Fisco区块链系统，所以对如何解决这个难题感到毫无头绪，无从下手。

幸运的是之后个人任务中涉及到了Java SDK的使用和业务逻辑的开发，我们这才决定了最终的项目框架：前端使用Vue和Element开发页面，后端采用Springboot开发工具，以合约和SDK相结合的方式与FISCO链进行交互。

在开发过程中，由于个人任务和小组任务的冲突，我们的开发时间并不充裕，甚至可以说较为仓促。而在这本就不充裕的时间中，我们又花费了相当长的时间在需求设计中。由于开发经验不足，前期没有定下一个共同的需求文档，导致实现起来充满了各种错误。大家对不同模块的实现方式有着不同的看法，不过幸运的是最终我们还是成功商量好了共同的需求功能和模块的实现，敲定了代码设计的具体框架。在代码实现阶段，由于开发人员对各自的模块都缺乏经验，甚至部分是从零开始，所以我们一开始上网查阅了许多资料，并相互交流，最终我们成功在5天内完成了所有代码的编写并进行了相应的测试。虽然总体上来说最终的结果较为简陋，但一开始定下的功能都基本实现了，总体来说完成度较高。

这次实训时间较为紧迫，知识掌握也不够充分，但我们组依然能够顺利地按照项目章程的规划如期完成，体现了大家的团队合作精神。我们不仅学习到了区块链的相关知识，了解了它在实际项目中的运用，还对vue框架和Springboot系统的开发更加熟练。在之后的实践中，我们将继续对这个系统进行更新和维护，争取开发出一个更加完善的区块链系统。

|  |
| --- |
| **教师评语：** |
| **团队项目实训报告成绩评定：** **分** |