**目 录**

1. [绪 论 1](#_bookmark0)
   1. [研究的背景及意义 1](#_bookmark1)
      1. [选题的背景 1](#_bookmark2)
      2. [国内外研究现状 1](#_bookmark3)
      3. [研究的意义 1](#_bookmark4)
   2. [系统目标 3](#_bookmark5)
2. [需求分析 4](#_bookmark6)
   1. [业务需求 4](#_bookmark7)
      1. [主要业务流程 4](#_bookmark8)
   2. [功能需求 8](#_bookmark9)
      1. [角色分析 8](#_bookmark10)
      2. [业务功能 8](#_bookmark11)
   3. [非功能需求 20](#_bookmark12)
      1. [环境需求 20](#_bookmark13)
      2. [性能需求 20](#_bookmark14)
      3. [安全需求 21](#_bookmark15)
3. [总体设计 23](#_bookmark16)
   1. [系统设计的原则 23](#_bookmark17)
   2. [系统体系结构设计 23](#_bookmark18)
   3. [系统功能结构设计 24](#_bookmark19)
4. [数据库设计 26](#_bookmark20)
   1. [概念结构设计 26](#_bookmark21)
      1. [设计思路 26](#_bookmark22)

[4.1.2 E-R 图 26](#_bookmark23)

* 1. [逻辑结构设计 30](#_bookmark24)
     1. [设计思路 30](#_bookmark25)
     2. [逻辑模型 30](#_bookmark26)
  2. [物理结构设计 33](#_bookmark27)
     1. [存取方式 33](#_bookmark28)
     2. [存储结构 33](#_bookmark29)

1. [界面设计 34](#_bookmark30)
   1. [界面关系图或工作流图 34](#_bookmark31)
   2. [界面设计成果 34](#_bookmark32)
      1. [主界面 34](#_bookmark33)

[5.2.2 子界面 1 35](#_bookmark34)

[5.2.3 子界面 2 35](#_bookmark35)

[5.2.4 子界面 3 36](#_bookmark36)

[5.2.5 子界面 4 36](#_bookmark37)

### [5.2.6 子界面 5 37](#_bookmark38)

### [5.2.7 子界面 6 37](#_bookmark39)

### [5.2.8 子界面 7 38](#_bookmark40)

1. [**详细设计 39**](#_bookmark41)

### [系统主要功能模块介绍 39](#_bookmark42)

### [功能模块设计 39](#_bookmark43)

### [心愿清单 39](#_bookmark44)

### [赠送清单 44](#_bookmark45)

### [书籍查询 46](#_bookmark46)

### [书籍管理 50](#_bookmark47)

[**7 编码 54**](#_bookmark48)

### [代码实现与核心算法 54](#_bookmark49)

### [关键代码实现 54](#_bookmark50)

### [核心算法部分 55](#_bookmark51)

### [代码优化分析 57](#_bookmark52)

### [Python 循环导入包问题的优化解决 57](#_bookmark53)

### [邮件发送时空时等待问题的优化 58](#_bookmark54)

[**8 测试 59**](#_bookmark55)

### [测试方案设计 59](#_bookmark56)

### [测试策略 59](#_bookmark57)

### [测试进度安排 59](#_bookmark58)

### [测试资源 60](#_bookmark59)

### [关键测试点 60](#_bookmark60)

### [测试用例构建 61](#_bookmark61)

### [测试用例编写约定 61](#_bookmark62)

### [测试用例设计 61](#_bookmark63)

### [关键测试用例 62](#_bookmark64)

### [测试用例维护 63](#_bookmark65)

1. [**总结与展望 65**](#_bookmark66)

### [设计工作总结 65](#_bookmark67)

### [未来工作展望 65](#_bookmark68)

[**谢 辞 67**](#_bookmark69)

[**参考文献 68**](#_bookmark70)

[**附录 A 外文翻译—原文部分 70**](#_bookmark71)

[**附录 B 外文翻译—译文部分 75**](#_bookmark72)

[**附录 C 软件使用说明书 79**](#_bookmark73)

[**附录 D 主要源代码 82**](#_bookmark74)

华

# 1 绪 论

## 研究的背景及意义

### 选题的背景

每个人都有自己的闲置书籍，而可能有部分闲置书籍其实是自己已经完全不需要了的，对于那些没有收藏书籍爱好的人来说，其实与其这样白白让书籍闲置，倒不如将书籍赠给其他需要这些书的人，让书籍产生更多价值，同时也能帮助到别人。

随着计算机的普及与互联网行业的蓬勃发展，我们现在有很多方法可以实现一个针对公益赠书的网站系统。为了实现这一想法，本文对赠书及索书业务进行了详细的分析，构思了系统的设计，从而实现了赠书者可以直接通过网页上传自己的闲置书籍信息，让其他用户可以向赠书者索要书籍，而用户也可以通过积分的形式，消耗积分来向其他用户索要他们的闲置书籍。

### 国内外研究现状

目前，国外已有许多出名的闲置书籍捐赠平台，深受大学生等群体的喜爱。而国内目前也出现了“享物说”这样的优秀闲置物品捐赠平台，但是其针对的不仅是闲置书籍，还有其他很多杂物。我希望能有一个专门提供闲置书籍捐赠服务的平台给广大互联网用户， 本人暂时还没有发现专门的捐赠书籍的平台出现在大众视野中，本课题的目的也是为了给国内网络用户提供一个比较有针对性的、公益性的书籍捐赠平台。本平台只面向一小部分真正有分享精神、热爱读书的人，因为纯公益性质的赠书，注定不会为多数人所喜爱。即使不会成为一种主流，本课题的开展仍然是非常有积极意义的，人们在捐赠书籍的同时， 不仅是传递了书籍，更是为中国社会注入正能量。

### 研究的意义

随着计算机以及计算机网络的普及和全国各地网络的日益完善、健全，生活与工作的计算机网络化将把人与人之间的距离拉的越来越近。公益活动也可以借助计算机以及计算机网络的发展得以在更广阔的范围内传播，因此，将公益活动网络化是可行的、有必要的， 公益活动的传播范围越广，其传递的正能量的范围也就越广。本课题“鱼书”公益赠书信息管理平台本质上就是一个公益项目，它将借助网络以及数据信息化的力量、为全国各地的爱书人士传播一种“爱书、读书、博爱、交流”的正能量，人们可以自由在该平台上互相赠送书籍，让旧书的价值，永不过时。

很多年轻人都希望拥有不再读书所需要的闲置书籍,而现在人们对于处理这些闲置书籍的另一种方式要么是将书堆在家中进行闲置,要么是将书当作二手的书籍进行卖掉,要么

是直接将书当作家中的废品进行卖掉。显然,以上这些的做法都已经无法真正充分体现出闲置书籍本身所可能具有的社会价值,于是,我们专门设计了"鱼书"的公益赠书数据库信息管理平台,让人们的这些闲置书籍重新收集起来体现出自身的社会价值,帮助社会更多的人读书学习。用户可以通过在这个平台直接免费向他人索取他们所捐赠的书籍, "鱼书"的公益赠书信息管理平台的整体设计与其实现,不仅充分体现了一种读书、爱书的积极精神,也充分体现了乐于助人、无私奉献的积极精神,为整个中国的社会进一步传递了"正能量"。

## 系统目标

设计一个在线公益性质的捐赠书籍或者向他人索要书籍的平台，用于在网络平台实现向他人赠书、向他人请求赠书、交流读书感悟等基本任务管理。

本系统具备如下基本功能：普通用户通过用户名及密码登陆到“鱼书”公益赠书信息管理系统，按照自己的需要选择捐赠书籍或者向他人请求赠送书籍。当用户选择捐书时， 通过网站首页搜索栏输入书籍的 ISBN 编号或者关键字，在搜索结果中找到自己想捐的书籍，之后选择确认赠书，其他想要此书的用户即可在该书籍的捐赠者列表中找到并联系你请求赠书；当用户选择向他人请求赠书时，通过网站首页搜索栏输入书籍的 ISBN 编号或者关键字，在搜索结果中找到自己想要的书籍，查看该书籍的捐赠者列表，与某一捐赠者联系并向他请求赠送该书籍；用户也可以选择在读书交流论坛页发表、删除、评论、帖子， 或者修改自己已发布帖子的内容。

后台系统管理员在普通权限用户的权限基础上还可以对普通用户的基本信息进行修改, 如增加或扣除向其他用户索取书籍时所需要消耗的鱼豆，或者用户的用户名等。。

# 需求分析

## 业务需求

* + 1. 能够实现书籍详细信息的展示、通过 ISBN 号或书名或作者搜索书籍、赠送书书籍、心愿清单、书籍交易等功能。
    2. 实现对书籍交易过程中产生的所有业务数据的管理，如交易时间、交易状态的管理与维护。
    3. 具有后台系统的管理维护功能，能够实现对用户信息、交易记录、书籍介绍等信息进行管理与维护。
    4. 响应速度快，安全性较高，能抵御一些常见 web 攻击。
    5. 系统管理能够保证运行稳定、易于使用和维护。

### 2.1.1 主要业务流程

1. 添加书籍到赠书清单，当用户希望赠送某本书籍时可通过如图 2-1 流程实现本操作：

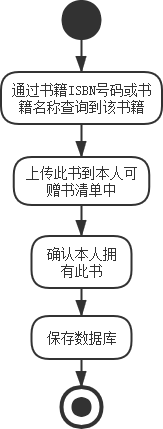


图 2-1 赠书清单活动图

1. 登录注册操作流程如图 2-2 所示：

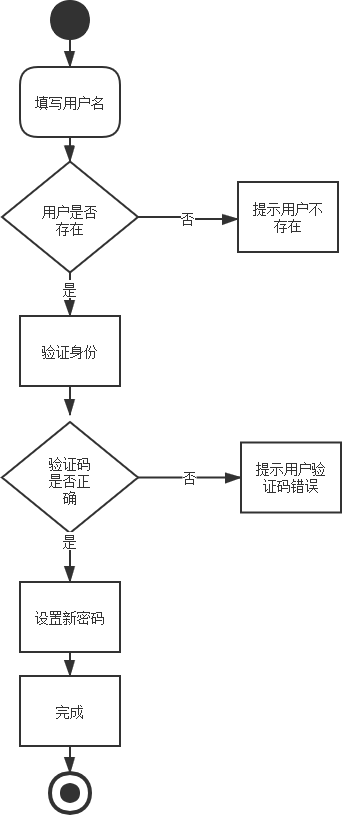


图 2-2 登录注册活动图

1. 申请交易活动图如图 2-3 所示：

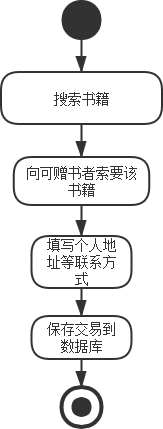


图 2-3 申请交易活动图

1. 添加心愿活动图如图 2-4 所示：

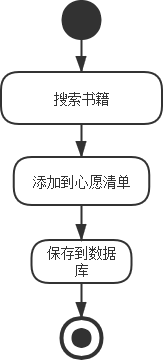


图 2-4 添加心愿活动图

1. 删除心愿清单中心愿活动图如图 2-5 所示：

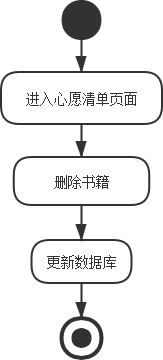


图 2-5 删除心愿活动图

## 功能需求

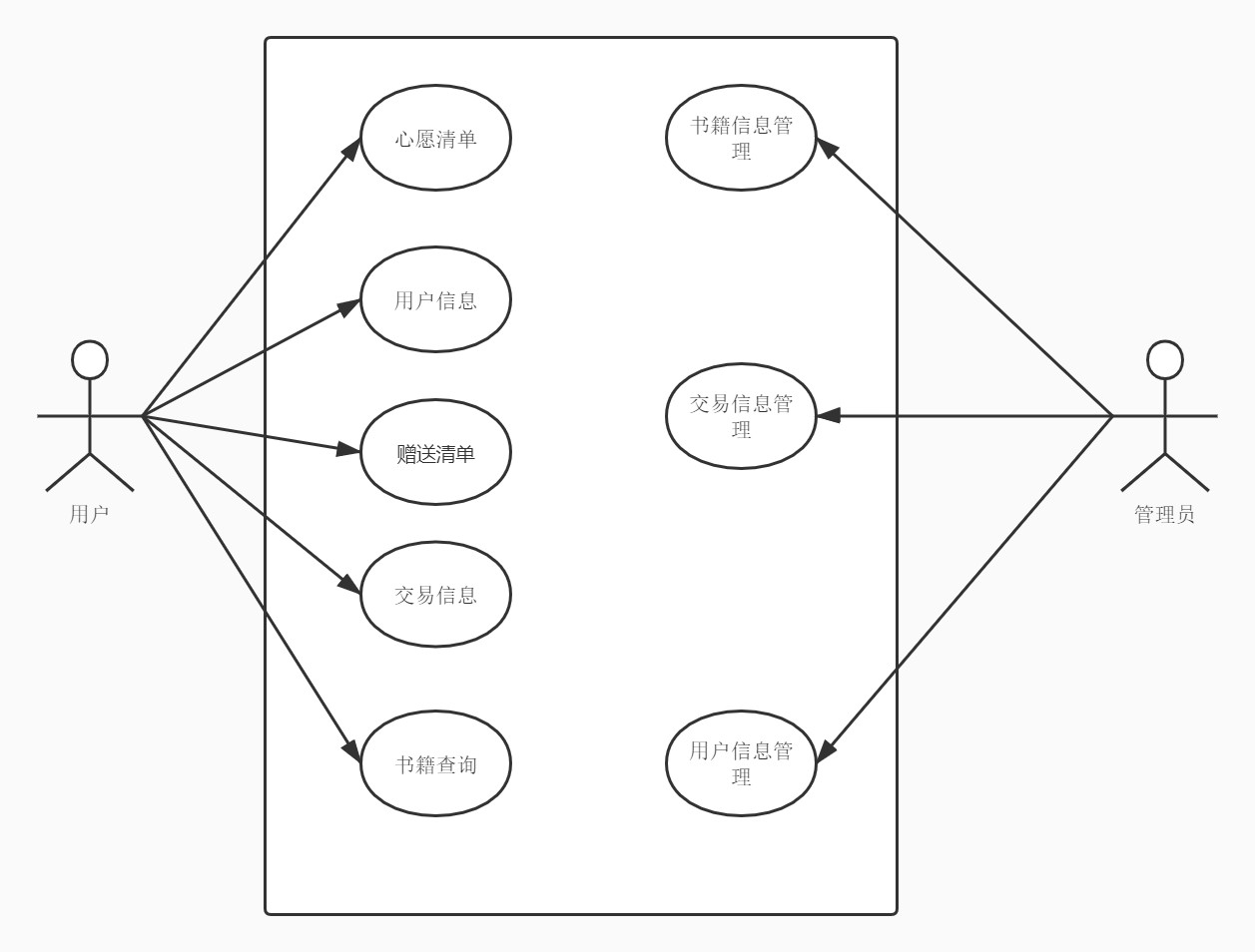
### 角色分析

系统角色表如表 2-1 所示：

表 2-1 系统角色表

|  |  |
| --- | --- |
| 角色 | 职责或功能 |
| 普通用户 | 赠送书籍与向他人索要书籍 |
| 系统管理员 | 系统管理员负责管理和监督维护整个系统的用户关系和组织的结构, 负责对用户、角色、用户级别的增、删、改、查等工作进行管理。 |

### 业务功能

以下从业务角度出发，给出了系统的总体功能用例图，包含了用户信息、交易信息、书籍查询等功能用例，如下图 2-6 所示：

1. 交易信息管理模块

图 2-6 系统总体用例图

交易信息管理的用例图如下图 2-7 所示：

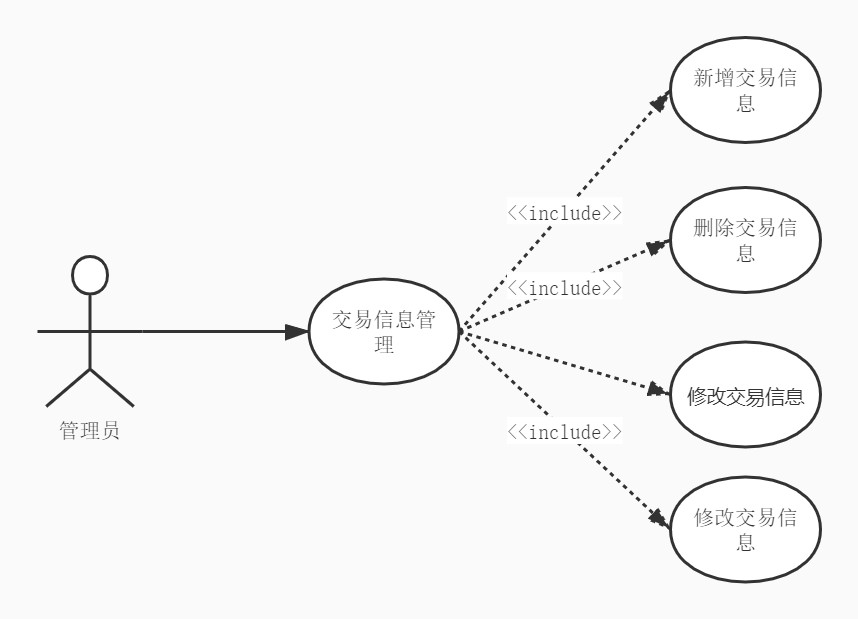


图 2-7 交易信息用例图

交易信息管理用例描述如表 2-2 所示：

表 2-2 交易信息用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 0001 | 用例名称 | 交易信息管理 |
| 用例描述 | 管理员进入交易信息管理页面，对用户的交易信息进行增删改查等操  作 | | |
| 执行者 | 管理员 | | |
| 前置条件 | 管理员必须已经登录 | | |
| 后置条件 | 将增、删、改、查的结果映射到数据库中 | | |
| 涉众利益 | 管理员希望可以返回操作成功或失败 | | |
| 用例场景 | 后台交易信息管理页 | | |

续表 2-2

|  |  |
| --- | --- |
| 基本事件流 | 基本流程（Base Flow）   1. 管理员进入交易信息管理页； 2. 找到需要修改的记录； 3. 进入信息修改页面； 4. 对交易信息进行增、删、改、查操作； 5. 数据库根据管理员的操作进行相对应的更新； 字段列表（Filed List）   交易信息=赠书者 id+赠书者昵称+索书者 id+索书者昵称+交易状态+交易创建时间+收书地址+索书者联系电话+书籍 ISBN 编号+书籍名称+书  籍作者 |
| 特殊需求  （Special Requirement） | 1. 管理员所有操作应在 3s 内得到响应； 2. 页面简洁，管理员能在 1min 内完成业务； |
| 编写人 | 周磊 |

1. 用户信息管理模块

用户信息管理用例图如图 2-8 所示：

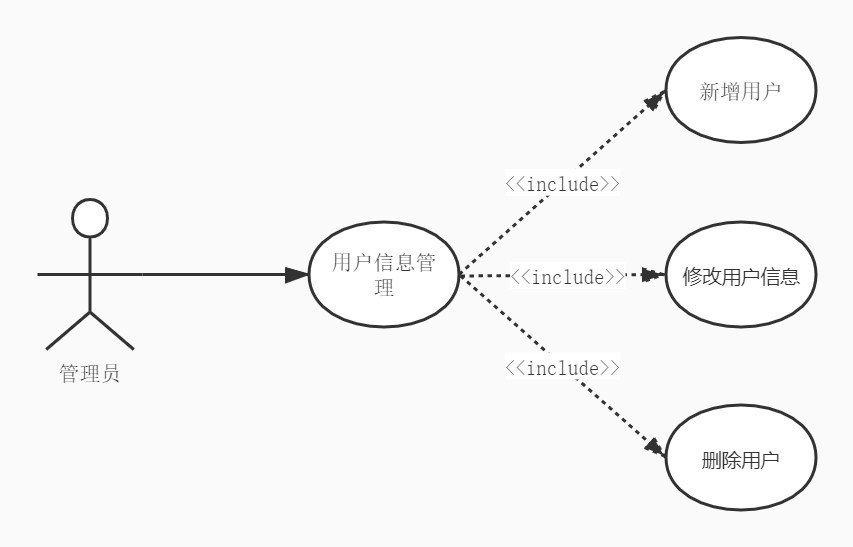


图 2-8 用户信息管理用例图

用户信息管理用例描述如表 2-3 所示：

表 2-3 用户信息管理用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 0002 | 用例名称 | 用户信息管理 |
| 用例描述 | 管理员进入用户信息管理页面，对用户个人信息进行增删改查等操作或新增用户、删除用户 | | |

续表 2-3

|  |  |
| --- | --- |
| 执行者 | 管理员 |
| 前置条件 | 管理员必须已经登录 |
| 后置条件 | 将对用户信息增、删、改、查与用户的新增与删除操作的结果映射到数据库中 |
| 涉众利益 | 管理员希望可以返回操作成功或失败 |
| 用例场景 | 后台用户信息管理页 |
| 基本事件流 | 基本流程（Base Flow）   1. 管理员进入用户信息管理页； 2. 对用户信息进行增、删、改、查操作，新增或删除用户； 字段列表（Filed List）   用户信息=用户 id+用户昵称+邮箱+电话号码+鱼豆数量 |
| 特殊需求  （Special Requirement） | 1. 管理员所有操作应在 3s 内得到响应； 2. 页面简洁，管理员能在 1min 内完成业务； |
| 编写人 | 周磊 |

1. 书籍信息管理模块

书籍信息管理用例图如图 2-9 所示：

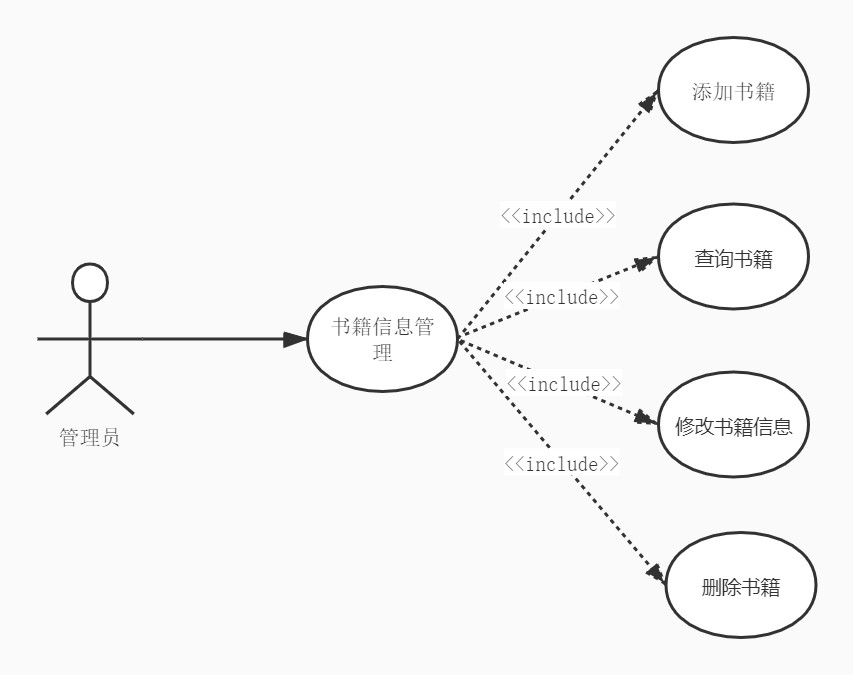


图 2-9 书籍信息管理用例图

书籍信息管理用例描述如表 2-4 所示：

表 2-4 书籍信息管理用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 0003 | 用例名称 | 书籍信息管理 |
| 用例描述 | 管理员进入书籍信息管理页面，对书籍信息进行增删改查等操作或  新增书籍、删除书籍，或查询书籍信息 | | |
| 执行者 | 管理员 | | |
| 前置条件 | 管理员必须已经登录 | | |
| 后置条件 | 将对书籍信息增、删、改、查与书籍的新增与删除操作的结果映射  到数据库中 | | |
| 涉众利益 | 管理员希望可以返回操作成功或失败 | | |
| 用例场景 | 后台书籍信息管理页 | | |

续表 2-4

|  |  |
| --- | --- |
| 基本事件流 | 基本流程（Base Flow）  1. 管理员进入书籍信息管理页；  对书籍信息进行增、删、改、查操作，新增或删除用户； 字段列表（Filed List）  书籍信息=书籍 id+书籍 ISBN 号码+书籍作者+出版日期+摘要信息 |
| 特殊需求  （Special Requirement） | 1. 管理员所有操作应在 3s 内得到响应； 2. 页面简洁，管理员能在 1min 内完成业务； |
| 编写人 | 周磊 |

1. 心愿清单模块

心愿清单用例图如图 2-10 所示：

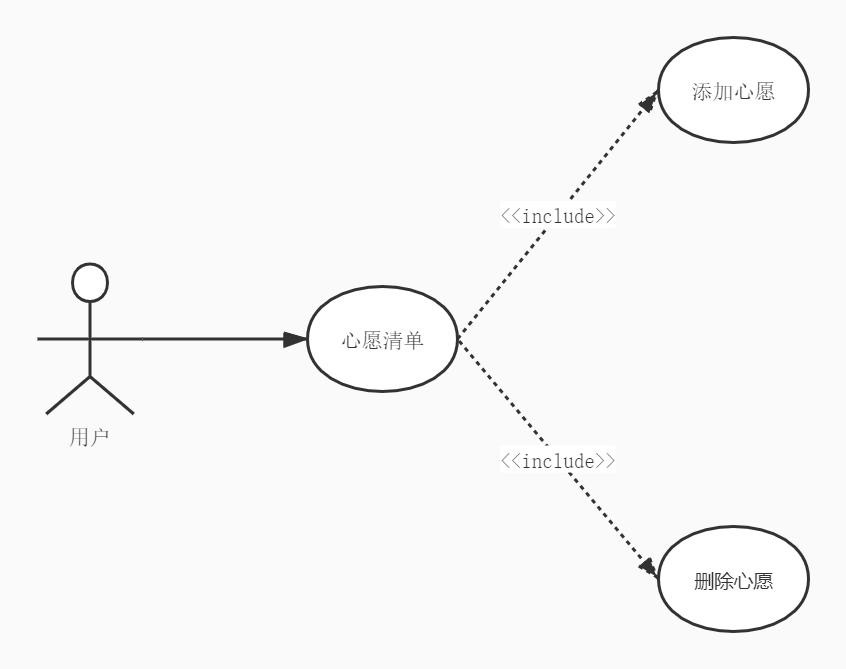


图 2-10 心愿清单用例图

心愿清单用例描述如表 2-5 所示：

表 2-5 心愿清单用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 0004 | 用例名称 | 心愿清单 |
| 用例描述 | 用户往心愿清单中添加或删除书籍 | | |

续表 2-5

|  |  |
| --- | --- |
| 执行者 | 用户 |
| 前置条件 | 用户必须已经登录 |
| 后置条件 | 将对心愿清单的操作结果映射到数据库中 |
| 涉众利益 | 用户希望系统给出操作成功或失败的提示信息 |
| 用例场景 | 心愿清单页面 |
| 基本事件流 | 基本流程（Base Flow）   1. 用户进入心愿清单页面； 2. 选择撤销书籍；   字段列表（Filed List）  书籍信息=书籍 id+书籍 ISBN 号码+书籍作者+出版日期+摘要信息心愿清单=创建时间+心愿 id+书籍 ISBN 号码 |
| 特殊需求  （Special Requirement） | 1. 管理员所有操作应在 3s 内得到响应； 2. 页面简洁，管理员能在 1min 内完成业务； |
| 编写人 | 周磊 |

1. 赠送清单模块

赠送清单用例图如图 2-11 所示：

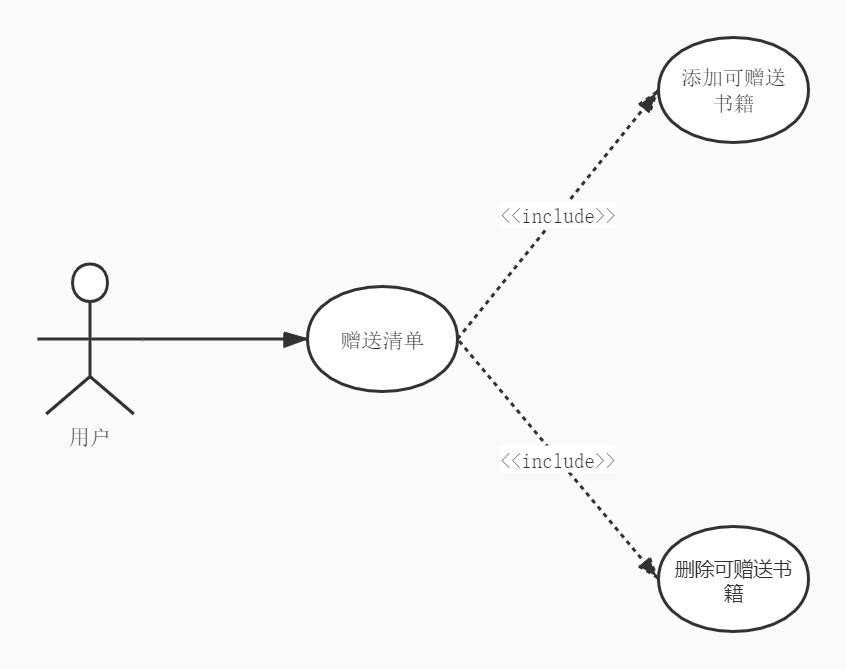


图 2-11 赠送清单用例图

赠送清单用例描述如表 2-6 所示：

表 2-6 赠送清单用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 0005 | 用例名称 | 赠送清单 |
| 用例描述 | 用户往清单清单中添加或删除书籍 | | |
| 执行者 | 用户 | | |
| 前置条件 | 用户必须已经登录 | | |
| 后置条件 | 将对赠送清单的操作结果映射到数据库中 | | |
| 涉众利益 | 用户希望系统给出操作成功或失败的提示信息 | | |
| 用例场景 | 赠送清单页面 | | |
| 基本事件流 | 基本流程（Base Flow）   1. 用户进入赠送清单页面； 2. 选择撤销书籍；   字段列表（Filed List）  书籍信息=书籍 id+书籍 ISBN 号码+书籍作者+出版日期+摘要信息心愿清单=创建时间+心愿 id+书籍 ISBN 号码 | | |
| 特殊需求  （Special Requirement） | 1. 管理员所有操作应在 3s 内得到响应； 2. 页面简洁，管理员能在 1min 内完成业务； | | |
| 编写人 | 周磊 | | |

1. 用户信息模块

用户信息用例图如图 2-12 所示：

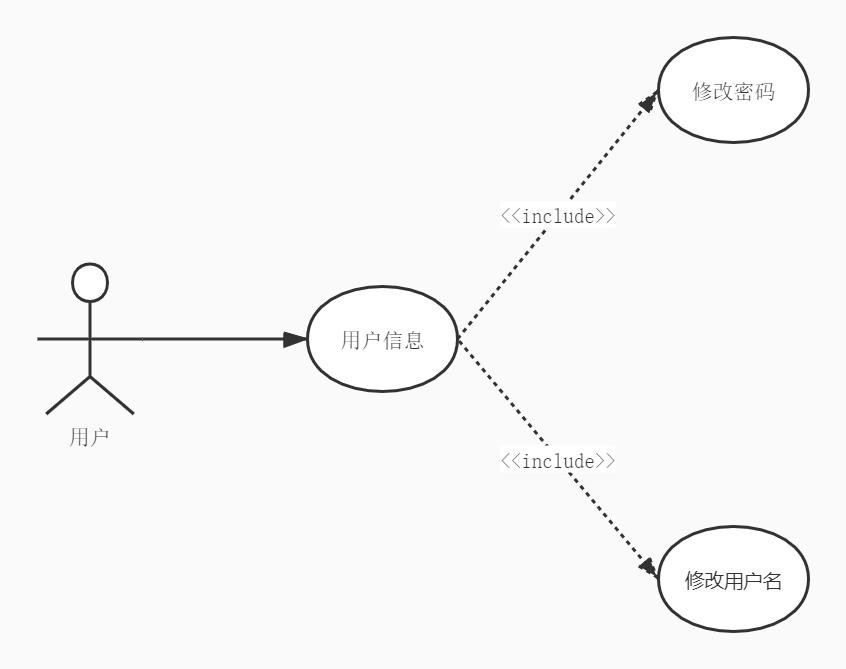


图 2-12 用户信息用例图

赠送清单用例描述如表 2-7 所示：

表 2-7 用户信息用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 0006 | 用例名称 | 用户信息 |
| 用例描述 | 用户修改个人基本信息 | | |
| 执行者 | 用户 | | |
| 前置条件 | 用户必须已经登录 | | |
| 后置条件 | 将对个人信息修改的操作结果映射到数据库中 | | |
| 涉众利益 | 用户希望系统给出操作成功或失败的提示信息 | | |
| 用例场景 | 个人信息页面 | | |

续表 2-7

|  |  |
| --- | --- |
| 基本事件流 | 基本流程（Base Flow）   1. 用户进入个人主页； 2. 选择需要修改的信息； 3. 修改个人信息；   字段列表（Filed List）  用户信息=用户 id+用户昵称+邮箱+电话号码+鱼豆数量 |
| 特殊需求  （Special Requirement） | 1. 管理员所有操作应在 3s 内得到响应； 2. 页面简洁，管理员能在 1min 内完成业务； |
| 编写人 | 周磊 |

1. 交易信息模块

交易信息用例图如图 2-13 所示：

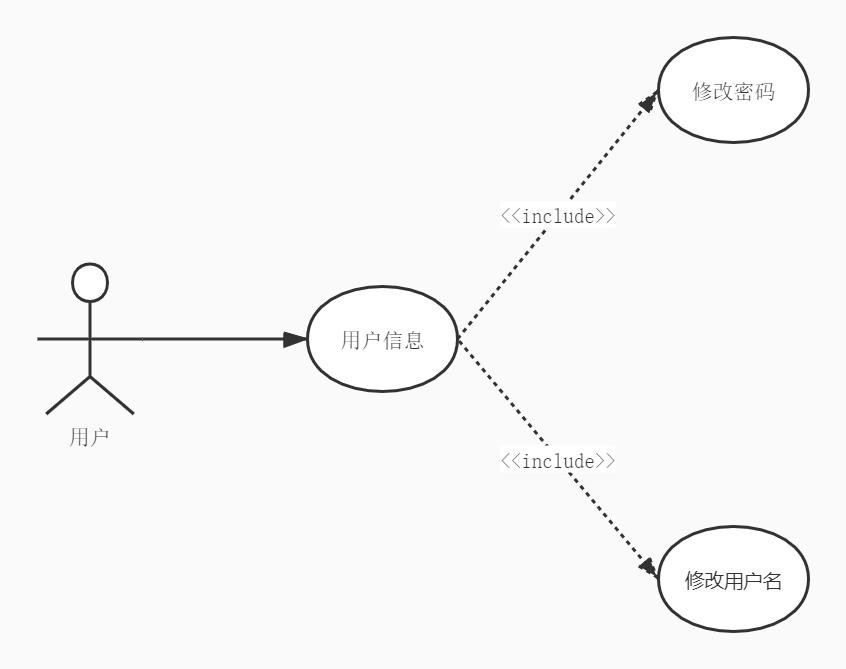


图 2-13 交易信息管理用例图

交易信息用例描述如表 2-8 所示：

表 2-8 交易信息用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 0007 | 用例名称 | 交易信息 |

续表 2-8

|  |  |
| --- | --- |
| 用例描述 | 用户对交易信息进行操作 |
| 执行者 | 用户 |
| 前置条件 | 用户必须已经登录 |
| 后置条件 | 将对个人信息修改的操作结果映射到数据库中 |
| 涉众利益 | 用户希望系统给出操作成功或失败的提示信息 |
| 用例场景 | 个人信息页面 |
| 基本事件流 | 基本流程（Base Flow）   1. 用户进入鱼漂页面； 2. 对未完成的交易选择已邮寄或者拒绝； 3. 查看已完成的交易记录； 字段列表（Filed List）   交易信息=赠书者 id+赠书者昵称+索书者 id+索书者昵称+交易状态+交易创建时间+收书地址+索书者联系电话+书籍 ISBN 编号+书籍名称+书籍作者 |
| 特殊需求  （Special Requirement） | 1. 管理员所有操作应在 3s 内得到响应； 2. 页面简洁，管理员能在 1min 内完成业务； |
| 编写人 | 周磊 |

1. 书籍查询模块

书籍查询用例图如图 2-14 所示：

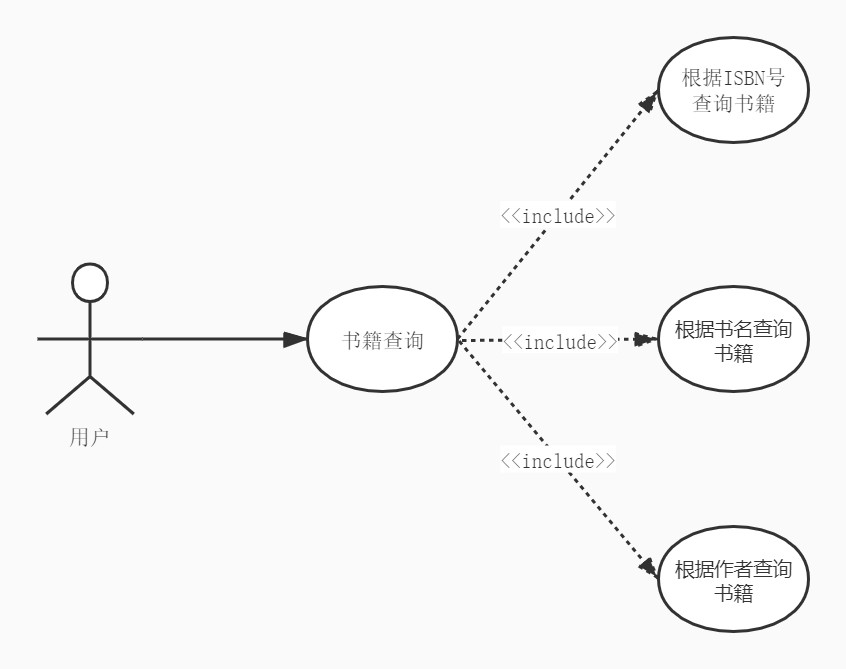


图 2-14 书籍查询用例图

书籍查询用例描述如表 2-9 所示：

表 2-9 书籍查询用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 0008 | 用例名称 | 书籍查询 |
| 用例描述 | 用户对交易信息进行操作 | | |
| 执行者 | 用户 | | |
| 前置条件 | 用户必须已经登录 | | |
| 后置条件 | 返回查询结果 | | |
| 涉众利益 | 无 | | |
| 用例场景 | 个人信息页面 | | |
| 基本事件流 | 基本流程（Base Flow）   1. 用户进入系统前台任意页面； 2. 在页面最上方搜索框内输入搜索关键字； 3. 查看系统返回结果； 字段列表（Filed List）   书籍信息=书籍 id+书籍 ISBN 号码+书籍作者+出版日期+摘要 | | |

续表 2-9

|  |  |
| --- | --- |
| 特殊需求  （Special Requirement） | 1. 管理员所有操作应在 3s 内得到响应； 2. 页面简洁，管理员能在 1min 内完成业务； |
| 编写人 | 周磊 |

## 非功能需求

### 环境需求

本系统服务器端环境如下表 2-10：

表 2-10 服务器端环境

|  |  |
| --- | --- |
| 系统所使用数据库 | MySQL |
| 操作系统 | Windows 2008 R2 |
| Web 服务器 | Flask 内置服务器 |
| 数据库 | MySQL5.7.26 |

本系统客户端环境如下表 2-11：

表 2-11 客户端环境

|  |  |
| --- | --- |
| 操作系统 | Windows XP 或其以上版本操作系统 |
| 浏览器 | IE 6.0 以及其以上版本（IE7.0、IE8.0）、Chrome、Firefox |
| 分辨率 | 最佳效果为 1024×768 像素 |

### 性能需求

1. 可维护性

维护的方法:系统在使用中遇到了数据库管理系统损坏、数据丢失的特殊情况下,能够根据要求进行数据库备份将丢失的数据完全找回,实现对重要数据的恢复,应对某些特殊的情况。

1. 易操作性

界面设计：本系统实现非常简洁明了的操作界面，用户能够快速熟悉系统的各个模块的功能及操作。

1. 可扩展性

功能扩展:根据系统从系统结构、功能设计、管理对象等各种功能方面的要求对功能进行扩展应用来设计和考虑,以更好地满足用户今后系统对于扩充和扩大使用系统功能范围的各种功能要求。

软硬件的升级:对系统应采取的措施包括硬件和软件平台,软件和硬件的升级应对负载平衡和管理机制的系统可扩展性进行充分考虑。软件系统升级需要用户具有灵活的系统扩展和管理能力,来帮助用户适应关键的技术软件和应用硬件的开发及其管理应用能力的上升。

1. 开放性

系统的所有应用程序系统数据传输格式技术标准要求应必须完全符合国家制定相关的行业标准及国际相关应的行业标准,以此也才能够有效确保应用系统的所有应用程序和系统数据之间具有良好的互操作性和能够减少被恶意移植的风险可能。

### 安全需求

1. 容错性

在用户可能操作错误的情况下必须保证系统不完全失效,仍然保证系统能够正常的工作,当发现用户计算机存在错误的输入或者错误操作导致非法输入数据的产生时,系统应保证具有一定的安全性和容错处理机制。在这种的情况下,系统应及时给出友好的提示,让输入错误的用户重新进行输入。

1. 系统的外在环境安全

安全系统要以能够充分考虑整个网络的安全较高级别,多层次的网络安全性能保护措施为主要技术前提,包括系统的数据备份,防火墙,用户的权限和其他的措施,以能够确保用户的数据安全和系统的机密信息不被恶意泄露;同时要考虑到系统的硬件和软件故障及时恢复等多层次的应急安全保护措施,以能够有效保障整个网络的安全和数据处理的安全性。同时要形成相对独立的安全数据保护机制,以能够有效防止对来自系统外的未经用户授权的数据非法访问。

1. 系统内部安全

在系统能够抵御外部攻击的同时，系统也需要能够满足所有用户正常的使用，不会发生一些明显的错误。

1. 系统运行安全

本网络系统的特点是能够有效应对一些常见的 web 网络攻击。本系统例如应对 ddos 网络攻击。ddos 网络攻击是一种指故意的通过攻击对象的网络协议所能实现的网络缺陷或直接通过野蛮的手段残忍地间接耗尽被侵入攻击对象的网络服务资源,目的主要是为了让被侵入目标的计算机或目标网络设备无法及时提供正常的服务或网络资源的访问,使被侵入目标系统或者服务对象的系统无法停止正常响应甚至导致系统崩溃,而在此类网络攻击中并不一定包括任何被侵入的目标计算机服务器或者侵入目标计算机网络的设备。这些被侵入服务对象的资源主要包括计算机网络的带宽,文件系统或者空间内存容量,开放的服务 器进程或者允许的网络连接。这种网络攻击可能会直接导致网络资源的严重匮乏,无论是

计算机的数据处理速度多快、内存容量多大、网络访问带宽的速度多快都导致系统无法完全避免这种网络攻击可能带来的严重后果。

# 总体设计

## 系统设计的原则

本系统遵循以下三个设计原则：

1. 实用性原则，将系统的实用性摆在首位，任何一个系统真正具有实用性才是最为关键点，系统需满足赠书用户与索书用户的基本需求。
2. 安全性原则。对于大部分常见的网络攻击方式，系统都应该具有阻挡他们的能力， 并且需要建立数据库备份，以防特殊情况下数据库丢失。
3. 可扩展性原则。预留接口可以大大提高系统的可扩展性，本系统就预留了非常多的数据接口，方便之后对功能的扩展。

## 系统体系结构设计

B/S 架构的全称为 Browser/Server，即浏览器/服务器结构。Browser 指的是Web 浏览器，极少数事务逻辑在前端实现，但主要事务逻辑在服务器端实现，Browser 客户端，WebApp 服务器端和 DB 端构成所谓的三层架构。B/S 架构的系统无须特别安装，只有Web 浏览器即可。

B/S 架构中，由 Web 浏览器来显示逻辑显示逻辑，而所有的事务处理逻辑在放在了 We 网络应用上，这样做可以大大减少客户端的压力。因为客户端本身并不包含大量业务逻辑， 因此也被成为瘦客户端。

B/S 架构优点如下：

* + 1. 客户端仅仅需要安装浏览器。
    2. B/S 模式架构下的客户管理可以直接将一个服务器平台放在一个大的广域网上, 通过一定的不同客户访问权限和服务控制权来实现一个多服务平台不同客户之间交互连接访问的主要功能和服务目的,交互访问性能比较强。
    3. B/S 架构无需升级多个客户端，升级服务器即可。

B/S 架构缺点如下：

1. 在某些跨平台浏览器上，B/S 架构不尽如人意。
2. 表现要达到C/S 程序的程度需要花费不少精力。
3. B/S 架构的最大问题在速度和安全性上需要花费巨大的设计成本。
4. 客户端服务器端的交互是请求-响应模式，通常需要刷新页面，这并不是客户乐意看到的。

正如前文所说，C/S 和 B/S 都可以进行同样的业务处理，但是 B/S 随着 Internet 技术的兴起，是对C/S 结构的一种改进或者扩展的结构。相对于C/S，B/S 具有如下优势：

1. 分布性：可以随时进行查询、浏览等业务。
2. 业务扩展方便：通过预留的接口可以方便扩展。
3. 维护简单方便:所有网站用户的同步和实时更新。
4. 开发简单，共享性强，成本低，数据可以持久存储在云端而不必担心数据的丢失。

## 系统功能结构设计

系统功能分析图如图 3-1：

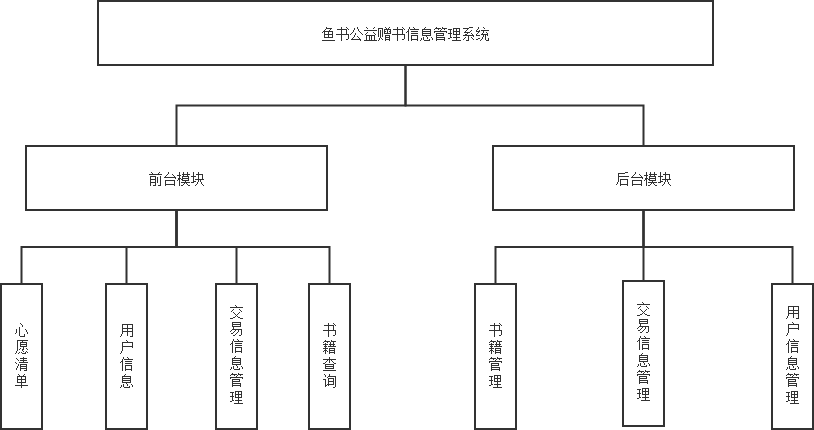


图 3-1 系统功能结构图

如上图所示，本系统一共分为两个核心子系统：前台子系统和后台管理子系统。各个模块的介绍如下：

1. 书籍查询：此模块有三个功能分别是根据 ISBN 号查询书籍、根据作者名称查询书籍、根据书名查询书籍。
   1. ISBN 号查询：用户/游客可在每个前台页面最上方的搜索框内输入想要搜索书籍ISBN 号码，系统将展示搜索结果。
   2. 书籍名称搜索：用户/游客可在每个前台页面最上方的搜索框内输入想要搜索书籍名称，系统将展示搜索结果。
   3. 书籍作者搜索：用户/游客可在每个前台页面最上方的搜索框内输入想要搜索书籍名称，系统将展示搜索结果
2. 心愿清单：该模块包含的功能为增加用户想要的书籍、删除用户想要的书籍。

用户可以将自己想要向其他用户索取的书籍添加到心愿清单，也可以从心愿清单中移除不再想要书籍。心愿清单页面显示用户已收入心愿清单的所有书籍，并可以显示自己收藏的书籍封面图片以及书籍摘要，当用户点击某个心愿清单中的书籍封面时，系统跳转到该书籍的详情页面，用户可以在搜索框通过书籍名称和作者名模糊查询出自己想找到的书籍，或者使用 ISBN 号精确查询到想找到的书籍，当用户取消其中一个的收藏书籍时，实

时在收藏夹列表里删除该书籍。

1. 用户信息：包含登录注册、修改个人信息和查看个人信息三个模块。
2. 登录与注册：用户通过自己的账号和注册密码进行登录,登录之后信息显示在网站首页。当系统发现用户的登录中没有注册账号的信息时候,点击登录进行注册,从首页的登录注册界面直接切换至后台的注册信息验证界面,填写正确的注册信息,后台的验证审核通过则表示注册成功,用户的注册信息写完后跳转至用户登录注册界面。
3. 修改个人信息：用户可在个人信息界面根据个人情况修改个人信息并在后台更新在页面同步显示。
4. 查看个人信息：用户可在个人信息界面查看信息，例如用户名、邮箱等。
5. 修改密码：用户可在个人信息页进行修改密码操作。
6. 赠送清单：该模块包含的功能为增加可赠送书籍、删除用户可赠送书籍。

用户可以将自己想要送出的书籍添加到赠送清单，赠送清单页面将会显示赠送清单中书籍的总数量，并且展示所有的书籍。用户可点击每个书籍的封面进入该书籍的详情页面， 查看书籍的具体信息，如：想要该书的人、书籍摘要等。每个书籍展示栏都有撤销按钮， 如果用户想取消赠送该书籍，可以点击该按钮，此书籍将从赠送清单中移除并更新到数据库。

1. 交易信息：该模块包含同意交易与拒绝交易两个主要功能。 1）当收到其他用户的索书申请，用户可以选择同意交易或拒绝交易。
2. 向其他用户发送索书申请，其他用户同意或者拒绝赠送的动作将显示在交易记录当中。
3. 书籍管理：主要业务功能范围包括书籍相关信息内容的新增、删除、修改、查询等功能。
4. 用户信息管理：主要包括用户的新增、删除、查询已经用户信息的修改等功能。
5. 交易信息管理：主要包括交易信息的新增、删除、修改、查询等功能。

# 数据库设计

## 概念结构设计

* + 1. **设计思路**

概念结构设计是整个数据库设计的关键，它通过对用户需求进行综合，归纳与抽象， 形成了一个独立于具体 DBMS 的概念模型。

设计概念结构通常有四类方法：

* + - 1. 自顶向下。即首先明确定义一个具有全局性的概念和有结构的理论框架,再逐步对其进行结构细化。
      2. 自底向上。即首先明确定义各种整体和局部应用的概念及其结构,然后再将他们的概念集成了起来,得到了全局概念的结构。
      3. 逐步扩张。首先明确定义最重要的几个核心基本概念和理论结构,然后向外发展扩张,以一个类似滚雪球的生长速度和发展方式逐步地发展生成其他的重要核心基本概念和理论结构,直至最终总体形成核心概念的基本理论结构。
      4. 混合策略。即自顶向下和自底向上相结合。

本系统开发时采用自顶向下的策略，先整理好整体架构，再逐渐细化每一个子模块的设计。

本系统数据表结构设计如下：

1. 书籍表（id,书名，作者，出版社，价格，页数，出版时间，摘要，封面地址， ISBN）
2. 交易信息表（创建时间，状态，索书者 id，赠书者 id，索书者名称，赠书者名称，地址，信息，书名，书籍作者，封面地址）
3. 礼物表（创建时间，状态，拥有者 id,ISBN，赠送情况）
4. 用户表（创建时间，id，昵称，状态，密码，邮箱，已收到书籍数量，已送出书籍数量，鱼豆数量）
5. 心愿表（创建时间，状态，拥有者 id,ISBN，赠送情况）
   * 1. **E-R 图**

系统总 E-R 图如下图 4-1 所示：

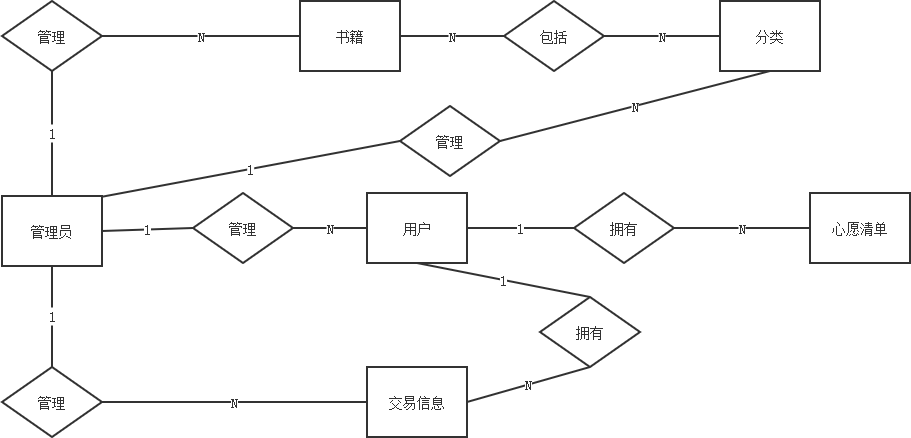


图 4-1 系统总 E-R 图

各部分实体图如下：

1. 书籍实体图如图 4-2： 实体：书籍

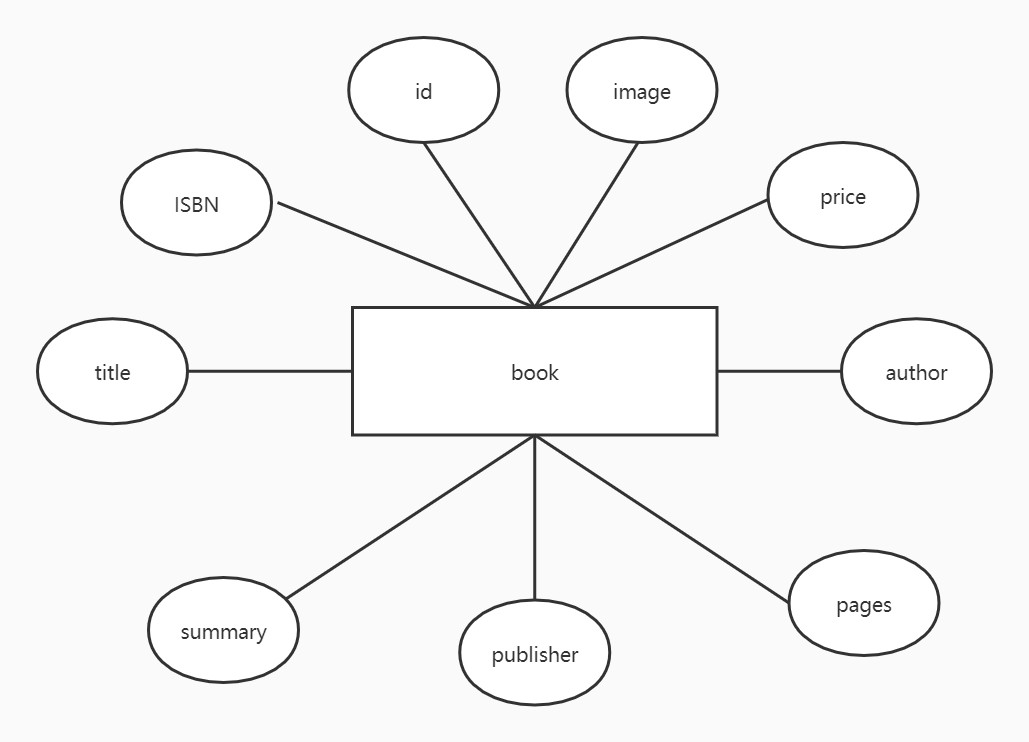
属性：id,书名，作者，出版社，价格，页数，出版时间，摘要，封面地址，ISBN

图 4-2 书籍实体图

1. 交易信息实体图如图 4-3：

实体：书籍

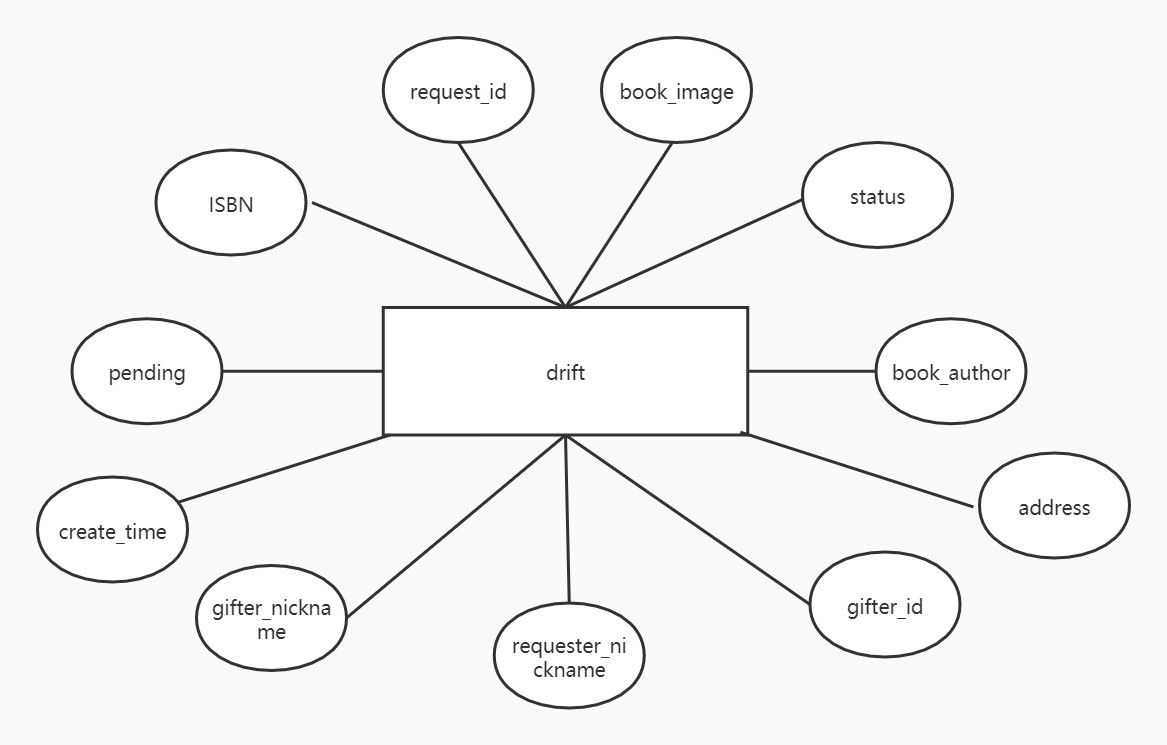
属性：创建时间，状态，索书者 id，赠书者 id，索书者名称，赠书者名称，地址，信息， 书名，书籍作者，封面地址

图 4-3 交易信息实体图

1. 礼物实体图如图 4-4： 实体：书籍

属性：创建时间，状态，礼物 id，拥有者 id,ISBN，赠送情况

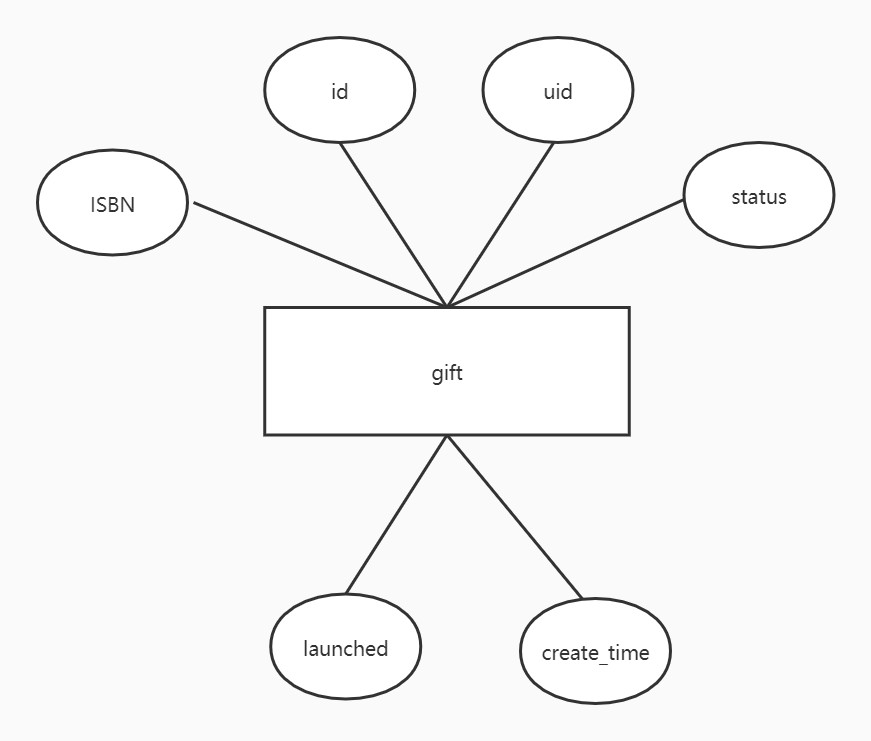


图 4-4 礼物实体图

1. 用户实体图如图 4-5： 实体：书籍

属性：创建时间，id，昵称，状态，密码，邮箱，已收到书籍数量，已送出书籍数量，鱼豆数量

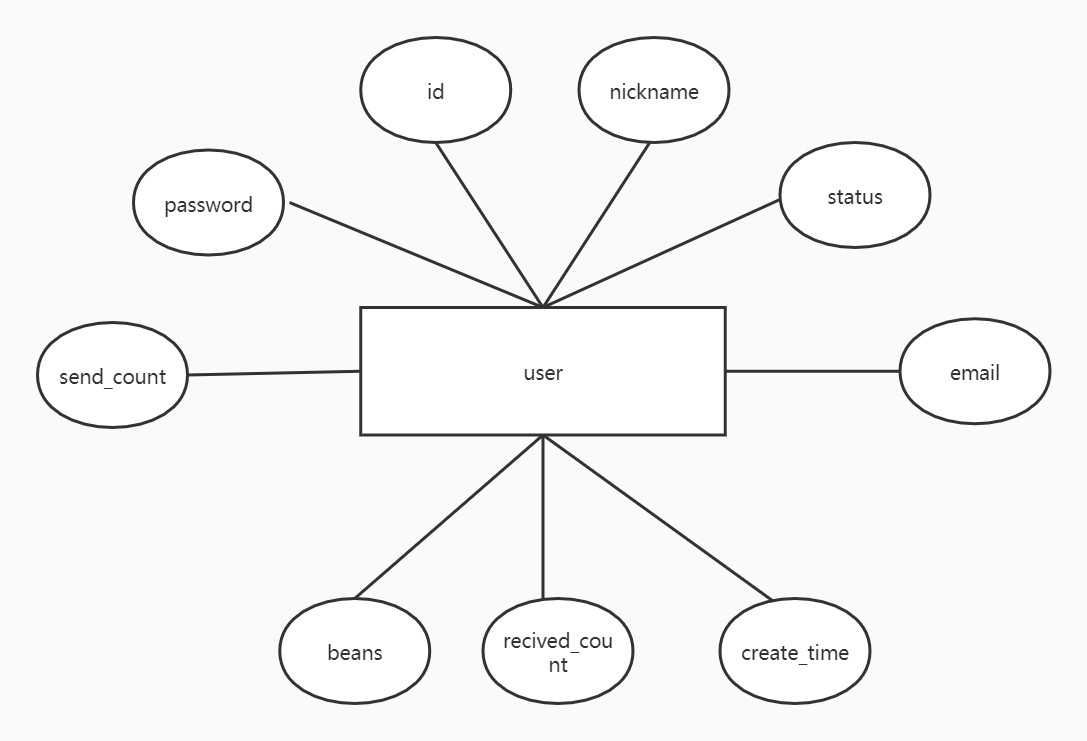


图 4-5 用户实体图

1. 心愿实体图如图 4-6： 实体：书籍

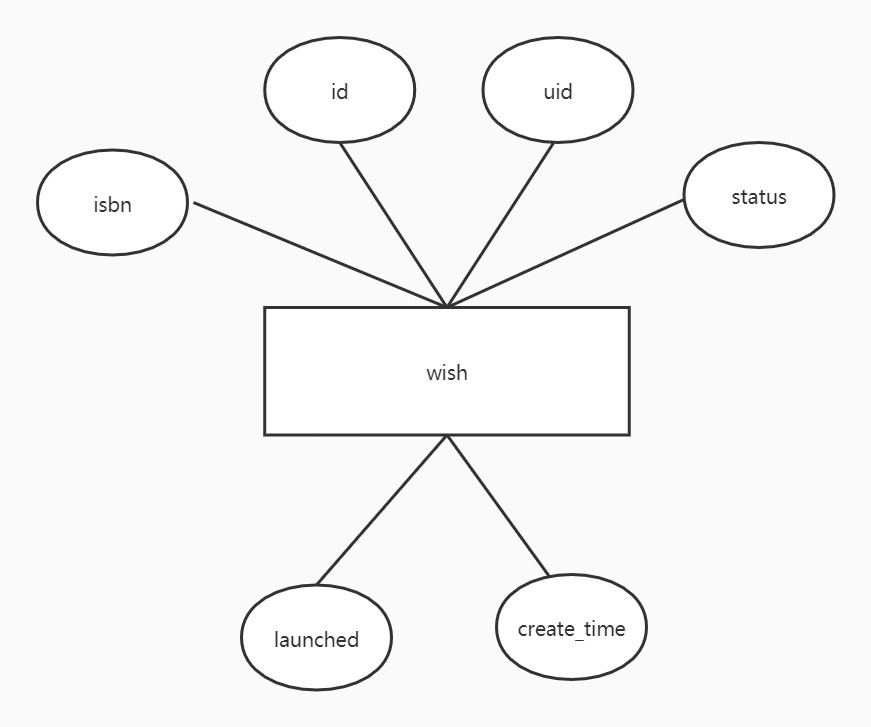
属性：创建时间，状态，拥有者 id,ISBN，赠送情况，心愿 id

图 4-6 心愿实体图

## 逻辑结构设计

* + 1. **设计思路**

逻辑结构设计主要流程就是将概念结构转换为某 DBMS 支持的数据模型，并且将其数据模型结构进行一些合理的优化。E-R 图的作用在这一阶段尤其重要。

E-R 图之间的冲突大致可分为三类：属性冲突，命名冲突，和结构冲突。E-R 图向关系模型的转换，如何将实体性和实体间的联系转换为关系模式、确定这些关系模式的属性和码是一个不可避免的技术型问题。

* + 1. **逻辑模型**

1. 书籍表如表 4-1 所示：

表 4-1 书籍表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 字段  长度 | 是否  主键 | 说明 |
| 1 | Id | 整型 | 11 | 是 | 书籍编号 |
| 2 | Title | 字符串 | 255 | 否 | 书籍名称 |

续表 4-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3 | Author | 字符串 | 255 | 否 | 书籍作者 |
| 4 | Publisher | 字符串 | 255 | 否 | 出版商 |
| 5 | Price | 浮点型 |  | 否 | 书籍定价 |
| 6 | Pages | 整形 |  | 否 | 书籍页数 |
| 7 | Pubdate | 日期 |  | 否 | 出版日期 |
| 8 | ISBN | 字符串 | 255 | 否 | 书籍 ISBN 编号 |
| 9 | Summary | 字符串 | 255 | 否 | 摘要 |
| 10 | Image | 字符串 |  | 否 | 书籍封面图地址 |

1. 交易信息表如表 4-2 所示：

表 4-2 交易信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 字段  长度 | 是否  主键 | 说明 |
| 1 | Id | 整型 | 11 | 是 | 交易信息 id |
| 2 | CreateTime | 日期 |  | 否 | 交易创建日期 |
| 3 | RecipientName | 字符串 | 255 | 否 | 收书者姓名 |
| 4 | Address | 字符串 | 255 | 否 | 收书地址 |
| 5 | Message | 字符串 | 255 | 否 | 索书者请求时附带信息 |
| 6 | Mobile | 字符串 | 255 | 否 | 电话号码 |
| 7 | ISBN | 字符串 | 255 | 否 | 书籍 ISBN 编号 |
| 8 | BookTitle | 字符串 | 255 | 否 | 书籍名称 |
| 9 | BookAuthor | 字符串 | 255 | 否 | 书籍作者 |
| 10 | BookImage | 字符串 | 255 | 否 | 书籍封面图地址 |
| 11 | RequesterId | 整形 | 11 | 否 | 索书者 id |
| 12 | RequersterNickn  ame | 字符串 | 255 | 否 | 索书者昵称 |
| 13 | GifterId | 整形 | 11 | 否 | 赠书者 id |
| 14 | GifterNickname | 字符串 | 255 | 否 | 赠书者昵称 |
| 15 | Pending | 整形 | 11 | 否 | 描述交易状态 |

（4）礼物表如表 4-3 所示：

表 4-3 礼物表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 字段  长度 | 是否  主键 | 说明 |
| 1 | Id | 整型 | 11 | 是 | 礼物编号 |
| 2 | Uid | 整形 | 11 | 否 | 礼物拥有者 id |
| 3 | Status | 整形 | 11 | 否 | 描述该段书籍是否被删除 |
| 4 | ISBN | 字符串 | 255 | 否 | 书籍 ISBN |
| 5 | CreateTime | 日期 |  | 否 | 礼物创建时间 |
| 6 | Launched | 整形 | 11 | 否 | 描述礼物是否被赠出 |

1. 心愿表如表 4-4 所示：

表 4-4 心愿表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 字段  长度 | 是否  主键 | 说明 |
| 1 | Id | 整型 | 11 | 是 | 心愿编号 |
| 2 | Uid | 整形 | 11 | 否 | 心愿拥有者 id |
| 3 | Status | 整形 | 11 | 否 | 描述该段书籍是否被删除 |
| 4 | ISBN | 字符串 | 255 | 否 | 书籍 ISBN |
| 5 | CreateTime | 日期 |  | 否 | 心愿创建时间 |
| 6 | Launched | 整形 | 11 | 否 | 描述心愿中的书籍是否已获得 |

1. 用户表如表 4-5 所示：

表 4-5 用户表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 字段  长度 | 是否  主键 | 说明 |
| 1 | Id | 整型 | 11 | 是 | 用户编号 |
| 2 | Nickname | 字符串 | 255 | 否 | 用户昵称 |
| 3 | Status | 整形 | 11 | 否 | 描述用户是否是被管理员删除 |
| 4 | PhoneNumber | 字符串 | 255 | 否 | 用户联系电话 |
| 5 | Password | 浮点型 |  | 否 | 登录密码 |
| 6 | Email | 字符串 |  | 否 | 登录邮箱 |
| 7 | CreateTime | 日期 |  | 否 | 用户注册时间 |
| 8 | Beans | 浮点型 | 255 | 否 | 鱼豆数量 |
| 9 | SendCounter | 整形 | 11 | 否 | 已赠书数量 |

续表 2-9

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 10 | ReceiveCounter | 整形 | 11 | 否 | 已收到书籍数量 |

## 物理结构设计

* + 1. **存取方式**

“鱼书”平台采用 MySQL 数据库作为数据的存储，采用 B+树索引存取方法。为了满足数据库实体完整性，本平台为数据库的每一张表都建立起了一个主键。在用户表中，为用户表中的用户名建立唯一索引，用于登录时使用；在书籍表中，为每一本书籍都设置 ISBN 号作为唯一索引。

* + 1. **存储结构**

设计数据库的物理结构的核心是确定数据的存放位置以及存储结构，主要包括确定索引、关系、聚簇、备份、日志等的存储安排和存储结构，以及确定系统配置。确定数据的存放位置和存储结构需要综合考虑存储空间、存取时间、和维护代价三方面的因素。这三个方面往往不能兼得，所有设计一个好的物理结构不是那么容易。本系统优先考虑存储空间来进行设计。

# 界面设计

## 界面关系图或工作流图

界面关系图如图 5-1：

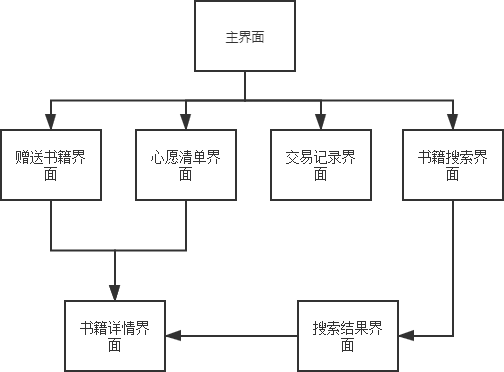


图 5-1 界面关系图

## 界面设计成果

* + 1. **主界面**

项目主页如图 5-2 所示：

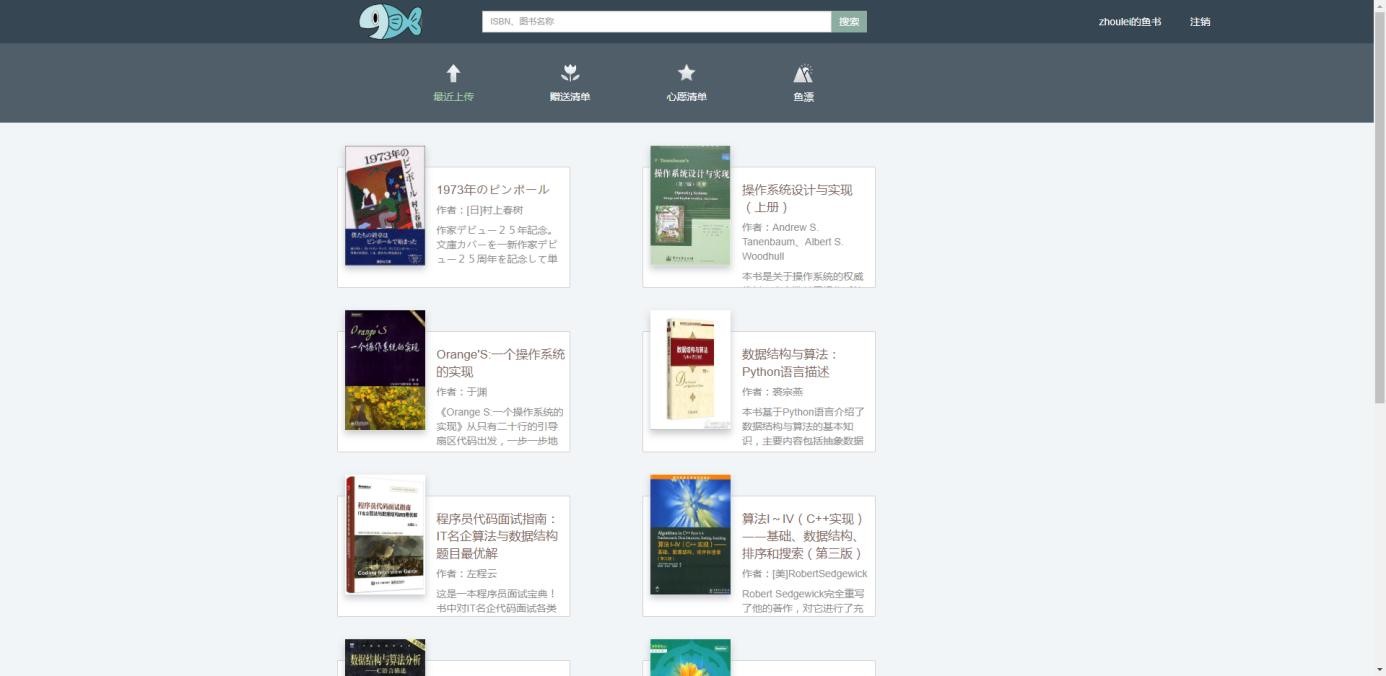


图 5-2 项目主页

* + 1. **子界面 1**

书籍详情页面如图 5-3 所示：

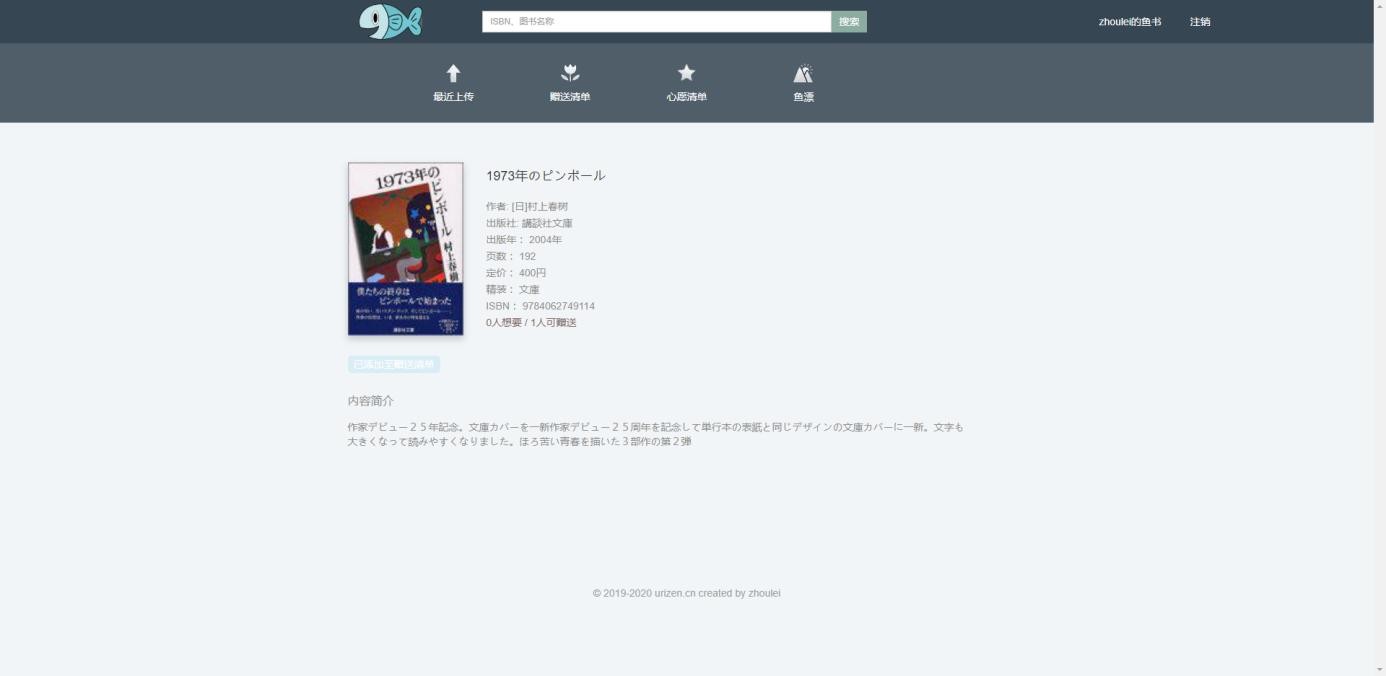


图 5-3 书籍详情页

* + 1. **子界面 2**

赠送清单页面如图 5-4 所示：

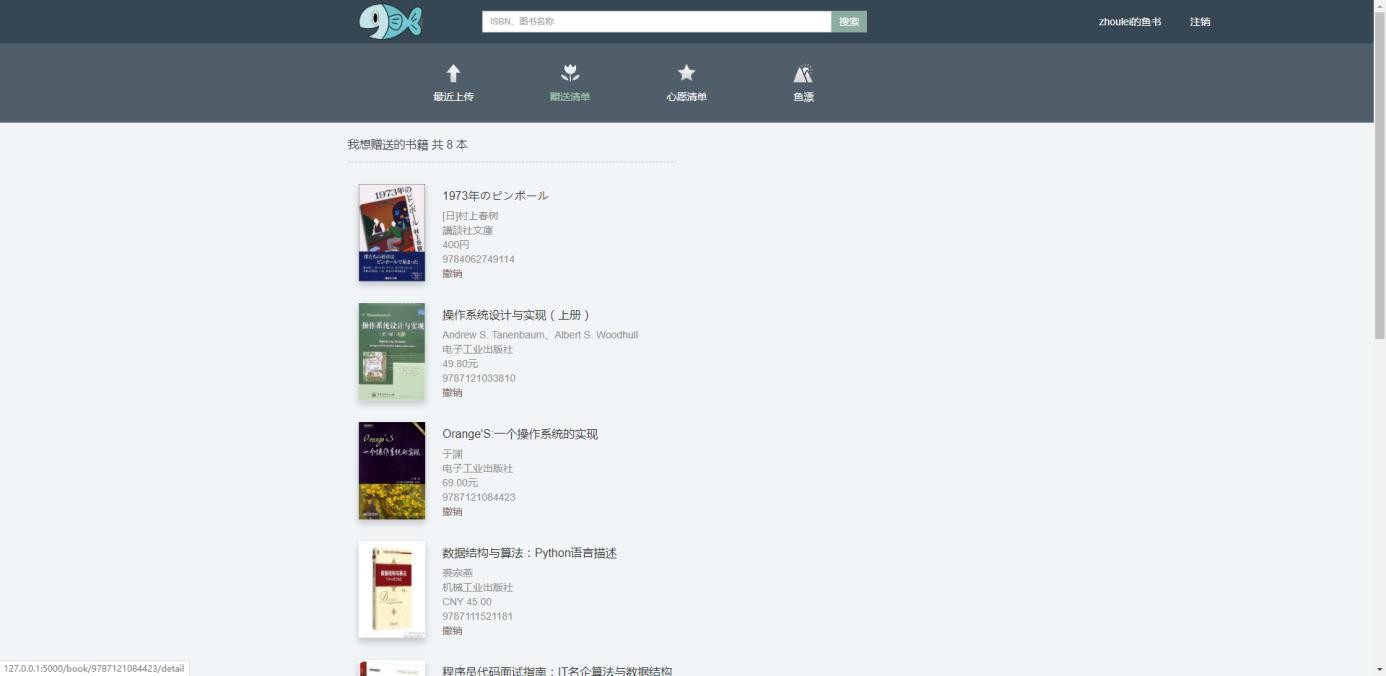


图 5-4 赠送清单页

* + 1. **子界面 3**

心愿清单页面如图 5-5 所示：

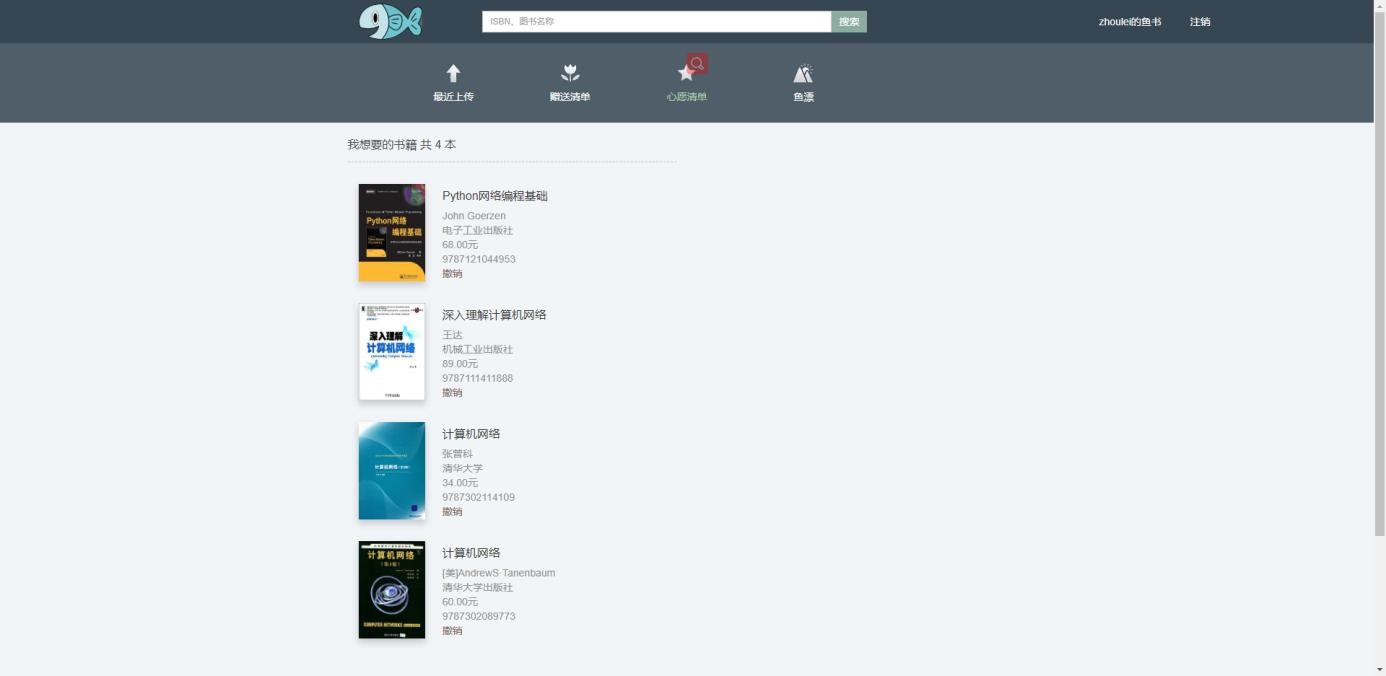


图 5-5 心愿清单页

* + 1. **子界面 4**

交易记录页面如图 5-6 所示：

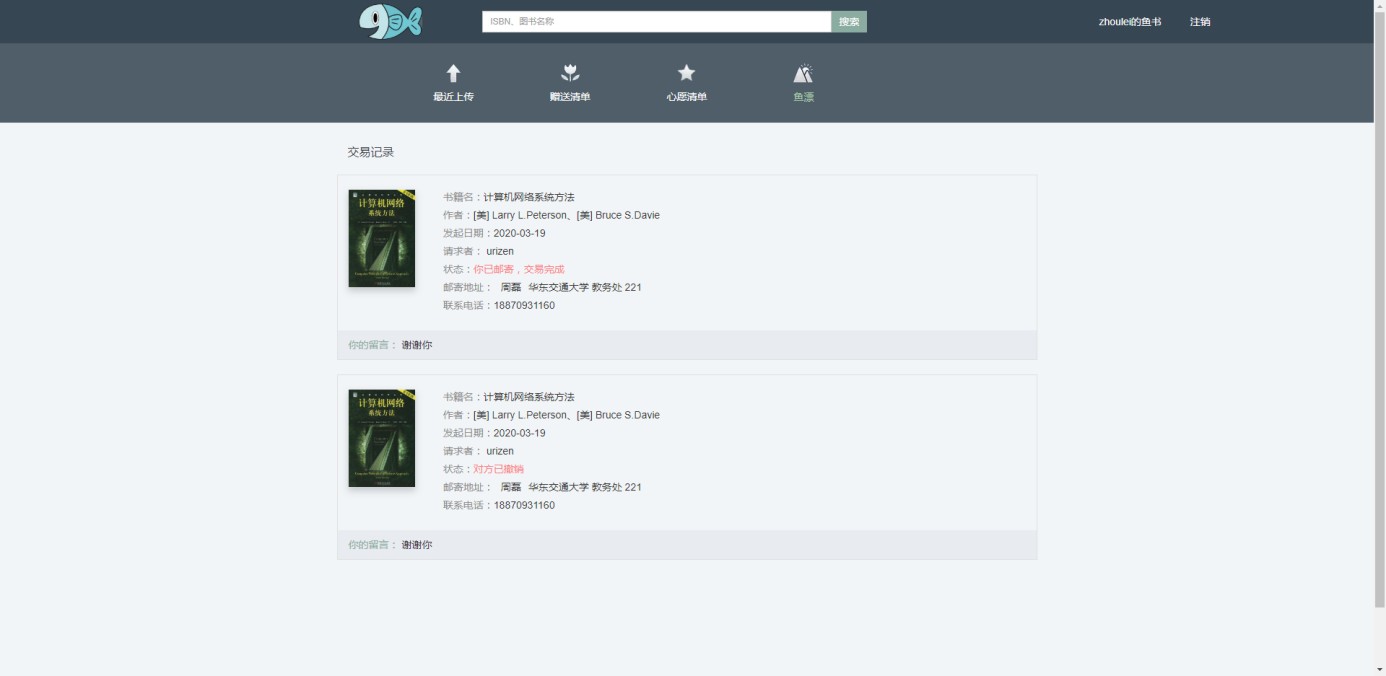


图 5-6 交易记录页

* + 1. **子界面 5**

个人信息页面如图 5-7 所示：



图 5-7 个人信息页

* + 1. **子界面 6**

重置密码页面如图 5-8 所示：



图 5-8 重置密码页

* + 1. **子界面 7**

用户登录页面如图 5-9 所示：

图 5-9 用户登录页

* + 1. **详细设计**
  1. **系统主要功能模块介绍**

子系统关系图如图 6-1 所示：

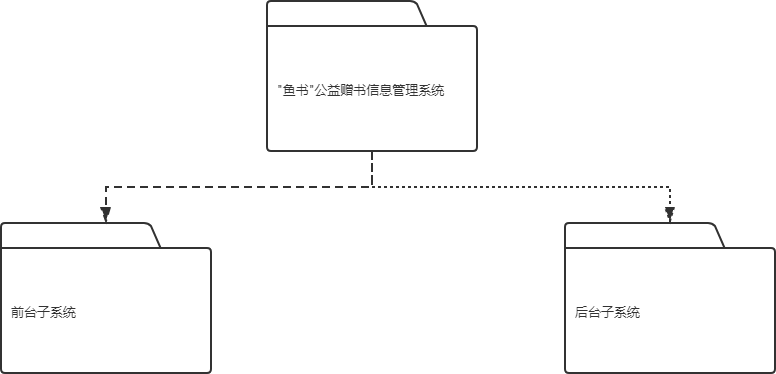


图 6-1 子系统关系图

前台子系统共包括心愿清单、用户信息、交易信息、书籍查询、赠送清单五个模块。

* + 1. 心愿清单：查看心愿清单、删除心愿清单中的书籍、新增心愿清单中的书籍。
    2. 用户信息：用户基本信息修改、查看等。
    3. 交易信息:满足查看交易信息、删除交易信息等操作。
    4. 书籍查询：满足书名查询、ISBN 号码查询、书籍作者查询。
    5. 赠送清单：包含查看赠送清单、删除赠送清单中的书籍、新增可赠送书籍等操作。后台子系统共包括：书籍管理、交易信息管理、用户信息信息管理三个模块。

1. 书籍管理：包含查找书籍、删除书籍、新增书籍、修改书籍信息等操作。
2. 用户账户信息管理：包含修改用户各项信息、新增或删除用户等操作。
3. 交易信息管理：包含查看交易信息、删除交易信息、修改交易信息、新增交易信息等操作。
   1. **功能模块设计**
      1. 心愿清单
4. 功能结构设计

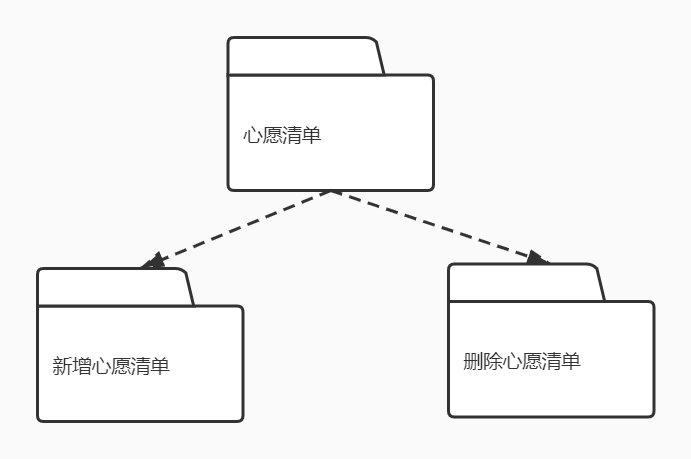
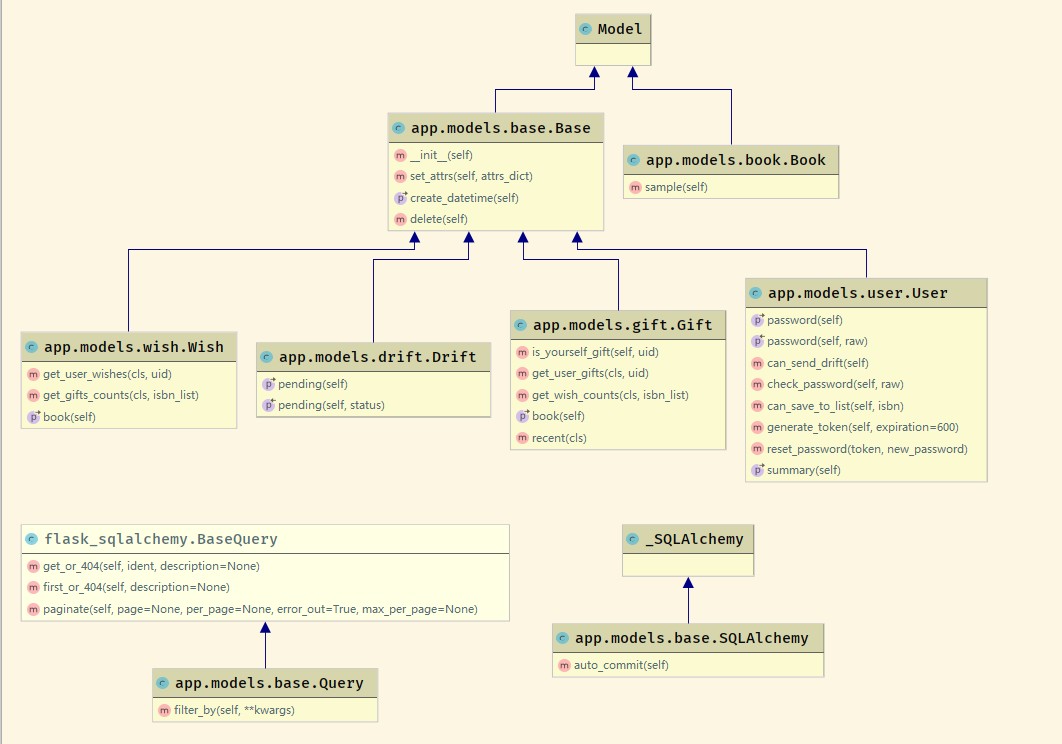
心愿清单：该模块包含的功能为增加用户想要的书籍、删除用户想要的书籍。用户可以将自己心仪的书籍添加到心愿清单，也可以查看已收藏到心愿清单中的书籍，也可以取消心愿从心愿清单中移除某书籍。心愿清单页面显示用户加入收藏的所有书籍，并且这些书籍的简要信息，当用户点击某个收藏的书籍封面图时，系统跳转到该物品的详情页，用户可以在收藏夹页面通过书籍名称模糊查询出自己想找到的书籍，或者使用 ISBN 号精确查询到想找到的书籍，当用户取消其中一个的收藏书籍时，实时在收藏夹列表里删除该书籍。心愿清单模块关系图如图 6-2 所示：

图 6-2 心愿清单模块关系图

1. 类图设计

心愿清单类图包含在下面的类图 6-3 中：



Base 类描述如表 6-1 所示:

图 6-3 心愿清单类图表 6-1 Base 类描述

Gift 类描述如表 6-2 所示:

|  |  |  |
| --- | --- | --- |
| Base 类 | | |
| 类的描述：所有模型类的基类，负责实现一些类的共有方法 | | |
| 操作描述 | init (self) | 初始化类的基本属性 |
| set\_attrs(self) | 给类的特殊属性赋值 |
| create\_datatime(self) | 生成心愿创建时间 |
| delete() | 删除记录 |

表 6-2 Gift 类描述

|  |
| --- |
| 礼物类 |
| 类的描述：购物车管理主功能类 |

续表 6-2

|  |  |  |
| --- | --- | --- |
| 操作描  述 | get\_user\_wish(cls,uid) | 得到用户的所有心愿 |
| get\_gift\_count(cls,isbn\_list) | 得到用户心愿总数量 |
| book(self) | 得到书籍列表的首本书 |

1. 顺序图设计

心愿清单模块顺序图如图 6-4 所示：

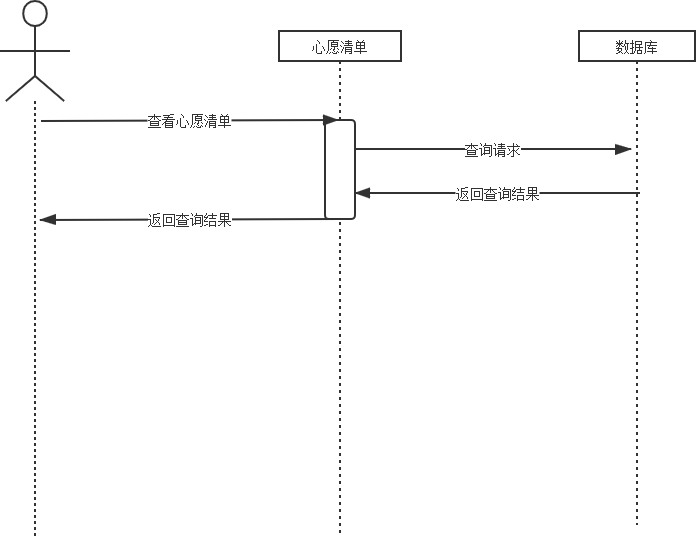


图 6-4 心愿清单模块顺序图

1. 核心处理流程设计

心愿清单模块核心处理流程图如图 6-5：

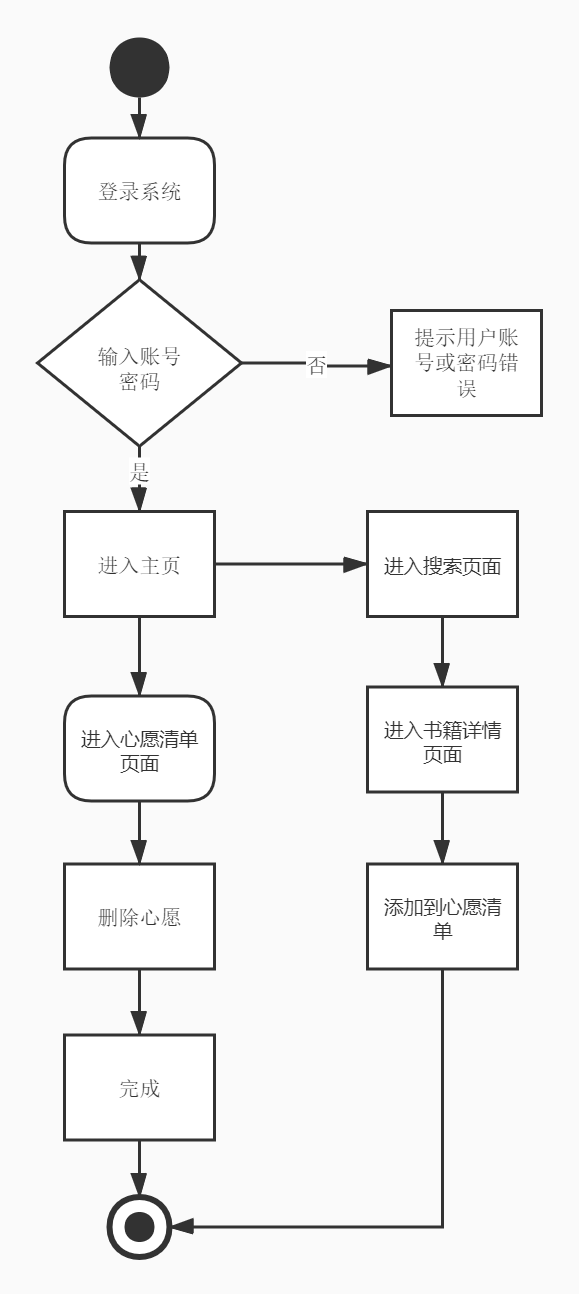


图 6-5 心愿清单核心处理流程图

* + 1. 赠送清单

1. 功能结构设计

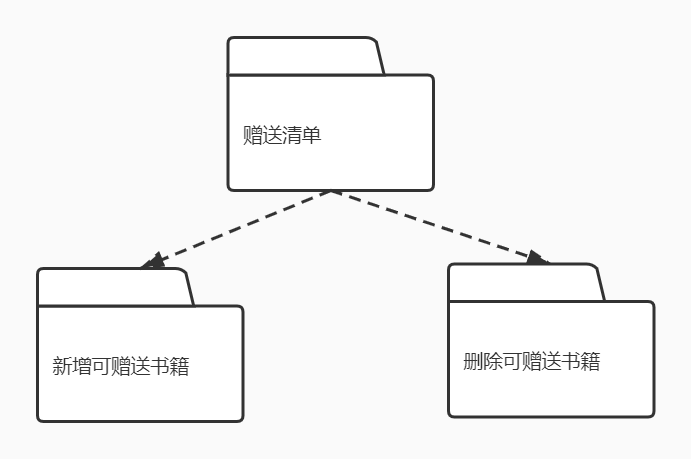
赠送清单：该模块包含的功能为添加用户自己的可赠送书籍、删除用户的可赠送书籍。用户可以将自己闲置的书籍添加到赠送清单，用户可以前台赠送清单页面看到自己的可赠送书籍列表以及赠书总数，也可以在该页面选择撤销某本书籍以让该书籍从赠送清单中删除。该页面能较为简略的显示每一本的的简短信息，用户也能通过图书封面指向的链接进入该书的书籍详情页面，更为详细地了解该书籍，并且可以在该页面查看到有哪些用户想要该本书籍。赠送清单模块关系图如图 6-6 所示：

图 6-6 赠送清单模块关系图

1. 类图设计

心愿清单类图包含在上面的类图 6-3 中。

Base 被描述表如表 6-4 所示：

表 6-4 base 类描述

|  |  |  |
| --- | --- | --- |
| Base 类 | | |
| 类的描述：所有模型类的基类，负责实现一些类的共有方法 | | |
| 操作描述 | init (self) | 初始化类的基本属性 |
| set\_attrs(self) | 给类的特殊属性赋值 |
| create\_datatime(self) | 生成心愿创建时间 |
| delete() | 删除记录 |

Wish 类描述表如表 6-5 所示：

表 6-5 wish 类描述

|  |  |  |
| --- | --- | --- |
| 心愿类 | | |
| 类的描述：购物车管理主功能类 | | |
| 操作描述 | get\_user\_gift(cls,uid) | 得到用户的所有礼物 |
| get\_gift\_count(cls,isbn\_list) | 得到用户心愿总数量 |
| book(self) | 得到书籍列表的首本书 |
| recent(cls) | 得到整个系统最近上传的书籍 |
| is\_yours\_gift(self,uid) | 判断某本书是否在某个用户的赠送清单中 |

1. 顺序图设计

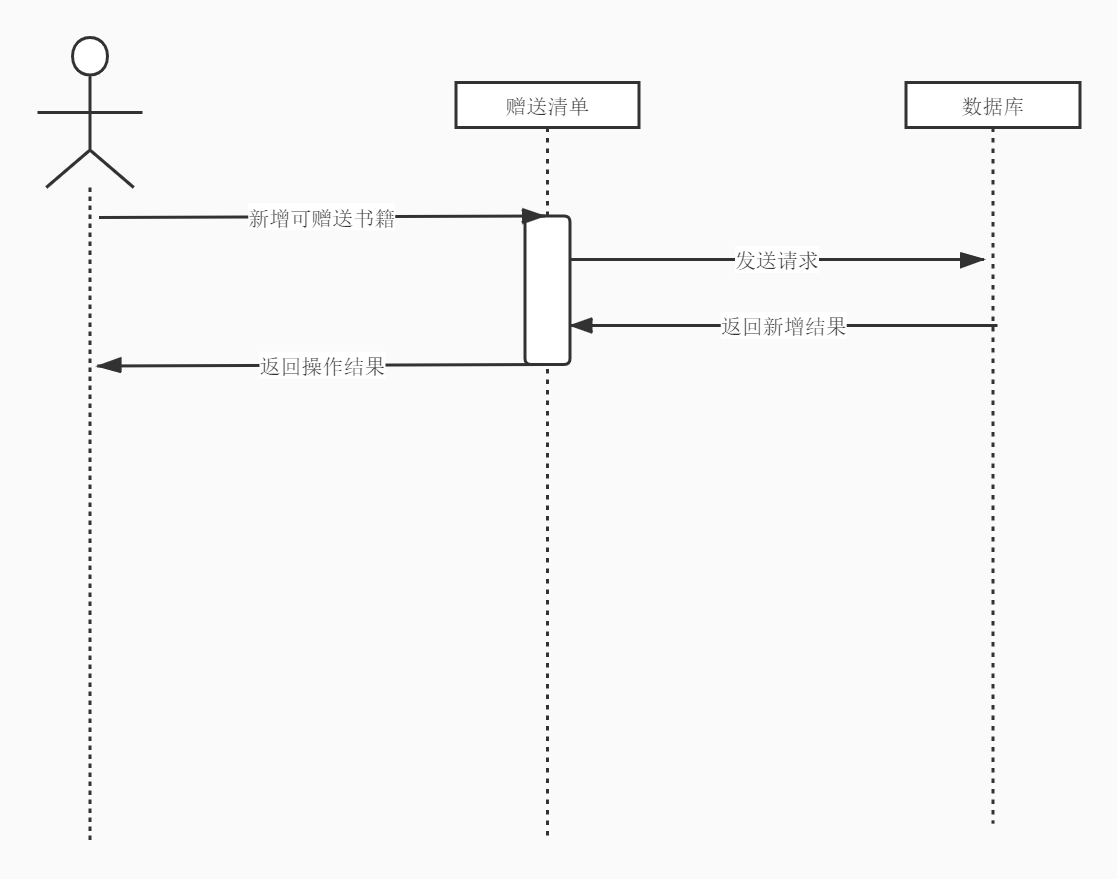
赠送清单模块顺序图如图 6-7：

图 6-7 赠送清单模块顺序图

1. 核心处理流程设计

赠送清单核心处理流程图如图 6-8：

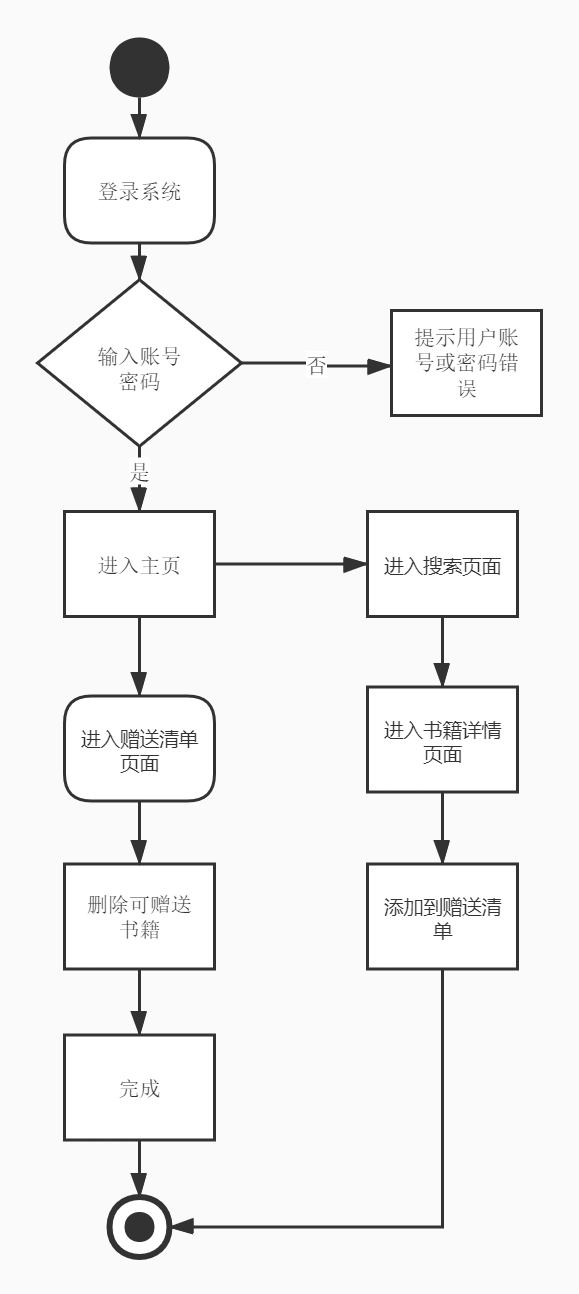


图 6-8 赠送清单核心处理流程图

* + 1. 书籍查询

1. 功能结构设计

书籍查询：该模块包含的功能为通过 ISBN 号查询书籍、通过作者名称查询书籍、通过书名查询书籍。书籍查询模块关系图如图 6-9 所示：

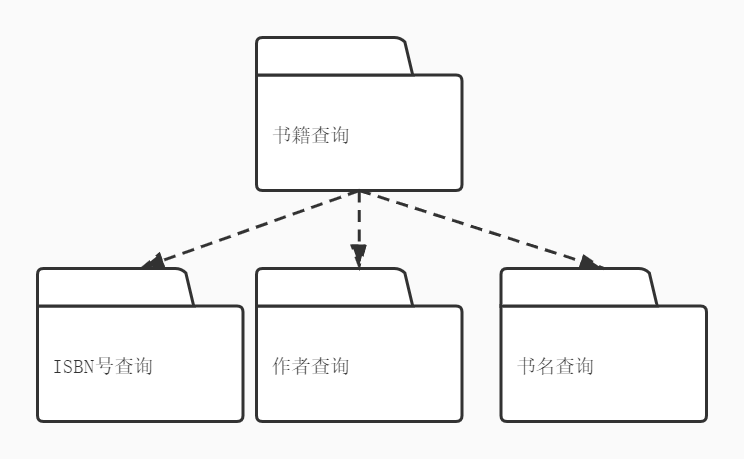


图 6-9 书籍查询模块关系图

1. 类图设计

书籍查询类图包含在下面的类图 6-10 中：

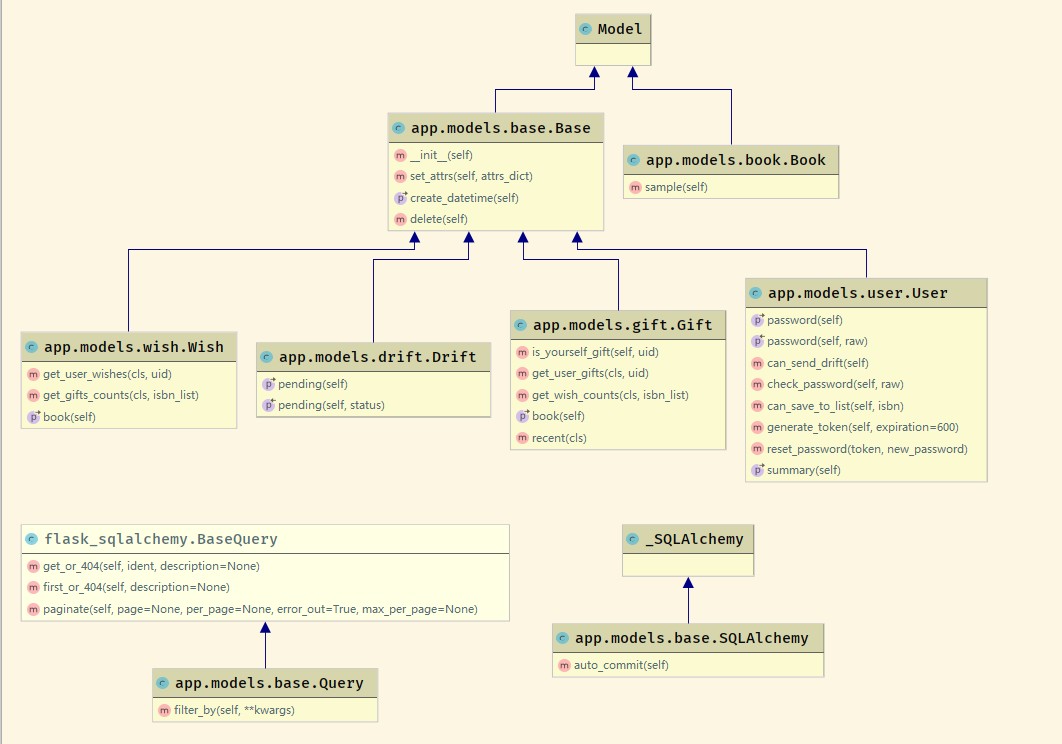


图 6-10 书籍查询类图

Base 类描述如表 6-6 所示:

表 6-6 Base 类描述

|  |  |  |
| --- | --- | --- |
| Base 类 | | |
| 类的描述：所有模型类的基类，负责实现一些类的共有方法 | | |
| 操作描述 | init (self) | 初始化类的基本属性 |
| set\_attrs(self) | 给类的特殊属性赋值 |
| create\_datatime(self) | 生成心愿创建时间 |
| delete() | 删除记录 |

Book 类描述如表 6-7 所示:

表 6-7 书籍类描述

|  |  |  |
| --- | --- | --- |
| 书籍类 | | |
| 类的描述：书籍类 | | |
| 操作描  述 | Sample（self) | 从 API 返回结果中得到书籍数据 |

1. 顺序图设计

心愿清单模块顺序图如图 6-11 所示：

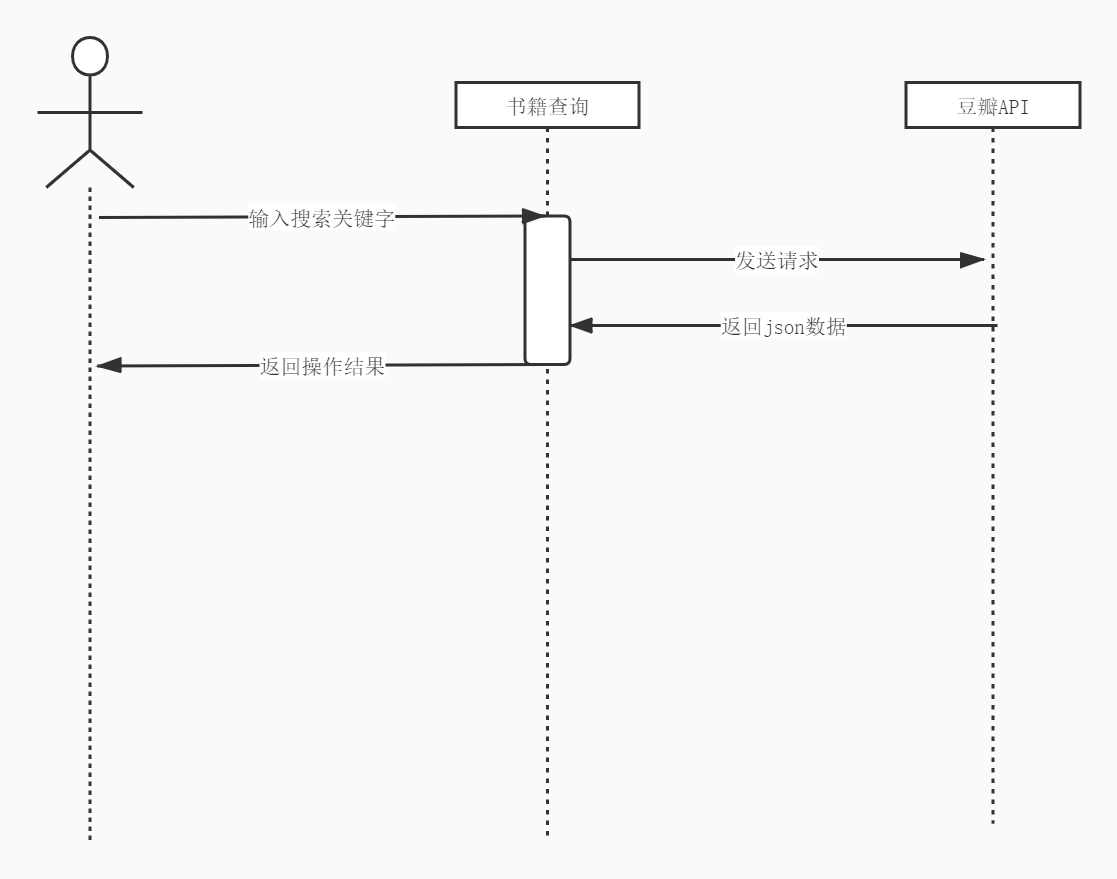


图 6-11 书籍查询模块顺序图

1. 核心处理流程设计

心愿清单模块核心处理流程图如图 6-12：

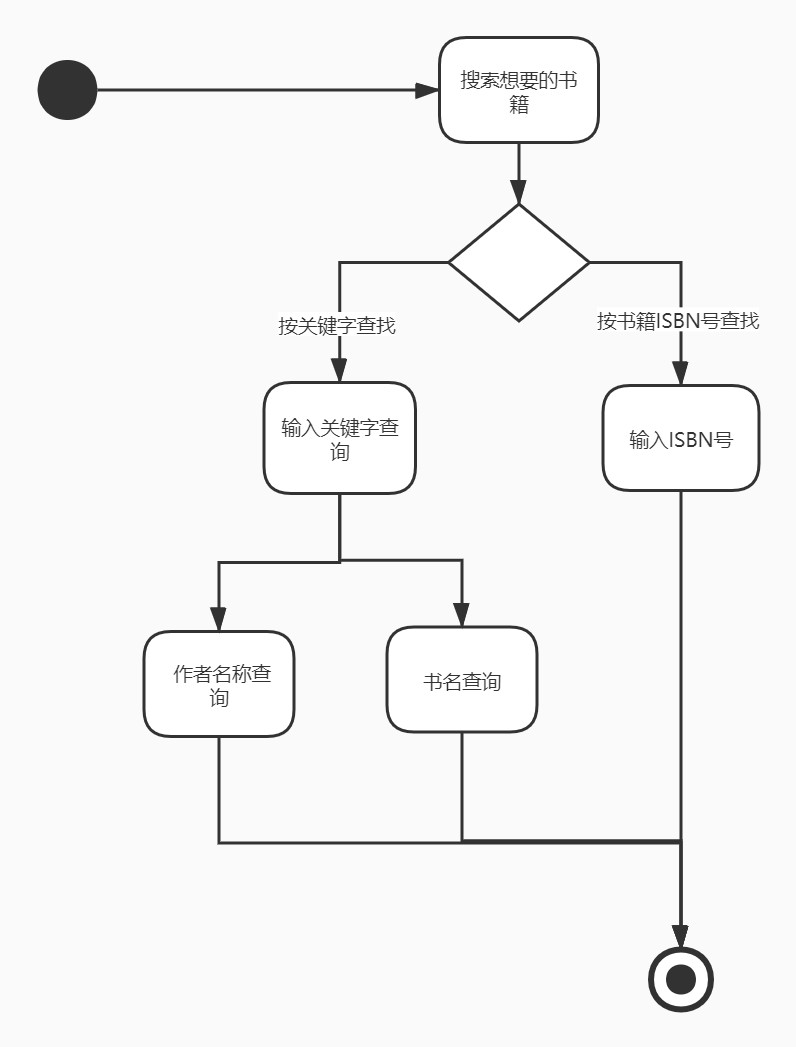


图 6-12 书籍查询核心处理流程图

* + 1. 书籍管理

1. 功能结构设计

书籍管理：该模块包含的功能为对数据库中的书籍数据进行新增、修改、删除、查找。书籍查询模块关系图如图 6-13 所示：

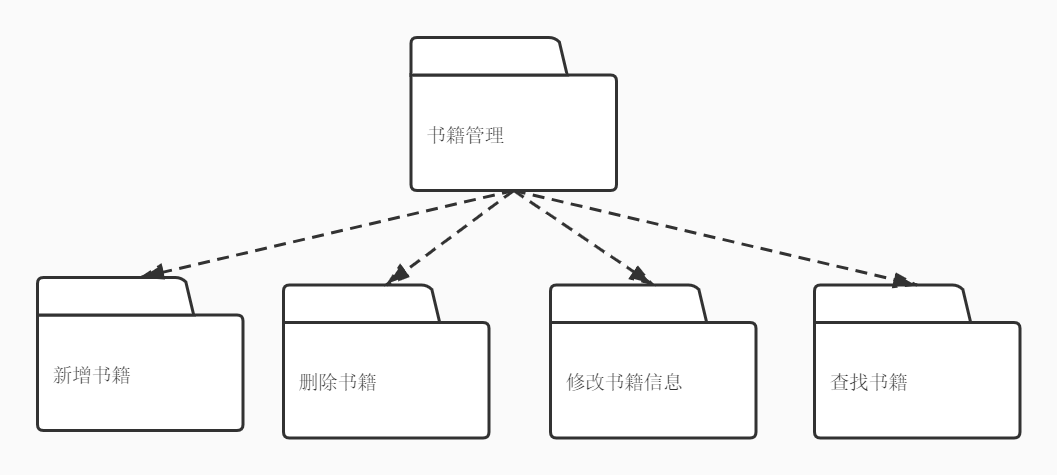


图 6-13 书籍查询模块关系图

1. 类图设计

书籍管理类图包含在下面的类图 6-10 中：

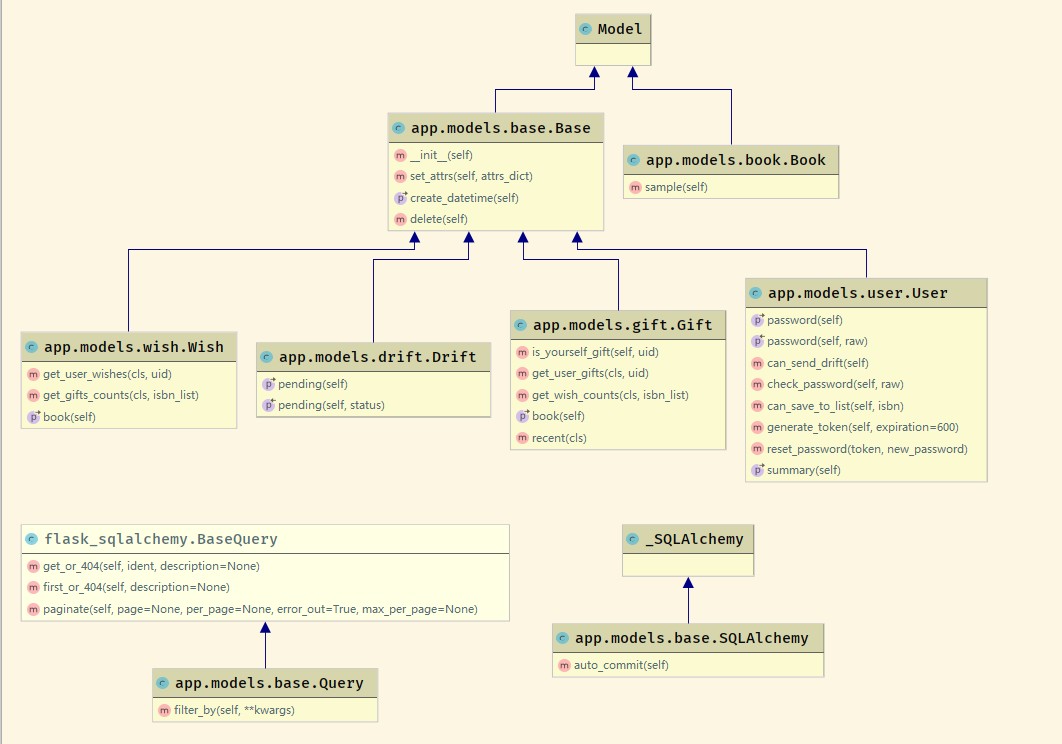


图 6-14 书籍管理类图

Book 类描述如表 6-8 所示:

表 6-8 书籍类描述

|  |  |  |
| --- | --- | --- |
| 书籍类 | | |
| 类的描述：书籍类 | | |
| 操作描  述 | Sample（self) | 从 API 返回结果中得到书籍数据 |

1. 顺序图设计

书籍管理模块顺序图如图 6-11 所示：

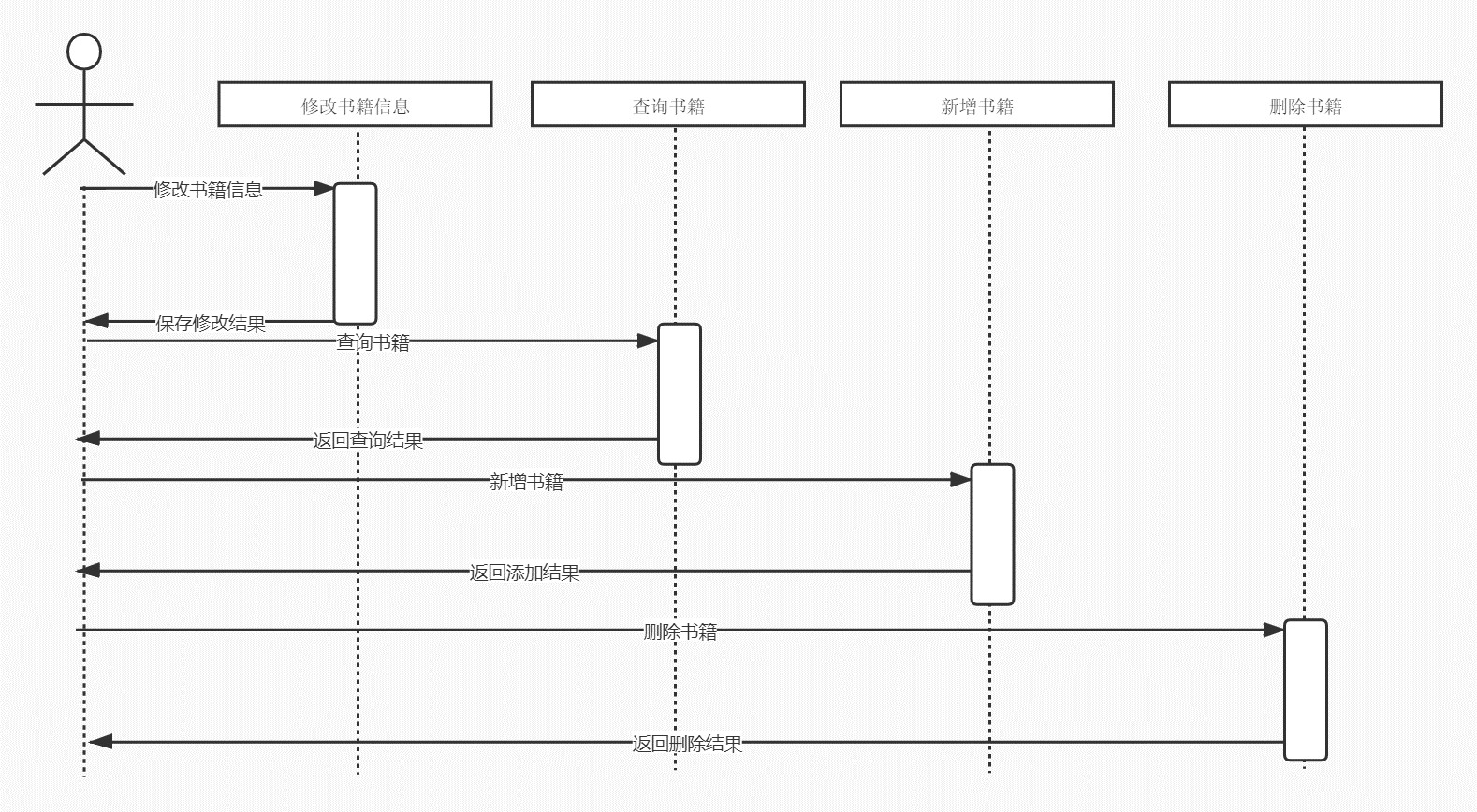


图 6-11 书籍查询模块顺序图

6.2.4.4 核心处理流程设计

书籍管理模块核心处理流程图如图 6-12：

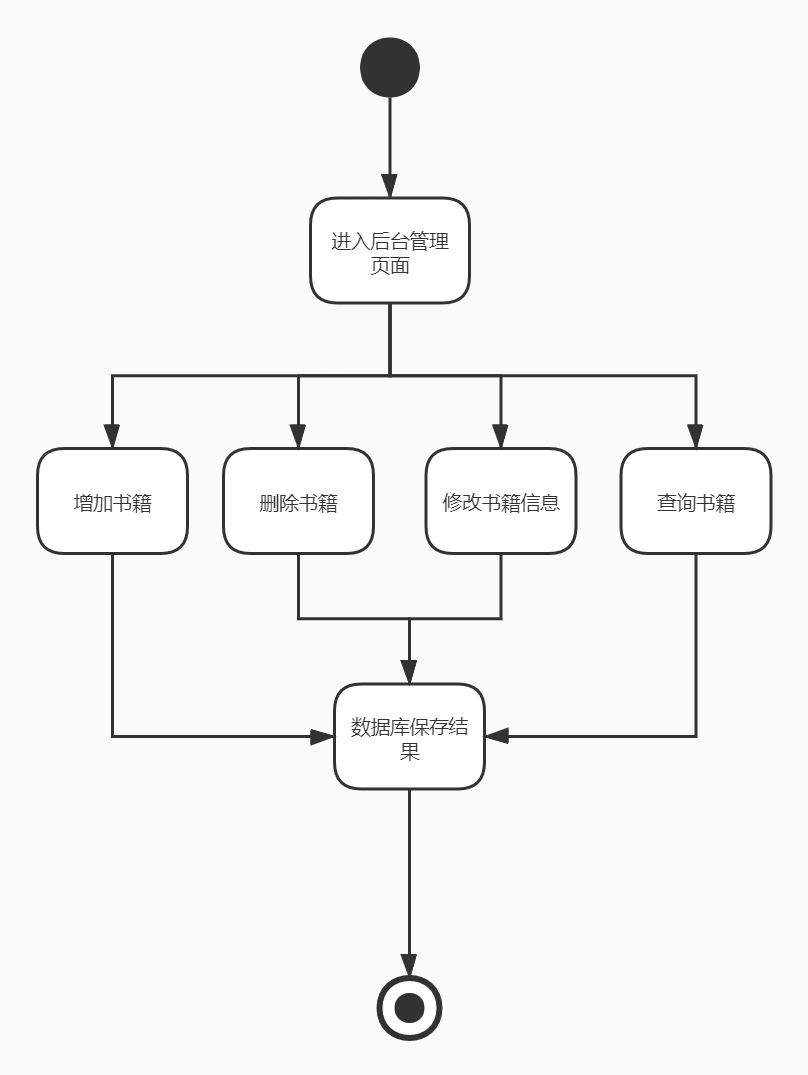


图 6-12 书籍管理核心处理流程图

* + 1. **编码**
  1. **代码实现与核心算法**
     1. **关键代码实现**

项目初始化：

login\_manager = LoginManager() mail = Mail()

admin = Admin(name="Fishbook 后台管理系统")

def create\_app():

app = Flask( name ) app.config.from\_object('app.secure') app.config.from\_object('app.setting') register\_blueprint(app)

db.init\_app(app) login\_manager.init\_app(app) admin.init\_app(app) login\_manager.login\_view = 'web.login'

login\_manager.login\_message = '请先登录或注册' register\_admin(admin)

mail.init\_app(app)

with app.app\_context(): db.create\_all()

return app

def register\_blueprint(app):

from app.web.book import web app.register\_blueprint(web)

def register\_admin(admin):

from flask\_admin.contrib.sqla import ModelView from app.models.user import User

from app.models.drift import Drift from app.models.gift import Gift from app.models.wish import Wish

# 在这里初始化Flask Flask-SQLAlchemy Flask-Admin

admin.add\_view(ModelView(User, db.session,"用户")) admin.add\_view(ModelView(Book, db.session,"书籍")) admin.add\_view(ModelView(Gift, db.session,"礼物")) admin.add\_view(ModelView(Drift, db.session,"赠书")) admin.add\_view(ModelView(Wish, db.session,"心愿清单"))

* + 1. **核心算法部分**

本项目涉及爬虫部分的关键算法：深度优先和广度优先算法：代码如下：

class Gragh():

def init (self,nodes,sides): self.sequense = {} self.side=[]

for node in nodes: for side in sides:

u,v=side

if node ==u:

self.side.append(v) elif node == v:

self.side.append(u) self.sequense[node] = self.side self.side=[]

#print self.sequense

深度优先算法，是一种用于遍历或搜索树或图的算法，也是爬虫技术中最为常见的算法之一。以深度优先的方式遍历数的每一个的节点，即若节点存在子节点则优先继续往下遍历子节点。当节点的的遍历已经到底，深度优先搜索将回溯到此节点的那条边的起始节点。

def DFS(self,node0):

#queue 本质上是堆栈，用来存放需要进行遍历的数据#order 里面存放的是具体的访问路径queue,order=[],[]

#首先将初始遍历的节点放到queue 中，表示将要从这个点开始遍历

queue.append(node0) while queue:

#从 queue 中 pop 出点 v，然后从 v 点开始遍历了，所以可以将这个点pop

出，然后将其放入order 中

#这里才是最有用的地方，pop（）表示弹出栈顶，由于下面的 for 循环不断的访问子节点，并将子节点压入堆栈，

#也就保证了每次的栈顶弹出的顺序是下面的节点v = queue.pop()

order.append(v)

#这里开始遍历v 的子节点

for w in self.sequense[v]:

#w 既不属于 queue 也不属于 order，意味着这个点没被访问过，所以讲起放到 queue 中，然后后续进行访问

if w not in order and w not in queue: queue.append(w)

return order

广度优先搜索 BFS 是从根节点开始，以一层一层的方式遍历数或者图每一层次，一般借助队列进行。如果某一层次所有节点均被访问，则广度优先算法继续遍历下一层次。。

def BFS(self,node0):

#queue 本质上是堆栈，用来存放需要进行遍历的数据

#order 里面存放的是具体的访问路径

queue,order = [],[]

#首先将初始遍历的节点放到queue 中，表示将要从这个点开始遍历

# 由于是广度优先，也就是先访问初始节点的所有的子节点，所以可以

queue.append(node0) order.append(node0)

while queue:

#queue.pop(0)意味着是队列的方式出元素，就是先进先出，而下面的 for 循环将节点v 的所有子节点

#放到 queue 中，所以 queue.pop(0)就实现了每次访问都是先将元素的子节点访问完毕，而不是优先叶子节点

v = queue.pop(0)

for w in self.sequense[v]: if w not in order:

# 这里可以直接order.append(w) 因为广度优先就是先访问节点的所有下级子节点，所以可以

# 将 self.sequense[v]的值直接全部先给到order order.append(w)

queue.append(w)

return order

* 1. **代码优化分析**
     1. **Python 循环导入包问题的优化解决**

在 Flask 项目开发过程当中，每个程序员都会面临的一个问题就是包的循环导入问题， 简单来说就是一个文件从导入另一个文件，而另一个文件刚好通过直接或者间接的方式也导入了这个文件，这就是 Flask 项目以及 Python 开发当中经常会出现的经典问题。解决这个问题有很多方案，其中大多数人推荐的是利用 python 不会重复导入第二次包的机制来固定包的引入顺序来解决。但本人觉得这种方法较为笨重，且难以第一时间阅读代码时

def register\_admin(admin):

from flask\_admin.contrib.sqla import ModelView from app.models.user import User

from app.models.drift import Drift from app.models.gift import Gift from app.models.wish import Wish

admin.add\_view(ModelView(User, db.session,"用户")) admin.add\_view(ModelView(Book, db.session,"书籍")) admin.add\_view(ModelView(Gift, db.session,"礼物")) admin.add\_view(ModelView(Drift, db.session,"赠书"))

admin.add\_view(ModelView(Wish, db.session,"心愿清单"))

就能理解，故采用我个人比较喜欢的函数内部导入包的方法。代码如下所示：

* + 1. **邮件发送时空时等待问题的优化**

本项目中多次使用 Flask-mail 实现发送邮件的功能，但是普通的发送邮件可能因为第三方邮件系统的影响导致用户第一时间收不到邮件，此时用户必须等待邮件发送成功才能进行下一步操作，故本项目将邮件模全脱离主线程，实现异步发送邮件，代码如下：

def send\_async\_email(app, msg): with app.app\_context():

try:

mail.send(msg) except Exception as e:

pass

def send\_mail(to, subject, template, \*\*kwargs): msg = Message('[FishBook]' + ' ' + subject,

sender=current\_app.config['MAIL\_USERNAME'], recipients=[to])

msg.html = render\_template(template, \*\*kwargs) # current\_app app = Flask()

app = current\_app.\_get\_current\_object()

thr = Thread(target=send\_async\_email, args=[app, msg]) thr.start()

* + 1. **测试**
  1. **测试方案设计**
     1. **测试策略**

1. 单元测试

我们将对需要测试的软件中的最小可测试单元进行检验称为单元测试。一般来说，要根据实际情况去才能去判定单元测试中单元代表的是什么，例如在 C 语言程序中单元指的是某个函数，而 Java 语言程序中单元指的却是一个类。但是从总体而言，单元其实是被人为所规定的，单元指的是测试模块中的最小的被测功能模块。软件开发过程中需要进行的最低级别的测试活动就是单元测试，被测软件的每一个独立小单元都将在与软件的其他部分完全隔开的情况下进行测试。

1. 黑盒测试

数据驱动功能测试、数据驱动功能测试或基于其需求和规格说明的功能测试都是黑盒测试的简称,黑盒测试是几种常用的程序功能测试方法之一。黑盒功测试指的是在知到用户对软件的需求的基础上,来检测软件是否完全符合用户的要求。黑盒测试顾名思义,就是把产品的程序功能看作一个一个黑盒子,外界无法看见里面的细节，测试人员并不需要知道盒子里面具体是什么，也就是软件的具体实现细节,而是在程序提供的外在接口上进行测试的方法,黑盒测试只会关注检查程序的功能是否按照其用户群体需求以及软件的规格说明书的要求和规定运行。黑盒测试完全不用考虑程序实现其功能的各种方法或者技术, 只关注程序是否能在正确接收到外界传来的数据时能否产生正确的输出信息。黑盒功能测试一般只针对于其程序的外部,不针对程序的内部逻辑结构。黑盒测试是一种主要针对程序的用户界面和外部功能进行的测试。

1. 白盒测试。

按照被测程序内部的结构进行测试的方式称为白盒测试,在一般情况下，白盒测试都是用来分析程序的源代码和内部结构的,测试者由此就可以清楚地看到被测程序的源代码, 并且能够完全了解被测试软件源代码的内部结构。正是因为这样,白盒测试通常也被称为结构驱动测试和逻辑驱动结构测试。人们通过白盒测试来确定和检测一个产品内部的动作程序是否按照产品设计的规格和产品说明书的要求和规定正常地运行,并检验被测程序电路中的每条通路结构是否都完全能按照设计所预定的要求正确地工作。

* + 1. **测试进度安排**

测试进度安排如表 8-1 所示：

表 8-1 测试进度安排表表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试阶段 | 测试任务 | 工作量估计 | 人员分配 | 起止时间 |
| 第一阶段功能测试 | 赠书业务测试登录注册测试索书业务测试注销登录测试  心愿清单模块测试赠书清单模块测试 | 4 日 | 周磊 | 2019 年 4 月 11 日  至 2019 年 4 月 14 日 |
| 第二阶段系统测试 | 完成所有模块的组合测试  确定所有业务流向  和数据都是正确的 | 3 日 | 周磊 | 2019 年 4 月 15 日  至 2019 年 4 月 17 日 |
| 第三阶段  性能测试 | 多用户访问，交替进  行测试 | 1 日 | 周磊 | 2019 年 4 月 18 日 |
| 第四阶段  安装部署测试 | 对与项目部署进行测试 | 2 日 | 周磊 | 2019 年 4 月 19 日  至 2019 年 4 月 20 日 |
| 第五阶段  兼容测试 | 软件的运行情况 | 1 日 | 周磊 | 2019 年 4 月 21 日 |

* + 1. **测试资源**

测试资源表如表 8-2 所示：

表 8-2 测试资源表

|  |  |
| --- | --- |
| 测试环境 | 具体环境 |
| 浏览器环境 | 火狐浏览器，谷歌浏览器，QQ 浏览器，IE 浏览器，360 浏览器 |
| 操作系统 | Windows10、Windows7 |

* + 1. **关键测试点**

1. 兼容性测试

显示在浏览器是否正常。重点测试网站分辨率在各个主流浏览器上是否能正常显示， 分辨率兼容测试是为了验证页面布局、界面显示以及相关网页字符在不同分辨率下显示的具体情况。用户使用的计算机的分辨率多种多样，在测试过程中不太可能全部覆盖到所有的分辨率，而且测试大量数据成本上也是一个很大的挑战，所以一定在说明书中注明系统所支持的最佳分辨率。常用的分辨率为 1024×768、1440×900、1280×800 和 1366×768，

这些可能是用户最可能使用的几种分辨率，系统是一定要支持这些常见分辨率的，其他的分辨率可以尽量去兼容。有时为了降低风险，在启动系统时，系统会对当前的分辨率进行判断，如果当前的分辨率不是最佳分辨率，系统则会将分辨率强制转换为系统所支持的最佳分辨率。

1. 系统功能测试 1）登陆注册测试：验证登录时当用户名或密码错误系统能给与预期的反应，当用户

注册时系统能正确验证用户输入的邮箱是否规范以及是否数据库中有重复邮箱。2）个人信息修改测试：个人信息的修改是否能够正确反应到数据库中。

1. 赠送书籍的上传与删除：验证赠书清单中书籍的上传和删除是否符合预期结果。
2. 心愿清单中书籍的上传与删除：验证心愿清单中书籍的上传和删除是否符合预期结果。
3. 书籍交易测试：验证书籍交易的流程如取消交易、确认书籍已赠送等操作是否正确反应到数据库交易双方的信息表中。
   1. **测试用例构建**
      1. **测试用例编写约定**

测试用例编写约定：如表 8-3 所示：

表 8-3 测试用例编写约定表

|  |  |
| --- | --- |
| 用例编号 | 用例编号： 唯一标识，与需求编号对应，为多对一关系 |
| 用例名称 | 对测试项简短的描述 |
| 前置条件 | 执行用例时需要的预置条件 |
| 操作步骤 | 执行该动作需要完成的操作 |
| 预期结果 | 执行完该动作后程序的表现结果 |
| 实际结果 | 实际输出的结果 |
| 验证结果 | 该测试用例是否执行通过 |
| 测试结果与结论 | 填写此次测试是否通过 |
| 测试执行者 | 按照该用例执行测试的人员 |

* + 1. **测试用例设计**

登录测试如表 8-4：

表 8-4 登录测试用例表

|  |  |
| --- | --- |
| 用例编号 | 1 |
| 用例名称 | 登录 |
| 前置条件 | 用户进入登录界面 |
| 操作步骤 | 输入账号密码 |

续表 8-4

|  |  |
| --- | --- |
| 预期结果 | 账号密码正确则登录成功，反之则失败 |
| 实际结果 | 与预期结果相符合 |
| 问题描述 | 暂无 |
| 验证结果 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 | 周磊 |

注册测试如表 8-5：

表 8-5 注册测试用例表

|  |  |
| --- | --- |
| 用例编号 | 2 |
| 用例名称 | 注册 |
| 前置条件 | 用户进入注册界面 |
| 操作步骤 | 输入账号密码 |
| 预期结果 | 账号密码符合格式规范则注册成功，反之则失败 |
| 实际结果 | 与预期结果相符合 |
| 问题描述 | 暂无 |
| 验证结果 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 | 周磊 |

* + 1. **关键测试用例**
       1. 添加书籍到心愿清单测试如表 8-6 所示：

表 8-6 添加书籍到心愿清单用例表

|  |  |
| --- | --- |
| 用例编号 | 3 |
| 用例名称 | 添加书籍到心愿清单 |
| 前置条件 | 用户进入书籍详情界面 |
| 操作步骤 | 点击添加到心愿清单 |
| 预期结果 | 心愿清单没有该书籍则添加成功 |
| 实际结果 | 与预期结果相符合 |
| 问题描述 | 暂无 |
| 验证结果 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 | 周磊 |

* + - 1. 从心愿清单删除书籍测试如表 8-6 所示：

表 8-7 添加书籍到心愿清单用例表

|  |  |
| --- | --- |
| 用例编号 | 4 |
| 用例名称 | 从心愿清单删除书籍 |
| 前置条件 | 用户进入书籍详情界面 |
| 操作步骤 | 点击删除按钮 |
| 预期结果 | 心愿清单没有该书籍则删除成功 |
| 实际结果 | 与预期结果相符合 |
| 问题描述 | 暂无 |
| 验证结果 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 | 周磊 |

* + 1. **测试用例维护**

根据测试产品的特性和测试用例的一致性,分下面几种特殊情况分别进行处理:：

1. 如果系统的基本特性没变，只是根据 Late Discovery Bug 或 Remedy Ticket 来进一步优化测试用例，只有这时候可以修改测试用例，也就意味着当前修改的测试用例，对目前和以前的版本都有效。
2. 没有产生新的特性，但是系统的原有功能有进一步优化，这时候的初始测试用例对新的版本是无效的，但是对上个版本依然有效，那么在这个时候不能修改或者删除之前测试用例，而是为新版本添加新的测试用例，这一点很重要。之前的测试用例 依然对上一个版本仍然有效。
3. 系统某些之前版本存在的功能取消了，这时候将新版本的对应之前版本多余的测试用例设置为无效。
4. 系统完全新增的特性，增加新的、对应新特性测试用例。通过这样的操作可以使得新旧版本的相同测试用例得到完全一致的管理与维护，测试用例数也不会成几何倍数的增加，可以完全保证测试用例的完整性与有效性。

以下原因可能导致测试用例变更：

1. 软件需求变更：一旦发生软件需求的变更，就有可能会导致软件功能的增加、删除、修改等变化，这时一定要及时变更测试用例。
2. 测试需求的遗漏和误解：当测试需求分析不到位时，就可能导致测试需求遗漏，此时相应的测试用例也要进行变更。很多软件存在隐性需求，在测试需求分析阶段是非常容易遗漏的，在测试执行过程中也很难被测试人员发现发现，这种情况一旦发生需要及时补充测试用例。
3. 测试用例遗漏：发现测试用例未覆盖全部需求，需要补充相应的测试用例。
4. 用户反馈的缺陷：在软件发布之后用户如果反馈缺陷则表明测试不全面，软件仍存在尚未发现的缺陷，需要补充或者修改测试用例。

大多数软件产品都是同时多个版本共同存在的，所以对应的测试用例也是共存的，故测试用例需要定期维护。

并遵循以下原则：

1. 及时删除过时的测试用例

需求的变更也可能导致原有部分测试用例不再能够适合新的用户需求要求。例如,删除了某个需求功能,那么针对该功能的测试用例也不再适合新的需要。所以随着用户对需求的每一次需求变更,都可能需要及时删除那些不再适合使用的测试用例。

1. 及时删除冗余的测试用例

在我们设计测试用例时,可能因为存在两个或者多个冗余测试用中的例集无法同时测试相同的数据内容,降低了二次回归的冗余测试用例工作效率,所以我们常常需要定期的重新整理每个测试用中的例集,及时删除冗余的测试用例。

1. 增加新的测试用例

由于软件需求的变更、用例功能出现遗漏或者软件版本更新发布后在测试中发现的缺陷等多种原因,原有的测试用例集没有完全的覆盖用户对软件的需求,需要增加新的测试用例。

1. 改进测试用例

随着系统不断更新,测试用例也会不断的增加,某些测试用例随着系统的输入和当前系统运行状态的变化而逐渐变得不再适用,这些测试用例难以得到重用,影响到当前回归系统系统测试的质量和效率,需要及时进行更新和改进,使之达到可重用可控制。

* + 1. **总结与展望**
  1. **设计工作总结**

在技术方面，由于我缺乏实际开发经验，这次毕设项目只能算是我的一个自我挑战， 不可能与那些技术大牛做的项目们相提并论，所以，项目中肯定存在这一些不太完善的地方，在今后的学习过程中，我不会遗忘这次的毕设项目，当我的技术水平与项目分析能力进一步提升时，我会回过头来继续完善这个项目。

完成本系统后，本人对于工程有了一个初步的理解，虽然这个系统做的比较粗糙简陋， 功能也不够完善，但是当我以后参考工作学习到更多技术之后，我一定会将本系统进一步完善，毕竟这是我的大学，也倾注了我的很多心血。经过这次，我对Python 更加熟练，对 Flask 的路由机制理解更加透彻，对很多 Flask 插件也有一进一步的理解。我也体会到了快速精通某个技术的最佳方式就是通过不断的尝试与实践，实践才能出真知，在未来的工作中我会继续贯彻这一正确思想，通过实践来学习技术。

* 1. **未来工作展望**

取得的成果与付出的努力是成正比的，经过长久的努力，我终于完成了这一人生中的一大重要考验，对于即将毕业的我，这是一件有无比纪念意义与价值的大事在完成的过程中，我一次又一次反复查看论文的正文，希望能够做自己能力范围内最完美的毕业论文，这是对我自己在四年学习生涯的一个交代。在完成本项目的过程中， 我接触到了了非常多的软件设计思想，也第一次独自的去做需求分析与详细设计报告，这对我而言是一个不小的挑战，开始时，我是觉得非常困难的，但当我真正去实现他们了之后，我的内心无比开心，战胜困难带来喜悦的难以言表的。经过这次，我的学习能力、思维、技术等各个方面都得到了非常大的提高。在做项目的过程当中，我是过得非常痛苦的，因为本身项目经验并不算很充足，一个又一个难以发现的 bug，让我挠头抓耳， 心烦意乱。通过查阅各本专业书籍以及到网上去搜索问题的解决办法，我一点一点成长， 不停的学习与试验，一个又一个 bug 得到解决，这种成就感让我非常满足。每次 bug 解决， 我的项目离成功就近了一步，我也从中又成长了一步。最终，我的项目完成了， 虽然其肯定还有很多缺点，也有很多功能并没有得到完美的诠释，但是我已经非常满足了， 毕竟通过它我得到了许多成长，而且它也是我第一次做的如此完整的项目。

对比之前需求分析，开发的这个公益赠书系统还存很多的缺陷。这些问题主要表现在下面的几个方面：

1. 用户在交易过程当中，并不能得到充分的协商机会，只能通过一次邮件的形式进行沟通。
2. 这个项目仅仅使用了 Flask 的基本技术，但是并没有处理并发的情况，后续可以通过多线程与协程进一步提高性能。
3. 这个项目的积分系统并不是完善，只是通过简单的设立数字来实现用户之间的基本交易，得到有效的数学分析之后，或许可以设计一些非常合理的积分制度，让整个项目的实际应用性大大提高，也可以解决用户交易当中潜在的很多问题，在问题没有解决之前， 潜在问题是一定存在的
4. 该系统还缺少一个用户与用户在线交流的平台。通过这个平台，用户于用户之间能够交流一些读书上的经验或者见解，用户也可以反馈一些意见给他人，只有这样平台的用户黏性才会高，能够吸引更多的用户来到平台。

**谢 辞**

**参考文献**

1. Rongrong Gong, Shijian Luo, Ji He.Use Case Based Innovative Design of E-commerce Website ， Journal of Computer Science and Technology 2012，5：32-35
2. Jan Hatzius，Goldman Sachs Investment Research. Internet B2B eCommerce B2B:2B or Not 2B? Version1.1,2010,12: 68-71.
3. 王晓燕，潘开灵，邓旭东.我国 B2C 电子商务发展现状研究.武汉科技大学管理学院,2011 年 29 期. 第 57 页
4. 黄梯云.管理信息系统 3 版[M].高等教育出版社 2015 年版，第 199 页
5. 孙祥,张硕阳,陈毅文,王二平.B2C 电子商务中消费者的风险来源及其影响中国科学院心理研究所,2006 年 04 期.
6. 萨师煊,王珊.数据库系统概论.高等教育出版社 2016 年版，第 86 页
7. 张基温.计算机网络技术.高等教育出版社 2012 年版，第 104-111 页
8. 陈国良,董荣胜.计算思维与大学计算机基础教育[J].中国大学教学,2011(01):9-13+34. [9]孟小峰,周龙骧,王珊.数据库技术发展趋势[J].软件学报,2004(12):74-88.

[10]吴溥峰,张玉清.数据库安全综述[J].计算机工程,2006(12):91-94.

[11]刘桂.分析软件工程中数据挖掘技术的应用方式[J].中国新通信，2017,19（13）：119. [12]解新星.软件工程数据挖掘研究进展[J].通讯世界，2017,（14）：71-72.

[13]张朋.软件工程技术在系统软件开发中的应用[J].电子测试，2017,（13）：129-130+111. [14].电子技术与软件工程[J].电子技术与软件工程，2017,（14）：266.

[15].电子技术与软件工程[J].电子技术与软件工程，2017,（13）：266.

[16]赵成章，刘元监.软件工程标准化的问题分析与对策研究[J].电子世界，2017,（13）：75. [17]张文瑞.论高可信软件工程技术[J].电子技术与软件工程，2017,（13）：59.

[18]孔晓.软件工程中的常用软件生命周期模型[J].电子技术与软件工程，2017,（14）：58. [19]刘琴.软件工程中程序设计方法的比较[J].计算机时代，2017,（07）：53-55.

1. 李亚丽，梁华国，张国平.基于 ExtJS 和 SSH2 的软件工程资源共享平台[J].信息技术，2017,（07）： 99-102.

[21]段翰林.计算机软件工程的自动化管理[J].科技资讯，2017,15（22）：12-13.

1. 崔萌萌，安强.项目管理对软件工程开发的必要性研究[J].科技风，2017,（17）：65. [23]刘春玉.计算机软件工程项目自动化管理探究[J].中国教育学刊，2017,（S1）：5-7.

[24]谷科.浅谈计算机专业的软件工程课程教学[J].科教导刊（中旬刊），2017,（07）：131-132. [25]张肖然.浅析借鉴软件工程的思想来设计远程教育课程[J].中国高新区，2017,（16）：88-89. [26]陈丛.独立学院《软件工程导论》课程教学改革与研究[J].福建电脑，2017,33（07）：164+58. [27]黄伟国.基于 CDIO 的软件工程课程教学改革[J].信息与电脑（理论版），2017,（13）：225-226+229. [28]沈海波，朱雄泳，周如旗.基于产教融合的软件工程专业转型发展研究与实践[J].教育教学论坛， 2017,（29）：266-268.

1. 葛文庚.编码驱动法在软件工程课程教学中的应用研究[J].教育现代化，2017,4（27）：123-124. [30]聂作财，刘鑫.民办本科院校软件工程专业工学结合人才培养模式的实践与研究[J].教育现代化， 2017,4（27）：1-2.

附录 A 外文翻译—原文部分

外文出处：The History of the Internet

The History of the Internet

Today - The Internet

Today, the Internet has become one of the most important technological advancements in the history of humanity. Everyone wants to get 'on line' to experience the wealth of information of the Internet. Millions of people now use the Internet, and it's predicted that by the year 2003 every single person on the planet will have Internet access. The Internet has truly become a way of life in our time and era, and is evolving so quickly its hard to determine where it will go next, as computer and network technology improve every day.

HOW IT WORKS:

It's a standard thing. People using the Internet. Shopping, playing games,conversing in virtual Internet environments.

The Internet is not a 'thing' itself. The Internet cannot just "crash." It functions the same way as the telephone system, only there is no Internet company that runs the Internet.

The Internet is a collection of millioins of computers that are all connected to each other, or have the means to connect to each other. The Internet is just like an office network, only it has millions of computers connected to it.

The main thing about how the Internet works is communication. How does a computer in Houston know how to access data on a computer in Tokyo to view a webpage?

Internet communication, communication among computers connected to the Internet, is based on a language. This language is called TCP/IP. TCP/IP establishes a language for a computer to access and transmit data over the Internet system.

But TCP/IP assumes that there is a physical connecetion between one computer and another. This is not usually the case. There would have to be a network wire that went to every computer connected to the Internet, but that would make the Internet impossible to access.

The physical connection that is requireed is established by way of modems,phonelines, and other modem cable connections (like cable modems or DSL). Modems on computers read and transmit data over established lines,which could be phonelines or data lines. The actual hard core connections are established among computers called routers.

A router is a computer that serves as a traffic controller for information.

To explain this better, let's look at how a standard computer might view a webpage.

* 1. The user's computer dials into an Internet Service Provider (ISP). The ISP might in turn be connected to another ISP, or a straight connection into the Internet backbone.
  2. The user launches a web browser like Netscape or Internet Explorer and types in an internet location to go to.
  3. Here's where the tricky part comes in. First, the computer sends data about it's data request to a router. A router is a very high speed powerful computer running special software. The collection of routers in the world make what is called a "backbone," on which all the data on the Internet is transferred. The backbone presently operates at a speed of several gigabytes per-second. Such a speed compared to a normal modem is like comparing the heat of the sun to the heat of an ice-cube.

Routers handle data that is going back and forth. A router puts small chunks of data into packages called packets, which function similarly to envelopes. So, when the request for the webpage goes through, it uses TCP/IP protocols to tell the router what to do with the data, where it's going, and overall where the user wants to go.

* 1. The router sends these packets to other routers, eventually leading to the target computer. It's like

whisper down the lane (only the information remains intact).

* 1. When the information reaches the target web server, the webserver then begins to send the web page back. A webserver is the computer where the webpage is stored that is running a program that handles requests for the webpage and sends the webpage to whoever wants to see it.
  2. The webpage is put in packets, sent through routers, and arrive at the users computer where the user can view the webpage once it is assembled.

The packets which contain the data also contain special information that lets routers and other computers know how to reassemble the data in the right order.

With millions of web pages, and millions of users, using the Internet is not always easy for a beginning user, especially for someone who is not entirely comfortale with using computers. Below you can find tips tricks and help on how to use main services of the Internet.

Before you access webpages, you must have a web browser to actually be able to view the webpages. Most Internet Access Providers provide you with a web browser in the software they usually give to customers; you. The fact that you are viewing this page means that you have a web browser. The top two use browsers are Netscape Communicator and Microsoft Internet Explorer. Netscape can be found at and MSIE can be found at [www.microsoft.com/ie.](http://www.microsoft.com/ie)

The fact that you're reading this right now means that you have a web browser.

Next you must be familiar with actually using webpages. A webpage is a collection of hyperlinks, images, text, forms, menus, and multimedia. To "navigate" a webpage, simply click the links it provides or follow it's own instructions (like if it has a form you need to use, it will probably instruct you how to use it). Basically, everything about a webpage is made to be self- explanetory. That is the nature of a webpage, to be easily navigatable.

"Oh no! a 404 error! 'Cannot find web page?'" is a common remark made by new web-users. Sometimes websites have errors. But an error on a website is not the user's fault, of course.

A 404 error means that the page you tried to go to does not exist. This could be because the site is still being constructed and the page hasn't been created yet, or because the site author made a typo in the page.

There's nothing much to do about a 404 error except for e-mailing the site administrator (of the page you wanted to go to) an telling him/her about the error.

A Javascript error is the result of a programming error in the Javascript code of a website. Not all websites utilize Javascript, but many do. Javascript is different from Java, and most browsers now support Javascript. If you are using an old version of a web browser (Netscape 3.0 for example), you might get Javascript errors because sites utilize Javascript versions that your browser does not support. So, you can try getting a newer version of your web browser.

E-mail stands for Electronic Mail, and that's what it is. E-mail enables people to send letters, and even files and pictures to each other.

To use e-mail, you must have an e-mail client, which is just like a personal post office, since it retrieves and stores e-mail.

Secondly, you must have an e-mail account. Most Internet Service Providers provide free e-mail account(s) for free. Some services offer free e-mail, like Hotmail, and Geocities.

After configuring your e-mail client with your POP3 and SMTP server address (your e-mail provider will give you that information), you are ready to receive mail.

An attachment is a file sent in a letter. If someone sends you an attachment and you don't know who it is, don't run the file, ever. It could be a virus or some other kind of nasty programs. You can't get a virus just by reading e-mail, you'll have to physically execute some form of program for a virus to strike.

A s ignature is a feature of many e-mail programs. A signature is added to the end of every e-mail you send out. You can put a text graphic, your business information, anything you want.

Imagine that a computer on the Internet is an island in the sea. The sea is filled with millions of islands. This is the Internet. Imagine an island communicates with other island by sending ships to other islands and receiving ships. The island has ports to accept and send out ships.

A computer on the Internet has access nodes called ports. A port is just a symbolic object that allows the computer to operate on a network (or the Internet). This method is similar to the island/ocean symbolism above.

Telnet refers to accessing ports on a server directly with a text connection. Almost every kind of Internet function, like accessing web pages,"chatting," and e-mailing is done over a Telnet connection.

Telnetting requires a Telnet client. A telnet program comes with the Windows system, so Windows users can access telnet by typing in "telnet" (without the "'s) in the run dialog. Linux has it built into the command line; telnet. A popular telnet program for Macintosh is NCSA telnet.

Any server software (web page daemon, chat daemon) can be accessed via telnet, although they are not usually meant to be accessed in such a manner. For instance, it is possible to connect directly to a mail server and check your mail by interfacing with the e-mail server software, but it's easier to use an e-mail client (of course).

There are millions of WebPages that come from all over the world, yet how will you know what the

address of a page you want is?

Search engines save the day. A search engine is a very large website that allows you to search it's own database of websites. For instance, if you wanted to find a website on dogs, you'd search for "dog" or "dogs" or "dog information." Here are a few search-engines.

1. Altavista (http://www.altavista.digital.com) - Web spider & Indexed
2. Yahoo (http://www.yahoo.com) - Web spider & Indexed Collection

（3）. Excite (http://www.excite.com) - Web spider & Indexed

1. Lycos (http://www.lycos.com) - Web spider & Indexed
2. Metasearch (http://www.metasearch.com) - Multiple search

A web spider is a program used by search engines that goes from page to page, following any link it can possibly find. This means that a search engine can literally map out as much of the Internet as it's own time and speed allows for.

An indexed collection uses hand-added links. For instance, on Yahoo's site. You can click on Computers & the Internet. Then you can click on Hardware. Then you can click on Modems, etc., and along the way through sections, there are sites available which relate to what section you're in.

Metasearch searches many search engines at the same time, finding the top choices from about 10 search engines, making searching a lot more effective.

Once you are able to use search engines, you can effectively find the pages you want.

With the arrival of networking and multi user systems, security has always been on the mind of system developers and system operators. Since the dawn of AT&T and its phone network, hackers have been known by many, hackers who find ways all the time of breaking into systems. It used to not be that big of a problem, since networking was limited to big corporate companies or government computers who could afford the necessary computer security.

The biggest problem now-a-days is personal information. Why should you be careful while making purchases via a website? Let's look at how the internet works, quickly.

The user is transferring credit card information to a webpage. Looks safe, right? Not necessarily. As the user submits the information, it is being streamed through a series of computers that make up the Internet backbone. The information is in little chunks, in packages called packets. Here's the problem: While the information is being transferred through this big backbone, what is preventing a "hacker" from intercepting this data stream at one of the backbone points?

Big-brother is not watching you if you access a web site, but users should be aware of potential threats while transmitting private information. There are methods of enforcing security, like password protection, an most importantly, encryption.

Encryption means scrambling data into a code that can only be unscrambled on the "other end." Browser's like Netscape Communicator and Internet Explorer feature encryption support for making on-line transfers. Some encryptions work better than others. The most advanced encryption system is called DES (Data

Encryption Standard), and it was adopted by the US Defense Department because it was deemed so difficult to 'crack' that they considered it a security risk if it would fall into another countries hands.

A DES uses a single key of information to unlock an entire document. The problem is, there are 75 trillion possible keys to use, so it is a highly difficult system to break. One document was cracked and decoded, but it was a combined effort of 14,000 computers networked over the Internet that took a while to do it, so most hackers don't have that many resources available.

附录 B 外文翻译—译文部分

Internet 的历史

起源——ARPAnet

Internet 是被美国政府作为一项工程进行开发的。这项工程的目的，是为了建立远距离之间点与点的通信，以便处理国家军事范围内的紧急事件，例如核战争。这项工程被命名为 ARPAnet，它就是Internet 的前身。建立此工程的主要应用对象就是军事通讯，那些负责 ARPAnet 的工程师们当时也没有想到它将成为“Internet”。

根据定义，一个“Internet”应该由四或者更多的计算机连接起来的网络。

ARPAnet 是通过一种叫 TCP/IP 的协议实现连网工作的。此协议最基础的工作原理是：如果信息在网络中的一条路径发送失败，那么它将找到其他路径进行发送，就好象建立一种语言以便一台计算机与其他计算机“交谈”一样，但不注意它是 PC，或是 Macintosh。

到了 20 世纪 80 年代，ARPAnet 已经开始变成目前更为有名的 Internet 了，它拥有 200 台在线主机。国防部很满意 ARPAnets 的成果，于是决定全力将它培养为能够联系很多军事主机，资源共享的服务网络。到了 1984 年，它就已经超过 1000 台主机在线了。

在 1986 年 ARPAnet 关闭了，但仅仅是建立它的机构关闭了，而网络继续存在与超过 1000 台的主机之间。由于使用 NSF 连接失败，ARPAnet 才被关闭。NSF 是将 5 个国家范围内的超级计算机连入 ARPAnet。

随着 NSF 的建立，新的高速的传输介质被成功的使用，在 1988 年，用户能通过 56k 的电话线上网。在那个时候有 28，174 台主机连入 Internet。到了 1989 年有 80，000 台主机连入 Internet。到 1989 年末，就有 290，000 台主机连入了。

另外还有其他网络被建立，并支持用户以惊人的数量接入。于 1992 年正式建立。现状——Internet

如今，Internet 已经成为人类历史上最先进技术的一种。每个人都想“上网”去体验一下 Internet 中的信息财富。成千上百的人都用 Internet。预计，到了 2003 年世界上的每个人，都将拥有 Internet 接入。Internet 已经真正成为我们这个年代生活的一部分。由于计算机技术和网络技术每天都在改变， 我们很难想象 Internet 下一步将发展成什么样子。

工作原理：

现在，人们用 Internet 是一件很平常的事。他们通过 Internet 进行购物、玩游戏、聊天等娱乐活动。

Internet 不仅仅是一件事物。Internet 也会崩溃。它的工作原理如同电话通信系统，只不过没有专门的 Internet 公司来经营 Internet。

Internet 是成千上万台计算机互相连接的集合。Internet 就像是办公网络一样，不同的是，它有成千上万台计算机相连接。

其中最主要的是 Internet 是如何进行通信的。位于 Houston 的一台计算机是如何通过浏览网页而能与位于 Tokyo 的计算机进行数据通信的呢？

Internet 信息，拥有信息的计算机连接到 Internet，是基于语言。这种语言叫做 TCP/IP。TCP/IP

建立了一种语言，能使计算机在 Internet 系统中传送数据。

但是 TCP/IP 的取得也必须具备两台电脑之间的物理连接。当然也未必都是这样。但也必须存在一根网络线将主机与 Internet 连接起来，但做到这样，还是不可能与 Internet 连接的。

物理连接要求通过 MODEM，电话线和其他类似 MODEM 的连接（如 DSL）来建立。计算机上的 MODEM 通过已建立的通信线进行收发数据，通信线可以是电话线或是数据线。事实上计算机之间建立连接的硬核被成为路由器。

路由器就是计算机中进行信息交互的管理器。

为了更好的对它说明，让我们来看看一台标准的计算机是怎样浏览网页的？

1. 用户计算机拨号进入 ISP。而此 ISP 可能还要连接入其他 ISP，或是直接进入 Internet 主干。
2. 用户打开网页浏览器如 NETSCAPE 或是 IE。
3. 接下来是进入 Internet 的棘手部分。首先，用户计算机相路由器发出请求。路由器是一种高速高效的计算机运行的专门软件。世界上所有路由的连接便形成了 Internet 的主干，在这里传送Internet 上的所有数据。目前主干网上的处理速度为每秒几千兆字节。这样的速度分配到一只 MODEM 上，就好比太阳光的热量分配到一块冰上的热量一样。
4. 路由器发送或接受数据。它将一小段数据分别打包，形成数据报，就像包裹一样。因此，当请求网页浏览是，就用 TCP/IP 协议告诉路由器如何处理这些数据，将这些数据发送去哪里，用户主要想去哪里。
5. 路由器将这些数据报发送给其他的路由器，最终转到目标主机上。就像传耳语的游戏一样（ 当然，只有完整的信息才能被传送）。
6. 当信息到达目标网页的服务器是，服务器就开始将网页信息发送回去。一台网页服务器，就是网页存储所在的计算机，它能对网页进行编辑，并将它发送给用户。网页被分成数据报，通过路由器， 最终到达用户计算机，这样，用户就能浏览网页了。数据报中含有相关的数据以及一些必须的信息让路由器或其他计算机知道如何将数据报按正确的顺序重新组装成原始的数据段。

有了成千上万的网页和成千上万的用户，对于初学者来说使用 Internet 将不再那么容易，尤其是那些不太精通电脑的人。接下来，你将能找到一些上网的小技巧和使用 Internet 主要服务的帮助。

在你打开网页之前，你必须有一个网页浏览器用于浏览网页。大部分网络服务商都会给用户提供一个网络浏览器。当你在浏览网页时，其实就是在使用浏览器。目前使用最广泛的网页浏览器是 Netscape 和 MSIE。Netscape 能自动连接到 [www.netscape.com](http://www.netscape.com/)，MSIE 能自动连接 [www.microsoft.com](http://www.microsoft.com/)。

接下来你就必须熟悉如何使用网页。网页是超链接、图片、文本、表格、按钮以及多媒体的集合。只需点击网页提供的连接或是按照网页的步骤（比如，如果你需要用网页中的一张表，旁边就会有使用这张表的帮助）做，你就可以进行网上冲浪了。基本上，网页中的每个元素都可以自我移植。

“哦，不！又是 404 出错！‘不能找到相关网页’”这是上网初学者中很普通的言论。有时网站也会出错。当然网站的错误跟用户的操作没有关系。

404 出错意思是你想找的那个网页不存在。这有可能是因为网站仍在建设中，页面还没有被创建，

或者是因为网站的制作者正在对页面进行修改。当出现 404 出错时，除了对网站管理发电子邮件，告诉他/她关于出错的问题，就别无他法了。

Javascript 出错是由于网站中的 Javascript 的程序代码出错造成的。并非所有的网页都使用Javascript，但有很多是用 Javascript 的。Javascript 不同于 JAVA，目前大部分浏览器都支持Javascript。如果你现在用的是旧版的网页浏览器，那么就有可能出现 Javascript 出错，那是因为你的浏览器的版本低于站点使用的 Javascript 版本。所以，你应该为你的浏览器升级到新的版本。

电子邮件即电子方式的邮件。电子邮件能使人们互相收发信件，甚至是文件和图片。

要使用电子邮件，那你就必须拥有一个电子邮件客户端，它就像是邮递员，为你收发电子邮件。其次，你必须有一个电子邮件帐户。大部分网络服务供应商都会提供给用户一个免费的电子邮件帐

户。有的网站也提供免费的电子邮箱，如 Hotmail 和 Geocities。

配置好电子邮件客户端，包括 POP3 和 SMTP 服务器地址（电子邮件供应商会给你相关信息）之后， 你就可以准备收邮件了。

附件是与信一起发过来的文件，如果有人发给你带有附件的邮件，但你不知道他是谁，就不要打开那个附件。它有可能是病毒或是其他恶意的程序。用户不会通过阅读邮件而被传染病毒，你不得不运行杀毒软件来防止病毒感染。

签名是很多电子邮件程序的一个特征。签名加在邮件的末尾。你可以将生动的文本，你职业信息或是其他你喜欢的东西作为签名。

设想在国际互联网中的电脑就像海中的岛屿。那么海洋中就充满了数百万的岛屿。这就是国际互联网。设想岛与岛之间是通过发送和接收船只来联络的。那么岛屿就有接收和发送船只的港口。

在国际互联网中的电脑有被叫做“港口”的端口。端口只是一个象征性的对象，它使得电脑能在网络（或是国际互联网）上运转。这种方法与上面提到的“岛屿、海洋”形象论类似。

远程登陆涉及到在服务器上直接用文本连接来使用端口。

几乎每种互联网活动，如访问网页、聊天、以及发电子邮件都是依托远程登陆来实现的。

远程登陆需要一个远程登陆客户服务器。远程登陆程序兼容于 Windows 系统，所以 Windows 用户可以通过在对话窗口键入 telnet（不加's）来接入远程登陆软件。Linux 系统则将它创建在命令行里：键入 telnet。用于 Macintosh 的普通远程登陆程序是 NCSA 远程登陆软件。

任何一个服务软件（网页端口监督程序，聊天端口监督程序）都可以通过远程登陆被接入，即使这些软件不经常以这种方法被接入。

举例来说，直接与邮件服务器连接并通过与邮件服务软件交流来校对你的邮件是可能的，但用电子邮件客户服务器却更简单（当然如此）。

世界各地有数百万的网页，那么你如何知道哪个网页的地址是你需要的呢？

搜索工具能节约时间。搜索工具是一个非常庞大的网站，它使你能够搜寻它自有的网站数据库。举例来说，如果你要找关于“狗”的网站，你可以搜索“狗”或“狗的信息”。这里有一些搜索引

擎：

1. Altavista (http://www.altavista.digital.com) - Web spider & Indexed
2. Yahoo (http://www.yahoo.com) - Web spider & Indexed Collection
3. Excite (http://www.excite.com) - Web spider & Indexed
4. Lycos (http://www.lycos.com) - Web spider & Indexed
5. Metasearch (http://www.metasearch.com) - Multiple search

网络蜘蛛是一种搜索引擎使用的程序，它随着可能找到的任何链接从一个网页到另一个网页。

这意味着只要它自己的时间和速度允许，搜索引擎就能尽可能多地在网络上逐字搜寻信息。编入索引的收集使用手动附加链接。比如雅虎网站。你可以点击“电脑和网络”，再点击“硬件”，然后点击“猫”（调制解调器）等等。那么通过分类的过程，与你要的类别相关的网站就找到了。

Metasearch 同时搜索许多搜索引擎，从十个引擎中找出最上面的搜索结果，使得搜索变得有效很多。一旦你能够使用搜索引擎，你就能有效的找到你要的网页。伴随着网络系统和多用户系统时代的来临，安全总是系统开发和系统操作人员考虑的问题。

从美国电报电话公司和它的电话网组建开始，黑客已为许多人所知。黑客就是一直寻找途径侵入系统的人。这过去不是一个大问题，因为网络系统对于那些能够提供必要的电脑安全措施的大型法人公司和政府的电脑来说是有限的。

现今最大的问题是个人信息。为什么当你通过网络购物时应该小心？让我们快速地看一看国际互联网时如何工作的。用户将信用卡的信息传递到网页上。看起来安全，对吗？其实未必。当用户提交信息时，它就会流过组成互联网主干网的一系列的电脑。信息是小块的一包包的，被称作信息包。

这里有个问题：当信息通过这巨大的支柱被传递时，怎样防止黑客在支柱的某一点上拦截数据流？ 你接入网站时哥哥不会看着你，但用户在传递私人信息时应该清楚潜在的威胁。

加强安全有很多方法，如密码保护，一种最重要的措施：加密。加密指将数据搅乱成只能在另一终端才能译出的密码。像 Netscape Communicator 和 Internet Explore 这样的浏览器为在线传递提供了特色的加密支持。

一些加密措施比其他工作地更好。最先进的加密系统被称作 DES（数据加密准则），且它被美国安全局采纳，因为它被认为是如此难以非法侵入以至他们认为如果它落入其它国家手中会有安全风险。

DES 用单独的信息钥匙来开启一整个文件。问题是有 75 万亿可能的钥匙可用，所以它是一个很难破坏的系统。一个文件被侵入并被解码，但这是一个需 14000 台电脑进入互联网系统的综合的努力过程， 并需要花一段时间来做。所以大部分黑客没有那么多的资源。

附录 C 软件使用说明书

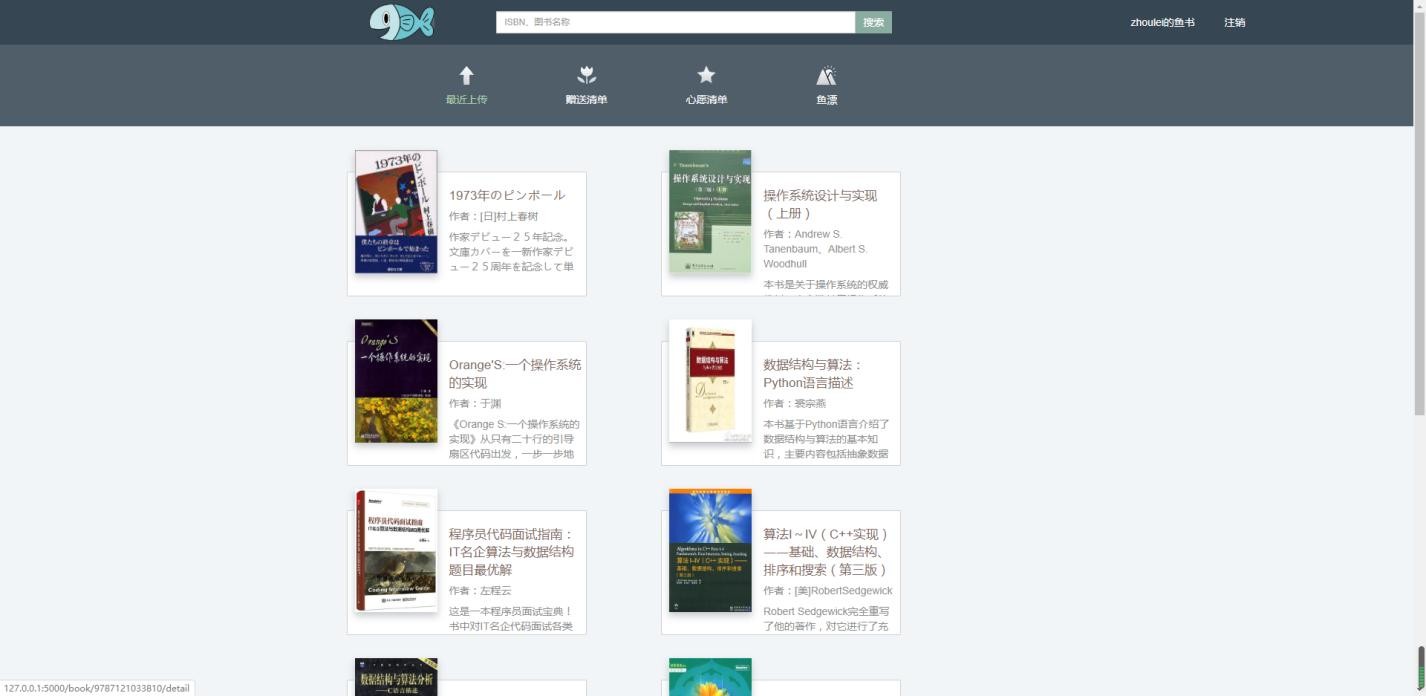
项目首页如图 1 所示，在该页面中能看到所有最近用户上传的可赠送的书籍：

图 1 主页面

点击任意书籍封面即可进入书籍详情页面，书籍详情页面包含书籍的作者、摘要、价格、想要此书的人数、想要获得此书的人数等关键信息。页面展示结果如第图 2 所示。

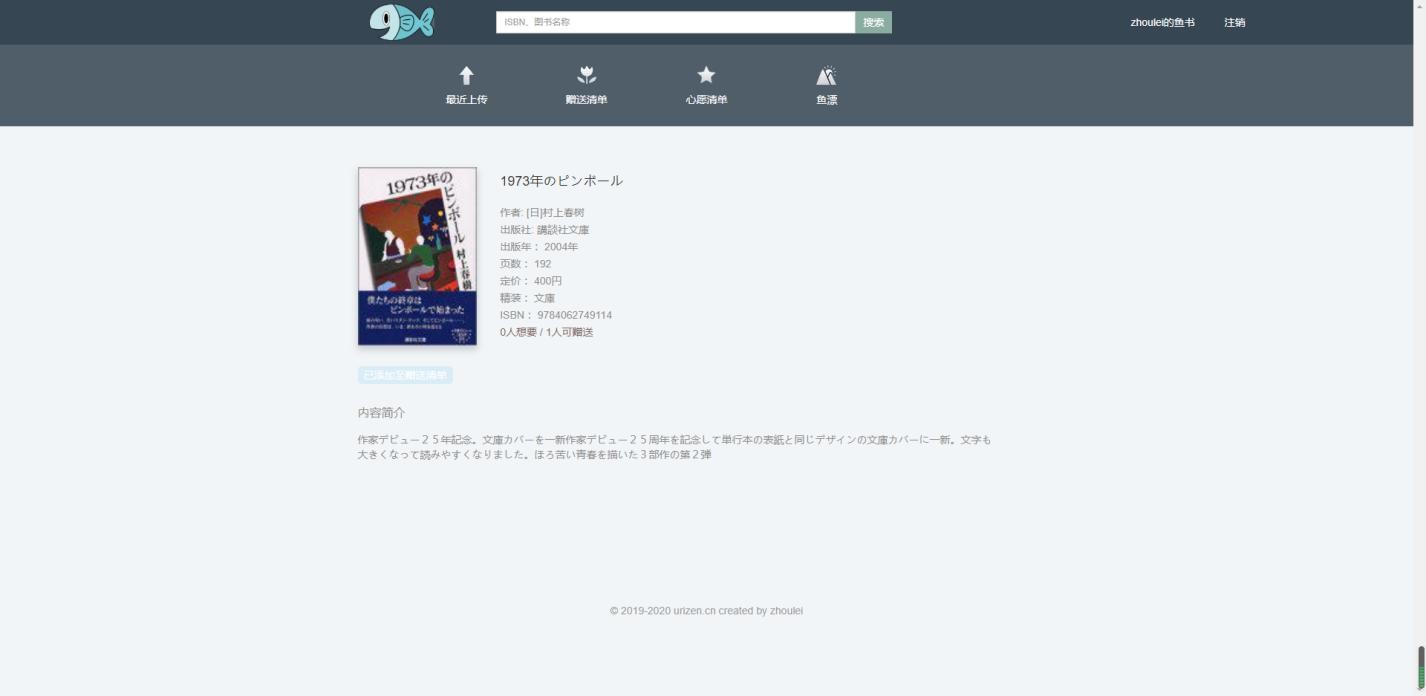


图 2 书籍详情页

从首页点击进入赠送清单页面，赠送清单页面记录了用户所有的想要赠出的书籍，在此页面也可以通过封面图进入书籍的详情页面，页面展示结果如第图 3 所示：

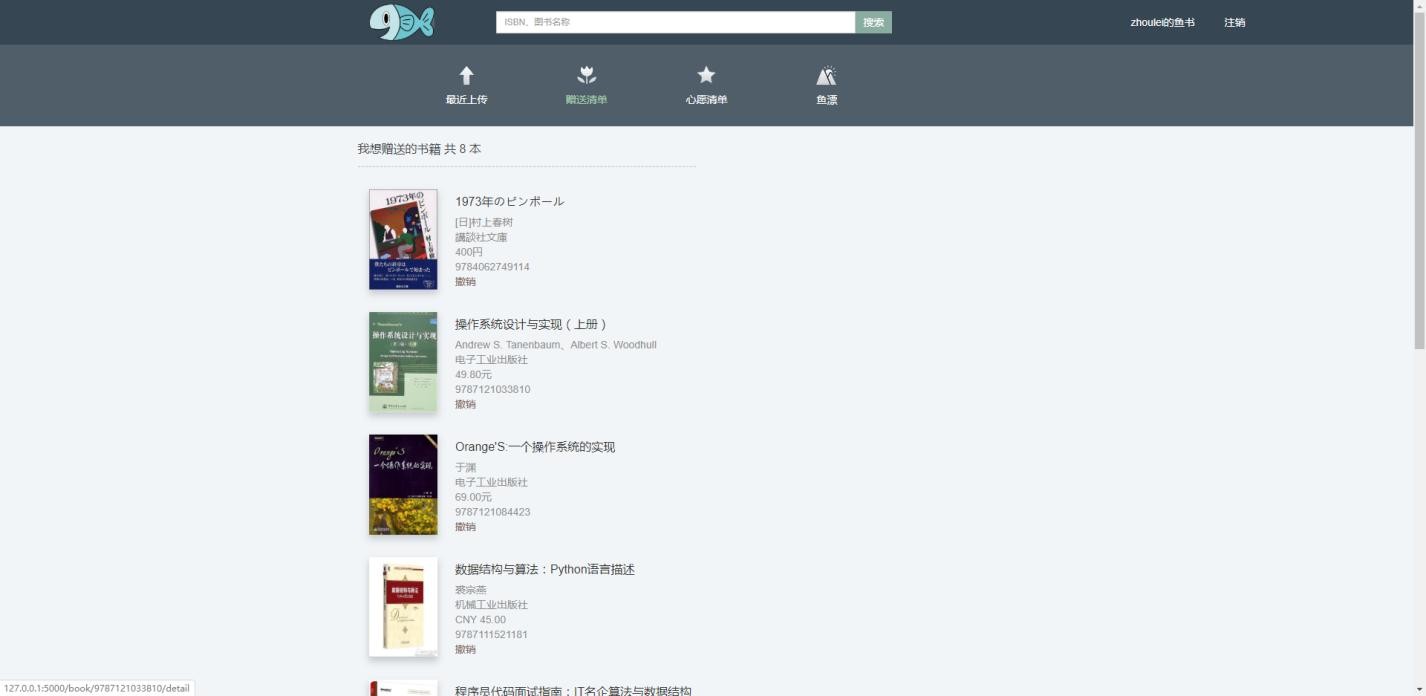


图 3 赠送清单页

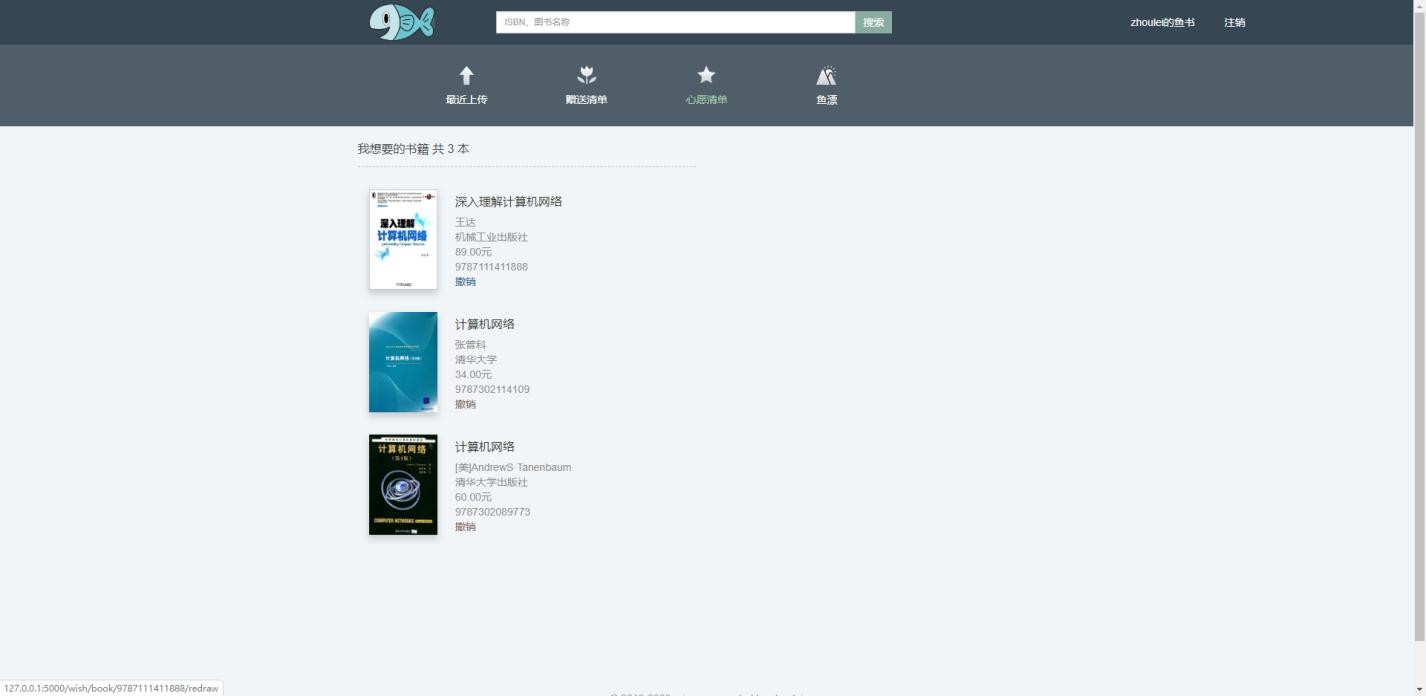
从首页点击进入心愿清单页面，心愿清单页面记录了用户所有的想要的书籍，在此页面也可以通过封面图进入书籍的详情页面，页面展示结果如第图 4 所示：

图 4 心愿清单页

从首页点击进入鱼漂页面，交易信息记载了最近进行的交易，并能够看到交易的状态，如果是赠书方可以在此页面选择同意或者拒绝交易。交易信息展示结果如第图 5 所示：

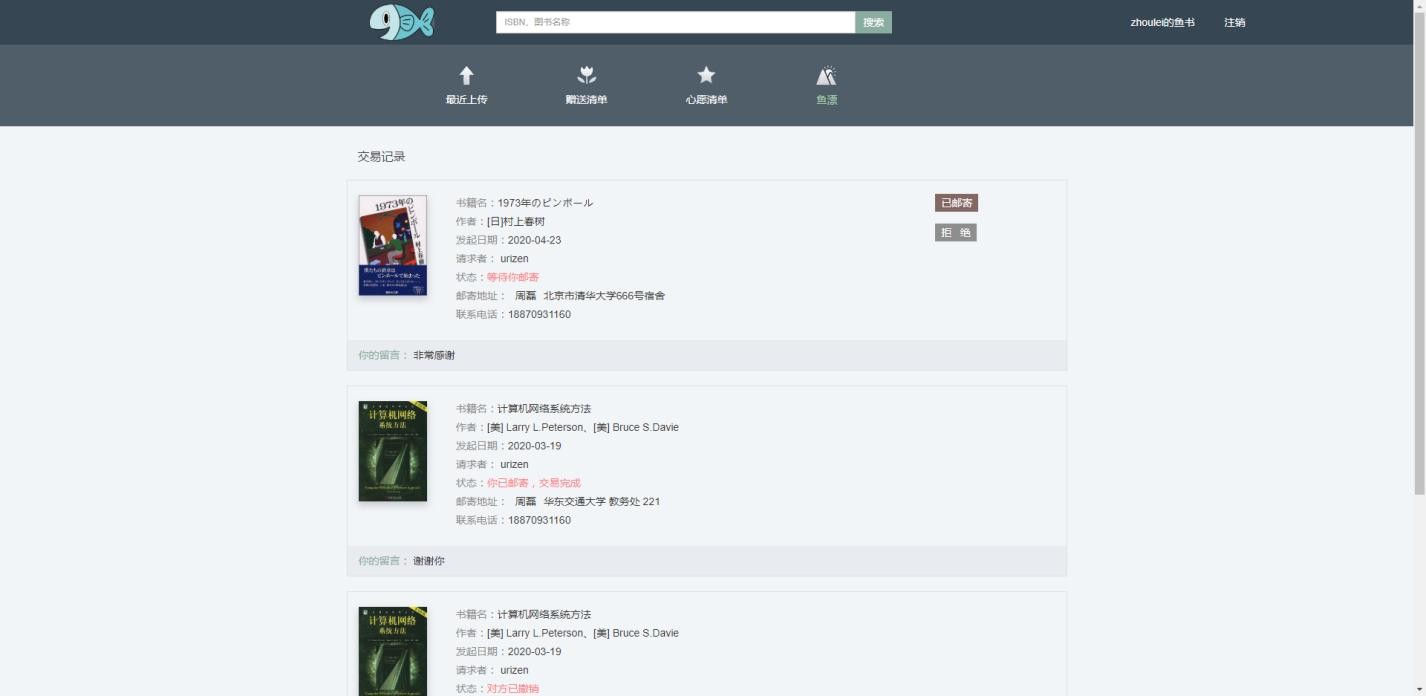


图 5 书籍详情页

附录 D 主要源代码

1. 项目初始化代码如下：

login\_manager = LoginManager() mail = Mail()

admin = Admin(name="Fishbook 后台管理系统")

def create\_app():

app = Flask( name ) app.config.from\_object('app.secure') app.config.from\_object('app.setting') register\_blueprint(app) db.init\_app(app)

login\_manager. init\_app(app) admin.init\_app(app)

login\_manager. login\_view = 'web.login' login\_manager. login\_message = '请先登录或注册' register\_admin(admin)

mail.init\_app(app)

with app.app\_context(): db.create\_all()

return app

def register\_blueprint(app):

from app.web.book import web app.register\_blueprint(web)

def register\_admin(admin):

from flask\_admin.contrib.sqla import ModelView from app.models.user import User

from app.models.drift import Drift from app.models.gift import Gift

from app.models.wish import Wish

# 在这里初始化Flask Flask-SQLAlchemy Flask-Admin

admin.add\_view(ModelView(User, db.session)) admin.add\_view(ModelView(Book, db.session)) admin.add\_view(ModelView(Gift, db.session)) admin.add\_view(ModelView(Drift, db.session))

admin.add\_view(ModelView(Wish, db.session))

1. 发送邮件相关代码：

from app import mail

from threading import Thread from flask\_mail import Message

from flask import current\_app, render\_template

def send\_async\_email(app, msg): with app.app\_context():

try:

mail.send(msg) except Exception as e:

pass

def send\_mail(to, subject, template, \*\*kwargs): msg = Message('[FishBook]' + ' ' + subject,

sender=current\_app.config['MAIL\_USERNAME'], recipients=[to])

msg.html = render\_template(template, \*\*kwargs) # current\_app app = Flask()

app = current\_app.\_get\_current\_object()

thr = Thread(target=send\_async\_email, args=[app, msg]) thr.start()

1. 部分业务代码：

class PendingStatus(Enum): """交易状 4 态"""

Waiting = 1

Success = 2

Reject = 3

Redraw = 4

@classmethod

def pending\_str(cls, status, key): key\_map = {

cls.Waiting: {

'requester': '等待对方邮寄', 'gifter': '等待你邮寄'

},

cls.Reject: {

'requester': '对方已拒绝', 'gifter': '你已拒绝'

},

cls.Redraw: {

'requester': '你已撤销',

'gifter': '对方已撤销'

},

cls.Success: {

'requester': '对方已邮寄', 'gifter': '你已邮寄，交易完成'

}

}

return key\_map[statu]

1. 关键基类：

class SQLAlchemy(\_SQLAlchemy): @contextmanager

def auto\_commit(self): try:

yield self.session.commit()

except Exception as e: db.session.rollback() raise e

class Query(BaseQuery):

def filter\_by(self, \*\*kwargs):

if 'status' not in kwargs.keys(): kwargs['status'] = 1

return super(Query, self).filter\_by(\*\*kwargs)

db = SQLAlchemy(query\_class=Query)

class Base(db.Model):

abstract = True

create\_time = Column('create\_time', Integer) status = Column(SmallInteger, default=1)

def init (self):

self.create\_time = int(datetime.now().timestamp())

def set\_attrs(self, attrs\_dict):

for key, value in attrs\_dict.items():

if hasattr(self, key) and key != 'id': setattr(self, key, value)

@property

def create\_datetime(self): if self.create\_time:

return datetime.fromtimestamp(self.create\_time)

else:

return None

def delete(self):

self.status = 0

self.status = 0

1. 核心逻辑代码：

@web.route('/register', methods=['GET', 'POST']) def register():

form = RegisterForm(request.form)

if request.method == 'POST' and form.validate(): with db.auto\_commit():

user = User() user.set\_attrs(form.data) db.session.add(user)

return redirect(url\_for('web.login'))

return render\_template('auth/register.html', form=form)

@web.route('/login', methods=['GET', 'POST']) def login():

form = LoginForm(request.form)

if request.method == 'POST' and form.validate():

user = User.query.filter\_by(email=form.email.data).first() if user and user.check\_password(form.password.data):

login\_user(user, remember=True) next = request.args.get('next')

if not next or not next.startswith('/'): next = url\_for('web.index')

return redirect(next)

else:

flash('账号不存在或密码错误')

return render\_template('auth/login.html', form=form)

@web.route('/reset/password', methods=['GET', 'POST']) def forget\_password\_request():

form = EmailForm(request.form) if request.method == 'POST':

if form.validate():

account\_email = form.email.data

user = User.query.filter\_by(email=account\_email).first\_or\_404() send\_mail(form.email.data, '重置你的密码',

'email/reset\_password.html', user=user,

token=user.generate\_token())

flash('一封邮件已发送到邮箱' + account\_email + '，请及时查收') # return redirect(url\_for('web.login'))

return render\_template('auth/forget\_password\_request.html', form=form)

@web.route('/reset/password/<token>', methods=['GET', 'POST']) def forget\_password(token):

form = ResetPasswordForm(request.form)

if request.method == 'POST' and form.validate():

success = User.reset\_password(token, form.password1.data) if success:

flash('你的密码已更新,请使用新密码登录')

return redirect(url\_for('web.login'))

else:

flash('密码重置失败')

return render\_template('auth/forget\_password.html', form=form)

@web.route('/change/password', methods=['GET', 'POST']) @login\_required

def change\_password():

form = ChangePasswordForm(request.form)

if request.method == 'POST' and form.validate(): with db.auto\_commit():

current\_user.password = form.new\_password1.data flash('密码已更新成功')

return redirect(url\_for('web.personal\_center'))

return render\_template('auth/change\_password.html', form=form)

@web.route('/logout') def logout():

logout\_user()

return redirect(url\_for('web.index'))

@web.route('/book/search') def search():

"""

q :普通关键字 isbn page

?q=金庸&page=1

"""

form = SearchForm(request.args) books = BookCollection()

if form.validate():

q = form.q.data.strip() page = form.page.data

isbn\_or\_key = is\_isbn\_or\_key(q) yushu\_book = YuShuBook()

if isbn\_or\_key == 'isbn': yushu\_book.search\_by\_isbn(q)

else:

yushu\_book.search\_by\_keyword(q, page)

books.fill(yushu\_book, q)

else:

flash('搜索的关键字不符合要求，请重新输入关键字') # return jsonify(form.errors)

return render\_template('search\_result.html', books=books, form=form)

@web.route('/book/<isbn>/detail') def book\_detail(isbn):

has\_in\_gifts = False has\_in\_wishes = False

yushu\_book = YuShuBook() yushu\_book.search\_by\_isbn(isbn)

book = BookViewModel(yushu\_book.first)

if current\_user.is\_authenticated:

if Gift.query.filter\_by(uid=current\_user.id, isbn=isbn,

launched=False).first():

has\_in\_gifts = True

if Wish.query.filter\_by(uid=current\_user.id, isbn=isbn,

launched=False).first():

has\_in\_wishes = True

trade\_gifts = Gift.query.filter\_by(isbn=isbn, launched=False).all() trade\_wishes = Wish.query.filter\_by( isbn=isbn, launched=False).all()

trade\_wishes\_model = TradeInfo(trade\_wishes) trade\_gifts\_model = TradeInfo(trade\_gifts)

return render\_template('book\_detail.html',

book=book, wishes=trade\_wishes\_model,

gifts=trade\_gifts\_model, has\_in\_wishes=has\_in\_wishes, has\_in\_gifts=has\_in\_gifts)

@web.route('/drift/<int:gid>', methods=['GET', 'POST']) @login\_required

def send\_drift(gid):

current\_gift = Gift.query.get\_or\_404(gid)

if current\_gift.is\_yourself\_gift(current\_user.id):

flash('这本书是你自己的^\_^, 不能向自己索要书籍噢')

return redirect(url\_for('web.book\_detail', isbn=current\_gift.isbn))

can = current\_user.can\_send\_drift() if not can:

return render\_template('not\_enough\_beans.html', beans=current\_user.beans)

form = DriftForm(request.form)

if request.method == 'POST' and form.validate(): save\_drift(form, current\_gift)

# send\_mail()

send\_mail(current\_gift.user.email, '有人想要一本书', 'email/get\_gift.html', wisher=current\_user,

gift=current\_gift)

return redirect(url\_for('web.pending')) # email 短信

gifter = current\_gift.user.summary return render\_template('drift.html',

gifter=gifter, user\_beans=current\_user.beans, form=form)

@web.route('/pending') @login\_required

def pending():

drifts = Drift.query.filter( or\_(Drift.requester\_id == current\_user.id,

Drift.gifter\_id == current\_user.id)) \

.order\_by(desc(Drift.create\_time)).all()

views = DriftCollection(drifts, current\_user.id)

return render\_template('pending.html', drifts=views.data)

@web.route('/drift/<int:did>/reject') @login\_required

def reject\_drift(did):

with db.auto\_commit():

drift = Drift.query.filter(Gift.uid == current\_user.id,

Drift.id == did).first\_or\_404() drift.pending = PendingStatus.Reject

requester = User.query.get\_or\_404(drift.requester\_id) requester.beans += 1

return redirect(url\_for('web.pending'))

@web.route('/drift/<int:did>/redraw') @login\_required

def redraw\_drift(did):

with db.auto\_commit():

drift = Drift.query.filter\_by( requester\_id=current\_user.id, id=did).first\_or\_404()

drift.pending = PendingStatus.Redraw current\_user.beans += 1

return redirect(url\_for('web.pending'))

@web.route('/drift/<int:did>/mailed') @login\_required

def mailed\_drift(did):

with db.auto\_commit():

drift = Drift.query.filter\_by( gifter\_id=current\_user.id, id=did).first\_or\_404()

drift.pending = PendingStatus.Success current\_user.beans += 1

gift = Gift.query.filter\_by(id=drift.gift\_id).first\_or\_404() gift.launched = True

Wish.query.filter\_by(isbn=drift.isbn, uid=drift.requester\_id,

launched=False).update({Wish.launched: True}) return redirect(url\_for('web.pending'))

def save\_drift(drift\_form, current\_gift): with db.auto\_commit():

drift = Drift()

# drift.message = drift\_form.message.data drift\_form.populate\_obj(drift)

drift.gift\_id = current\_gift.id drift.requester\_id = current\_user.id

drift.requester\_nickname = current\_user.nickname drift.gifter\_nickname = current\_gift.user.nickname drift.gifter\_id = current\_gift.user.id

book = BookViewModel(current\_gift.book)

drift.book\_title = book.title drift.book\_author = book.author drift.book\_img = book.image drift.isbn = book.isbn

current\_user.beans -= 1

db.session.add(drift)

@web.route('/my/gifts') @login\_required

def my\_gifts():

uid = current\_user.id

gifts\_of\_mine = Gift.get\_user\_gifts(uid) isbn\_list = [gift.isbn for gift in gifts\_of\_mine]

wish\_count\_list = Gift.get\_wish\_counts(isbn\_list)

view\_model = MyTrades(gifts\_of\_mine, wish\_count\_list)

return render\_template('my\_gifts.html', gifts=view\_model.trades)

@web.route('/gifts/book/<isbn>') @login\_required

def save\_to\_gifts(isbn):

if current\_user.can\_save\_to\_list(isbn): # 事务

# rollback

with db.auto\_commit(): gift = Gift() gift.isbn = isbn

gift.uid = current\_user.id

current\_user.beans += current\_app.config['BEANS\_UPLOAD\_ONE\_BOOK'] db.session.add(gift)

else:

flash('这本书已添加至你的赠送清单或已存在于你的心愿清单，请不要重复添加') return redirect(url\_for('web.book\_detail', isbn=isbn))

@web.route('/gifts/<gid>/redraw') @login\_required

def redraw\_from\_gifts(gid):

gift = Gift.query.filter\_by(id=gid, launched=False).first\_or\_404() drift = Drift.query.filter\_by(

gift\_id=gid, pending=PendingStatus.Waiting).first() if drift:

flash('这个礼物正处于交易状态，请先前往鱼漂完成该交易')

else:

with db.auto\_commit():

current\_user.beans -= current\_app.config['BEANS\_UPLOAD\_ONE\_BOOK'] gift.delete()

return redirect(url\_for('web.my\_gifts'))

@web.route('/') def index():

recent\_gifts = Gift.recent()

books = [BookViewModel(gift.book) for gift in recent\_gifts] return render\_template('index.html', recent=books)

@web.route('/personal') @login\_required

def personal\_center():

return render\_template('personal.html', user=current\_user.summary)

def limit\_key\_prefix():

isbn = request.args['isbn'] uid = current\_user.id

return f"satisfy\_wish/{isbn}/{uid}"

@web.route('/my/wish') @login\_required

def my\_wish():

uid = current\_user.id

wishes\_of\_mine = Wish.get\_user\_wishes(uid) isbn\_list = [wish.isbn for wish in wishes\_of\_mine] gift\_count\_list = Wish.get\_gifts\_counts(isbn\_list)

view\_model = MyTrades(wishes\_of\_mine, gift\_count\_list)

return render\_template('my\_wish.html', wishes=view\_model.trades)

@web.route('/wish/book/<isbn>') @login\_required

def save\_to\_wish(isbn):

if current\_user.can\_save\_to\_list(isbn): with db.auto\_commit():

wish = Wish()

wish.uid = current\_user.id wish.isbn = isbn db.session.add(wish)

else:

flash('这本书已添加至你的赠送清单或已存在于你的心愿清单，请不要重复添加') return redirect(url\_for('web.book\_detail', isbn=isbn))

@web.route('/satisfy/wish/<int:wid>') @login\_required

def satisfy\_wish(wid):

wish = Wish.query.get\_or\_404(wid)

gift = Gift.query.filter\_by(uid=current\_user.id, isbn=wish.isbn).first() if not gift:

flash('你还没有上传此书，'

'请点击“加入到赠送清单”添加此书。添加前，请确保自己可以赠送此书')

else:

send\_mail(wish.user.email,

'有人想送你一本书', 'email/satisify\_wish.html', wish=wish, gift=gift)

flash('已向他/她发送了一封邮件，如果他/她愿意接受你的赠送，你将收到一个鱼漂') return redirect(url\_for('web.book\_detail', isbn=wish.isbn))

@web.route('/wish/book/<isbn>/redraw') @login\_required

def redraw\_from\_wish(isbn):

wish = Wish.query.filter\_by(isbn=isbn, launched=False).first\_or\_404() with db.auto\_commit():

wish.delete()

return redirect(url\_for('web.my\_wish'))

def satifiy\_with\_limited(): isbn = request.args['isbn']

flash('你已向他发送过赠送邀请，请不要频繁发送')

return redirect(url\_for('web.book\_detail', isbn=isbn)) class YuShuBook:

# 模型层 MVC M 层

isbn\_url = '<http://t.yushu.im/v2/book/isbn/>{}'

keyword\_url = '[http://t.yushu.im/v2/book/search?q=](http://t.yushu.im/v2/book/search?q){}&count={}&start={}'

def init (self): self.total = 0 self.books = []

def search\_by\_isbn(self, isbn):

url = self.isbn\_url.format(isbn) result = HTTP.get(url)

self. fill\_single(result)

def fill\_single(self, data): if data:

self.total = 1 self.books.append(data)

def fill\_collection(self, data): self.total = data['total'] self.books = data['books']

def search\_by\_keyword(self, keyword, page=1):

url = self.keyword\_url.format(keyword, current\_app.config['PER\_PAGE'],

self.calculate\_start(page))

result = HTTP.get(url)

self. fill\_collection(result)

def calculate\_start(self, page):

return (page - 1) \* current\_app.config['PER\_PAGE']

@property def first(self):

return self.books[0] if self.total >= 1 else None