|  |
| --- |
| S2023-B105 |
| 品读二手书交易平台 |
| 前台概要设计说明书 |

|  |
| --- |
| 乔轲 夏润成 王晗頔 李精文 王子骏  2023-5-9 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **《“品读”二手书交易平台前端概要设计说明书》** | | | | | |
| **编号** | S2023-QK105 | | | **版本** | 2.0 |
| **编制** | 乔轲 夏润成 王子骏 李精文 王晗頔 | **审核** | 乔轲 | **批准** | 乔轲 |
| **日期** | 2023.2.26 | **日期** | 2023.2.27 | **批准日期** | 2023.2.27 |

**前台概要设计说明**

**0 目的**

本概要设计说明书意在为读者便于后期前端页面做UI设计和页面之间跳转关系做概要说明，及其便于与前端相关数据库的搭建。是后期“详细设计说明书” 的编写依据。

预期的读者为：审核人员、小组成员及其指导老师。

* 1. 项目背景

项目名称：品读图书交易平台。  
项目代号：S2023-QK105

项目来源：小组成员根据 “多抓鱼”图书平台的发展为灵感，共同商讨得出。

客户名称：读书爱好者、

* 1. 参考文献

需求说明书，立项任务书，软件开发计划书。

* 1. 总体设计
     1. 需求规定：说明本系统的功能，性能要求。

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **功能名称** | **功能概要** | **性能要求** |
| 1 | 购买 | 加入购物车，提交订单 | 购物车到生成订单（排除订单量大的条件）的响应速度小于5s |
| 2 | 出售 | 筛选出售的书籍，提交订单 | 正确识别卖书库中用户勾选要卖的书籍，并且生成相应正确的订单，响应速度小于5s |
| 3 | 首页推送 | 平台展示日常推送，源于豆瓣接口 | 进入页面渲染推送内容，推送内容完整，点击模块进入豆瓣对应的内容。 |
| 4 | 推荐 | 随机展示图书，按豆瓣评分排序推荐 | 根据豆瓣接口进行图书展示，响应速度小于2s。 |
| 5 | 卖书库 | 添加、删除书库书目。查看卖书库，生成预出售订单 | 用户查看卖书库内书籍时，响应速度小于3s。  添加书籍时，与外部接口链接后返回书籍信息响应速度小于2s。 |
| 6 | 我的订单 | 查询订单状态 | 数据库应及时更新用户订单信息。 |
| 7 | 个人信息 | 查询、修改个人信息 | 个人信息更新后及时刷新页面及其更新数据库 |
| 8 | 登录注册 | 注册查询数据库是否有重复，登录需账号密码匹配。 | 登录及其注册的响应速度均需小于2s。 |
| 9 | 搜索 | 查询书目 | 相应速度小于3s。 |

* + 1. 运行环境：

软件系统运行环境：Windows 10，Windows11，MacOS

硬件环境：MacBook Air Apple M1 8GB

MacBook Pro (13-inch, 2017) 3.1 GHz 双核Intel Core i5 8 GB

小新13pro AMD Ryzen 7 4800U with Radeon Graphics 1.80 GHz

11th Gen Intel(R) Core(TM) i5-11300H @ 3.10GHz 3.11 GHz

Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz

* + 1. 本项目设计概念和处理流程

本项目采用C2B2C的设计理念，即用户可以购买图书也可以销售二手图书。具体而言，用户可以将图书销售给平台，也可以从平台购书。项目主要涉及的功能可以分为：用户买书、用户卖书。

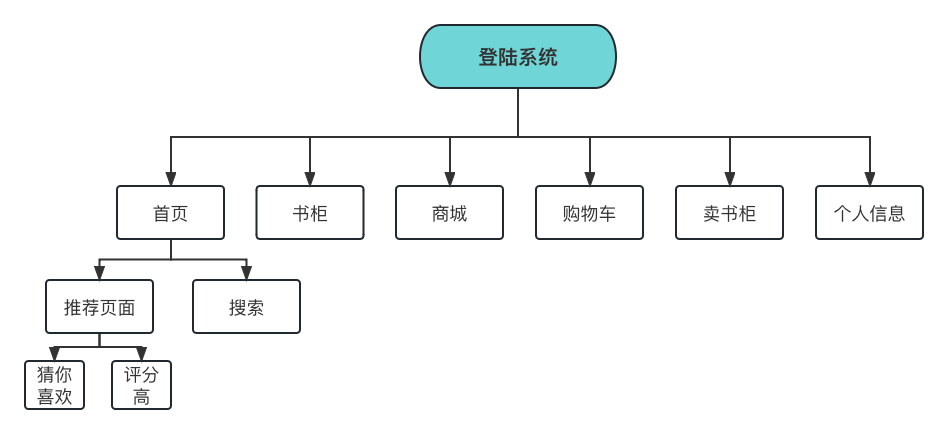
用户买书流程：

用户浏览图书→选图书品相→加入购物车→下单图书→确认订单→缴费

用户卖书流程：

用户录入欲售图书的ISBN→图书进入卖书库→用户选择卖书库内图书进行销售→下单→确认订单→确认账款

* + 1. 结构：对处理流程或功能模块形式化（表格化），说明每个系统元素的标识符和功能，分层次地给出各元素间的控制与被控制关系。

****

* + 1. 功能需求与程序的关系：采用矩阵图说明各项功能需求的实现同各块程序的分配关系：

|  |  |  |
| --- | --- | --- |
| 功能 | 用户 | 系统管理员 |
| 购买 | True | True |
| 出售 | True | True |
| 推荐 | True | True |
| 卖书库 | True | True |
| 我的订单 | True | True |
| 个人信息修改 | True | True |
| 登录注册 | True | True |
| 搜索 | True | True |
| 书籍折旧规则 |  | True |
| 书籍信息更新 |  | True |
| 更新图书库存 |  | True |
| 管理员账号注册 |  | True |

* 1. 接口设计

4.1公共接口

（1）用户的登录与注册

（2）管理员的登录

4.2外部接口：

pymysql：Django链接mysql的接口

ISBN接口：接口地址：https://api.gugudata.com/text/isbn

豆瓣接口：获取豆瓣图书信息

4.3内部接口：

urlpatterns = [

path('admin/', admin.site.urls),

# people

path('api/userlist/', people.views.UserList.as\_view()),

path('api/userregister/', people.views.UserRegister.as\_view()),

path('api/userlogin/', people.views.UserLogin.as\_view()),

path('api/useraddress/', people.views.UserAddressListCreate.as\_view()),

# 下面的url：pk=uid并发送get，返回此人的所有记录；pk=useraddressid并发送patch,delete，修改、删除该条记录

path('api/useraddress/<int:pk>/', people.views.UserAddressRetrieveUpdateDestroy.as\_view()),

path('api/userdetail/<int:pk>/', people.views.UserDetail.as\_view()),

path('api/userchangepassword/', people.views.UserChangePw().as\_view()),

path('api/userverifypw/', people.views.UserPwVerify.as\_view()),

path('api/adminlist/', people.views.AdminListCreate.as\_view()),

path('api/adminlogin/', people.views.AdminLogin.as\_view()),

# book

path('api/book/', book.views.BookListCreate.as\_view()),

path('api/book/<int:isbn>/', book.views.BookRetrieveUpdateDestroy.as\_view()),

path('api/bookisflow/', book.views.BookIsFlowList.as\_view()),

# shop的精髓之处

path('api/bookcatepresspricesearch/<str:pk>/<str:qk>/<str:ok>/<str:rk>/<str:sk>/',

book.views.BookRetrieveByCatePressPriceSearch.as\_view()),

path('api/bookcate/<str:pk>/<str:qk>/<str:ok>/<str:rk>/<str:sk>/',

book.views.BookCategoryListByCatePressPriceSearch.as\_view()),

path('api/bookpress/<str:pk>/<str:qk>/<str:ok>/<str:rk>/<str:sk>/',

book.views.BookPresshouseListByCatePressPriceSearch.as\_view()),

# conditionrate

path('api/conditionrate/', book.views.ConditionRateListCreate.as\_view()),

path('api/conditionrate/<int:pk>/', book.views.ConditionRateRetreiveUpdateDelete.as\_view()),

# inventory

path('api/inventory/', book.views.InventoryListCreate.as\_view()),

path('api/inventory/<int:pk>/', book.views.InventoryRetrieveUpdateDestroy.as\_view()),

path('api/inventorydetail/<int:pk>/', book.views.InventoryDetailRetrieve.as\_view()),

# cart

path('api/cart/', buy.views.CartListCreate.as\_view()),

# get方法时pk为uid，patch、delete方法时pk为自增id

path('api/cart/<int:pk>/', buy.views.CartRetrieveUpdateDestory.as\_view()),

# 购物车减少一件

path('api/cartminus/', buy.views.CartMinus.as\_view()),

# 获得一个用户的一个inventory的数量

path('api/cartinventorynum/<int:pk>/<int:qk>/<str:ok>/', buy.views.CartInventoryNumRetrieve.as\_view()),

# sell

path('api/warehouse/', sell.views.WarehouseListCreate.as\_view()),

path('api/warehouse/<int:pk>/', sell.views.WarehouseRetrieveDestroy.as\_view()),

path('api/warehouse/uid=<int:pk>/', sell.views.WarehouseOfUserRetrieve.as\_view()),

# buy\_order

path('api/buyorder/', buy.views.BuyOrderListCreate.as\_view()),

path('api/buyorder/<int:pk>/', buy.views.BuyOrderRetrieveUpdateDelete.as\_view()),

# buyorderdetail

path('api/buyorderdetail/', buy.views.BuyOrderDetailListCreate.as\_view()),

path('api/buyorderdetail/<int:pk>/', buy.views.BuyOrderDetailRetrieveUpdateDestory.as\_view()),

# sellorder

path('api/sellorder/', sell.views.SellOrderListCreate.as\_view()),

path('api/sellorder/<int:pk>/', sell.views.SellOrderRetrieveUpdateDestroy.as\_view()),

path('api/sellorder/status=<int:pk>/', sell.views.SellOrderStatusList.as\_view()),

# sellorderdetail

path('api/sellorderdetail/', sell.views.SellOrderDetailListCreate.as\_view()),

path('api/sellorderdetail/<int:pk>/', sell.views.SellOrderDetailRetrieveUpdateDestroy.as\_view()),

path('api/sellorderdetailpatch/<int:pk>/', sell.views.SellOrderDetailPatch.as\_view()),

# feedback 这两个是for user

path('api/feedback/', feedback.views.FeedbackListCreate.as\_view()),

# # GET方法时pk=uid，返回此用户的所有历史问题，包括问题和回复及处理状态

path('api/feedback/<int:pk>/', feedback.views.FeedbackRetrieveUpdateDestroy.as\_view()),

path('api/feedbackpending/', feedback.views.FeedbackListPending.as\_view()),

path('api/feedbackpendingfromasset/', feedback.views.FeedbackListPendingFromAsset.as\_view()),

path('api/feedbackfromaftersale/', feedback.views.FeedbackListFromAftersale.as\_view()),

path('api/feedbackpendingfromaftersale/', feedback.views.FeedbackListPendingFromAftersale.as\_view())

* 1. 系统数据结构设计

图示, 示意图

描述已自动生成

* + 1. 逻辑结构设计：

电子商务网数据存放在SQL Server数据库表中，本系统用到的数据库1个，数据表14个，数据库名称为DP。（注：下述数据表中字段带\*且标黄的为主键，字段带$且标绿的为外键。）

**1. people**

**1.1 user（用户）**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*u\_id | \*用户码 | int | 非空 |
| u\_name | 用户名 | Varchar(100) | 非空 |
| u\_pwd | 密码(SHA256加密) | Varchar(256) | 非空 |
| u\_ma | 余额 | float | 非空 |
| u\_birth | 生日 | Varchar(10) |  |
| u\_gender | 性别 | Varchar(2) |  |

**1.2 useraddress（用户地址）**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*u\_id | \*用户码 | int | 非空 |
| receiver\_name | 用户名 | Varchar(100) | 非空 |
| receiver\_phone | 收货人电话 | Varchar(20) | 非空 |
| receiver\_region | 收货人区域 | Varchar(10) | 非空 |
| receiver\_place | 收货人地址 | Varchar(100) | 非空 |

**1.3 administrator（管理员）**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*a\_id | \*管理员id | int | 非空 |
| a\_name | 管理员名 | varchar(20) | 非空 |
| a\_pwd | 管理员密码 | varchar(20) | 非空 |
| a\_level | 管理员权限(0为超级管理员；1为图书管理员；2为资金专员；3为售后人员；4为购售专员；5为物流对接人员) | int | 非空 |
| a\_gender | 性别 | varchar(2) |  |
| a\_birth | 生日 | varchar(10) |  |
| a\_content | 简介 | varchar(1000) |  |
| a\_phone | 手机 | int |  |

**2. feedback（售后反馈表）**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 解释 | 类型 | 非空否 |
| fba2u\_id | 表记录的主键 | int | 非空 |
| $u\_id | 用户id | int | 非空 |
| $as\_id | 售后人员id | int | 非空 |
| $as\_id | 售后人员id | int | 非空 |
| $a\_id | 资金人员id | int | 非空 |
| fba2u\_condition | 处理进度：  0:未处理,  1:售后反馈资金,  2:资金反馈售后,  3:售后反馈用户 | enum(0,1,2,3) | 非空 |
| fba2u\_content\_u2as | 用户问题 | varchar(300) | 非空 |
| fba2u\_content\_as2u | 售后给用户反馈内容 | varchar(300) | 非空 |
| fba2u\_content\_as2a | 售后给资金专员的问题 | varchar(300) | 非空 |
| fba2u\_content\_a2as | 资金人员给售后人员的反馈 | varchar(300) | 非空 |

**3. book**

**3.1 book（图书库）**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*b\_isbn | ISBN | Varchar(100) | 非空 |
| b\_name | 书名 | Varchar(100) | 非空 |
| b\_writer | 作者 | Varchar(20) | 非空 |
| b\_intro | 简介 | Varchar(1000) |  |
| b\_category | 类别 | Varchar(20) | 非空 |
| b\_presshouse | 出版社 | Varchar(50) | 非空 |
| b\_pressdate | 出版时间 | Varchar(7) |  |
| b\_price | 官方价格 | float | 非空 |
| b\_flow | 流通状态 | enum(0,1) | 非空 |

**3.2 inventory(库存)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*i\_id | 库存图书id | int | 非空 |
| $b\_ISBN | ISBN | Varchar(100) | 非空 |
| $cr\_condition | 品相 | Varchar(20) | 非空 |
| I\_book\_num | 数量 | Int | 非空 |

**3.3 mane\_regulation (图书定价规则)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*cr\_id | 自增主键 | int | 非空 |
| cr\_condition | 品相 | Varchar(20) | 非空 |
| cr\_rate | 买书折旧率 | Float | 非空 |
| S\_rate | 卖书折旧率 | float | 非空 |

**4. buy(购物相关)**

**4.1 cart(购物车)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*c\_id | \*购物车码 | int | 非空 |
| $uid | $用户id | int | 非空 |
| c\_book\_num | 图书数量 | int | 非空 |
| $i\_id | 库存图书id | Int | 非空 |

**4.2 buy\_order(购书订单)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*bo\_id | 买书订单号 | int | 非空 |
| $uid | $用户id | int | 非空 |
| Bo\_status | 订单状态(0-待处理，1-已发货，2-未审核,3已删除,4已完成) | int | 非空 |
| Bo\_submit\_time | 订单提交时间 | Datetime | 非空 |
| Bo\_finish\_time | 订单完成时间 | Datetime | 非空 |

**4.3 buy\_order\_detail(购书详细表)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*bod\_id | 主键 | int | 非空 |
| $bo\_id | ISBN | Varchar(100) | 非空 |
| $I\_id | 库存图书id | int | 非空 |
| Bod\_book\_num | 购书数量 | Int | 非空 |
| Bod\_order\_book\_price | 价格 | Float | 非空 |

**5. sell（卖书相关）**

**5.1 warehouse(卖书库)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*w\_id | \*卖书码 | int | 非空 |
| $u\_id | $用户id | int | 非空 |
| $B\_isbn | ISBN | Varchar(20) | 非空 |
| w\_customer\_condition | 用户选的图书品相 | Varchar(20) | 非空 |

**5.2 sell\_order(卖书订单)**

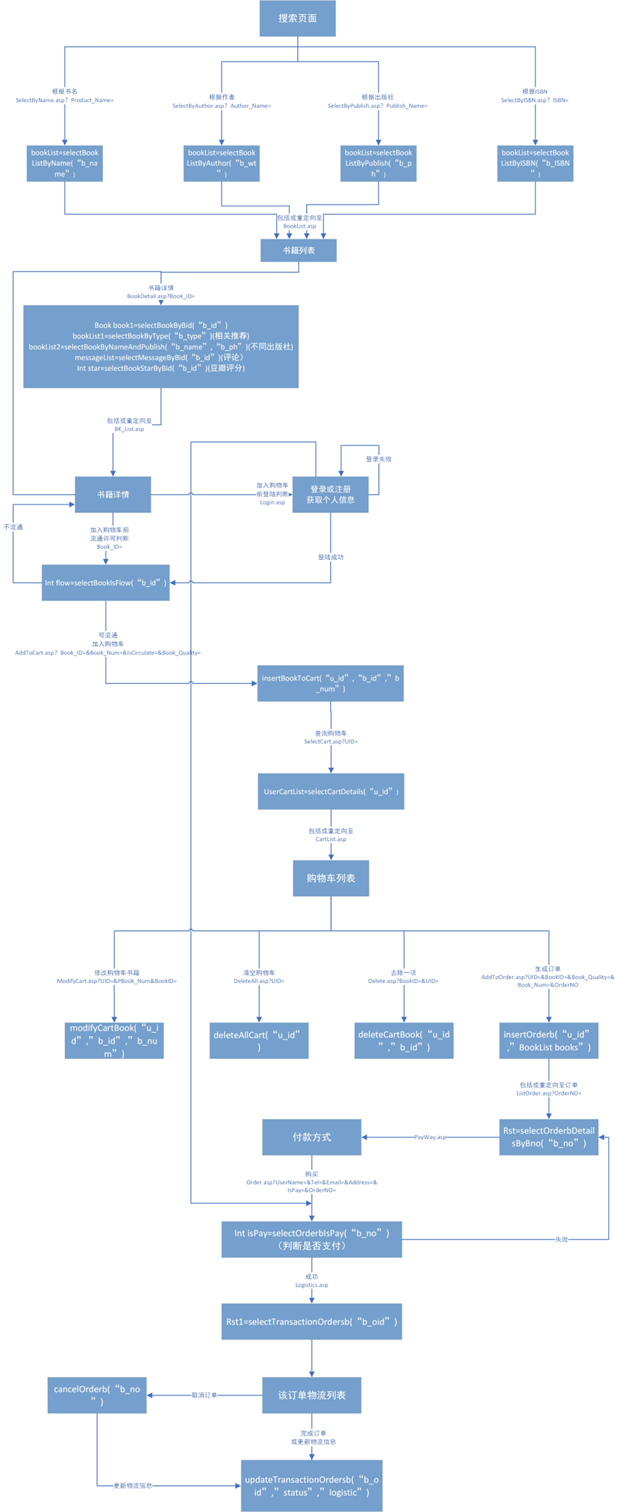
|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*so\_id | 卖书订单号 | int | 非空 |
| $u\_id | $用户id | int | 非空 |
| So\_status | 订单状态(0-待审核；1-已审核) | int | 非空 |

**5.3 sell\_order\_detail**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **解释** | **类型** | **非空否** |
| \*sod\_id | 主键 | int | 非空 |
| $so\_id | ISBN | Varchar(100) | 非空 |
| $b\_isbn | isbn | Varchar(20) | 非空 |
| W\_customer\_condition | 用户定的品相 | Varchar(20) | 非空 |
| Sod\_final\_condition | 购售定的品相 | Varchar(20) | 非空 |
| Sod\_predict\_prize | 预期价格 | Float | 非空 |
| Sod\_final\_prize | 最终价格 | Float | 非空 |

预期价格：标价x比率(s\_rate)

6.2 数据结构与程序的关系：说明各个数据结构与访问这些数据结构的各个程序之间的对应关系。



* 1. 系统出错（异常）处理

1. def selectBookListByName(“b\_name”)

功能：根据书名查询相关书籍

参数：b\_name 用户检索的书籍名字

返回：0 成功 （返回BookList）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectBookListByAuthor(“b\_wt”)

功能：根据作者查询相关书籍

参数：b\_wt 用户检索的书籍作者

返回：0 成功 （返回BookList）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectBookListByPublish(“b\_ph”)

功能：根据出版社查询相关书籍

参数：b\_ph 用户检索的书籍出版社

返回：0 成功 （返回BookList）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectBookListByISBN(“b\_ISBN”)

功能：根据ISBN号查询相关书籍

参数：b\_ISBN 用户检索的书籍ISBN号

返回：0 成功 （返回BookList）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectBookByBid(“b\_id”)

功能：根据b\_id查询具体书籍

参数：b\_id 图书数据库中每类书唯一性字段

返回：0 成功 （返回Book类）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectBookByType(“b\_type”)

功能：根据b\_type查询与之相同类型的图书

参数：b\_type 该书的类型

返回：0 成功 （返回书籍List）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectBookByNameAndPublish(“b\_name”,”b\_ph”)

功能：查询该图书的不同出版社书籍

参数：b\_name 图书名

b\_ph 图书出版社

返回：0 成功 （返回图书List）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectMessageByBid(“b\_id”)

功能：查询该图书的评论

参数：b\_id 某一类图书唯一性字段

返回：0 成功 （返回评论的List）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectBookStarByBid(“b\_id”)

功能：查询该书的豆瓣评分

参数：b\_id 某一类图书唯一性字段

返回：0 成功 （返回图书List）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectBookIsFlow(“b\_id”)

功能：判断图书是否流通

参数：b\_id 某一类图书唯一性字段

返回：0 图书可流通

-1 图书不可流通

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def insertBookToCart(“u\_id”,“b\_id”，”b\_num”)

功能：将图书加入到购物车中

参数：b\_id 某一类图书唯一性字段

u\_id 用户唯一性字段

b\_num 图书数量

返回：0 成功插入

-1 购物车已满

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectCartDetails(“u\_id”)

功能：查询该用户购物车中商品

参数：u\_id 用户唯一性字段

返回：0 成功 （返回购物车List）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def modifyCartBook(“u\_id”,”b\_id”,”b\_num”)

功能：修改购物车书籍信息

参数：u\_id 用户唯一性字段

b\_id 某一类图书唯一性字段

b\_num 购物车图书数量

返回：0 成功 （修改成功）

-1 修改失败（库存不足）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def deleteAllCart(“u\_id”)

功能：清空购物车

参数：u\_id 用户唯一性字段

返回：0 成功 （删除成功）

-1 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def deleteCartBook(“u\_id”,”b\_id”)

功能：删除购物车某一书籍

参数：u\_id 用户唯一性字段

b\_id 某一类图书唯一性字段

返回：0 成功 （修改成功）

-1 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def insertOrderb(“u\_id”,”BookList books”)

功能：生成订单

参数：u\_id 用户唯一性字段

Books 图书列表

返回：0 成功 （插入成功）

-1 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectOrderbDetailsByBno(“b\_no”)

功能：根据b\_no查询订单

参数： b\_no 订单唯一性字段

返回：0 成功 （返回订单Rst）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectOrderbIsPay(“b\_no”)

功能：判断订单是否支付

参数：b\_id 订单唯一性字段

返回：0 订单支付成功

-1 订单未支付成功

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def selectTransactionOrdersb(“b\_oid”)

功能：检索订单具体物流信息

参数：b\_oid 订单物流唯一性字段

返回：0 成功 （返回订单具体物流信息Rst）

-1 无此存储过程（即未检索到）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def cancelOrderb (“b\_no”)

功能：取消订单

参数：b\_no 订单唯一性字段

返回：0 删除成功

-1 删除失败（即已发货等不能取消订单情况）

-2 创建存储过程需要的参数时失败

>0 VID报告的错误

1. def updateTransactionOrdersb(“b\_oid”,”status”,”logistic”)

功能：修改订单物流信息

参数：b\_oid 订单物流唯一性字段

Status 订单完成状态

Logistic 订单物流状态

返回：0 修改成功

-1 创建存储过程需要的参数时失败

>0 VID报告的错误

**7. 界面设计**

**7.1 网站首页页面示意图**



**7.2 商店示意图**



**7.3 图书页面**



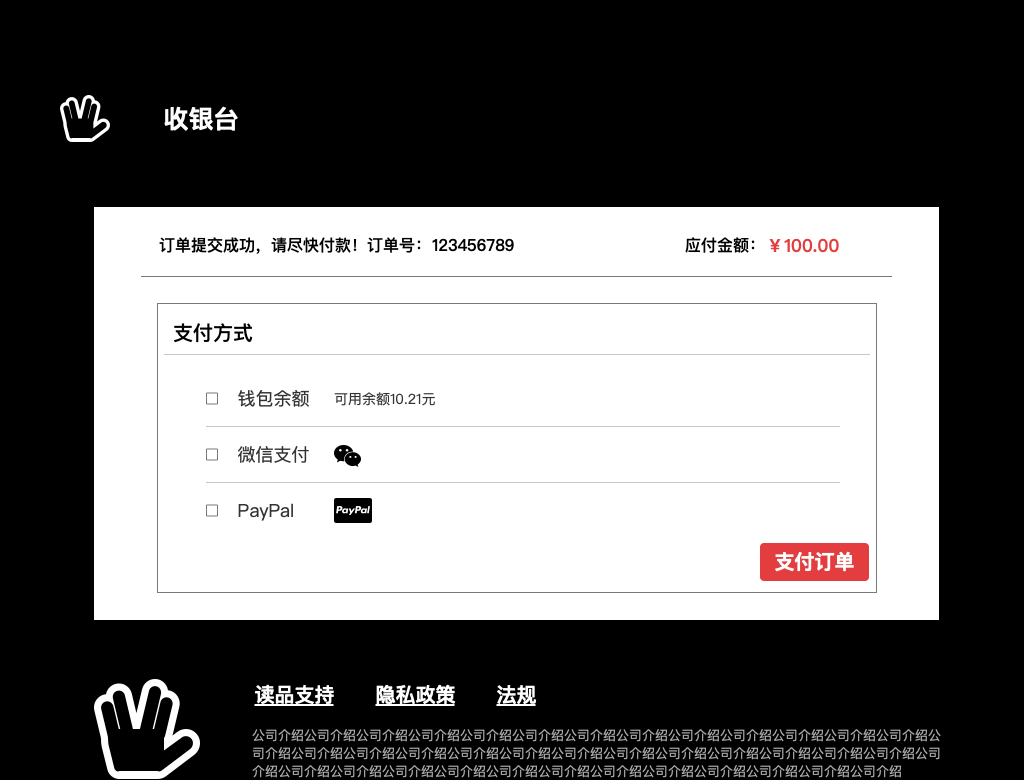
**7.4 书柜示意图**



**7.5 购物车示意图**



**7.6 购买账单页面**

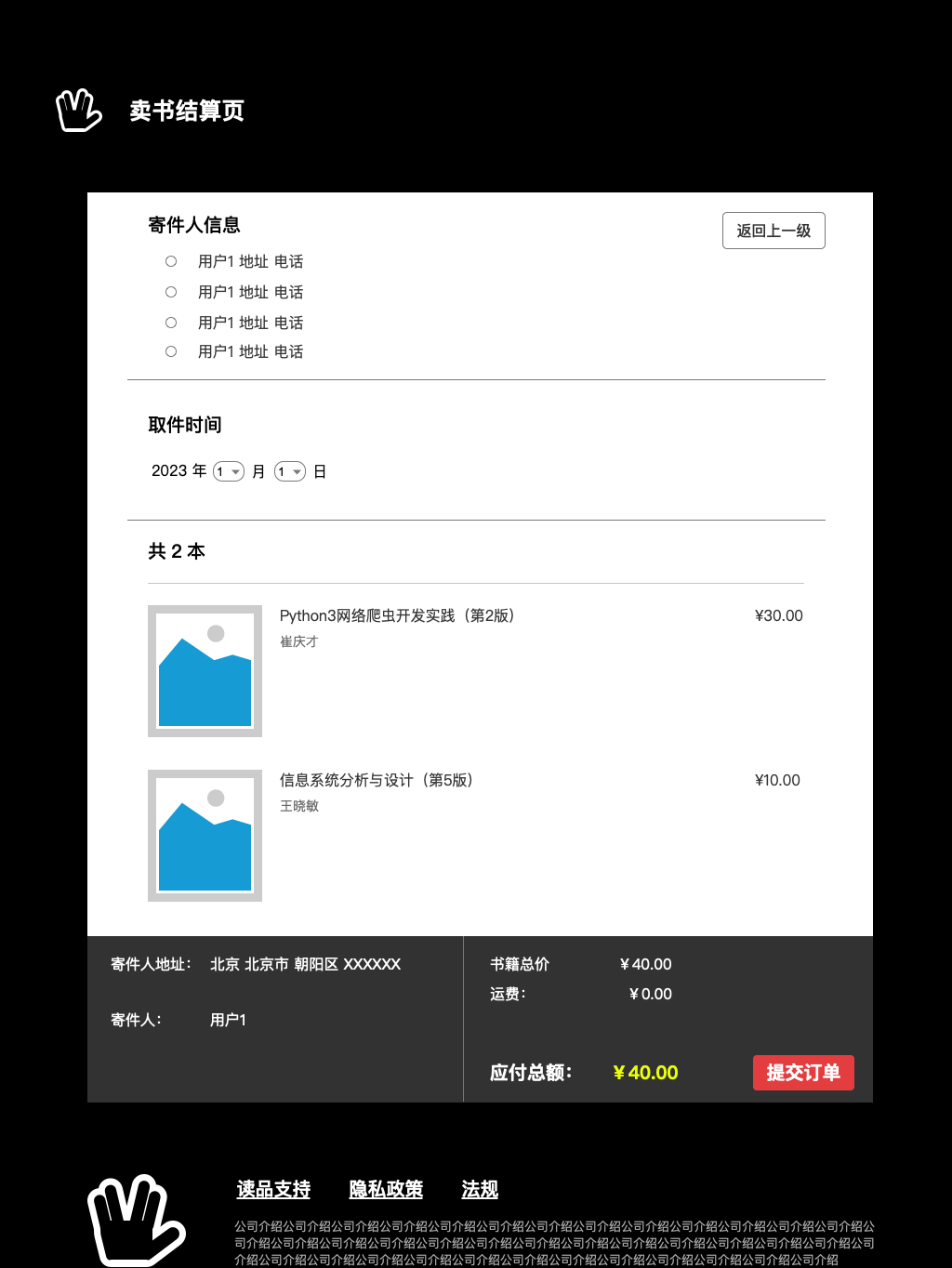




**7.7 卖书库示意图**

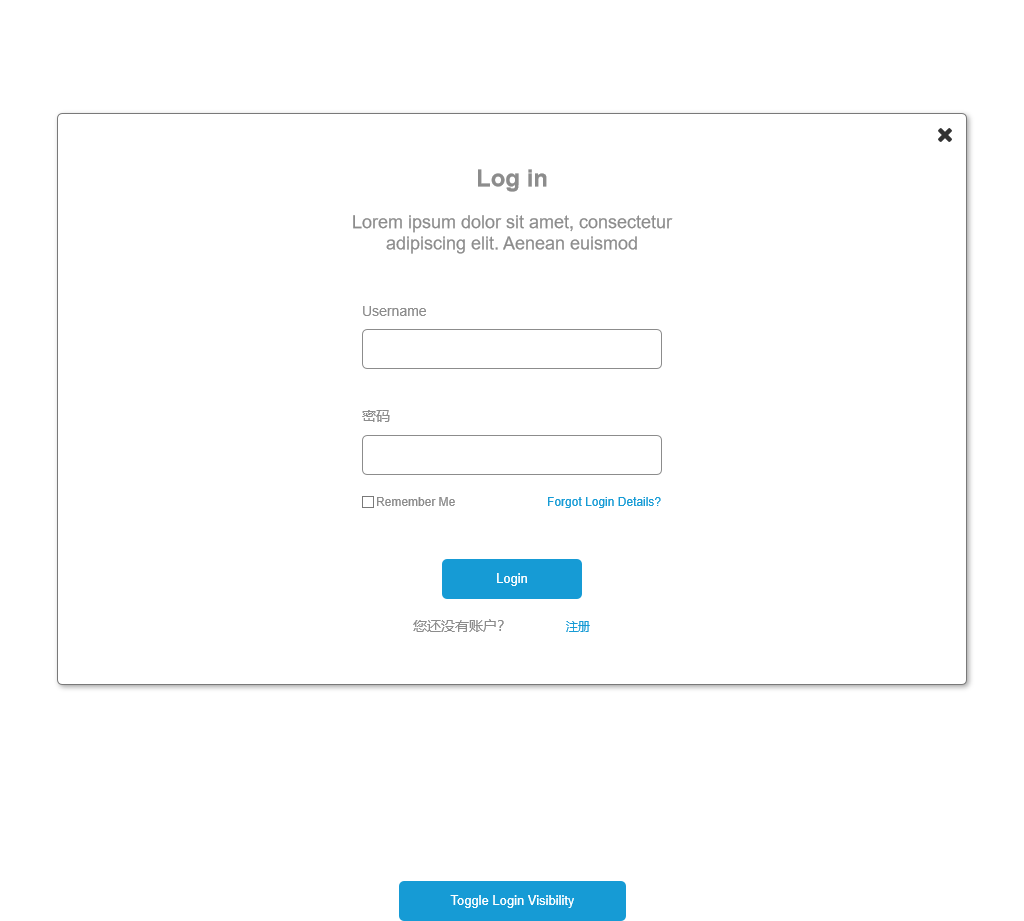


**7.8 卖书账单页面**





**7.9 用户登录**



**7.10 用户信息**

用户基本信息页面



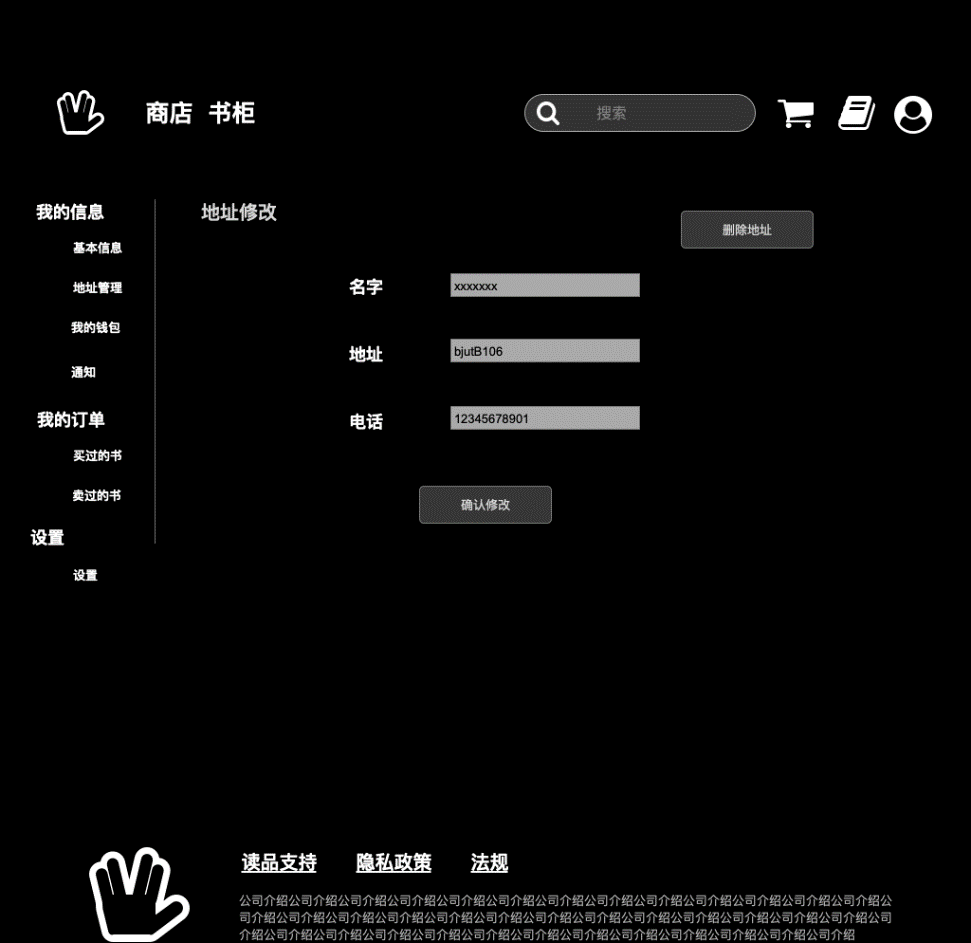
更改个人信息页面



地址管理页面



地址信息修改页面



我的钱包



买过的书订单页面



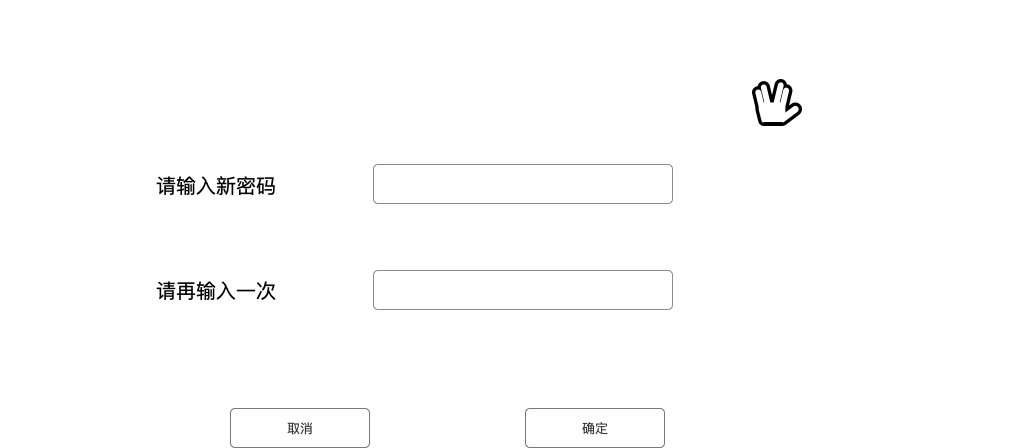
卖过的书订单页面



设置总页面



修改密码界面



修改密码失败



注销页面



其他设置页面



通知页面

