**小组编号：** **A08**

**项目名称： 静读书城**

**小组负责人： 吴扎西**

**小组成员： 龙波燚**

**范大荣**

**周兢**

**吴扎西**

《JavaWeb编程》

期末小组项目文档

2021-2022学年第一学期

目 录

[1 需求分析 1](#_Toc90556756)

[2 系统分析 9](#_Toc90556757)

[3 系统设计 19](#_Toc90556758)

[4 测试分析报告 23](#_Toc90556759)

# 需求分析

本章主要包含需求定义、用例图文档。

## 需求定义

### 需求概述

#### 背景

书籍是人类进步的阶梯，把书籍比喻成人类进步的阶梯，意思是说人类的进步离不开书籍，书籍为人类提供了很多的知识还有经验，所以我们一定要多读书，读好书。当今网络的发展形式，网络越来越深的渗入到人们的生活中，在网上购物已成为人们的一大乐趣,所以网上开店顺应大家的需求应运而生，虽电子书已成日常趋势，实体书城门店日益减少，但屏幕上闪烁着荧光的文字怎么也抵不过那一纸书页所带来的宁静与韵味。静读书城为传统书店与书籍爱好者之间架起了一道绿色桥梁。本系统的主要目的是实现图书的在线销售，实现书籍预定、结帐、查看订单等一系列功能，实现用户与书店的双赢局面。

#### 目标

系统的目标是为书虫服务—热爱实体书本，喜欢收藏名著，广阅古典现代书籍之人，主要通过开通网上书本售卖提供的方便之路使书虫能够通过网络，快速方便的与那些需要书本却又无法在当地新华书店等线下实体销售渠道购买的买家进行交流达成协议，进而形成供求关系长期下来是形成产业链，形成物资的充分流动与吸收吸引，并且省去了书虫许多的寻觅书籍的精力。

在很多城市、镇子之中、新华文轩等各种线下书本销售店里的各式书本都比较昂贵，根据近10年的实体书行情，这些实体商店的目标用户已经越来越少，很多的实体书店存在目前来看，有2种意义，一种是象征意义，一种是精神寄托。象征意义很简单，城市的脸面，文化的脸面。精神寄托，就是目前主流的方向，做书店人文综合体，然后靠关系吃补贴，弄投资，做成一份生意。

当你又需要某些书本却不知道到哪家书店去找，可能只能亲自或找人代去才能买到不然的话就要花大价钱，并且还不一定能找到所需要的。静读书城网站所提供的更加方便、更加简单的书本的查询查找；它让人们更加方便的获取自己想要的书籍，从而在这黄昏的市场中存活的更久。

#### 运行环境

1. 硬件环境 ：Inter i5-7300+4G内存+100G硬盘+百兆网卡+百兆局域网
2. 软件环境 ：Windows xp sp3/Windows7/Windows10
3. 支持环境 ：Tomcat 9.\*

#### 用户特点

喜欢阅读的人群，喜欢纸质书本，热爱收藏人群，高校学生

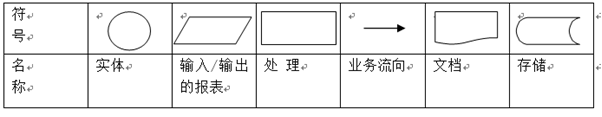
#### 约束条件

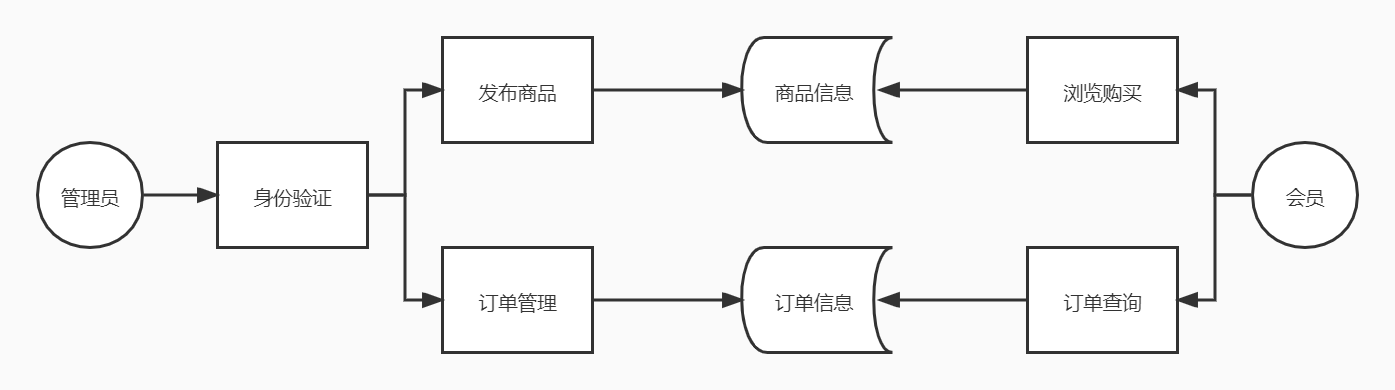
消费环境：目前，我国一些民众热衷于境外旅游和购物，说明了国内的消费环境还存在一些问题，制约了实体书本需求。首先，人们日常的生活被互联网手机所充斥，随着工作时间的增加，物价上涨，人们受迫仅能满足一些低级消费娱乐，诸如互联网短视频，阅文、起点等轻小说。这些都限制了实体书的销售。

阅读环境：据媒体报道，中国人年均读书0.7本，与韩国的人均7本、日本的40本、俄罗斯的55本相比，中国人的阅读量少得可怜。他说：“或许不应过分苛责。但我只是忧虑，如果就此疏远了灵魂，未来的中国可能会为此付出代价”，当前中国国民阅读量少，限制了阅读行业整体发展。

### 功能需求

#### 业务流程





### 非功能需求

响应时间：小于2s

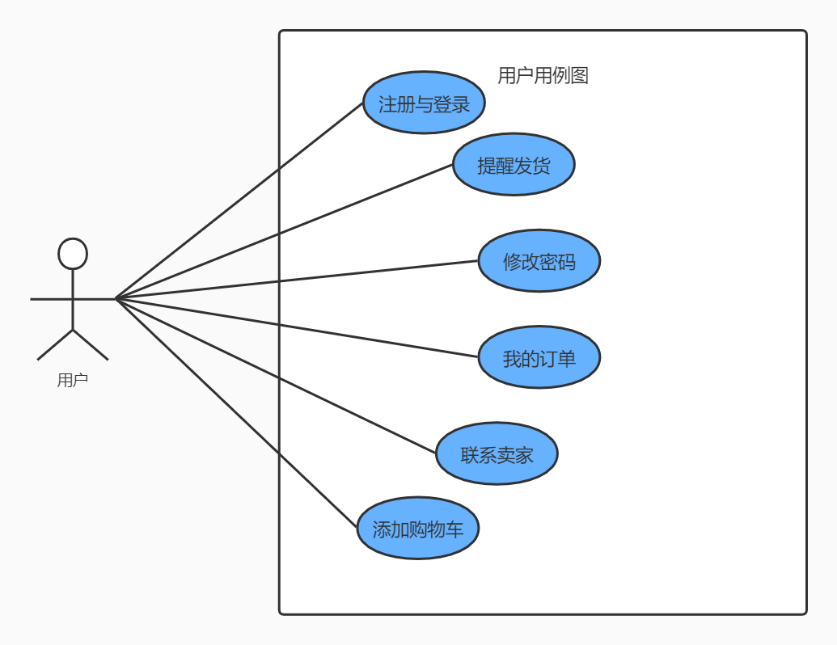
存储空间：100M

保密性：高

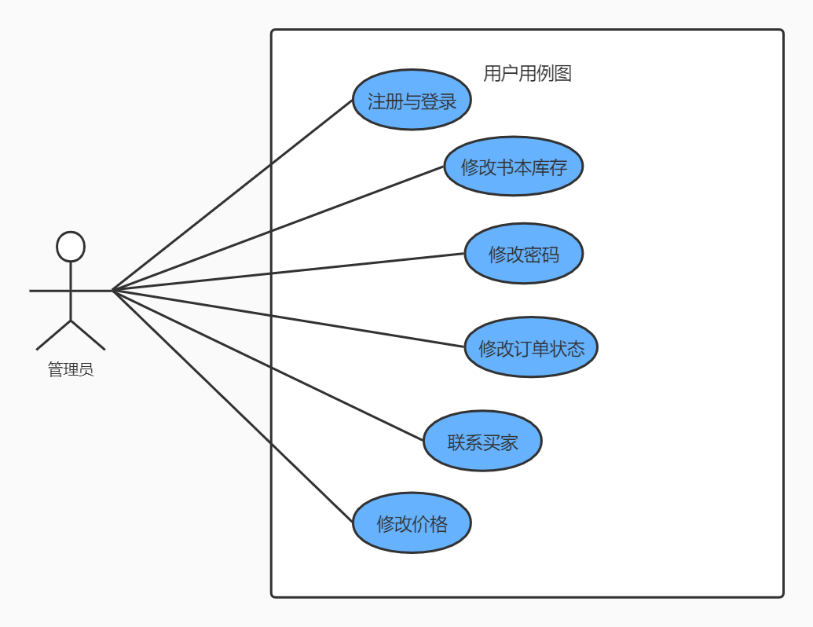
## 用例图文档

### 图形文档

1. 用户用例图



1. 管理员用例图



### 文字说明

#### 用例图综述

用户用例图主要为两大模块，个人信息、书城购买。

个人信息模块，用户在使用本系统时，首先需要注册一个账号，在注册界面中填写，用户名称、用户密码、电子邮件、验证码即可完成注册。同时用户可以在个人信息中修改自己的用户名以及密码。

书城购买模块，用户在主页面浏览书本，同时可以根据价格区间来筛选用户所需要的书籍。选好所需要的书本后，用户可以添加购物车，下单后商品转至订单模块，用户可以在我的订单中查询自己下单的书本。当对于物流以及售后出现问题，可以提醒卖家发货，或者直接联系卖家。

管理员用例图主要为两大模块，管理员信息、书城信息。

管理员信息模块，系统在初步建成时，有初始化超级管理员admin： 账号：admin 密码：admin，管理员首先登录，可以在个人的信息界面中修改密码。

书城信息模块，管理员可以根据库存数量，在后台管理中的图书馆里进行修改，诸如新增书籍，修改库存，删除书籍等操作。当用户在书城下单之后，管理员可以通过订单系统及时跟进当前的物流情况。同时也可以联系买家

#### 参与者描述

列出用例图中每个参与者的名称，可按字母顺序或其他某种有规律的次序排列。对参与者附有必要的文字说明。

参与者：

个人用户

管理员

#### 用例描述

对用例图中的每个用例进行详细说明。要按某种顺序排列用例。

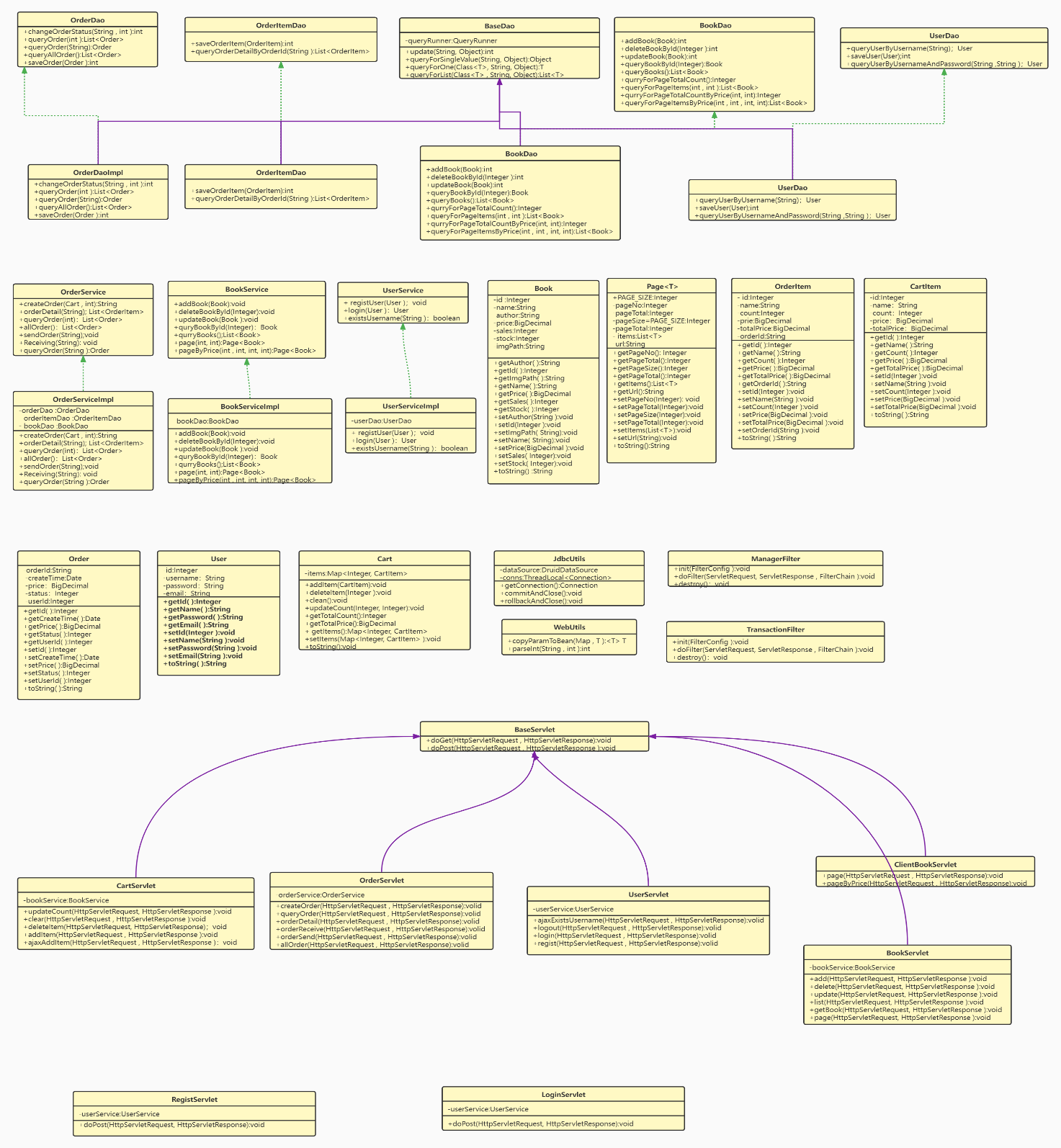
以下为用例描述示例。

|  |  |  |
| --- | --- | --- |
| **用例名称** | **【001】注册登录** | |
| **概述** | **用户在本系统注册账号** | |
| **参与者** | **用户** | |
| **基本流程** | **Actor** | **System** |
| 1. 用户点击主页面注册按钮  3. 用户输入对应的用户名称、用户密码、确认密码、电子邮箱、验证码，并按下注册  8. 用户在{注册成功}点击[转到主页]。 | 2.系统显示为注册界面  4 系统保存用户信息。  5 系统保存注册日期。  7 系统显示**{注册成功界面}，并提示“注册成功”**  9 系统显示{主界面} |
| **分支流程** | **no** | |
| **用例名称** | **【002】修改密码** | |
| **概述** | **用户在个人界面中修改密码** | |
| **参与者** | **用户** | |
| **基本流程** | **Actor** | **System** |
| 1 用户点击{主页面}上的[用户名称]。  3. 用户输入旧密码和新密码并提交请求。  8. 用户在{修改成功页面}点击[返回主页]。 | 2. 系统显示为用户个人信息的**{个人信息页面}。**  4 系统保存新密码信息。  5 系统修改密码的日期，保存日志。  6 系统显示**{修改成功界面}，并提示“密码修改成功！”**  9 系统显示{用户主界面} |
| **分支流程** | **no** | |
| **用例名称** | **【003】我的订单** | |
| **概述** | **用户进入订单界面，查询自己下单的商品** | |
| **参与者** | **用户** | |
| **基本流程** | **Actor** | **System** |
| 1 用户点击{用户主页面}上的[我的订单]。 | 2. 系统显示为用户购买书本信息的**{我的订单}。** |
| **分支流程** | **no** | |
| **用例名称** | **【004】提醒发货** | |
| **概述** | **用户在订单中的已下单商品界面提醒商家发货** | |
| **参与者** | **用户** | |
| **基本流程** | **Actor** | **System** |
| 1 用户点击{用户主页面}上的[我的订单]。  3. 用户选择任意一订单进行查看，点击{我的订单界面}上的{查看详情}。  5. 用户在{查看详情界面}点击[提醒发货]。 | 2. 系统显示为用户购买书本信息的**{我的订单}。**  4 系统显示当前书本商品的订单状态。  6 系统向商家发送关于发货的催促提示。  7 系统重新显示**{用户主界面}** |
| **分支流程** | **no** | |
| **用例名称** | **【005】联系卖家** | |
| **概述** | **用户在订单中的已下单商品界面拉起聊天界面联系商家** | |
| **参与者** | **用户** | |
| **基本流程** | **Actor** | **System** |
| 1 用户点击{用户主页面}上的[我的订单]。  3. 用户选择任意一订单进行查看，点击{我的订单界面}上的{查看详情}。  5. 用户在{查看详情界面}点击[提醒发货]。 | 2. 系统显示为用户购买书本信息的**{我的订单}。**  4 系统显示当前书本商品的订单状态。  6 系统向商家发送关于发货的催促提示。  7 系统显示**{与商家的聊天界面}** |
| **分支流程** | **no** | |
| **用例名称** | **【006】添加购物车** | |
| **概述** | **用户在商城选定商品之后，将其添加至购物车** | |
| **参与者** | **用户** | |
| **基本流程** | **Actor** | **System** |
| 1 用户点击{主页面}上的[加入购物车]。  3. 用户点击{主页面}上的[购物车]。  5. 用户在{购物车}点击[删除]。  9. 用户点击{购物车界面}的{清空购物车}按钮 | 2.系统提示为{您刚刚将xxx加入到了购物车中}  4 系统显示为{购物车界面}。  6 系统删除添加的书本。  8 系统保存用户与职位之间的关系。  10 系统将购物车所有商品删除  11 系统显示{用户主界面} |
| **分支流程** | **no** | |
| **用例名称** | **【007】注册登录** | |
| **概述** | **管理员在初始界面进行登录** | |
| **参与者** | **管理员** | |
| **基本流程** | **Actor** | **System** |
| 1. 管理员点击主页面{登录}按钮  3. 管理员输入对应的管理员名称、管理员密码并按下登录  6. 管理员在{登录成功}点击[转到主页]。 | 2.系统显示为登录界面  4 系统保存管理员信息。  5 系统显示**{登录成功界面}，并提示“登录成功”**  7 系统显示{主界面} |
| **分支流程** | **no** | |
| **用例名称** | **【008】修改书本信息** | |
| **概述** | **管理员在后台管理中修改书本的库存，种类等信息** | |
| **参与者** | **管理员** | |
| **基本流程** | **Actor** | **System** |
| 1 管理员点击{主页面}上的[后台管理]。  3.管理员点击{后台管理欢迎界面}上的{图书管理}。  5.管理员在{图书管理页面}点击[添加图书]。  7 管理员输入图书的名称、 价格 、作者、 销量 、库存 信息并点击{提交}按钮 | 2. 系统显示为管理员操作的**{后台管理欢迎界面}。**  4 系统显示{图书管理系统}界面。  6 系统显示{添加图书}界面  8 系统重新显示**{图书管理系统}界面，并提示“图书增添成功！”** |
| **分支流程** | **no** | |
| **用例名称** | **【009】修改密码** | |
| **概述** | **管理员在个人界面中修改密码** | |
| **参与者** | **管理员** | |
| **基本流程** | **Actor** | **System** |
| 1 用户点击{主页面}上的[用户名称]。  3. 用户输入旧密码和新密码并提交请求。  8. 用户在{修改成功页面}点击[返回主页]。 | 2. 系统显示为用户个人信息的**{个人信息页面}。**  4 系统保存新密码信息。  5 系统修改密码的日期，保存日志。  6 系统显示**{修改成功界面}，并提示“密码修改成功！”**  9 系统显示{用户主界面} |
| **分支流程** | **no** | |
| **用例名称** | **【010】修改订单状态** | |
| **概述** | **管理员在后台管理中修改买家的订单状态** | |
| **参与者** | **管理员** | |
| **基本流程** | **Actor** | **System** |
| 1 管理员点击{主页面}上的[后台管理]。  3.管理员点击{后台管理欢迎界面}上的{订单管理}。  5.管理员在{订单管理页面}点击[添加快递单号]。 | 2. 系统显示为管理员操作的**{后台管理欢迎界面}。**  4 系统显示{订单管理系统}界面。  6 系统显示{添加单号}界面  8 系统重新显示**{订单管理系统}界面，并提示“订单跟进完成！”** |
| **分支流程** | **no** | |
| **用例名称** | **【011】修改价格** | |
| **概述** | **管理员在后台系统中修改商品的价格** | |
| **参与者** | **管理员** | |
| **基本流程** | **Actor** | **System** |
| 1 管理员点击{主页面}上的[后台管理]。  3.管理员点击{后台管理欢迎界面}上的{图书管理}。  5.管理员在{图书管理页面}点击[修改]。  7 管理员输入图书的名称、 价格 、作者、 销量 、库存 信息并点击{提交}按钮 | 2. 系统显示为管理员操作的**{后台管理欢迎界面}。**  4 系统显示{图书管理系统}界面。  6 系统显示{修改图书}界面  8 系统重新显示**{图书管理系统}界面，并提示“图书修改成功！”** |
| **分支流程** | **no** | |

# 系统分析

## 类图文档

### 图形文档



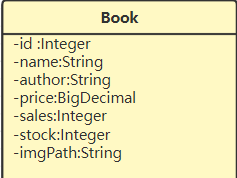
### 文字说明

#### 类图综述

在面向对象的建模中，建立类图是最基本的任务。对于整个系统而言，我们设计出了utils的工具类，其对应了1个类，JDBC类用于对数据库进行连接操作，提交和回滚事务的一个工具类。在控制层的逻辑之中我们设计出一个Base类来控制整个服务流程中解决post请求中中文乱码问题，并且解决相应过程中的中文乱码。在filter中过滤了是否是登录的用户，对于没有登录的用户，不能进行下一步的操作。在整个的业务流程中，我们都采用控制层的8个servlet类来进行整个流程的控制，请求转发，贴合MVC的开发模式，对于系统的用户者，不能看见JSP页面。在业务服务中，我们提供了对系统用户和图书操作的一切业务服务类。在DAO中提供的是业务事务处理后数据变化，相应在数据库变化之中。对于用户而言，有实体类User，对于用户登录及注册方面有相对应的control类分别是点击登录和注册、用户信息验证、验证是否是已登录和注册的用户。在系统管理员中，其业务逻辑其实与普通户用是一样的，只是在注册，在数据库中显示判断是否是管理员。

#### 类描述

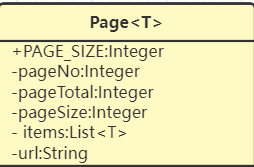
类的整体：Book类、该类是书籍的实体类、无主动性。



属性：id：（书籍的id号）、name : String（书籍的名字）、author :String（书籍的作者）、price :BigDecimal（书籍的价格）、sales : Integer（书籍的售价）、stock ：Integer（库存）、imgPath :String（数据图片存放的路径）。

服务：说明书籍这个实体的一切信息。描述这个实体的所有信息，并且设置这个实体属性的get和set方法。

类的整体：Page类、该类是书籍页面的实体类、无主动性。



属性：PAGE\_SIZE:Integer（页面信息条数

pageNo:Integer（页面数）

pageTotal:Integer（总的页面数）

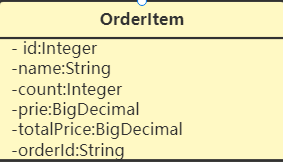
pageSize:Integer（页面大小）

items:List<T>（订单列表）

url:String（地址字符）

服务：主要负责每一个页面的大小和信息数目的定义和说明。

类的整体：OderItem类、该类中定义了订单项的实体类、无主动性。



属性：id:Integer（数据库中的ID号）

name:String（订单号的名字）

count:Integer（订单的数量）

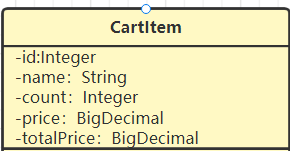
prie:BigDecimal（书籍的价格）

totalPrice:BigDecimal（该订单的总价格）

orderId:String （订单号的id值）

服务：该类主要是用来描述订单的信息。

类的整体：CartItem类：该类是描述购物商品项里面的属性定义，无主动性。



属性：id:Integer（购物车id）

name：String （购物车商品名字）

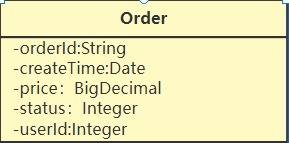
count：Integer （购物车数量）

price：BigDecimal（价格）

totalPrice：BigDecimal（总的价格）

服务：该类主要是用来描述购物车商品项属性。

类的整体：Order类、该类是订单的实体类。



属性：orderId:String （订单id）

createTime:Date（订单的创建时间）

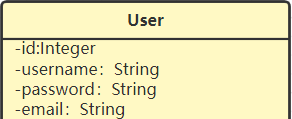
price：BigDecimal（订单的价格）

status：Integer （表示发货的状态）

userId:Integer（用户的id号）

服务：该类只要是用来描述订单里面的属性。

类的整体：User类，该类用来定义用户的信息，在注册和登录会用到其中定义的信息。



属性：-id:Integer （私有：用户的id）

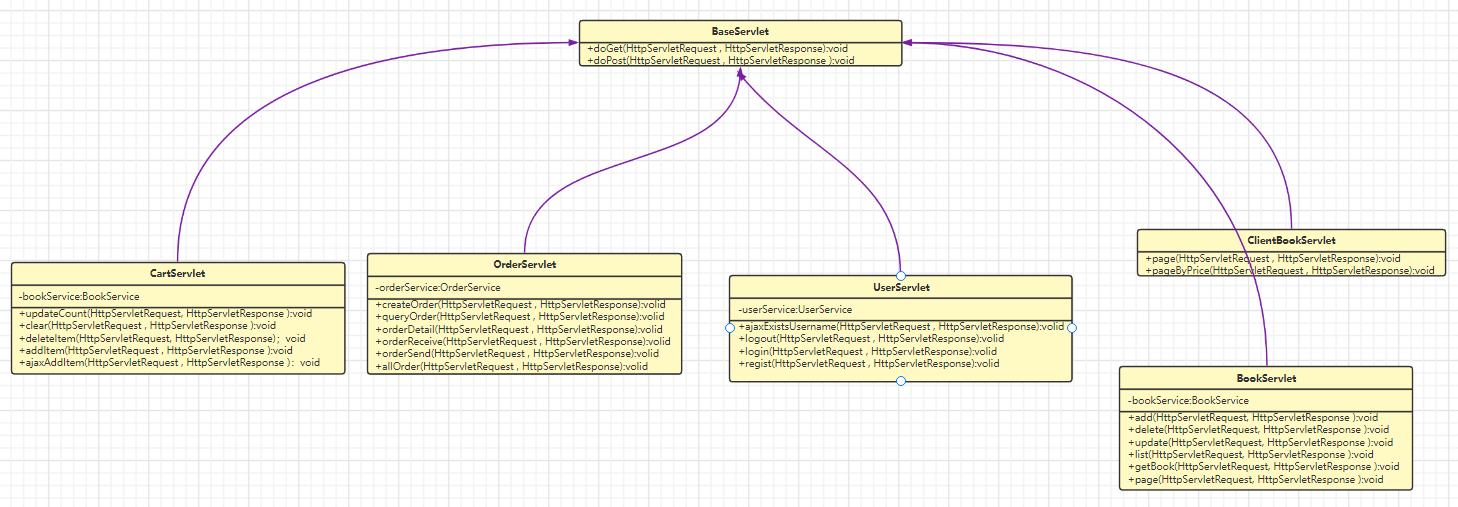
-username：String （私有：用户名字）

-password：String （私有：用户密码）

-email：String （私有：用户的邮件号）

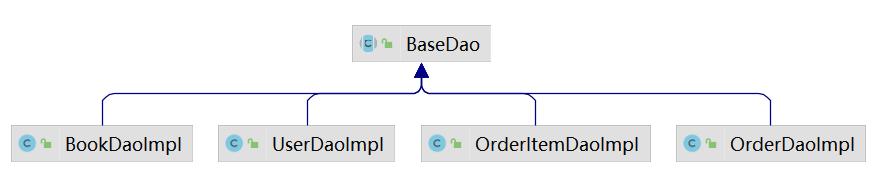
服务：定义了用户这个实体的参数及信息。

#### 关联描述

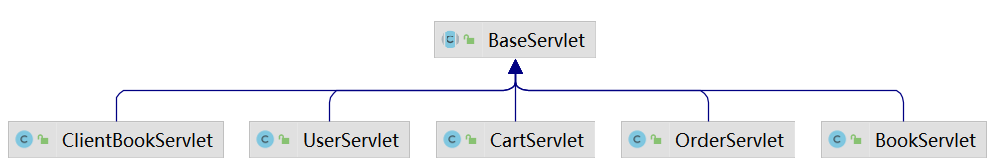


在上述这些类中在下面的类的实例化过程中，调用中都对BaseServlet中的实例进行了引用，所以是关联关系。在RegistServlet、LoginServlet、UserServlet这三个类中，他们都使用了UserService类中的实例，在登录注册的时候，使用的是用户的业务服务，这些类是存在关联关系。

#### 泛化描述

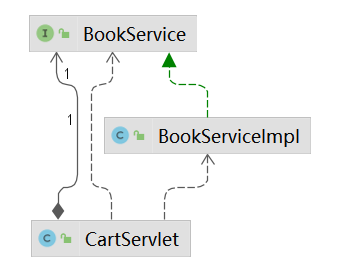


通过继承关系，我们得知类：BookDaoImpl、UserDaoImpl、OrderItemDaoImpl、OrderDaoImpl继承了类BaseDao，我们在这4个类中使用了BaseDao类的属性和一些方法，并且做出了在这个类中我们需要的一些功能。



同样的在上图所示的类中，我们的继承关系，在创建自己的类时，我们继承了我们的最基本的BaseServlet类。并且在下面的5个类中我们特化了父类BaseServlet的一些功能。

#### 依赖描述

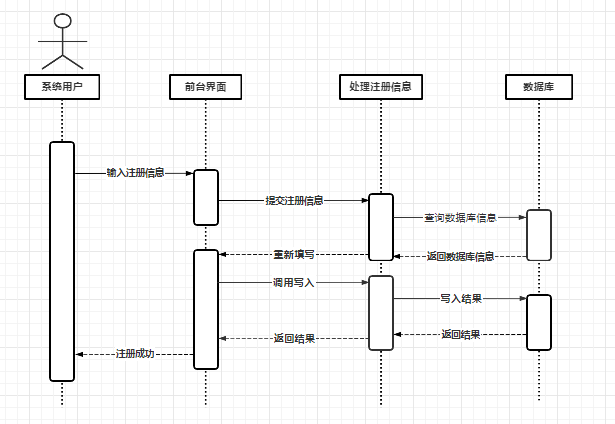


在我们进行创建订单的业务逻辑时，我们是需要使用书籍的业务处理的接口，并且在接口中去实现这个业务逻辑的处理，返回我们需要的结果。，

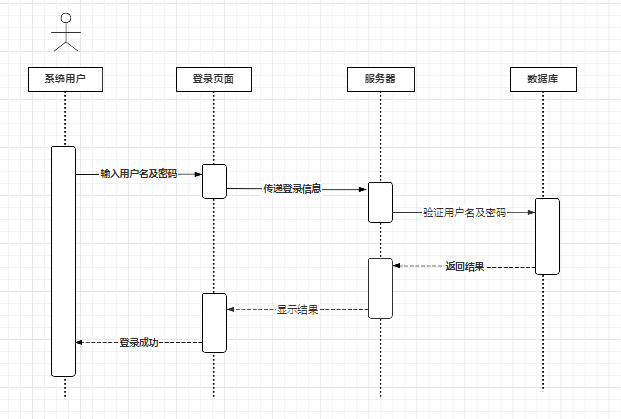
## 顺序图文档

### 图形文档

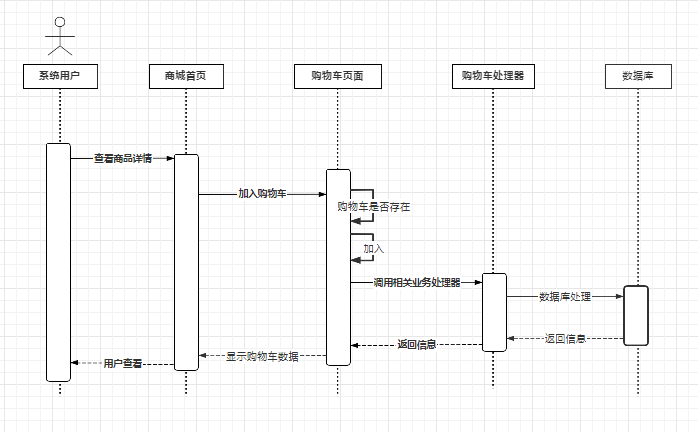
1.用户注册



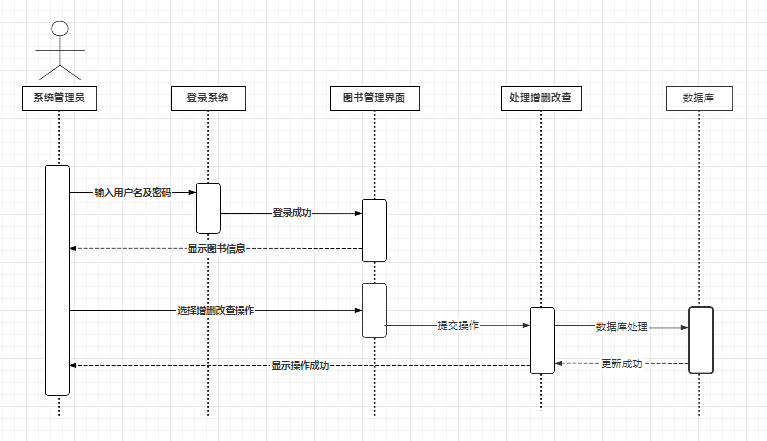
2.用户登录



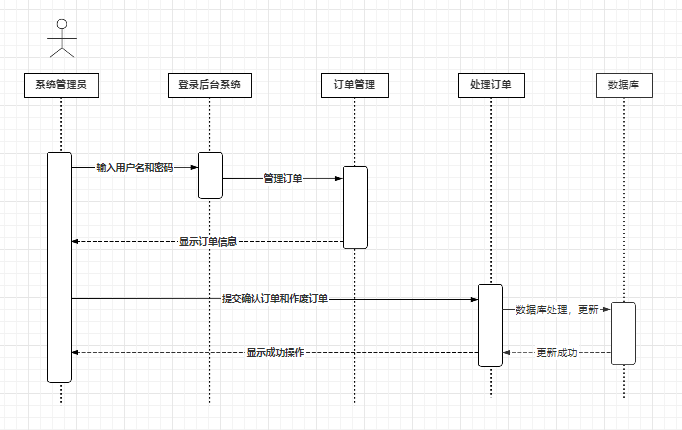
3.添加购物车



4.图书管理



5.订单管理



### 文字说明

#### 顺序图综述

从总体上描述该顺序图的目的，以及所涉及到对象和参与者。

1. **用户注册时序图：**

在本系统中，用户可以输入个人信息进行用户注册，符号要求的用户将被允许注册登录。

1. **用户登录时序图：**

成功注册的用户可以通过用户名及密码登录至本书城系统。

1. **添加购物车时序图：**

在本书城系统中用户可以在商城首页查看相关图书，并能完成添加购物车功能，系统会在购物车页面显示出用户添加进购物车的图书信息。

1. **图书管理时序图：**

在本书城系统中，系统管理员可以在图书管理界面对图书信息进行管理和维护（添加、删除、查询、修改）。

1. **订单管理时序图：**

在本书城系统中，系统管理员可以在订单管理界面对订单信息进行查询及处理，确定相关订单是否正常，是否已作废。

#### 顺序图中的对象与参与者描述

对顺序图中的所有对象和参与者，依次进行如下描述：

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 对象类型 | 对象名称 | 是否为主动对象  （参与者不描述此内容） |
| 1 | 人 | 系统用户/管理员 | 是 |
| 2 | 页面 | 前台界面 | 否 |
| 3 | 页面 | 注册页面 | 否 |
| 4 | 页面 | 登录页面 | 否 |
| 5 | 页面 | 商城首页 | 否 |
| 6 | 页面 | 购物车页面 | 否 |
| 7 | 页面 | 图书管理页面 | 否 |
| 8 | 页面 | 订单管理页面 | 否 |
| 9 | 页面 | 购物车处理器 | 否 |
| 10 | 页面 | 用户修改器 | 否 |
| 11 | 服务器 | 服务器端 | 否 |

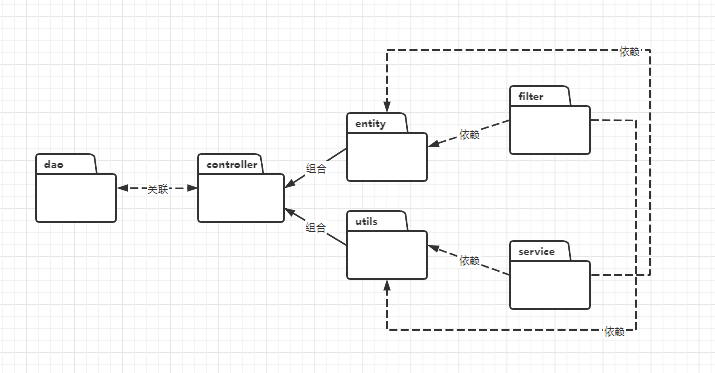
## 包图文档

### 图形文档

即所绘制的包图

### 文字说明

#### 包图综述



本项目共6个包，包括controller控制层、dao持久层、service业务层、entity模型层、filter过滤器和utils层。Dao层介于业务逻辑层和数据库之间,进行数据的访问和操作；service是业务层，负责完成数据库的调用；控制层controller，主要负责业务调度；模型层entity主要用于存放我们的实体类，与数据库中的属性值基本保持一致；utils层属于工具类层，通用的、与业务无关的，可以独立出来，供其他项目使用。

#### 包图中的包描述

对包图中的每个包进行如下描述：

1. 包的名称：controller、dao、service、entity、filter、utils
2. 包的种类：类包、用例包或其他
3. 与该包有关系的其他包：filter包与entity包、utils包的关系为依赖关系，由输入端指向输出端；service层的包类似；controller层与数据库dao包相关联，与entity层和utils层组合，且依赖于服务层service包

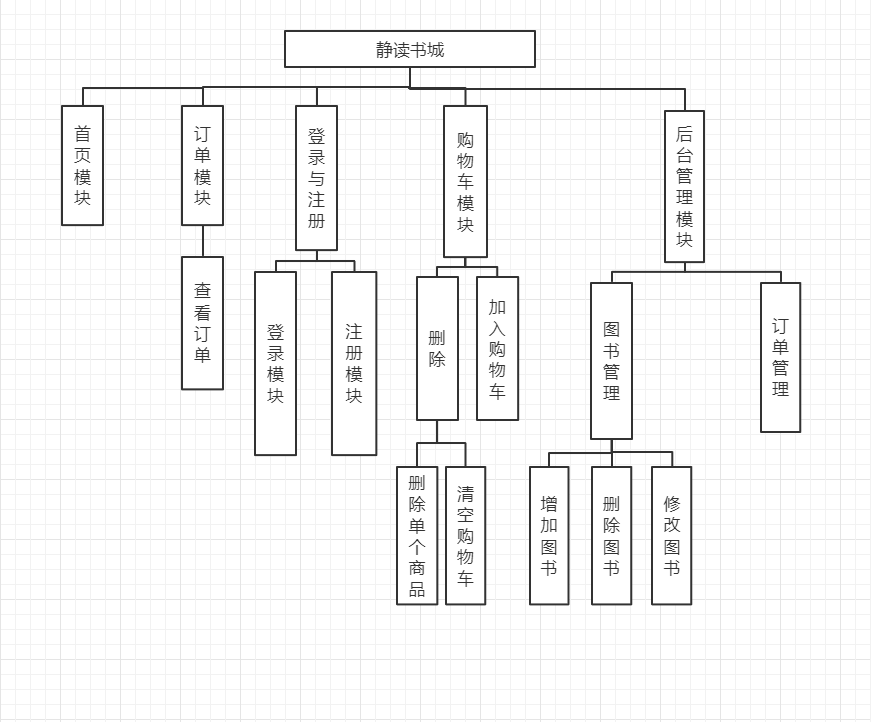
# 系统设计

本章主要包含系统体系结构设计、数据库设计、用户界面设计。

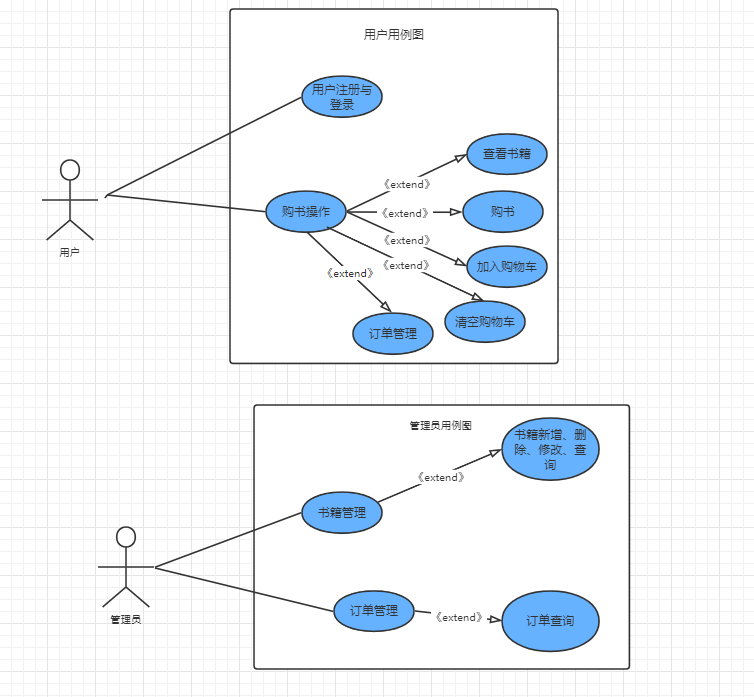
## 系统体系结构设计

### 程序（模块）划分

**即给出功能模块和层次图**



**用例图**



用户功能机构分析：

1. 登录模块：用户进行登录操作
2. 注册模块：用户进行注册操作
3. 购物车模块：用户可以将书籍加入购物车、删除购物车、结算购物车里面的商品、也可以清空购物车。
4. 订单模块：用户可以查看自己的订单，以及订单的状态（是否下单、付款）

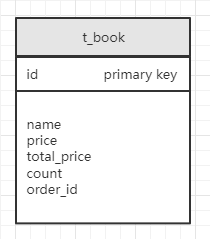
管理员功能结构分析：

1. 图书管理：增加图书，删除图书，修改图书信息
2. 订单管理：查看订单，对订单进行管理。

## 数据库设计

### 

t\_book:书籍信息表：



Id:书籍编号

Name：书籍名称

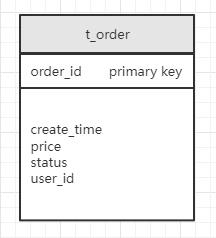
Price：书籍价格

Total\_price:书籍总价

Count：书籍数量

Order\_id:外键，订单号

### 



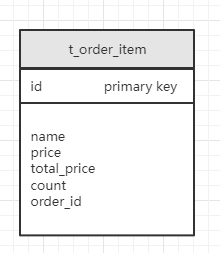
Order\_id:主键，订单编号

Create\_time:订单创建时间

Status：订单状态：是否付款

User\_id:外键，创建该订单的用户id

### 



Id：编号id

Name：订单里面的书籍名称

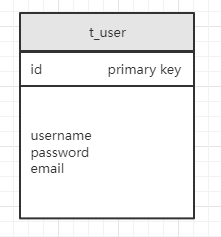
Price：订单里的书籍价格

Total\_price：订单总价格

Count：书籍的数量

Order\_id：订单id，外键

### 



Id：用户id，主键

Username：用户名

Email：用户的邮箱

## 用户界面设计

# 测试分析报告

## 引言

### 编写目的

本测试计划从属于静读书城系统。用户为广大的读者书友，用于日常购买图书。执行本测试前，应完成软件计划，需求分析，设计及编码工作。

### 项目背景

古人说，“书中自有颜如玉”，“书中自有黄金屋”。可见，书籍对于人类的进步有着巨大的意义。读史可以让我们深邃，读诗歌使人浪漫，读数理使人明智…….

多读书，是一种好习惯，也是人类进步离不开的手段。在那一部部经典的著作当中，完美可以学习先辈的聪明智慧，学习异邦的优秀文化，正所谓“运筹帷幄之中，决胜千里之外”，如果没有深厚的学识做基础，怎会有如此“胸有成竹”。

所以说，读书是大家读的一种生活必须，但是获取书籍需要通过一定的渠道，为了方便大家及时获得想要阅读的书籍，我们A08小组开发这个“静读书城”系统，提供一个在线购各类书籍的平台。我们将初心不改，只卖图书，争取让静读书城越来越好。

### 定义

1. 黑盒测试：黑盒测试，它是通过测试来检测每个功能是否都能正常使用。在测试中，把程序看作一个不能打开的黑盒子，在完全不考虑程序内部结构和内部特性的情况下，在程序接口进行测试，它只检查程序功能是否按照需求规格说明书的规定正常使用，程序是否能适当地接收输入数据而产生正确的输出信息。黑盒测试着眼于程序外部结构，不考虑内部逻辑结构，主要针对软件界面和软件功能进行测试。
2. 冒烟测试：这一术语源自硬件行业。对一个硬件或硬件组件进行更改或修复后，直接给设备加电。如果没有冒烟，则该组件就通过了测试。在软件中，“冒烟测试”这一术语描述的是在将代码更改嵌入到产品的源树中之前对这些更改进行验证的过程。在检查了代码后，冒烟测试是确定和修复软件缺陷的最经济有效的方法。冒烟测试设计用于确认代码中的更改会按预期运行，且不会破坏整个版本的稳定性。
3. 传统上所谓压力测试（stress testing）是指将整个金融机构或资产组合置于某一特定的（主观想象的）极端市场情况下，如假设利率骤升100个基本点，某一货币突然贬值30%，股价暴跌20%等异常的市场变化，然后测试该金融机构或资产组合在这些关键市场变量突变的压力下的表现状况，看是否能经受得起这种市场的突变。

### 参考资料

[**1**] 王琨。嵌入式计算机软件测试关键技术探讨[J].科技创新与应用，2016,（07）：87-88.

[**2**] 李明艳。基于移动互联网的软件测试技术研究及应用[J].电子技术与软件工程，2016,（04）：42.

[**2**] 李翘婕。基于反馈控制的软件测试方法研究与实现[D].昆明理工大学，2016.

[**3**] 丁森科。基于PetriNet软件测试系统的研究[D].浙江大学，2016.

[**4**] 陈佳丽，陈晓洁。静态代码分析在软件测试中的应用研究[J].嘉应学院学报，2016,（02）：35-39.

[**5** ]张洪春。软件测试工程化模型及应用研究[J].中央民族大学学报（自然科学版），2016,（01）：40-45.测试计划执行情况

### 测试项目

（列出每一测试项目的名称、内容和目的。）

### 测试机构和人员

测试机构：成都大学javaWeb’ A08小组

测试人员：吴扎西、周兢

### 测试结果

（登录测试

（1）对登录系统的用户进行身份合法性检查，要求输入帐号和密码。禁止身份信息不正确的用户登录；

（2）每次登录将随机生成验证码，如果验证码错误将无法登录，以防止非法穷举入侵；

（3）将成功的登录操作记入操作日志文件

1．有效等价类：输入为用户信息表中保存的用户名和对应的密码。测试用例为：

|  |  |  |  |
| --- | --- | --- | --- |
| 等价类编号 | 用户名 | 密码 | 输出 |
| 1 | 小红 | xh | “登录正确” |
| 2 | 小李 | xl | “登录正确” |
| 3 | 张强 | zq | “登录正确” |
| 4 | 小张 | xz | “登录正确” |
| 5 | 李明 | lm | “登录正确” |
| 6 | 系统管理员 | admin | “登录正确” |

符合预期结果。

通过“查询”功能检查操作日志，发现已登记以上“登录数据库”操作及相应用户名密码。

符合预期结果。



2．无效等价类：

（1）用户名不在用户信息表中，密码在用户信息表中。测试用例为：

用户名：张三

密码： xd

点“确定”后，弹出“用户名和密码错误”对话框。

符合预期结果。

（2）用户名在用户信息表中，密码不在用户信息表中。测试用例为：

用户名：李明

密码： cb

点“确定”后，弹出“用户名和密码错误”对话框。

符合预期结果。

（3）用户名和密码均在用户信息表中，但两者不匹配。测试用例为：

用户名：李明

密码： xh

点“确定”后，弹出“用户名和密码错误”对话框。

符合预期结果。



注册测试

对注册的用户信息进行验证，用户的不同信息都要求有固定的格式，例如电话号码，电子邮箱，姓名格式，密码格式等。



测试用例：

姓名：丽丽

账号：lili

密码：lili123

核对密码：lili123

联系电话：19822928151

联系地址：四川成都

邮政编码：112500

电子邮箱：1234@qq.com

1. 当密码与核对密码不同时，提示：前后密码不一致

符合预期效果

1. 当联系电话少于11位，提示：无效的电话号码

符合预期效果

1. 当电子邮箱不是类似于“123456@qq.com”的格式时，提示：邮箱格式不规范

符合预期效果

（1）用户子系统模块测试

表5－1用户子系统模块的测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **功能** | **基本要求** | **测试情况** | **测试通过** | |
| **是** | **否** |
| 登录 | 输入正确的用户名和密码可以登录系统。  输入错误的用户名和密码系统给出明确提示。 | 功能实现。 |  |  |
| 注册 | 注册用户的时候具有对用户输入数据合法性的判断，电话号码位数，邮箱后缀名是否正确。在用户输入不合法数据，给出提示。 | 功能实现。 |  |  |

（2）购物车功能测试



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **功能** | **基本要求** | **测试情况** | **测试通过** | |
| **是** | **否** |
| 加入购物车 | 将图书加入购物车 | 功能实现。 |  |  |
| 从购物车删除 | 将图书从购物车删除 | 功能实现。 |  |  |
| 清空购物车 | 将购物车里面的书籍清空 | 功能实现。 |  |  |

1. 修改图书信息测试



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **测试内容** | **基本要求** | **测试情况** | **测试通过** | |
| **是** | **否** |
| 修改名称 | 修改图书名称 | 实现要求。 |  |  |
| 修改价格 | 修改图书价格 | 实现要求。 |  |  |
| 修改作者 | 修改作者姓名 | 实现要求 |  |  |
| 修改销量 | 修改书籍的销量 | 实现要求 |  |  |
| 修改库存 | 修改书籍的库存数量 | 实现要求 |  |  |
| 提交按钮 | 点击提交，提交修改信息 | 实现功能 |  |  |

（4）购物车测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **测试内容** | **基本要求** | **测试情况** | **测试通过** | |
| **是** | **否** |
| 添加购物车 | 用户选择商品后，点击加入购物车，商品成功加入购物车内。 | 实现要求。 |  |  |
| 删除购物车 | 在购物车中，用户对于不需要的商品可进行删除操作，点击删除符号，可将商品删除购物车。 | 实现要求。 |  |  |

（5）订单管理测试

表7－6订单测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **测试内容** | **基本要求** | **测试情况** | **测试通过** | |
| **是** | **否** |
| 查看订单 | 用户提交订单后，订单保存至数据库，用户可在用户界面查看相关订单，了解信息。 | 实现要求 |  |  |
| 删除订单 | 用户在用户界面拥有删除订单功能，在订单界面，点击删除，订单成功被删除。 | 实现要求 |  |  |

## 评价

### 软件能力

经过一系列测试，对各个模块的功能进行了检测，系统并没有出现影响使用的bug，能够顺利的完成查看书籍、加入购物车、删除书籍、修改书籍信息、注册账户、登录账户等功能。

系统能够初步实现模仿购买书籍的功能，基本实现设计目的，满足预计需求。

### 缺陷和限制

1. 系统偶尔会出现一些小bug，比如页面丢失。
2. 系统没有实现支付功能，不能够正真地实现购买书籍的功能。
3. 系统没有实现书籍检索功能。

### 建议

1. 优化页面转发逻辑
2. 增加支付功能
3. 增加书籍检索功能

### 测试结论

在经过可操作的检测之后，系统基本上达到了设计初期的预期，实现了设计的需求，能够顺利的运行，达到了开发的目的。但是任然由许多地方需要优化。

登录

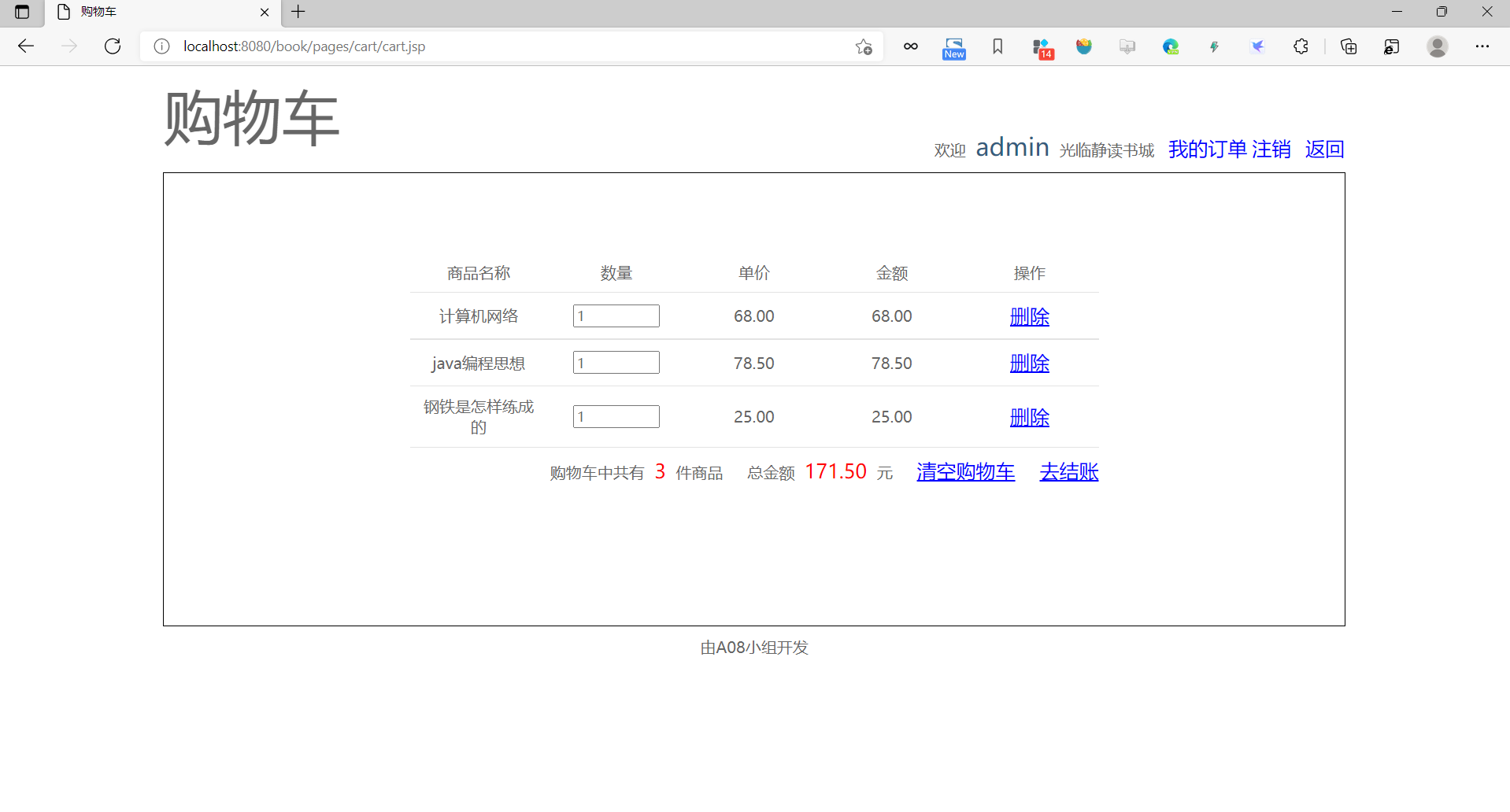
注册



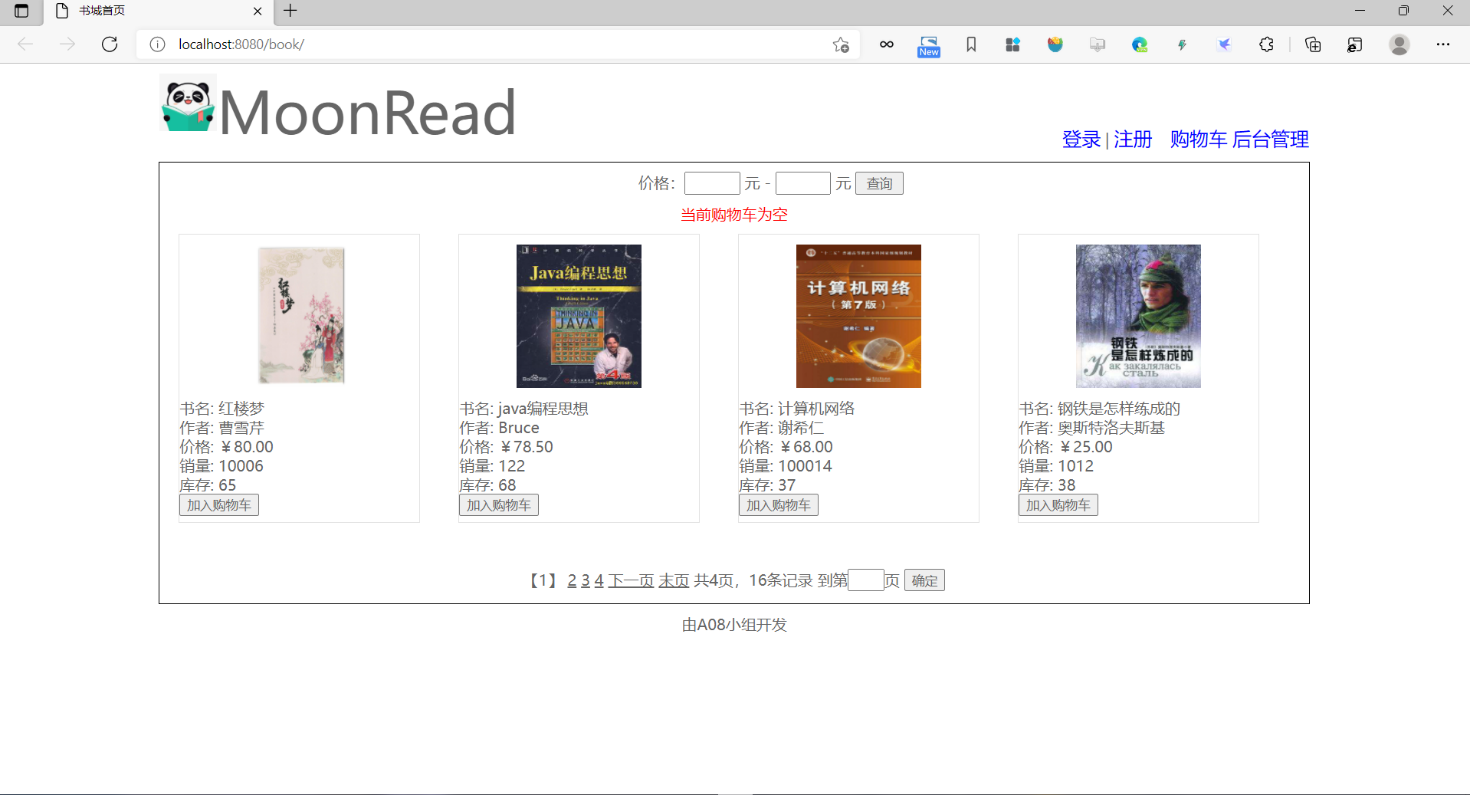
订单



购物车



首页



后台管理

