

浙江工业大学

《软件工程》实验报告

大学校园网上书店系统开发文档

组长：

组员：

组员：

组员：

指导老师：

2020 年 X 月 X 日

目录

大学校园网上书店系统开发文档 1

前言 5

1 可行性研究 6

1.1 系统概述 6

1.2 初步的系统分析设计 6

1.2.1 业务流程图 7

1.2.2 数据流图 11

1.2.3 数据字典 13

1.3 技术可行性分析	18
1.4 经济/效益可行性分析	18
1.4.1 开发成本	18
1.4.2 运行费用	19
1.4.3 经济效益	19
1.5 系统开发计划	20
2 需求分析	20
2.1 总体目标	20
2.2 具体目标	20
2.3 系统数据建模	21
2.4 系统功能建模	27
2.5 数据字典	28
3 总体设计	33
3.1 系统结构图	33
3.2 数据库设计	35
3.2.1 逻辑结构设计	35
3.2.2 物理结构设计	40
3.3 系统模块 IPO 表	41
4 详细设计	61
4.1 顾客注册模块	61
4.2 顾客个人中心模块	62
4.3 顾客浏览图书模块	64

4.4 顾客查看订单功能模块	65
4.5 顾客查看购物车功能模块	65
4.6 顾客付款功能模块	66
4.7 店员修改个人信息功能模块	67
4.8 店员注册功能模块	68
4.9 店员查看图书信息功能模块	69
4.10 店员查看订单/退单信息功能模块	70
4.11 用户注销登陆模块	71
4.12 店员确认专业课程书单信息功能模块	72
4.13 店员生成报表并查看报表	73
4.14 用户登录模块	74
4.15 管理员修改个人信息模块	75
4.16 管理员查看订单功能模块	76
4.17 管理员管理店员信息功能模块	77
5 系统实现	79
5.1 开发平台和开发环境介绍	79
5.2 编码	80
5.3 前台界面与后台数据库连接代码	80
5.4 各模块功能的实现。	82
5.4.1 主界面模块实现	82
5.4.2 注册模块实现	83
5.4.3 登陆模块实现	83

5.4.4 顾客浏览图书模块实现	84
5.4.5 快速检索图书模块实现	85
5.4.6 用户注销登陆状态模块实现	85
5.4.7 顾客查看购物车模块实现	85
5.4.8 顾客查询订单和申请退款模块实现	86
5.4.9 用户修改个人信息模块实现	87
5.4.10 店员确认订单模块实现	87
5.4.11 店员确认退单模块实现	88
5.4.12 店员管理图书信息模块实现	88
5.4.13 书店报表生成模块实现	89
5.2 测试	90
5.2.1 软件测试的目标	90
5.2.2 软件测试的步骤	90
5.2.3 单元/集成/验收测试	90
5.2.4 测试过程	92
6 维护	99
6.1 系统维护过程	99
6.2 系统维护策略	99
7 总结与体会	101
7.1 总结	101
7.2 体会	101
8 附录	106

前言

本文档是根据软件工程的基本原理、开发方法以及开发过程进行了多次的讨论与实践的成果，本系统按照软件生命周期的各个阶段相应的任务进行开发，主要包括可行性研究、需求分析、总体设计、详细设计、编码、测试等内容，每一章节都图文并茂地阐述了具体的设计过程，使得该文档具有较高的可读性，易于开发人员进行设计和维护。

1 可行性研究

1.1 系统概述

大学校园网上书店系统是一个基于 Java 和 SQL 语言、在 BS 架构模式下运行以及应用 SSH 开发框架和数据库技术实现的 Web 应用系统。

该系统将校园实体书店线下售卖图书模式升级为网上在线销售模式，其功能主要有两个方面。一方面通过导入专业年级相应的课程表，实现快速便捷查找和购买所需教材；另一方面实现学生与书商之间线上图书交易。

从书店的角度看，使用该系统能够使员工更方便地查看商品的库存情况、商品是否仍具时效性等基本信息，而基于记录的销售信息数据以及针对性的课程书目，商家也能够制定更合适的打折方案和进货策略，对数量的把握更为精准，也减少了进不必要数目造成库存堆积的损失，大大减少了成本与损失，提高了利润空间。从学生的角度看，使用该系统能方便同学们迅速地浏览和定位自己所需的图书、查看各个图书的价格信息、减少买书所需的等待时间，从而使学生的日常生活更为便捷，节约了学生的时间和金钱。此举方便了学生和书商，提高了校园信息化水平，且利于校方更新教学书目，也能间接提高教学质量。

1.2 初步的系统分析设计

网上购书系统包括以下几个系统模块：

1. 浏览查阅模块：该系统模块描述了游客端的功能，具有查看图书信息、搜索图书、查看图书具体信息、查看专业课程书目表、登陆/注册功能。
2. 图书购买模块：该系统模块描述了注册用户端特有功能，具有购买图书、查看购买信息、积分打折功能。
3. 书店管理员模块：该系统模块描述了书店管理员端特有功能，具有添加、删除、修改图书信息，查看订单信息、修改订单属性（是/否发货、确认订单等属性）、根据订单信息产生报表功能、专业课程书单销售确认。
4. 网站管理员模块：该系统模块描述了系统管理员端的功能，具有对顾客信息、员工信息进行增、删、改、查，录入课程用书信息功能。

1.2.1 业务流程图

业务流程图是概括得描绘物理系统的传统工具。其基本思想是利用图形符号以黑匣子的形式描绘组成系统的各个部件。其强调的是数据在系统各部件间的流动情况，选择性地暂时忽略对各数据加工处理的控制过程。以一种物理数据流图的形式展现系统的各个功能运作流程，为人们初步理解项目系统提供帮助。基于上述分析的四个系统模块，本项目系统的业务流程图如下图 1.1-图 1.11 所示。

(1) 顾客浏览-搜索图书信息业务

图 1.1 顾客浏览-搜索图书信息业务流图

(2) 用户登录-注册业务

图 1.2 顾客登录-注册业务流图

(3) 顾客教材查询和订购业务

图 1.3 顾客教材查询和订购业务流图

(4) 顾客查看订单业务

图 1.4 顾客查看订单业务流图

(5) 顾客购买商品以及积分运作业务

图 1.5 顾客购买商品以及积分运作业务流图

(6) 店员未完成交易订单处理业务

图 1.6 店员未完成交易订单处理业务流图

(7) 店员账单查询以及生成报表业务

图 1.7 店员账单查询以及生成报表业务流图

(8) 店员图书增删改查业务

图 1.8 店员图书增删改查业务流图

(9) 店员课程教材查询链接书籍业务

图 1.9 店员课程教材查询链接书籍业务

(10) 管理员对用户和图书信息的管理业务

图 1.10 管理员对用户和图书信息的管理业务

(11) 管理员学校专业课程导入业务

图 1.11 管理员学校专业课程导入业务

1.2.2 数据流图

数据流图（DFD）是一种图形化技术，其描绘了信息流和数据从输入移动到输出的过程中所经受的变换。在数据流图中没有任何具体的物理部件，它只是描绘数据在软件中流动和被处理的逻辑过程，构建的是系统的逻辑模型。画数据流图的目的是利用它作为交流信息、分析和设计的工具。首先，基于上述业务流程图，绘制出系统的顶层数据流图，如下图 1.12 所示。

图 1.12 顶层数据流图

该顶层数据流图仅用一个加工过程 P0 来表示整个系统，用 E0、E1、E2 表示该系统全部的数据输入输出实体。顶层数据流图作用是在最高层次上展示数据流动情况，为后续进一步弄清系统数据流动和处理的过程夯实基础。

其次，在上述顶层数据流图的基础上，项目将对系统加工过程 P0 进行初步细化，得到 0 层数据流图。该 0 层数据流图将系统加工过程按之前的模块分成相应的几个部分，同时为这几个部分链接了对应的数据库，展现出了更为详细且更接近于现实的系统逻辑结构模型，该 0 层数据流图如下图 1.13 所示。

图 1.13 0 层数据流图

1.2.3 数据字典

数据字典是关于数据的信息的集合，也就是对数据流图中包含的所有元素的定义的集合，其主要功能是在软件分析和设计的过程中给人提供关于数据的描述信息。经过分析，该项目需要建立顾客表、课程教材单、订单、店员表、图书表、管理员表、(销售) 报表、退款单等等，其初步的数据字典如下表 1.1-表所示：

表 1.1 顾客数据字典

名称: 顾客

描述: 在网上购书系统消费的用户

定义: 顾客 = 姓名 + 性别 + 用户名 + 密码 + 联系方式 + 地址 + 所在学校 + 积分

字段: 姓名 = 字符/20 位

性别 = 【男/女】

用户名 = 字符/20 位

密码 = 字符/20 位

联系方式 = 字符/11 位

地址 = 字符/30 位

所在学校 = 字符/20 位

积分 = 整型

位置: 顾客表

表 1.2 店员数据字典

名称: 店员

描述: 管理网上书店的用户

定义: 店员 = 用户名 + 密码 + 姓名 + 性别 + 联系方式 + 书店名 + 书店的地址 + 书店负责的学校的名称

字段: 用户名 = 字符/20 位

密码 = 字符/20 位

姓名 = 字符/20 位

性别 = 【男/女】

联系方式 = 字符/11 位

书店名 = 字符/20 位

书店的地址 = 字符/30 位

书店负责的学校的名称 = 字符/20 位

位置: 店员表

表 1.3 管理员数据字典

名称: 管理员

描述: 负责系统的运营、维护与管理专业人员

定义: 管理员账号 = 管理员用户名 + 管理员用户密码 + 姓名 + 性别 + 联系方式

字段: 管理员用户名 = 字符/20 位

管理员用户密码 = 字符/20 位

姓名 = 字符/20 位

性别 = 【男/女】

联系方式 = 数字/11 位

位置: 管理员表

表 1.4 图书数据字典

名称: 图书

描述: 本系统销售的商品类型

定义: 图书 = 书籍号 + 书名 + 版本号 + 书籍出版社 + 售价 + 库存量 + 折扣 + 销售书店 + 所在学校

字段: 书籍号 = 字符/10 位

书籍名 = 字符/20 位

版本号 = 字符/20 位

书籍出版社 = 字符/20 位

售价 = 浮点型

库存量 = 整型

折扣 = 浮点型

销售书店 = 字符/20 位

所在学校 = 字符/20 位

位置: 图书表

表 1.5 订单数据字典

名称: 订单
描述: 记录退款相关信息的表单
定义: 订单 = 订单号 + 顾客姓名 + 顾客用户名 + 书籍名 + 书籍号 + 数量 + 书店名 + 书店地址 + 送货地址 + 下单时间 + 金额总数 + 备注 + 是否确认 + 是否退货
字段: 订单号 = 字符/20 位
顾客姓名 = 字符/20 位
顾客用户名 = 字符/20 位
书籍名 = 字符/20 位
书籍号 = 字符/10 位
数量 = 整型
书店名 = 字符/20 位
书店地址 = 字符/30 位
送货地址 = 字符/30 位
下单时间 = 字符/10 位
金额总数 = 浮点型
备注 = 字符/30 位
是否确认 = 【是/否】
是否退货 = 【是/否】
位置: 订单表

表 1.6 退款单数据字典

名称: 退款单
描述: 记录退款相关信息的表单
定义: 退款单 = 订单号 + 顾客姓名 + 顾客用户名 + 书籍名 + 书籍号 + 数量 + 书店名 + 书店地址 + 送货地址 + 下单时间 + 申请退款时间 + 受理退款时间 + 备注 + 金额总数 + 退款原因 + 是否确认

名称: 退款单

字段: 订单号 = 字符/20 位

顾客姓名 = 字符/20 位

顾客用户名 = 字符/20 位

书籍名 = 字符/20 位

书籍号 = 字符/10 位

数量 = 整型

书店名 = 字符/20 位

书店地址 = 字符/30 位

送货地址 = 字符/30 位

下单时间 = 字符/10 位

申请退款时间 = 字符/10 位

受理退款时间 = 字符/10 位

备注 = 字符/30 位

金额总数 = 浮点型

退款原因 = 字符/30 位

是否确认 = 【是/否】

位置: 退款单表

表 1.7 销售情况报表数据字典

名称: 销售情况报表

描述: 返回一个月内书店图书销售情况的统计数据

定义: 销售情况报表 = 书籍名 + 书籍号 + 图书售价 + 单种图书销售总量
+ 单种图书销售金额总数 + 收入总额

字段: 书籍名 = 字符/20 位

书籍号 = 字符/20 位

图书售价 = 字符/5 位

单种图书销售总量 = 整型

单种图书销售金额总数 = 浮点型

收入总额 = 浮点型

位置: 销售报表

表 1.8 课程教材书单数据字典

名称: 课程教材书单

描述: 记录各学校各专业学生课程所需教材名称的表单
定义: 课程教材单 = 序号 + 专业 + 学年 + 课程 + 书籍名称 + 书籍出版社 + 主要作者 + 出版号 + 使用学校校名 + 链接书籍号 + 价格
字段: 序号 = 数字/5 位
专业 = 字符/20 位
学年 = 数字/1 位
课程 = 字符/20 位
书籍名称 = 字符/20 位
书籍出版社 = 字符/20 位
主要作者 = 字符/20 位
出版号 = 字符/20 位
使用学校校名 = 字符/20 位
链接书籍号 = 字符/20 位
价格 = 数字/5 位
位置: 课程教材书单

1.3 技术可行性分析

本实验采用的开发工具为 MySQL、Subline Text 3 和 Eclipse, 实现基于 SSH 的 JavaEE 完全能够胜任此次开发。因为本系统服务的用户阈值为一个学校的学生数量, 考虑到用户查看商品和下单的并发性不是很高, 而 MySQL 是一个小型关系型数据库管理系统, 被广泛地应用在 Internet 上的中小型网站中。由于其体积小、速度快、总体拥有成本低, 尤其是开放源码这一特点, 所以本项目选择了 MySQL 作为网站数据库。SublimeText 3 是一款具有代码高亮、语法提示、自动完成而且反应快速的编辑器软件, 用其开发前端网页是一个极其方便的事情, 是当前网页开发的主流编辑器之一。而 SSH 三者各自的作用及优点如下:

Struts 是一个很好的 MVC 框架, 主要技术是 Servlet 和 Jsp。Struts 的 MVC 设计式可以使项目设计的逻辑变得更清晰, 让程序层次分明。

spring 提供了管理业务对象的一致方法，并鼓励注入对接口编程而不是对类编程的良好习惯，使本系统在最大程度上解耦。

Hibernate 是用来持久化数据的，提供了完全面向对象的数据库操作。Hibernate 对 JDBC 进行了非常轻量级的封装，它使得与关系型数据库打交道变得轻松。

同时，为了高质量完成本软件系统开发工作，还需要遵循如下准则：

- (1) 做好数据的规划，建立稳定的信息模型；
- (2) 在功能模型的划分上，按组织机构来划分子系统或模块；
- (3) 应用软件的开发设计，要充分考虑应用软件的适应性，建立友好的人机界面。

综上所述，系统的实现技术上完全可行。

1.4 经济/效益可行性分析

1.4.1 开发成本

大学校园网上书店系统的开发隶属于一般性软件项目开发工作，其主要涉及的开发成本来源于前期项目准备、软件设计、编码实现以及产品测试人员的人力成本，其所有使用的开发软件、开发框架均属于开源性质。

1.4.2 运行费用

该项目的运行费用主要来自两个方面，后期系统运营人员的工资支出和网站服务器的租赁费用。

后期系统运营人员的工作主要涉及定期的数据备份、网站安全的维护以及对使用者的基本培训等方面，是项目正常运行必不可少的一环。按当前市场行情来看，考虑到地区之间的差异性，维护与本项目相类似的小型网站的月工资大概在 2500-5000 人民币左右。

网站服务器的租赁费用属于基本硬件的开销。针对一所大学来讲，与一般网站访问人群基数相比，其校内学生人数相对较少。因此，该项目运行所需服务器的性能要求并不会很高。按当前的市场价位估计，该项目每年在服务器上的开销大约在 800 元左右。

1.4.3 经济效益

该项目有助于提高书店经济收益的优势主要有以下两点：

一、当前网上购物已经成为一种常态，“网上订购 + 快递配送”越发成为消费主流。与一般的大学实体书店，网上售书可以为学生购书提供了非常大的便利，学生足不出户即可购得想要的书籍。

二、与一般的网上书店相比，虽然销售价格上可能相对偏高，但本项目在教材销售方面有着较大优势。考虑到学生在订购教材时很大可能是不知道本学校是使用哪些专业教材的情况，针对具体的一所大学而言，由于其校内书店掌握着该校各门课程所采用教材的准确信息，因此，本项目可以依靠上述有利条件，为不同专业不同年级的学生制定专属教材清单，解决学生在订购教材时各种不必要的麻烦。通过这样的方式，吸引更多的学生前来购书。

借助以上两个优势，相比原先实体店直接销售模式，学校书店的书籍销售水平将会提高不止一个档次。由于该网站运行模式适用于各个大学校园书店的书籍销售，在一定销售推广工作的基础上，开发与运营成本可由各个注册商家进行平摊。一般来讲，不需要多长的时间，项目将可迎来收益。

1.5 系统开发计划

在开始我们正式的系统开发前，我们小组进行了大量的讨论与思索，所以在前两周我们完成了系统的可行性分析，需求分析和总体设计部分。把基本结构搭出来后，后面四周我们进行了代码的编写和系统设计的详细话。最后两周我们完成了程序的测试，实现了各模块的功能，完成了我们最终的实验报告。

2 需求分析

2.1 总体目标

本项目的目标意在依托当前网上购物的大好形势和借助校园书店独有的地理优势和相关学校资源，实现学生与大学校园书店两者之间的互利共赢。对学生主体而言，如何更方便、更快捷地购买书籍、如何花费更少的时间和精力选购专业课程教材是该项目主要考虑的实现目标；而对大学校园书店实体而言，本项目主要的实现目标考虑的是如何提高图书的销售量来增加商家利润、如何扩大书店的影响力以及如何更方便地统计销售数据来实现动态地制定销售计划和合理地引进相应书籍的数量。

2.2 具体目标

(1) 浏览查阅模块：

查看图书信息，搜索图书，查看图书具体信息，查看专业课程书目表（仅限注册用户），登陆/注册功能，加入购物车（仅限注册用户）。

2) 图书购买模块：

选购图书放入购物车，编辑图书购买数量，提交订单，查询历史订单，积分换取折扣，修改地址，付款，生成积分，申请退款。

(3) 书店管理员模块：

注册，登录，注销账户，添加图书信息，删除图书信息，修改图书信息，查看订单信息，确认订单，确认退单，根据订单信息产生报表，确认专业课程书单。

(4) 网络管理员模块

查询、修改顾客信息，查询、修改店员信息，查询、修改图书信息，查询、修改教材信息，导入教材书单。

2.3 系统数据建模

系统数据模型包含三种互相关联的信息：数据对象、数据对象的属性和数据对象彼此间相互连接的关系。本项目使用实体-联系图来清晰、准确地描述用户的数据要求。E-R 图属于面向问题的数据模型，即概念性数据模型。它从用户角度出发看待数据，反映用户的现实环境，与软件系统中的实现方法无关。

全局的 E-R 图设计如图 2.1 所示：

图 2.1 全局 E-R 图

对全局 E-R 图 2.1 进行分解，得到以下局部 E-R 图

图 2.2 顾客浏览商品

图 2.3 顾客查看课程教材单

图 2.4 顾客提交订单

图 2.5 图书形成订单

图 2.6 顾客申请退款单

图 2.7 订单形成退款单

图 2.8 店员确认订单

图 2.9 店员确认退款单

图 2.10 店员出售图书

图 2.11 店员链接课程教材

图 2.12 管理员管理顾客信息

图 2.13 管理员管理店员信息

图 2.14 店员查阅销售情况报表

图 2.15 订单形成销售情况报表

图 2.16 管理员导入课程教材单

图 2.17 管理员管理图书

2.4 系统功能建模

如下图 2.7 顶层数据流图所示，该系统整体上可以分为三个实体和一个处理过程。其实体分别为 E0 顾客、E1 店员、E2 管理员，处理过程为 P0 大学

校园网上书店系统。由于系统主要功能在可行性分析阶段已经进行了严谨的分析讨论，因此在该阶段可以直接沿用之前的顶层数据流图。该数据流图以人机交互思想为指导，从整体观念出发，展示了流入和流出系统的所有数据信息。

图 2.18 顶层数据流图

0 层数据流图 2.8 将 E0 顾客、E1 员工、E2 管理员之间的消费信息数据流进行第一步细化。其中，将处理过程 P0 大学校园网上书店系统细分为 P0 顾客与员工信息管理、P1 图书信息管理、P2 图书信息查询、P3 导入专业课程书单、P4 专业书单查询、P5 订单处理、P6 书单确认；并引入了多个数据存储，如顾客表、店员表、图书表、订单表、专业课程书单。由于本系统在可行性研究阶段已经进行了详细的分析论证，因此 0 层数据流图可在需求分析阶段直接沿用。

图 2.19 0 层数据流图

2.5 数据字典

数据字典最重要的用途是作为分析阶段的工具，典型的情况是，在数据字段中记录数据元素的下列信息：一般信息（名称，描述），定义（数据类型，长度，结构等），使用特点（值的范围，使用频率，使用方式—输入、输出、本地、条件值等），控制信息（来源，用户，使用它的程序，改变权，使用权等）等。根据可行性研究中的初步分析，我们需要建立顾客表，员工表，商品表，订单表和管理员表等。对在可行性分析研究阶段得到的数据字典进行补充完善如下：

表 2.1 顾客数据字典

名称: 顾客
描述: 在网上购书系统消费的用户
定义: 顾客 = 姓名 + 性别 + 用户名 + 密码 + 联系方式 + 地址 + 所在学校 + 积分

名称: 顾客

说明: 姓名 = 字符/20 位

性别 = 【男/女】

用户名 = 字符/20 位

密码 = 字符/20 位

联系方式 = 字符/11 位

地址 = 字符/50 位

所在学校 = 字符/30 位

积分 = 整型

位置: 顾客表

表 2.2 店员数据字典

名称: 店员

描述: 管理网上书店的用户

定义: 店员 = 用户名 + 密码 + 姓名 + 性别 + 联系方式 + 书店名 + 书店
的地址 + 书店负责的学校的名称

说明: 用户名 = 字符/20 位

密码 = 字符/20 位

姓名 = 字符/20 位

性别 = 【男/女】

联系方式 = 字符/11 位

书店名 = 字符/30 位

书店的地址 = 字符/50 位

书店负责的学校的名称 = 字符/30 位

位置: 店员表

表 2.3 管理员数据字典

名称: 管理员

描述: 负责系统的运营、维护与管理专业人员

名称: 管理员

定义: 管理员账号 = 管理员用户名 + 管理员用户密码 + 姓名 + 性别 + 联系方式

说明: 管理员用户名 = 字符/20 位

管理员用户密码 = 字符/20 位

姓名 = 字符/20 位

性别 = 【男/女】

联系方式 = 数字/11 位

位置: 管理员表

表 2.4 图书数据字典

名称: 图书

描述: 本系统销售的商品类型

定义: 图书 = 书籍号 + 书名 + 版本号 + 书籍出版社 + 原价 + 库存量 + 书店折扣 + 销售书店 + 所在学校

说明: 书籍号 = 字符/20 位

书籍名 = 字符/20 位

版本号 = 字符/20 位

书籍出版社 = 字符/30 位

原价 = 浮点型

库存量 = 整型

书店折扣 = 浮点型

销售书店 = 字符/30 位

所在学校 = 字符/30 位

售价 = 原价 × 书店折扣

位置: 图书表

表 2.5 课程教材书单数据字典

名称: 课程教材书单
描述: 记录各学校各专业学生课程所需教材名称的表单
定义: 课程教材单 = 序号 + 专业 + 学年 + 课程 + 书籍名称 + 书籍出版社 + 主要作者 + 出版号 + 使用学校校名 + 链接书籍号 + 原价 + 书店折扣
说明: 序号 = 数字/20 位
专业 = 字符/20 位
学年 = 数字/1 位
课程 = 字符/20 位
书籍名称 = 字符/20 位
书籍出版社 = 字符/30 位
主要作者 = 字符/20 位
出版号 = 字符/20 位
使用学校校名 = 字符/30 位
链接书籍号 = 字符/20 位
原价 = 浮点型
书店折扣 = 浮点型
售价 = 原价 × 书店折扣
位置: 课程教材书单

表 2.6 购物车数据字典

名称: 购物车
描述: 记录顾客购买的书籍相关信息的表单
定义: 购物车 = 顾客姓名 + 顾客用户名 + 书籍名 + 书籍号 + 售价 + 积分折扣 + 数量 + 金额总数 + 备注
说明: 售价 = 书籍原价 × 书店折扣
金额总数 = 售价 × 积分折扣 × 数量

表 2.7 订单表数据字典

名称: 订单
描述: 记录购书相关信息的表单
定义: 订单 = 订单号 + 顾客姓名 + 顾客用户名 + 书籍名 + 书籍号 + 售价 + 积分折扣 + 数量 + 书店名 + 书店地址 + 送货地址 + 下单时间 + 金额总数 + 备注 + 是否确认 + 是否付款 + 是否退货
说明: 订单号 = 字符/20 位
顾客姓名 = 字符/20 位
顾客用户名 = 字符/20 位
书籍名 = 字符/20 位
书籍号 = 字符/20 位
书籍价格（折后）= 浮点型
数量 = 整型
书店名 = 字符/30 位
书店地址 = 字符/50 位
送货地址 = 字符/50 位
下单时间 = 字符/20 位
金额总数 = 浮点型
备注 = 字符/30 位
是否确认 = 【是/否】
是否付款 = 【是/否】
是否退货 = 【是/否】
售价 = 书籍原价 × 书店折扣
金额总数 = 售价 × 积分折扣 × 数量
位置: 订单表

表 2.8 退款单数据字典

名称: 退款单
描述: 记录退款相关信息的表单
定义: 退款单 = 订单号 + 顾客姓名 + 顾客用户名 + 书籍名 + 书籍号 + 付款金额 + 数量 + 书店名 + 书店地址 + 送货地址 + 下单时间 + 申请退款时间 + 受理退款时间 + 备注 + 金额总数 + 退款原因 + 是否确认

名称: 退款单
说明: 付款金额 = 售价 × 积分折扣

表 2.9 销售情况报表数据字典

名称: 销售情况报表
描述: 返回一个月内书店图书销售情况的统计数据
定义: 销售情况报表 = 书籍名 + 书籍号 + 售价 + 单种图书销售总量 + 单种图书销售金额总数 + 收入总额
说明: 书籍名 = 字符/20 位
书籍号 = 字符/20 位
售价 = 浮点型
单种图书销售总量 = 整型
单种图书销售金额总数 = 浮点型
收入总额 = 浮点型
售价 = 原价 × 书店折扣
单种图书销售金额总数 = 售价 × 单种图书销售总量
收入总额 = 单种图书销售金额累加
位置: 销售情况报表

3 总体设计

3.1 系统结构图

因为本系统在前面需求分析阶段已经得到了较为完善可行的顶层数据流图、0 层数据流图以及 1 层数据流图，所以在本总体设计阶段直接沿用上述得到的图即可，具体详见图 2.7-图 2.11。

层次图可以用来描述软件的层次结构，适用于在自顶向下设计软件的过程中使用，本系统分别从顾客端、店员端、网站管理员端三个方面绘制系统结构图，具体如下所示：

图 3.1 总体层次结构图

图 3.2 顾客端层次结构图

图 3.3 店员端层次结构图

图 3.4 网络管理员端层次结构图

3.2 数据库设计

3.2.1 逻辑结构设计

(1) 关系模式设计 系统的关系模式可由系统数据建模中的 E-R 图分析设计而得。其中 E-R 图模式中的实体和联系都可以转化成关系型，其属性可以转化为关系型的属性，实体型的主键也可以转化为关系型的主键。而对实体集之间的联系转换受到其自身类型的影响，需要具体问题具体分析。针对该项目中的联系转换，本项目作出以下几点调整：

(1) 对于 1:1 型的联系转换：先将两个实体型分别转换为对应的关系模式后，再将其中一个实体的主键属性加入另一个实体当中，即可完成相关转换。但是，对于本项目中订单和退款单的 1:1 型联系，由于它们可以对应于同一个订单号以及它们大部分属性是相同的，因此，为了便于查询，项目将退款单并入订单之中。

(2) 对于 1:n 型的联系转换：除了需要将实体自身的属性直接进行转换，对在 n 端的实体，需要将在 1 端实体的主键放入其中。需要按其修改的联系有顾客和订单的提交联系、店员与订单间的确认关系、图书与订单的确认关系。

(3) 对于某些特殊的联系转换：销售情况报表的作用是方便店员对一个月或某段时间下书籍销售情况的统计，由于时间范围的不准确性，本项目不对该报表进行存储，只对该操作进行运算显示。因此，对于该实体型以及其与相关实体型间的联系本项目不对其进行转换。

(4) 对于 $n:m$ 型的联系转换：除了需要将实体自身的属性直接进行转换，在设计时还需要将两者间的联系转换为一个关系型。该关系型的属性需要包含其对应两个实体的主键属性和联系自身的属性，需要进行该操作的联系有店员与图书的出售联系、店员和课程教材单的链接图书关系。

(5) 对于某些特殊的 $n:m$ 型的联系转换：对于上述管理员对各实体的管理联系，虽然其符合 $n:m$ 型的联系模式，但是这里考虑的是任何一个管理员可以对所有实体进行操作。这样的模式更类似于一种权限而已，因此无需将该联系转换为一个新的关系型。对于一些图书查阅联系，其本质是个体对于图书查找的权限，这样的 $n:m$ 型关系也无需转换为一个新的关系型。需要进行上述处理的关系有顾客和图书的浏览关系、顾客与课程教材单的查看关系、管理员与顾客的管理关系、管理员与店员的管理关系、管理员与图书的管理关系、管理员导入课程教材单。

按上述要求建立系统的关系模式，其主要内容如下所示，其中，下划线表示该关系型的主键，波浪线表示该关系型的外键。

顾客表 (用户名, 密码, 姓名, 性别, 所在学校, 联系方式, 默认地址, 积分)

店员表 (用户名, 密码, 姓名, 性别, 联系方式, 书店名, 书店地址, 书店对应学校名称)

管理员表 (用户名, 密码, 姓名, 性别, 联系方式)

图书表 (出版号, 书名, 主要作者, 出版社, 原价)

导入表 (出版号, 店员用户名, 折扣, 库存量)

订单表 (订单号, 顾客用户名, 店员用户名, 出版号, 送货地址, 数量, 下单时间, 金额总数, 备注, 是否付款, 是否确认, 确认时间, 退货申请, 是否退货, 退款理由, 是否积分打折, 获得积分)

课程教材单 (序号, 学年, 专业, 课程, 使用学校名称, 出版号, 书籍名称)

上架表 (店员用户名, 序号, 是否上架)

(2) 数据类型定义 接下来对关系模式设计中的属性定义类型、长度和约束，得到表 3.1-表 3.12。

表 3.1 顾客表

数据项名	英文名	数据类型	长度	实体完整性约束	参照完整性约束	用户自定义性约束
用户名	Cusername	VARCHAR	20	√	/	/
密码	Cpassword	VARCHAR	20	/	/	默认为123456
姓名	Cname	VARCHAR	20	/	/	not null
性别	Csex	VARCHAR	2	/	/	男/女
所在学校	Cschool	VARCHAR	30	/	/	not null
联系方式	Cphone	VARCHAR	11	/	/	not null
默认地址	Clocal	VARCHAR	50	/	/	not null
积分	Cvalue	TINYINT UN-SIGNED	/	/	/	初始为0

表 3.2 店员表

数据项名	英文名	数据类型	长度	实体完整性约束	参照完整性约束	用户自定义性约束
用户名	Susername	VARCHAR	20	√	/	/
密码	Spassword	VARCHAR	20	/	/	默认为123456

姓名	Sname	VARCHAR	20	/	/	not null
性别	Ssex	VARCHAR	2	/	/	男/女
联系方式	Sphone	VARCHAR	11	/	/	not null
书店名	Sshopname	VARCHAR	30	/		not null
书店地址	Slocal	VARCHAR	50	/	/	not null
书店对应学校名称	Sschool	VARCHAR	30	/	/	/

表 3.3 管理员表

数据项名	英文名	数据类型	长度	实体完整性约束	参照完整性约束	用户自定义性约束
用户名	Ausername	VARCHAR	20	√	/	/
密码	Apassword	VARCHAR	20	/	/	默认为123456
姓名	Aname	VARCHAR	20	/	/	not null
性别	Asex	VARCHAR	/	/	/	男/女
联系方式	Aphone	VARCHAR	11	/	/	not null

表 3.4 图书表

数据项名	英文名	数据类型	长度	实体完整性约束	参照完整性约束	用户自定义性约束
出版号	Bnumber	TINYINT UN-SIGNED	/	√	/	自增

书名	Bbookname	VARCHAR	30	/	/	not null
主要作者	Bwriter	VARCHAR	30	/	/	not null
出版社	Bpublic	VARCHAR	30	/	/	not null
原价	Bprice	DOUBLE	/	/	/	初始为0

表 3.5 导出表

数据项名	英文名	数据类型	长度	实体完整性约束	参照完整性约束	用户自定义性约束
出版号	Bnumber	VARCHAR	20	/	√	not null
店员用户名	Susername	VARCHAR	20	/	√	not null
折扣	Idiscount	DOUBLE	/	/	/	初始为0
库存量	Inumgoods	TINYINT UN-SIGNED	/	/	/	初始为0

表 3.6 订单表

数据项名	英文名	数据类型	长度	实体完整性约束	参照完整性约束	用户自定义性约束
订单号	Onumber	INTEGER UNSIGNED	/	√	/	/
顾客用户名	Cusername	VARCHAR	20	/	√	not null
店员用户名	Susername	VARCHAR	20	/	√	not null

出版号	Bnumber	VARCHAR	13	/	√	not null
送货地址	Olocal	VARCHAR	50	/	/	一般使用默认地址
下单时间	Otime	DATETIME	/	/	/	not null
数量	Oquantity	TINYINT UN- SIGNED	/	/	/	not null
金额总数	Osumprice	DOUBLE	/	/	/	not null
备注	Oremark	VARCHAR	30	/	/	/
是否付款	Opay	BIT	/	/	/	/
是否确认	Oaccept	BIT	/	/	/	/
确认时间	Ochecktime	DATE	/	/	/	not null
退货申请	Oreturn	BIT	/	/	/	/
退货理由	Oreason	VARCHAR	30	/	/	/
是否退货	Ocancel	BIT	/	/	/	/
是否积分打折	Odiscount	BIT	/	/	/	/
获得积分	Ogetvalue	TINYINT UN- SIGNED	/	/	/	/

表 3.7 课程教材单表

数据项 名	英文名	数据类 型	长度	实体完 整性约 束	参照完 整性约 束	用户自 定义性 约束
序号	Tid	TINYINT UN- SIGNED	/	√	/	/
学年	Tyear	VARCHAR	20	/	/	not null
专业	Tmajor	VARCHAR	20	/	/	not null
课程	Tcourse	VARCHAR	20	/	/	not null
使用学 校名称	Tschool	VARCHAR	20	/	/	not null
出版号	Bnumber	TINYINT UN- SIGNED	/	/	√	not null
书籍名 称	Bbookname	VARCHAR	30	/	/	/

表 3.8 上架表

数据项 名	英文名	数据类 型	长度	实体完 整性约 束	参照完 整性约 束	用户自 定义性 约束
店员用 户名	Susername	VARCHAR	20	/	√	/
序号	Fid	TINYINT UN- SIGNED	/	√	/	not null
是否上 架	Fonshelf	INT	/	/	/	默认为 0

3.2.2 物理结构设计

数据库最终是存储在物理设备上的，数据库在物理设备上的存储结构和存取方法就称为数据库的物理结构，它依赖于具体的计算机系统。所谓数据库的物理结构设计就是为一个给定数据库的逻辑结构选取一个最适合应用环境的物理结构和存取方法的过程，其目的是为了提高数据库的访问速度并有效地利用存储空间。

(1) 聚簇设计 聚簇是将有关数据元组集中存放在一个物理块内或若干相邻物理块内或同一柱面内，以提高查询效率的数据存储结构。建立聚簇有如下原则：

1. 当对一个关系的某些属性列的访问是该关系的主要应用，而对其他属性的访问很少或是次要应用时，可以考虑对该关系在这些属性列上建立聚簇。
2. 如果一个关系在某些属性列上的值重复率很高，则可以考虑对该关系在这些组属性列上建立聚簇。
3. 如果一个关系一旦装入数据，某些属性列的值很少修改，也很少增加或删除元组，则可以考虑对该关系在这些组属性列上建立聚簇。

由于在 SQL server2016 中，每张基本表就已经默认在主键属性上建立聚簇索引，且由于每张基本表上，最多只能建立一个聚簇索引，所以不需要再进行聚簇索引的设计。

(2) 分区设计设计 磁盘分区设计的本质是确定数据库数据的存放位置，其目的是提高系统性能，是数据库物理设计的内容之一。

磁盘分区设计的一般原则：

1. 减少访问冲突，提高 I/O 并发性。多个事物并发访问同一磁盘时，会产生磁盘访问冲突而导致效率低下，如果事务访问数据均能分布于不同磁盘上，则 I/O 可并发执行，从而提高数据库访问速度。
2. 分散热点数据，均衡 I/O 负担。在数据库中数据访问的频率是不均匀的，那些经常被访问的数据成为热点数据，此类数据宜分散存在于不

同的磁盘上，以均衡各个磁盘的负荷，充分发挥多磁盘的并行操作的优势。

3. 保证关键数据快速访问，缓解系统瓶颈。在数据库中有些数据如数据字典等的访问频率很高，为保证对它的访问不直接影响整个系统的效率，可以将其存放在某一固定磁盘上，以保证其快速访问。

本实验的网上超市购物系统由于程序较小，所以不进行分区设计。

3.3 系统模块 IPO 表

3.3.1 顾客登录主页模块

(1) 功能描述

对登录系统的顾客进行身份验证或对需要注册的顾客提供注册服务。

(2) 模块 IPO 表

系统名称	网上书店购物系统		
模块名称	顾客登录	模块编号	01
作者		日期	2018/12/26
模块描述	对登录系统的顾客进行身份验证或对需要注册的顾客提供注册服务。		
调用模块	数据库模块		
被调用模块	无		
输入	项目 帐号，密码 格式 帐号：char 密码：char		
处理	1、顾客输入帐号、密码 2、模块获得帐号、密码等信息 3、模块调用数据库表进行比较校验 4、模块返回登录信息		
输出	用户登录成功/失败。		
局部数据元素	数据库表		

系统名称	网上书店购物系统
注释	需要数据库信息表

(3) 主要算法

【登录】按钮：验证顾客的合法性。

【取消】按钮：关闭顾客登录/注册窗口。

【注册】按钮：弹出用户注册功能窗口。

【忘记密码】按钮：弹出用户找回密码功能窗口。

3.3.2 店员登陆主页模块

(1) 功能描述

对登陆系统的店员进行身份验证

(2) 模块 IPO 表

系统名称	大学校园网上书店系统		
模块名称	员工登陆	模块编号	02
作者	沈洋	日期	2018/12/26
模块描述	用于系统对员工的登陆进行身份验证		
调用模块	数据库模块		
被调用模块	无		
输入	项目 格式	用户名，密码，验证码 用户名：char 密码：char 验证码： char	
处理	1. 用户输入用户名，密码 2. 模块获得用户名密码等信息 3. 模块调用数据库表进行比较 4. 模块返回登陆信息		
输出	用户登陆成功或登陆失败		

系统名称	大学校园网上书店系统
局部数据元素	数据库表
约束条件	验证码
注释	需要数据库员工信息表

(3) 主要算法

【登陆】按钮：验证用户的合法性

【取消】按钮：关闭用户登录界面

【注册】按钮：弹出新用户注册功能窗口

3.3.3 管理员登录主页模块

(1) 功能描述

对登录系统的管理员进行身份验证。

(2) 模块 IPO 表

系统名称	网上书店购物系统		
模块名称	管理员登录	模块编号	03
作者		日期	2018/12/26
模块描述	用于系统对管理员的登录进行身份验证		
调用模块	数据库模块		
被调用模块	无		
输入	项目	管理员用户名，密码	
	格式	管理员用户名：char，管理员密码：char	
处理	1、用户输入用户名、密码 2、模块获得用户名、密码等信息 3、模块调用数据库表进行比较校验 4、模块返回登录信息		
输出	用户登录成功或登录失败。		

系统名称	网上书店购物系统
局部数据元素	数据库表
注释	需要数据库信息表

(3) 主要算法

【登录】按钮：验证用户的合法性。

【取消】按钮：关闭用户登录窗口。

【注册】按钮：弹出用户注册功能窗口。

3.3.4 顾客注册页面登录主页模块

(1) 功能描述

对登录系统的顾客进行身份验证或对需要注册的顾客提供注册服务。

(2) 模块 IPO 表

系统名称	网上书店购物系统		
模块名称	顾客注册	模块编号	04
作者		日期	2018/12/26
模块描述	对登录系统的顾客进行身份验证或对需要注册的顾客提供注册服务。		
调用模块	数据库模块		
被调用模块	无		
输入	项目	帐号，密码，电话，默认地址，性别	
	格式	帐号：char；密码：char；电话：char；默认地址：char；性别：char	
处理	1、顾客输入注册信息 2、模块调用数据库表进行比较校验 3、模块返回登录信息		
输出	用户注册成功/失败。		

系统名称	网上书店购物系统
局部数据元素	数据库表
注释	需要数据库信息表

(3) 主要算法

【确认】按钮：验证顾客注册信息的合法性，合法则弹出注册成功页面。

【取消】按钮：关闭顾客注册窗口。

3.3.5 店员注册模块

(1) 功能描述

用于创建店员账号，并对店员基本信息初始化，如姓名，性别，密码

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	员工注册	模块编号	05
作者	沈洋	日期	2018/12/26
模块描述	用于员工注册账号		
调用模块	数据库模块		
被调用模块	无		
输入	项目	用户名，密码，联系方式，性别，姓名，书店名，书店地址	
	格式	用户名：char 密码：char 联系方式：char 性别：string 姓名：char 书店名：char 书店地址：char	
处理	1. 员工进入注册账号界面 2. 员工输入用户名，密码等信息 3. 系统接收信息进行检验，符合要求后写入数据库 4. 系统弹出注册成功的对话框		
输出	注册成功对话框		

系统名称	大学校园网上书店系统
局部数据元素	数据库表
约束条件	无
注释	需要数据库店员表

(3) 主要算法

【确定】按钮：员工账号注册完成

【取消】按钮：关闭注册用户界面

3.3.6 顾客个人中心功能模块

(1) 功能描述

用于顾客查看自己信息并进行修改。

(2) 模块 IPO 表

系统名称	网上书店系统		
模块名称	顾客管理顾客信息	模块编号	06
作者		日期	2018/12/26
模块描述	用于顾客对自己信息进行修改		
调用模块	数据库模块		
被调用模块	无		
输入	项目	顾客帐号，顾客密码，顾客姓名	
		顾客性别，顾客联系方式，顾客积分，地址	
	格式	顾客帐号：char 顾客密码：char 顾客姓名：char	
		顾客性别：char 顾客联系方式：char 顾客积分：int 顾客课程书目录：char 地址：char	

系统名称	网上书店系统
处理	1、顾客登录系统进入顾客界面 2、顾客修改自身信息 3、顾客添加自身信息 4、顾客提交更改后的自身信息 5、系统接收信息写入数据库中 6、系统弹出更新成功对话框
输出	更新成功对话框
约束条件	账号密码与数据库内信息匹配
注释	需要数据库 user 表、admin 表

(3) 主要算法

【修改】按钮：顾客修改自身信息。

【添加】按钮：顾客添加自身信息。

【提交】按钮：顾客提交更改的信息。

3.3.7 店员修改个人信息模块

(1) 功能描述

用于店员对个人信息的修改

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	店员信息修改	模块编号	07
作者	沈洋	日期	2018/12/26
模块描述	用于店员对自己个人信息的修改		
调用模块	数据库模块		
被调用模块	店员个人信息显示模块		
输入	项目	密码，联系方式，姓名，书店名，书店地址	

系统名称	大学校园网上书店系统		
处理	格式	密码: char 联系方式: char 姓名: char 书店名: char 书店地址: char	
输出	1、店员登录进入个人信息模块 2、店员选择要更改的个人信息 3、店员进行个人信息修改并提交 4、系统接收信息进行校验,符合要求后写入数据库 5、系统弹出信息修改成功对话框		
约束条件	无		
注释	需要数据库员工信息表		

3.3.8 管理员修改个人信息功能模块

(1) 功能描述

用于管理员对自己信息进行修改。

(2) 模块 IPO 表

系统名称	网上书店系统		
模块名称	管理员管理管理 员信息	模块编号	08
作者		日期	2018/12/26
模块描述	用于管理员对自己信息进行修改		
调用模块	数据库模块		
被调用模块	无		
输入	项目 格式	管理员用户名, 管理员密码, 管理员用户名: char, 管理员密码: char	

系统名称	网上书店系统
处理	1、管理员登录系统进入管理员界面 2、管理员修改自身信息 3、管理员提交更改后的自身信息 4、系统接收信息写入数据库中 5、系统弹出更新成功对话框
输出	更新成功对话框
约束条件	无
注释	需要数据库信息表

(3) 主要算法

【修改】按钮：管理员修改自身信息。

【提交】按钮：管理员提交更改的信息。

3.3.9 用户注销界面

(1) 功能描述

退出当前员工的登陆状态

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	用户信息修改	模块编号	09
作者	沈洋	日期	2018/12/26
模块描述	用于用户退出登陆		
调用模块	数据库模块		
被调用模块	无		
输入	项目	用户名	
	格式	用户名: char	
处理	1、用户登录进入服务界面 2、用户注销登录状态 3、系统返回登陆界面		

系统名称	大学校园网上书店系统
输出	注销成功界面
约束条件	无
注释	无

(3) 主要算法

【注销】按钮：注销用户

【取消】按钮：关闭注销界面

3.3.10 顾客浏览图书功能模块

(1) 功能描述

用于顾客浏览图书。

(2) 模块 IPO 表

系统名称	网上书店系统		
模块名称	顾客浏览图书功能模块	模块编号	10
作者		日期	2018/12/26
模块描述	用于顾客浏览图书		
调用模块	数据库模块		
被调用模块	无		
输入	项目	无	
处理	格式		
	1. 顾客搜索关键字		
	2. 顾客查看相关图书外观		
	3. 顾客点击查看详情		
	4. 顾客查看所选书详情		
	5. 顾客将书籍加入购物车		

系统名称	网上书店系统
输出	图书书名: char; 版本号: char; 出版社: char; 售价: float; 库存量: int; 折扣: char; 销售书店: char; 所在学校: char; 书籍号: char
约束条件	无
注释	需要数据库信息表

(3) 主要算法

【加入购物车】按钮: 顾客将书籍加入购物车。

【查看详情】按钮: 顾客查看书籍详情。

【搜索】按钮: 顾客搜索相关书籍。

3.3.11 购物车功能模块

(1) 功能描述

用于顾客查看及修改购物车信息。

(2) 模块 IPO 表

系统名称	网上书店系统		
模块名称	购物车/付款页面	模块编号	11
作者		日期	2018/12/26
模块描述	用于顾客购买图书。		
调用模块	数据库模块		
被调用模块	无		
输入	项目	账号, 密码	
	格式	账号: char; 密码: char	
处理	1. 顾客查看购物车 2. 顾客修改书籍数量 3. 顾客点击付款		
输出	付款码/支付成功 or 失败		

系统名称	网上书店系统
约束条件	无
注释	需要数据库信息表

(3) 主要算法

【付款】按钮：弹出付款页面

【查看订单】按钮：跳转订单页面

3.3.12 付款功能模块

(1) 功能描述

用于顾客购买图书。

(2) 模块 IPO 表

系统名称	网上书店系统		
模块名称	购物车/付款页面	模块编号	12
作者		日期	2018/12/26
模块描述	用于顾客购买图书。		
调用模块	数据库模块		
被调用模块	无		
输入	项目	书名，数量，单价，折扣，总价，积分、手机	
	格式	书名：char；数量：int；单价：float；折扣：float；总价：float；积分：int；手机：char	
处理	1. 用户是否修改地址等收件人信息 2. 用户确认信息 3. 弹出付款二维码 4. 用户付款 5. 付款完成		
输出	付款码/支付成功 or 失败		
约束条件	无		
注释	需要数据库信息表		

系统名称	网上书店系统
------	--------

(3) 主要算法

【付款】按钮：弹出付款二维码。

【修改信息】按钮：修改收件人信息。

3.3.13 订单模块

(1) 功能描述

用于顾客查看订单并做售后相关事务。

(2) 模块 IPO 表

系统名称	网上书店系统		
模块名称	订单	模块编号	13
作者		日期	2018/12/26
模块描述	用于顾客查看订单并做售后相关事务。		
调用模块	数据库模块		
被调用模块	无		
输入	项目	账号，密码	
	格式	账号：char；密码：char	
处理	1. 通过登陆验证 2. 顾客查看已购买书籍订单 3. 顾客查看未送达订单的物流进度 4. 顾客选择订单进行售后处理 5. 顾客对店家进行评价		
输出	订单详情（订单表）、物流详情、售后页面、评价页面		
约束条件	无		
注释	需要数据库信息表		

(3) 主要算法

【查看详情】按钮：顾客查看订单详情。

【查看物流】按钮：顾客查看物流信息。

【申请退款】按钮：顾客申请退款。

【评价】按钮：顾客对店家进行评价。

3.3.14 店员查看图书模块

(1) 功能描述

员工查看现有图书的信息

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	店员查看图书信息	模块编号	14
作者	沈洋	日期	2018/12/26
模块描述	用于店员查看系统中已有图书信息		
调用模块	数据库模块		
被调用模块	无		
输入	项目	用户名	
	格式	用户名：char	
处理	1、店员登录进入员工界面 2、店员选择查看图书信息按钮 3、系统跳转至查看图书信息界面 4、店员输入要查看书籍的书籍号或点击书籍查看书籍基本信息		
输出	书籍号，书名，版本号，出版社，售价，库存量		
约束条件	无		
注释	需要数据库图书信息表		

(3) 主要算法

【查询】按键：进入查询图书信息界面查看图书信息

【取消】 按键：关闭查询界面

3.3.15 店员修改图书信息模块

(1) 功能描述

店员修改当前已有图书的信息

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	店员修改图书信息	模块编号	15
作者	沈洋	日期	2018/12/26
模块描述	用于店员修改系统中已有图书信息		
调用模块	数据库模块		
被调用模块	无		
输入	项目	书籍号，书名，版本号，出版社，售价，库存量，	
	格式	书籍号：char，书名：char，版本号：char，出版社：char，售价：float，库存量：int	
处理	1、店员登录进入员工界面 2、店员选择修改图书信息按钮 3、系统跳转至修改图书信息界面 4、店员输入要修改书籍的书籍号或点击书籍显示书籍基本信息 5、店员输入要修改的信息，点击确认 6、系统弹出修改成功的对话框		
输出	修改成功的对话框		
约束条件	书籍号无法修改		
注释	需要数据库图书信息表		

(3) 主要算法

【修改】 按键：进入修改图书信息界面

【确认】 按键： 确认图书新修改的信息

【取消】 按键： 关闭图书信息修改界面

3.3.16 店员添加书籍信息模块

(1) 功能描述

店员添加新的书籍信息

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	店员添加图书信息	模块编号	16
作者	沈洋	日期	2018/12/26
模块描述	用于店员添加系统中没有的图书信息		
调用模块	数据库模块		
被调用模块	无		
输入	项目	书籍号，书名，版本号，出版社，售价，库存量，	
	格式	书籍号：char，书名：char，版本号：char，出版社：char，售价：float，库存量：int	
处理	1、店员登录进入员工界面 2、店员选择添加图书信息按钮 3、系统跳转至添加图书信息界面 4、店员输入要添加书籍的书籍号和书名等基本信息 5、店员添加书籍信息，点击确认 6、系统弹出书籍信息添加成功对话框		
输出	添加成功对话框		
约束条件	书籍号不能重复		
注释	需要数据库图书信息表		

(3) 主要算法

【添加】 按键： 进入添加图书信息界面

【确认】 按键： 确认添加的图书信息

【取消】 按键： 关闭图书信息添加界面

3.3.17 店员删除书籍信息模块

(1) 功能描述

店员删除已有书籍信息

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	店员删除图书信息	模块编号	17
作者	沈洋	日期	2018/12/26
模块描述	用于店员删除系统中已有的图书信息		
调用模块	数据库模块		
被调用模块	无		
输入	项目 格式	书籍号 书籍号: char	
处理	1、店员登录进入员工界面 2、店员选择删除图书信息按钮 3、系统跳转至删除图书信息界面 4、店员输入要删除书籍的书籍号 5、店员删除书籍信息，点击确认 6、系统弹出书籍信息删除成功对话框		
输出	删除成功对话框		
约束条件			
注释	需要数据库图书信息表		

(3) 主要算法

【删除】 按键： 进入删除图书信息界面

【确认】 按键： 确认删除的图书信息

【取消】 按键：关闭图书信息删除界面

3.3.18 店员确认专业课程书籍信息模块

(1) 功能描述

员工确认专业课程书籍的信息

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	员工确认书籍信息	模块编号	18
作者	沈洋	日期	2018/12/26
模块描述	用于员工确认专业课程书籍的信息		
调用模块	数据库模块		
被调用模块	无		
输入	项目 格式	用户名 用户名：char	
处理	1、员工登录进入员工界面 2、员工选择确认专业书籍信息按钮 3、系统跳转至确认专业书籍信息界面 4、员工点击确认出售按钮 5、系统弹出确认成功的对话框		
输出	确认成功的对话框		
约束条件	无		
注释	需要专业课程书单		

(3) 主要算法

【确认】 按键：确认专业图书信息

【出售】 按键：确定出售专业图书

3.3.19 店员审查订单模块

(1) 功能描述

员工确认订单

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	员工确认订单	模块编号	19
作者	沈洋	日期	2018/12/26
模块描述	用于员工确认订单		
调用模块	数据库模块		
被调用模块	无		
输入	项目	用户名, 订单号	
	格式	用户名: char 订单号: char	
处理	1、员工登录进入订单界面 2、点击查询历史订单按钮查询历史订单 3、员工点击接单订单按钮 4、系统弹出接单成功的对话框及修改数据库		
输出	接单成功的对话框或输出历史订单		
约束条件	无		
注释	需要订单表和图书表		

(3) 主要算法

【历史订单】按钮: 查询历史订单

【确认接单】按钮: 同意接单

【取消】按钮: 拒绝接单

3.3.20 店员审查退单模块

(1) 功能描述

员工确认退单

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	店员确认退单	模块编号	20
作者	沈洋	日期	2018/12/26
模块描述	用于店员确认退单		
调用模块	数据库模块		
被调用模块	无		
输入	项目	点击按钮，	
	格式	无	
处理	1、店员登录进入退单界面 2、点击查询历史订单按钮查询历史退单 3、店员点击同意退款按钮 4、系统弹出退款成功的对话框及修改数据库		
输出	退款成功的对话框或输出历史退单		
约束条件	无		
注释	需要订单表和图书表		

(3) 主要算法

【历史退单】按钮：查询历史退单

【确认退款】按钮：同意退款

【取消】按钮：拒绝退款

3.3.21 店员查看报表模块

(1) 功能描述

店员查看报表

(2) 模块 IPO 图

系统名称	大学校园网上书店系统		
模块名称	员工查看报表	模块编号	21
作者	沈洋	日期	2018/12/26

系统名称	大学校园网上书店系统		
模块描述	用于店员查看报表		
调用模块	数据库模块		
被调用模块	无		
输入	项目 格式	用户名 用户名: char	
处理	1、店员登录进入查看报表界面 2、点击查询历史报表按钮查询历史报表 3、系统弹出查询的报表		
输出	输出查询的报表		
约束条件	无		
注释	需要订单表		

(3) 主要算法

【历史报表】按钮：查询历史报表

【取消】按钮：关闭查看界面

3.3.22 管理员查看订单功能模块

(1) 功能描述

用于管理员掌握用户和店家的交易信息。

(2) 模块 IPO 表

系统名称	网上书店系统		
模块名称	管理员查看订单信息	模块编号	22
作者		日期	2018/12/26
模块描述	用于管理员掌握用户和店家的交易信息。		
调用模块	数据库模块		
被调用模块	无		

系统名称	网上书店系统		
输入	项目	订单号	
	格式	订单号: char	
处理	1、管理员登录系统进入管理员界面 2、管理员查看订单信息 3、系统从数据库中提取所需信息		
输出	订单信息		
约束条件	无		
注释	需要数据库信息表		

(3) 主要算法

【查看】按钮：管理员查看订单信息

【搜索】输入框：管理员查询所需信息

3.3.23 管理员管理店员信息功能模块

(1) 功能描述

用于管理员对店员信息进行查询、修改、删除账户。

(2) 模块 IPO 表

系统名称	网上书店系统		
模块名称	管理员管理店员信息	模块编号	23
作者		日期	2018/12/26
模块描述	用于管理员对店员信息进行查询、更改、删除账户		
调用模块	数据库模块		
被调用模块	无		
输入	项目	店员用户名	
	格式	店员用户名: char	

系统名称	网上书店系统
处理	1、管理员登录系统进入管理员界面 2、管理员修改店员信息 3、管理员添加店员信息 4、管理员提交更改后的店员信息 5、管理员删除店员账号 6、系统接收信息写入数据库中 7、系统弹出更新成功对话框
输出	更新成功对话框
约束条件	无
注释	需要数据库信息表

(3) 主要算法

【修改】按钮：管理员修改店员信息。

【添加】按钮：管理员添加店员信息。

【删除】按钮：管理员删除店员账号。

【提交】按钮：管理员提交更改的信息。

4 详细设计

详细设计阶段的根本目标是确定应该怎样具体地实现所要求的系统，即不仅在逻辑上正确地实现每个模块的功能，更重要的是设计出的处理过程应该尽可能简明易懂。描述程序处理过程的工具称为过程设计的工具，它们可以分为程序流程图、盒图、问题分析图（PAD）图等，由于这些工具各有其优缺点，我们将在下面使用各种不同的过程设计工具对各个模块进行详细设计。

4.1 顾客注册模块

该模块用于顾客进行注册本网上书店系统

图 4.1 顾客注册模块

伪代码设计

Begin

输入用户名，密码，手机号，性别，默认地址；

If(用户名/密码 == " ") 则给出相应提示信息；

Else 连接数据库

If(信息已存在) 则提示信息已存在，让用户重新输入

Else 提示注册成功，进入主界面

End

4.2 顾客个人中心模块

用于顾客查看自己信息并进行修改。

图 4.2 个人信息流程图

伪代码设计

Begin

验证顾客登陆合法性；

If (顾客点击 == “查询个人信息”)

连接数据库；

显示相关个人信息；

If (顾客点击 ==“确认”)

Returnbegin;

If (顾客点击 ==“修改个人信息”)

If (验证合法) 连接数据库；

允许用户修改部分信息；

If (顾客点击 ==“确认修改”)

验证修改后信息合法性；

If (合法) 提示修改完成；

Returnbegin;

Else 提示错误，returnbegin;

End

4.3 顾客浏览图书模块

用于顾客浏览图书。

图 4.3 顾客浏览图书流程图

伪代码设计

Begin

显示畅销书籍；

If (顾客输入搜索 ==“书名”)

连接至数据库；

显示相关书籍；

If（顾客点击 ==“某一书籍”）

显示相关书籍详情；

If（顾客点击 ==“加入购物车”）

反馈加入购物车成功；

提醒顾客购买；

Return 书籍详情页面；

If（顾客点击 ==“返回”）

返回相关书籍页面；

End

4.4 顾客查看订单功能模块

用于顾客查看订单。

图 4.4 顾客查看订单流程图

伪代码设计

Begin

进入顾客界面；

点击查看订单按钮；

进入订单选择界面；

点击想查看的订单；

进入详情界面；

点击确定；

End

4.5 顾客查看购物车功能模块

用于顾客查看购物车。

图 4.5 顾客查看购物车流程图

伪代码设计

Begin

进入顾客界面；

点击查看购物车按钮；

进入商品选择界面；

点击想查看的商品；

If（点击 ==“购买”

跳转付款模块

End

4.6 顾客付款功能模块

用于顾客付款。

图 4.6 顾客付款流程图

伪代码设计

Begin

生成收款二维码；

If（顾客付款成功）

反馈谢谢惠顾；

Else

Return 生成收款二维码；

End

4.7 店员修改个人信息功能模块

该模块用于实现店员修改个人信息服务

图 4.7 店员修改个人信息流程图

伪代码设计

Begin

进入修改信息模块

Switch case(选择要修改的个人信息)

输入个人信息并提交

If(修改信息是否符合标准)

{上传数据库}

Else 返回修改界面

弹出修改成功界面

4.8 店员注册功能模块

该模块用于实现店员的平台注册

图 4.8 店员注册流程图

伪代码设计

Begin

店员用户注册

输入注册信息

if(注册信息是否符合标准)

上传数据库

Else 返回注册界面

显示等待审批界面

If(审批通过)

{写入数据库}

弹出注册成功消息提示

4.9 店员查看图书信息功能模块

该模块用于店员查看图书的基本信息

图 4.9 店员查看图书信息流程图

伪代码设计

Begin

进入查看图书信息模块

While(是否继续查询)

{

Switch case()

1:{

输入书籍号查看信息

If(书籍号是否存在)

显示书籍信息

Else{重新输入书籍号}

}

2:

{

显示所选取的书籍的基本信息

}

}

4.10 店员查看订单/退单信息功能模块

该模块用于店员查看订单/退单的信息

图 4.10 店员查看订单/退单信息盒图

伪代码设计

Begin

进入查看订单退单信息模块

Switch case(选择进入订单/退单信息)

1:{

进入订单信息模块

查询订单信息

确认订单信息

上传数据库并显示接单成功通知

}

2:{

进入退单信息模块

查询退单信息

确认退单信息

上传数据库并显示退单成功通知

}

4.11 用户注销登陆模块

该模块用于实现用户在平台上的登陆注销服务

图 4.11 用户注销登录状态模块 PAD 图

伪代码设计

```
BEGIN
进入用户页面；
点击注销；
IF（确认）
{
清除用户登录状态；
反馈注销成功；
}
ELSE
{
返回用户界面；
}
END
```

4.12 店员确认专业课程书单信息功能模块

该模块用于店员确定是否出售专业课程相关图书

图 4.12 店员确认专业课程书单信息模块盒图

伪代码设计

```
BEGIN
点击查看专业课程书单;
IF(点击确认销售)
{
连接数据库;
IF (库存 != 0)
{
反馈上架成功信息;
}
ELSE
{
上架失败;
}
}
ELSE
{
返回查看专业课程书单页面;
}
END
```

4.13 店员生成报表并查看报表

该模块用于实现店员生成并查看报表服务

图 4.13 店员查看报表模块流程图

伪代码设计

BEGIN

点击查看报表;

进入报表页面;

点击生成报表;

反馈报表信息;

END

4.14 用户登录模块

该模块用于顾客、员工、管理员使用账号

图 4.14 登陆模块流程图

伪代码设计

Begin

选择登陆身份;

输入用户名, 密码;

if(用户名/密码 == " ") 则给出相应提示信息;

else 连接数据库

if(信息不匹配) 则提示用户名或者密码错误, 让用户重新输入

else 提示登陆成功, 进入主界面

End

4.15 管理员修改个人信息模块

该模块用于管理员对自己信息进行修改。

图 4.15 管理员修改个人信息模块流程图

伪代码设计

Begin

进入管理员界面；

点击修改信息按钮；

进入修改信息界面；

修改信息；

if(修改成功) 反馈修改成功信息；

else 重新修改；

End

4.16 管理员查看订单功能模块

该模块用于管理员掌握用户和店家的交易信息。

图 4.16 管理员查看订单信息模块流程图

伪代码设计

Begin

进入管理员界面；

点击查看订单按钮；

进入订单选择界面；

点击想查看的订单；

进入详情界面；

点击确定；

End

4.17 管理员管理店员信息功能模块

该模块用于管理员对店员信息进行查询、修改、删除账户。

图 4.17 管理员管理店员信息模块流程图

伪代码设计

Begin

Switch 点击操作

Case 点击查询店员信息：

{

跳转至店员查询界面；

点击店员信息查询按钮；

查看店员信息；

点击确认；

无反馈；

}

Case 点击修改店员信息：

{

跳转至修改店员界面；

点击店员信息修改按钮；

修改店员信息；

点击确认；

反馈修改成功信息；

}

Case 点击删除店员信息：

{

跳转至删除店员界面；

点击店员删除界面；

删除店员；

点击确认；

反馈修改成功信息；

}

5 系统实现

5.1 开发平台和开发环境介绍

开发平台：Eclipse8.0

开发环境：Eclipse 8.0+MySQL 5.6.41

(1)Eclipse 是一个开放源代码的、基于 Java 的可扩展开发平台。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。是目前最流行的 Windows 平台应用程序开发环境之一。Eclipse 附带了一个标准的插件集，包括 Java 开发工具（Java Development Kit，JDK）。Eclipse 还包括插件开发环境（Plug-in Development Environment，PDE），这个组件主要针对希望扩展 Eclipse 的软件开发人员，因为它允许他们构建与 Eclipse 环境无缝集成的工具。由于 Eclipse 中的每样东西都是插件，对于给 Eclipse

提供插件，以及给用户提供一个一致和统一的集成开发环境而言，所有工具开发人员都具有同等的发挥场所。

本实验中开发的网上书店系统就是在 Eclipse8.0 平台上采用 Java 语言开发的。

Java 是一门面向对象编程语言，不仅吸收了 C++ 语言的各种优点，还摒弃了 C++ 里难以理解的多继承、指针等概念，因此 Java 语言具有功能强大和简单易用两个特征。Java 语言作为静态面向对象编程语言的代表，极好地实现了面向对象理论，允许程序员以优雅的思维方式进行复杂的编程。

(2)MySQL 是一种关系数据库管理系统，由瑞典 MySQL AB 公司开发，目前属于 Oracle 旗下产品，是最流行的关系型数据库管理系统之一。MySQL 软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库。

运行环境：Tomcat 9.0.10

大学书店网上书店系统属于 Web 应用系统，运行时需要投放在相应服务器上，方可在浏览器中使用。对于正式投放市场的情况，平台需额外租用服务器投放系统程序来满足用户访问的条件。而对于当前的演示部分，用户仅需使用 Tomcat 应用软件（一个免费的开放源代码的 Web 应用服务器）即可在本地网页访问该平台。在测试时，用户需首先需要在本地 MySQL 数据库中建立 projectdb 数据库，该过程可以在命令行中操作，具体操作如下所示：

//启动服务器 (需在管理员权限下执行)

```
net start mysql
```

//打开数据库

```
mysql -u root -p
```

//输入密码

.....

//创建数据库

```
create database projectdb
```

然后, 在该源代码文件中的 “applicationContext.xml” 文件里修改本地 MySQL 的密码, 其修改操作如下所示, 其中 value 中的值即为密码:

```
<propertyname="password"value="ymj15345"/>
```

保存修改后, 将程序解压导入 Eclipse 中, 在其中配置相关的 Tomcat 服务器后, 点击启动按钮, 在本地浏览器中输入 “localhost:8080/InternetBookShop/index” 即可访问平台。Eclipse 中配置 Tomcat 详见网站 “<https://jingyan.baidu.com/article/851fbc378d19e03e1f15ab24.html>”

5.2 编码

通常把 Code 和 Test 统称为实现。

所谓 Code 就是把软件设计结果翻译成用某种程序设计语言书写的程序。作为软件工程过程的一个阶段, Code 是对设计的进一步具体化, 因此, 程序的质量主要取决于软件设计的质量。

5.3 前台界面与后台数据库连接代码

在 SSH 框架下实现的大学校园网上书店系统, 其数据库连接部分主要依靠 Hibernate 部件进行实现。在 Hibernate 中通过 “实体名 +.hbm.xml” 文件配置实体的基本属性, 其代码部分见附录; 然后, 利用 “applicationContext.xml” 文件读取实体配置信息并创建相应的数据库连接 (一种利用 Spring 部件代理 hibernate 连接数据库的方式)。

具体前端连接数据库的代码如下:

```
<?xmlversion="1.0"encoding="UTF-8"?>
```

```
<beansxmlns="http://www.springframework.org/schema/beans"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xmlns:context="http://www.springframework.org/schema/context"
```

```
xmlns:aop="http://www.springframework.org/schema/aop"
xmlns:tx="http://www.springframework.org/schema/tx"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-4.3.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.3.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-4.3.xsd
http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-4.3.xsd">
<context:component-scanbase-package="com.*"/>
<beanid="dataSource"class="com.mchange.v2.c3p0.ComboPooledDataSource"destroy-
method="close">
<propertyname="driverClass"value="com.mysql.jdbc.Driver"/>
<propertyname="jdbcUrl"value="jdbc:mysql://localhost:3306/projectdb?useUnicode=true&character
8"/>
<propertyname="user"value="root"/>
<propertyname="password"value="ymj15345"/>
<propertyname="initialPoolSize"value="1"/>
<propertyname="minPoolSize"value="1"/>
<propertyname="maxPoolSize"value="50"/>
<propertyname="maxIdleTime"value="60"/>
<propertyname="acquireIncrement"value="3"/>
```



```
<propertyname="idleConnectionTestPeriod" value="60"/>

</bean>

<beanid="sessionFactory" class="org.springframework.orm.hibernate5.LocalSessionFactoryBean">

<propertyname="dataSource" ref="dataSource"/>

<propertyname="hibernateProperties">

<props>

<propkey="hibernate.dialect">

org.hibernate.dialect.MySQL5Dialect

</prop>

<propkey="hibernate.hbm2ddl.auto">

update

</prop>

</props>

</property>

<propertyname="mappingResources">

<list>

<value>com/entity/Customers.hbm.xml</value>

<value>com/entity/Shops.hbm.xml</value>

<value>com/entity/Books.hbm.xml</value>

<value>com/entity/Orders.hbm.xml</value>

<value>com/entity/Imports.hbm.xml</value>

</list>
```

```
</property>

</bean>

<beanid="txManager"class="org.springframework.orm.hibernate5.HibernateTransactionManager">

<propertyname="sessionFactory"ref="sessionFactory"/>

</bean>

<tx:annotation-driventransaction-manager="txManager"/>

</beans>
```

5.4 各模块功能的实现。

5.4.1 主界面模块实现

建立大学生网上书店的主界面，左上角分别为登录入口和注册入口，右上角为购物车入口。

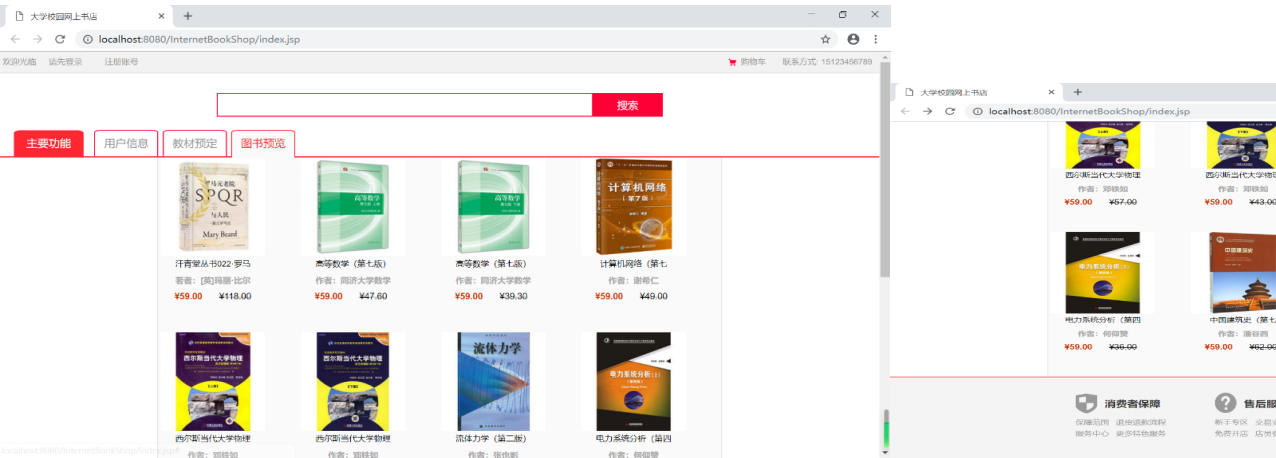


图 5.1 建立主界面模块实现效果图

5.4.2 注册模块实现

该模块用于实现顾客、店员的注册的功能，输入要写的信息，通过信息验证并申请提交后才能注册成功

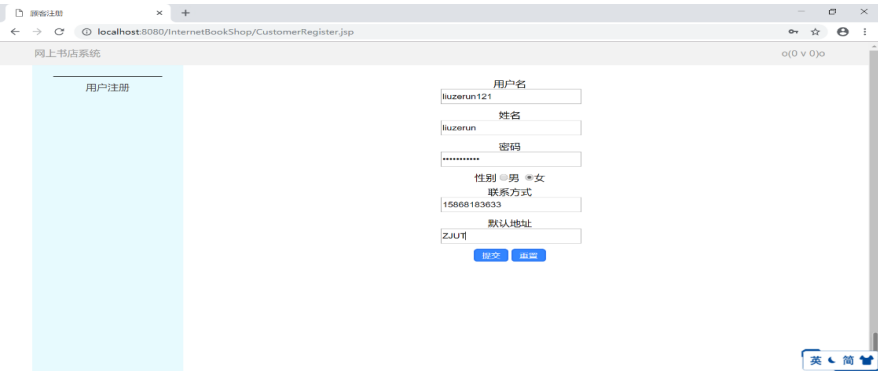


图 5.2 顾客注册模块实现效果图

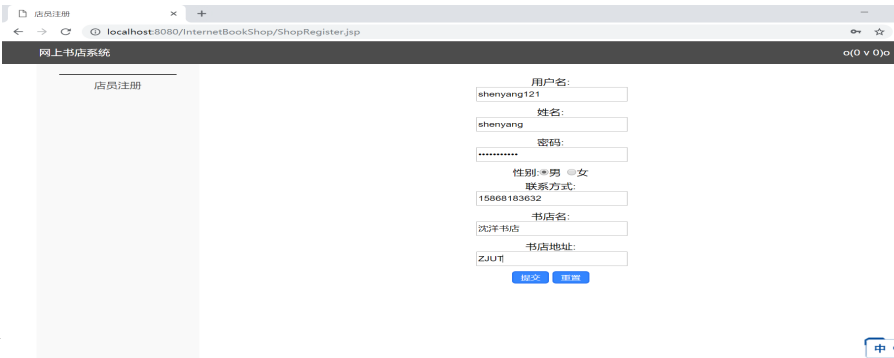


图 5.3 店员注册模块实现效果图

5.4.3 登陆模块实现

该模块用于实现顾客、店员登陆系统的功能，他们通过输入用户名、密码，并进行数据库校对，当输入的密码符合其用户名且输入正确时，可以成功进入本系统。登陆后左上角显示用户名与退出登陆接口，右上角有订单查询接口。



图 5.4 登陆模块实现效果图

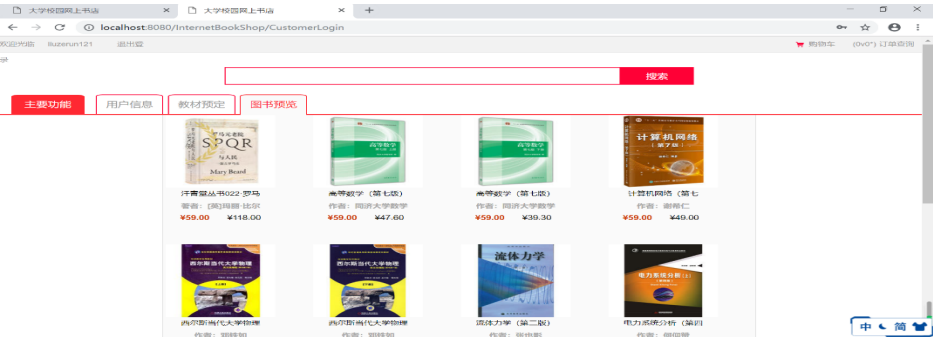


图 5.5 登陆后主界面模块实现效果图

5.4.4 顾客浏览图书模块实现

本模块实现了顾客浏览图书的功能，可以展示多种图书的图书名、售价。

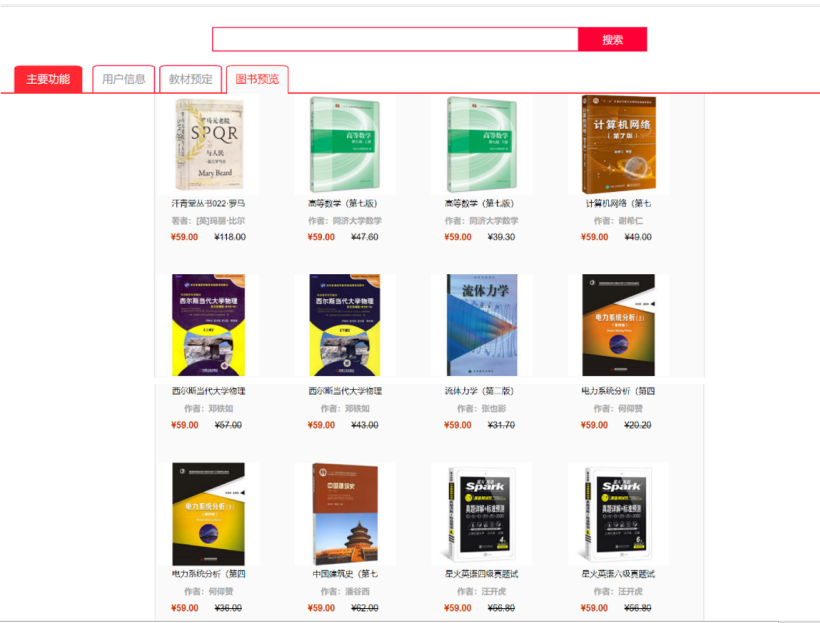


图 5.6 顾客浏览图书模块实现效果图

5.4.5 快速检索图书模块实现

该模块实现了用户快速检索图书的功能，当顾客输入书籍号以后，按下搜索按钮，页面中间就会返回该图书信息，点击查看详情后显示书籍的详细信息，加入购物车进行图书购买。当店员输入书籍号后，按下搜索按钮，页面中间就会返回该图书信息，方便店员进行图书信息修改。



图 5.7 顾客搜索图书模块实现效果图

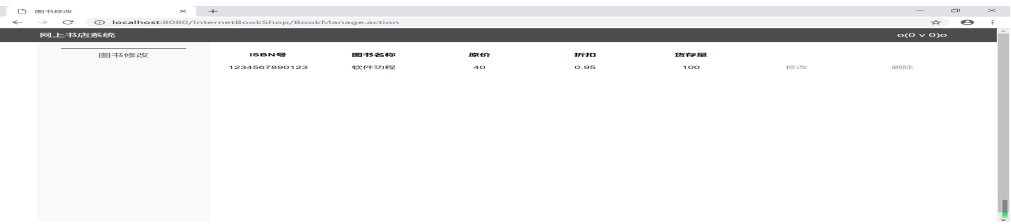


图 5.8 用户搜索图书模块实现效果图

5.4.6 用户注销登陆状态模块实现

该模块实现了用户注销自己登陆状态的功能。

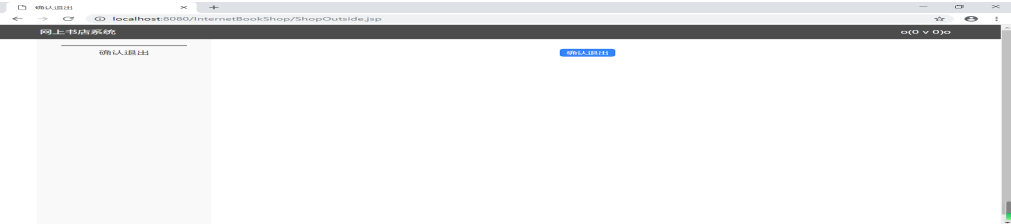


图 5.9 用户注销登陆状态模块实现效果图

5.4.7 顾客查看购物车模块实现

该模块用于查看已加入购物车中的书籍数量与信息，价格和与之相对应的订单号。确认信息下单后会在购物车一栏生成订单。

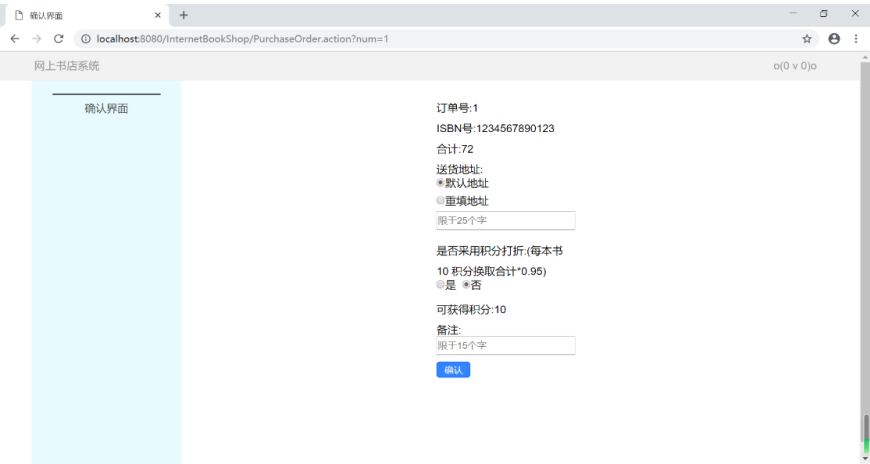


图 5.10 顾客查看购物车模块实现效果图



图 5.11 购物车生成订单模块实现效果图

5.4.8 顾客查询订单和申请退款模块实现

订单查询模块可现实当前已下订单的具体信息与物流状态。并可申请退款。退款可输入退款理由并提交申请由管理员审核完成退款。



图 5.12 顾客查询订单模块实现效果图

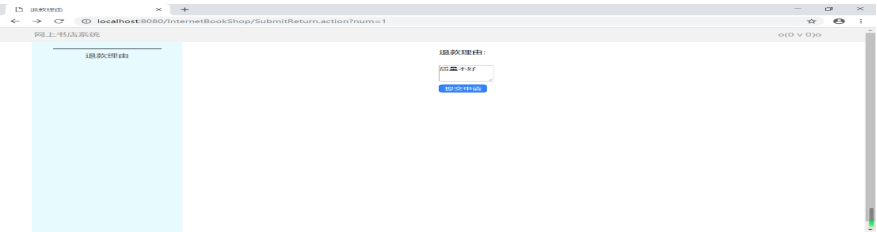


图 5.13 顾客输入退款理由模块实现效果图

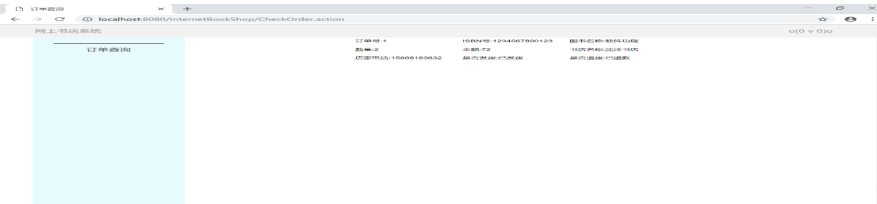


图 5.14 顾客查看退单模块实现效果图

5.4.9 用户修改个人信息模块实现

该模块用于用户修改个人信息，顾客可自行修改个人信息中的密码、联系方式、默认地址三项；店员可自行修改个人信息中的密码、联系方式、书店地址和书店名四项。点击修改后立即生效。

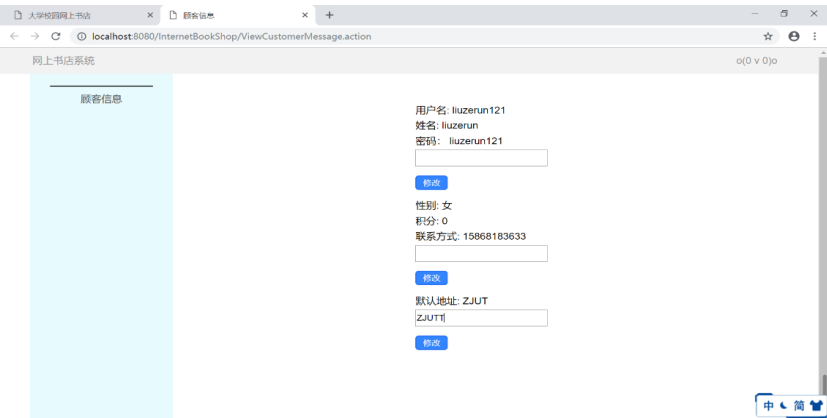


图 5.15 用户修改个人信息模块实现效果图

5.4.10 店员确认订单模块实现

该模块用于店员确认顾客提交的订单请求，包括“订单号”、“ISBN 号”、“图书名”、“数量”、“总价”、“顾客用户名”、“联系方式”等信息和“确认发货”的功能控件。



图 5.16 店员确认订单实现效果图

5.4.11 店员确认退单模块实现

该模块用于店员确认顾客提交的退单请求，包括“订单号”、“ISBN 号”、“图书名”、“数量”、“总价”、“顾客用户名”、“联系方式”等信息和“确认退货”、“拒绝退货”等功能控件。



图 5.17 店员确认退单实现效果图

5.4.12 店员管理图书信息模块实现

该模块用于店员管理图书的各类信息，包括“修改图书信息”、“添加图书”、“删除图书”等功能控件。修改图书信息实现了店员修改图书折扣和库存量的功能，增加图书信息实现了店员增加当前没有的图书的信息的功能，删除图书信息实现了店员删除当前已有图书信息的功能。

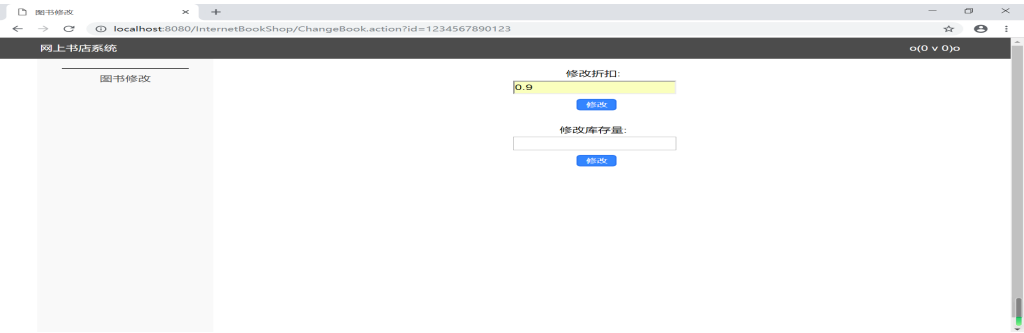


图 5.18 店员修改图书信息模块实现效果图

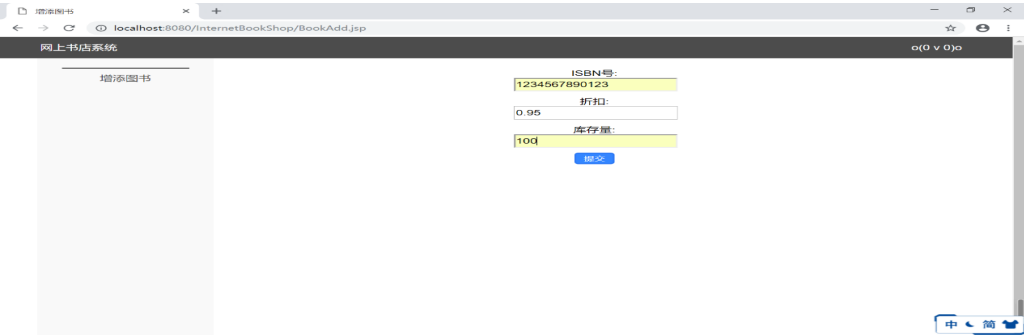


图 5.19 店员添加新图书基本信息模块实现效果图

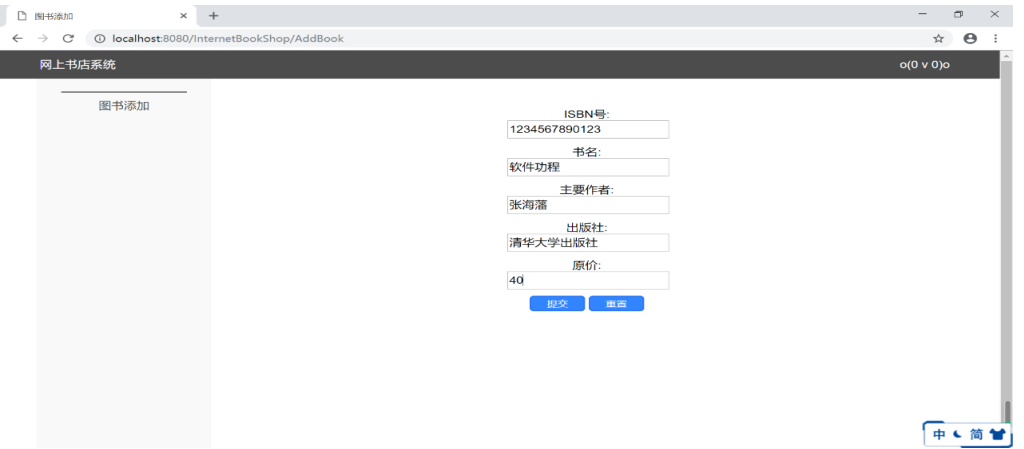


图 5.20 店员添加管理新图书具体信息模块实现效果图

5.4.13 书店报表生成模块实现

该模块用于生成书店销售报表以反映最近销售情况的功能,包括“ISBN 号”、“销售额”、“销量”等信息和“生成报表”、“返回主界面”等功能控件。查看报表页面可以通过输入时间来获得这段时间的销售报表信息,报表页面可以查看查询时间内销售的书的数量、种类和销售额。

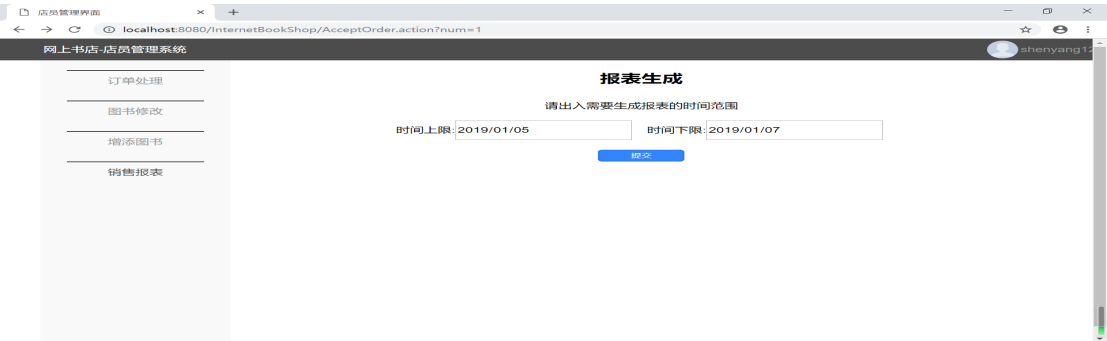


图 5.21 店员生成报表模块实现效果图

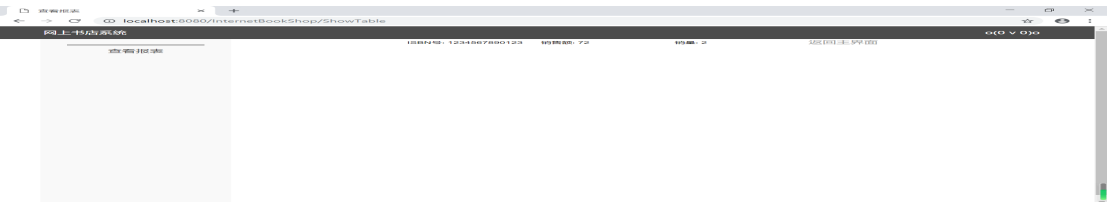


图 5.22 店员查看报表模块实现效果图

5.2 测试

5.2.1 软件测试的目标

软件测试是指对一个完成了全部或部分功能、模块的计算机程序在正式使用前的检测,以确保该程序能按预定的方式正确地运行。

有如下规则:

- 1. 测试是为了发现程序中的错误而执行程序的过程。

2. 好的测试方案是极可能发现迄今为止尚未发现的错误的测试方案
3. 成功的测试是发现了迄今为止尚未发现的错误的测试。

5.2.2 软件测试的步骤

测试过程必须分步骤进行，后一个步骤在逻辑上是前一个步骤的继续。大型软件系统通常由若干个子系统组成，每个子系统又由许多模块组成。从系统上来说，软件测试的方法主要包括单元测试，集成测试，系统测试，确认测试。

5.2.3 单元/集成/验收测试

单元测试（模块测试）

单元测试（模块测试）是开发者编写的一小段代码，用于检验被测代码的一个很小的、很明确的功能是否正确。通常而言，一个单元测试是用于判断某个特定条件（或者场景）下某个特定函数的行为。

单元测试的目的： 在于发现各模块内部可能存在的各种错误，主要是基于白盒测试。

- a) 验证代码是与设计相符合的；
- b) 发现设计和需求中存在的错误；
- c) 发现在编码过程中引入的错误。

单元测试策略： 孤立的测试策略、自顶向下、自底向上的单元测试策略

1) 孤立的测试策略：

· 方法：不考虑每个模块与其他模块之间的关系，为每个模块设计桩模块和驱动模块。每个模块进行独立的单元测试。

· 优点：该方法是最简单，最容易操作的。可以达到高的结构覆盖率。该方法是纯粹的单元测试。

- 缺点：桩函数和驱动函数工作量很大，效率低。

2) 自顶向下的单元测试策略：

- 方法：先对最顶层的单元进行测试，把顶层所调用的单元做成桩模块。其次对第二层进行测试，使用上面已测试的单元做驱动模块。如此类推直到测试完所有模块。

- 优点：可以节省驱动函数的开发工作量，测试效率较高。

- 缺点：随着被测单元一个一个被加入，测试过程将变得越来越复杂，并且开发和维护的成本将增加。

3) 自底向上的单元测试策略：

- 方法：先对模块调用层次图上最低层的模块进行单元测试，模拟调用该模块的模块做驱动模块。然后再对上面一层做单元测试，用下面已被测试过的模块做桩模块。以此类推，直到测试完所有模块。

- 优点：可以节省桩函数的开发工作量，测试效率较高。缺点：不是纯粹的单元测试，底层函数的测试质量对上层函数的测试将产生很大的影响。

单元测试是由程序员自己来完成，最终受益的也是程序员自己。可以这么说，程序员有责任编写功能代码，同时也就有责任为自己的代码编写单元测试。执行单元测试，就是为了证明这段代码的行为和我们期望的一致。

集成测试

集成测试（也叫组装测试，联合测试）是单元测试的逻辑扩展。它的最简单的形式是：两个已经测试过的单元组合成一个组件，并且测试它们之间的接口。从这一层意义上讲，组件是指多个单元的集成聚合。在现实方案中，许多单元组合成组件，而这些组件又聚合成程序的更大部分。方法是测试片段的组合，并最终扩展进程，将您的模块与其他组的模块一起测试。最后，将构成进程的所有模块一起测试。集成测试主要目的是针对详细设计中可能存在的问题，尤其是检查各单元与其它程序部分之间的接口上可能存在的错误。

集成测试策略：

1) 大爆炸集成 2) 自顶向下集成 3) 自底向上集成 4) 三明治（混合式）集成 5) 基于集成 6) 分层集成 7) 基于功能的集成 8) 基于消息的集成 9) 基于进度的集成 10) 基于风险的集成

系统测试是将经过测试的子系统装配成一个完整系统来测试。它是检验系统是否确实能提供系统方案说明书中指定功能的有效方法。系统测试主要针对概要设计，检查了系统作为一个整体是否有效地得到运行，例如在产品设置中是否达到了预期的高性能。

验收测试

验收测试是部署软件之前的最后一个测试操作。验收测试的目的是确保软件准备就绪，并且可以让最终用户将其用于执行软件的既定功能和任务。验收测试是向未来的用户表明系统能够像预定要求那样工作。经集成测试后，已经按照设计把所有的模块组装成一个完整的软件系统，接口错误也已经基本排除了，接着就应该进一步验证软件的有效性，这就是验收测试的任务，即软件的功能和性能如同用户所合理期待的那样。

5.2.4 测试过程

本系统采取以白盒测试方案为主，黑盒测试方案为辅的测试方法。先分别进行单元测试，再进行集成测试，遵循自底向上的策略。对于每一个测试用例都有考虑到合法和非法的测试情况。

5.2.4.1 单元测试（黑盒测试） 黑盒测试又称为功能测试、数据驱动测试或基于规格说明书的测试，是一种从用户观点出发的测试。测试人员一般把被测程序当作一个黑盒子。但是黑盒测试并不能取代白盒测试，它是与白盒测试互补的测试方法，它很可能发现白盒测试不易发现的其他类型的错误。

黑盒测试主要测到的错误类型有：不正确或遗漏的功能；接口、界面错误；性能错误；数据结构或外部数据访问错误；初始化或终止条件错误等等。

常用的黑盒测试方法有：等价类划分法；边界值分析法；因果图法；场景法；正交实验设计法；判定表驱动分析法；错误推测法；功能图分析法。

下面对各个模块进行测试如下：

用户注册模块测试：

表 5.1“注册模块” 输入条件等价类表

输入等价类	有效等价类	无效等价类
注册信息	1. 无空值且符合各项信息的输入要求	1. 无效用户名 2. 无效密码 3. 无效电话号码 4. 存在空值

表 5.22 有效等价类测试用例表

测试数据	期望结果	覆盖的有效等价类
1. 用户类型（顾客）用户名 (liuzerun123)、密码 (qwer1234)、电话号码 (15868809653)	输入有效	1

表 5.3 无效等价类测试用例表

测试数据	期望结果	覆盖的无效等价类
1.. 用户类型（顾客）用户名 (liuzerun)、密码 (qwer1234)、电话号码（15868809653）	无效输入	1
1.. 用户类型（顾客）用户名 (liuzerun123)、密码 (qwer)、电话号码（15868809653）	无效输入	2
1. 用户类型（顾客）用户名 (liuzerun123)、密码 (qwer1234)、电话号码 (84888324)	无效输入	3

测试数据	期望结果	覆盖的无效等价类
1. 用户类型（顾客）用户名 ()、 密码 (123456)、电话号码 (15868809653)	无效输入	4

用户登录模块测试：

表 5.4“登录模块” 输入条件等价类表

输入等价类	有效等价类	无效等价类
登录信息	1. 无空值且符合各项信息 的输入要求	1. 无效用户名 2. 无效密码 3. 存在空值

表 5.5 有效等价类测试用例表

测试数据	期望结果	覆盖的有效等价类
1. 用户类型（顾客）用户名 (liuzerun123)、密码 (qwer1234)	输入有效	1

表 5.6 无效等价类测试用例表

测试数据	期望结果	覆盖的无效等价类
1.. 用户类型（顾客）用户名 (liuzerun)、密码 (qwer1234)	无效输入	1
1.. 用户类型（顾客）用户名 (liuzerun123)、密码 (qwer)	无效输入	2
1. 用户类型（顾客）用户名 ()、 密码 (123456)	无效输入	3

店员导入图书模块测试：

表 5.7“店员导入图书模块测试” 输入条件等价类表

输入等价类	有效等价类	无效等价类
图书版本号	1. 无空值且为 13 位数字	1. 无效版本号 2. 存在空值

表 5.8 有效等价类测试用例表

测试数据	期望结果	覆盖的有效等价类
1234567890123	输入有效	1

表 5.9 无效等价类测试用例表

测试数据	期望结果	覆盖的无效等价类
ABCDEFGG	无效输入	1
1223124	无效输入	1
输入图书名为空	无效输入	2
123456789012A(13 位)	无效输入	1

5.3.3.2 集成测试（白盒测试） 设计测试方案是测试阶段的关键技术问题。所谓测试方案包括具体的测试目的，应该输入的测试数据和预取的结果。由于时间关系，小组对系统进行了部分功能的白盒测试。下面使用白盒测试方法对本系统进行测试。白盒测试包括：语句覆盖、判断覆盖、条件覆盖、判断/条件覆盖、条件组合覆盖、点覆盖、边覆盖、路径覆盖等。下面对本系统的几个模块进行测试：

登录模块的测试：

- 1. 选择用户类型，输入账号、密码，系统进行匹配。
- 2. 密码错误时，刷新页面，提醒用户密码需要重新输入。

3. 账号不存在时，刷新页面提醒用户账号需重新输入。

测试路径：

路径 1：1->2->3->4->6->7

路径 2：1->2->3->4->6->8

路径 3：1->2->3->4->5

图 5.23 登录模块白盒测试实现效果图

购物模块的测试：

1. 用户浏览图书时是否加入购物车
2. 用户将图书加入购物车后是否选择付款
3. 用户选择付款后是否使用积分

测试路径：

路径 1：1->2->3->4

路径 2：1->2->3->4->5->6->7

路径 3：1->2->3->4->5->6->7->8->9->10->11->12

路径 4：1->2->3->4->5->6->7->8->9->13->10->11->12

图 5.24 购物模块白盒测试实现效果图

5.3.3.3 系统测试与回归测试 完成了各个模块的测试后，要进行系统测试。系统测试是针对整个产品系统进行的测试，目的是验证系统是否满足了需求规格的定义，找出与需求规格不相符合或与之矛盾的地方。在系统测试的任何一个阶段，只要发现了错误，就要尽可能及时更正。更正后还要检验已经发现的缺陷有没有被正确的修改和修改过程中有没有引发新的缺陷，即

回归测试。另外，每当一个新的模块被当作集成测试的一部分加进来的时候，软件环境都很发生改变，即建立起新的数据流路径，还有可能激活了新的控制逻辑。这些改变可能会使原本工作得很正常的功能产生错误。因此在集成测试策略的环境中，要进行回归测试，就是对部分已通过测试的功能要再次进行测试，以保证系统在新环境下能正常工作。

5.3.3.4 系统测试结论 经过对该系统的需求分析的要求进行基本功能测试后，基本功能可以正常使用，数据库功能也正常，系统三种类型的用户互不干扰，数据更新未发生冲突，所以结果符合本大学校园网上书店系统的需求。

6 维护

在软件产品被开发出来并交付用户使用之后，就进入了软件的运行维护阶段。这个阶段是软件生命周期的最后一个阶段。

为了清除系统运行中发生的故障和错误，产品维护人员要对软件进行必要的修改与完善；为了使软件产品适应用户环境的变化，满足新提出的需要，也要对原软件产品做些局部的更新，这些工作称为软件的维护。其任务是改正软件系统在使用过程中发现的隐含错误，扩充在使用过程中用户提出的新的功能及性能要求，其目的是维护软件系统的“正常运作”。

6.1 系统维护过程

系统维护工作在整个系统生命周期中常常被忽视。人们往往热衷于系统开发，当开发工作完成以后，多数情况下开发队伍被解散或撤走，而在系统开始运行后并没有配置适当的系统维护人员。这样，一旦系统发生问题或环境发生变化，最终用户将无从下手，这就是为什么有些信息系统在运行环境中长期与旧系统并行运行不能转换，甚至最后被废弃的原因。

为了有效避免以上阐述的情况。我们在设计网上书店系统的时候主要从以下两个方面做好维护：

1. 对代码进行注释，便于开发维护人员理解
2. 提供说明文档，说明了开发、维护、更新的内容

每次在发布更新的时候，我们都要在说明文档中单独标记出更新的内容以方便使用者，让使用者更好的理解和使用新版本的软件。

6.2 系统维护策略

网上书店系统依赖于数据库数据的实时更新，一旦数据库的数据没有及时更新到每个对应的表或者出现触发器失效的情况，应该立即启用应急的措施。具体如下：

1. 建立一个备用数据库，在主数据库失灵的时候可以使用备用数据来保证系统的正确运行。
2. 备份数据库使用云备份
3. 软件在运行的时候每天自动从数据库中调用各项数据并保存在一个临时的文档中，每个小时都替换更新一次。
4. 每隔一周的时间会将云备份的数据转存到磁盘等固定设备上。

7 总结与体会

7.1 总结

通过为期 8 周的软件工程实验课程的训练，该团队五人在整个过程中都收获颇丰。我们小组每个成员都深刻意识到利用软件工程思想指导项目进行分析、设计、实现、测试和维护等过程中的必要性。8 周过后，由于时间的原因和系统的复杂性，我们虽然详细地完成了前期项目的所有设计过程，但并未实现该系统所具备全部功能，仅对其中的核心部分进行的编程实现。只能说略有遗憾，但是对于最后的成品我们还是非常满意的。我们基本实现了顾客的个人信息业务和购书业务以及店员的个人信息业务和图书基本销售业务的大部分功能，整个购书的操作流程还是可以在实现程序中有所体验。

该系统后端依托 JavaWeb 应用开发的 SSH 框架进行实现，可以处理用户在系统上的所有业务操作和相应的数据库操作；前端部分使用 HTML 标签+CSS 布局构成 JSP 文件，为用户提供视觉良好的服务界面。在该系统中还有必要的操作错误或输出信息错误的提示功能，可以帮助用户更好地掌握平台的基本使用。

7.2 体会

XXX 心得体会：

通过此次软件工程课程设计的训练，可以说，我个人对软件项目开发的过程更加清晰；对利用软件工程思想进行项目开发的流程掌握得更加熟练；对于软件工程这门课程对于项目设计开发的重要指导意义理解得也更加透彻。借助实验的形式，将理论付诸实践，不仅锻炼了我的实际操作能力和写作表达能力，同时也让我对之前所学的相关知识理解得更为深刻。

此处实验，我们小组选择的是完成大学校园网上书店系统的项目实现。在一开始的时候，大家并不是非常看重对于系统的可行性分析和需求分析部分，因为在那时大家总觉得如何编程实现才是我们在开发过程中研究的重点，因此，这一部分完成的非常粗略。然后，随着时间的推移，我们发现没有先前详细的分析设计会导致我们在实现过程中无休止的返工，从而造成大量的时间被浪费。作为组长的我果断调整计划，重新开始将需求分析清楚。而在此调整之后，我们确实发现返工的次数和程度比之前少了很多。而在此之后，我开始对大部分需要完成的项目步骤在正式开始之前都进行了具体的了解，包括它存在的作用、如何实现操作以及将各种相关的表示方式进行优缺点对比等等，我发现其实每一个步骤、每一种表示图在这个项目中都有它发挥作用的地方。它们并不是之前上课时所体会到的“好像没什么太大作用”，而是帮助这个项目更加顺利的去进行，并且尽可能地减少发生较大错误的可能性，例如 E-R 图的设计可以帮助我们快速地完成各种联系的基本表转化，硬想可能实体一多就搞不清哪些属性该是外键、哪些联系需要拆分。这是前人智慧的结晶啊，在此次软件工程项目中我深有体会，果然不停智者言，真得就是吃亏在眼前。

除此之外，由于个人队长的身份，在此次过程中我也深刻体会到团队协作的

重要性。如何使“1+1>2”，如何发挥队员的个人优势对于大工作量的项目是非常重要的。我非常感谢我的队友可以容忍我这几个礼拜的压榨，有些时候确实考虑的点实在太多，而且个人对项目进展的时间控制确实不当，造成后期有点来不及完成。同时，我也很自豪自己这次还算是担当了队长的重任。

最后，我希望自己可以将此次课程设计过程中学到的知识和体会到的感悟用在今后的各种大型项目实践当中，在一次次的的项目锻炼中不断成长，做越来越“完善”的自己。在该次实验中优秀组员为：刘泽润、赵懿辉。

XXX 心得体会：

在本次软件工程的课程设计中，我在锻炼了自己的动手能力、实践能力和团队合作能力的同时，对软件工程开发的各个流程有了更具体更深入的理解，初步掌握了软件开发工程的各项步骤，通过这样一个系统的训练，我收益良多，从中深深体会到了实践对于计算机专业的必要性。

本次大型实验我们经过充分的讨论定下了网上书店系统这个题目，开发经过了可行性分析、需求分析、总体设计、详细设计、系统实现、维护等多个步骤。

在整个过程中，我负责一部分文档的撰写，深刻感受到了开发的反复性，就拿各个模块的流程图来说，我前前后后就修改了多次。因此尽管很多东西看似简单，但是想要一次就能完善也很难。所幸我们团队能够在开始前仔细讨论，开发过程中不懂得和有分歧的地方能及时解决，避免发生更大的问题。

软件开发的确是一个需要不断进行迭代的过程，这样才能得到一个功能完善，运行稳定的系统。我们做的系统由于规模比较小，在可行性分析和需求分析部分基本已经把整个程序的各种数据流图、数据字典清楚地表达了出来，但我们多次在进行到下一个步骤时仍然会发现前面几个步骤没有做好，如在设计 IPO 图时，突然想起来在数据流图里面有某个功能尚未实现，这时就要返回去进行修改。在本次实验中，我体会到了前人总结的一步步开发步骤的确是非常有意义的，按照这样的步骤做下来以后，我发现从刚开始对本系统只有一个模糊的概念到后来竟然能够完善地将这个系统实现，软件工程的思想给了我巨大的帮助。

软件工程文档的编写是为了更好的规划整个项目，更方便的进行交流和编

程。在写实验报告的时候我体会到每一个模块的设计都会影响到整个项目的质量，当我们在设计时一定要积极地交流沟通，注意细节，问题发现的越早，修改起来越方便。

通过软件工程这门课，这项实验我真的学到了很多，在日后的学习过程中我在提高自己专业技能的同时会更加注重自己文档撰写能力和团队写作能力的培养。

XXX 心得体会：

通过这 8 周进行的本次软件工程的课程设计，我对软件工程这课有了更加深刻的认识，对系统开发的各个基本流程也有了初步的了解，从课堂上学到的知识在本次课程设计中也得到了充分的实践，我看到并感受到了一个系统的具体分析，具体设计步骤，思路方法都有了进一步提高，感触良深。

本次课程设计我们小组经过讨论定下了大学校园网上书店，开发经过了可行性分析、需求分析、总体设计、详细设计、系统实现，维护等多个步骤，在整个开发过程中，我全程参与了每个阶段的设计与完善，感受到了系统开发的不易与反复性，的确需要一个不断进行迭代的过程，只有不断进行返工才能得到功能较为完善的系统。虽然我们前期不断的完善返工，但队员都没有抱怨，默默做好本职工作，让我感受到了团结合作、相互交流对于系统开发的作用。一个好的团队往往可以有出色的团队合作让事情做起来事半功倍，大家积极参与，促进了思想的碰撞，增加了创新思维，避免了很多不必要的错误，少走了很多弯路。

在完成了本次系统的开发后，我对文档的编写有了更强的重视，对 Visio，Word 等办公软件有了更多的了解，在以后做项目时我一定会注意每一个模块设计都会影响到整个项目的质量，所以要在设计完成前积极讨论，争取及时发现问题，越早发现改动的越少，修改起来越方便。在日后的学习中我一定重视自己文稿撰写能力和团队协作能力，为以后做好准备。

XXX 心得体会：

软件工程的课程设计让我获益良多，不仅巩固丰富了我对于软件工程理论知识的掌握，还锻炼了我的动手能力，培养了我的团队合作能力，明确了计算机专业课程设置必要性，让我对于软件工程这一原本遥不可及的名词有了进一步的认识。

本次大型实验的从题目的确定开始就是一波三折，从最开始想要做的“数学建模网站”到中途想做的“网络商城”，再到最后确定的“网上书店系统”，就经过了激烈的讨论。

开发经过“可行性分析”、“需求分析”、“总体设计”、“详细设计”、“系统实现”、“测试”、“维护”等诸多步骤，终成最终的“网上书店系统”。工程不如单个代码，在多人合作、代码量累积的诸多不利条件下，让我深刻体会到了开发的反复性。某一模块的功能与其他同学负责的模块进行磨合，或是根据实用性增删改查，进行了前前后后多次修改，不断迭代。但归功于组长敏君的统筹安排与组员间热情的互相探讨，我们才及时解决了相关问题。

我在本次实验中主要负责了文档的编写，尤其是顾客部分及相关模块。虽然因为时间原因和能力因素，我未能像组长那样直接参与到代码的编写、功能的实现，但是我认为文档的编写也很是重要。文档的规范编写，讲求信达雅原则，能够更好地规划整个项目，便于队员交流与编程，一个个小小的模块，垒成一个大工程的框架。更是让我明白了，没有好的文档编写，是不可能合作写出巨大的工程的。软件工程规范的文档编写是极为重要的，更是必要的。

感谢我的老师李英龙老师，以及我同小组的成员，我才能顺利的完成软件工程课程安排和课程设计的学习与实现，我从中学到了许许多多，向着一个软件工作者又成长了一步，向着软件工程师又迈进了一步。

XXX 心得体会：

通过这一段时间软件工程实验的训练学习，我对软件工程这门课有了很多认识，对软件行业有了新的认识，对程序员这个角色有了一些新的认识。这门课和几周时间的小组配合，让我明白了，程序开发过程中的小组配合十分重要，一个优秀的团队，合理的任务分配可以极大加快工程项目的编写速度，并且在工程策划前期就可以将一些后期的 BUG 避免和根除。通过这次实验让我体会到对于一个软件项目，先写文档也是非常重要的，对程序员来说学会写文档也是一个非常重要的技能。软件系统开发不是一朝一夕就可以完成的。比如需求分析中的数据对象及属性、联系来说，就需要小组进行细致的商讨推敲与后期的反复修改。因此尽管很多东西看似简单，但是想要一次就能完善也很难。虽然我们在设计时进行了充分的讨论研究，但在实现过程中也出现了许多之前没有考虑到的问题。积累了经验，增加了见识，并

且有了团队合作的经验。

经过本次开发，我对 visio，word 等办公软件也有了更进一步的认识，能够更加熟悉的使用并辅佐工程开发。这对以后的学习工作来说都是一笔很大的隐性财富。

8 附录

实体对应数据库表的 Hibernate 配置文件具体内容如下所示：

1. 图书表

```
<?xmlversion="1.0"encoding="UTF-8"?>

<!DOCTYPEhibernate-mappingPUBLIC"-//Hibernate/Hibernate  Map-
ping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>

<classname="com.entity.Books"table="T_Books"catalog="projectdb">

<idname="Bnumber"type="java.lang.String">

<columnname="Bnumber"length="13"/>

<generatorclass="assigned"/>

</id>

<propertyname="Bname"type="java.lang.String">

<columnname="Bname"length="30"not-null="true"/>

</property>

<propertyname="Bwriter"type="java.lang.String">

<columnname="Bwriter"length="20"/>
```

```

</property>

<propertyname="Bpublish" type="java.lang.String">

<columnname="Bpublish" length="30"/>

</property>

<propertyname="Bprice" type="java.lang.Double">

<columnname="Bprice" not-null="true"/>

</property>

</class>

</hibernate-mapping>

```

2. 顾客表

```

<?xmlversion="1.0" encoding="UTF-8"?>

<!DOCTYPEhibernate-mappingPUBLIC"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>

<classname="com.entity.Customers" table="T_Customers" catalog="projectdb">

<idname="Cusername" type="java.lang.String">

<columnname="Cusername" length="20"/>

<generatorclass="assigned"/>

</id>

<propertyname="Cname" type="java.lang.String">

<columnname="Cname" length="20" not-null="true"/>

</property>

```

```

<propertyname="Cpassword"type="java.lang.String">
<columnname="Cpassword"length="20"default="123456"/>
</property>
<propertyname="Csex"type="java.lang.String">
<columnname="Csex"length="2"/>
</property>
<propertyname="Cphone"type="java.lang.String">
<columnname="Cphone"length="11"not-null="true"/>
</property>
<propertyname="Clocal"type="java.lang.String">
<columnname="Clocal"length="50"not-null="true"/>
</property>
<propertyname="Cvalue"type="java.lang.Integer">
<columnname="Cvalue"default="0"/>
</property>
</class>
</hibernate-mapping>

```

3. 导入表

```

<?xmlversion="1.0"encoding="UTF-8"?>
<!DOCTYPEhibernate-mappingPUBLIC"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>

```

```

<classname="com.entity.Imports"table="T_Imports"catalog="projectdb">
<composite-idname="key">
<key-propertyname="Iusername"length="20"type="java.lang.String"/>
<key-propertyname="Inumber"length="13"type="java.lang.String"/>
<generatorclass="assigned"/>
</composite-id>
<many-to-onename="shops"class="com.entity.Shops">
<columnname="Susername"not-null="true"/>
</many-to-one>
<many-to-onename="books"class="com.entity.Books">
<columnname="Bnumber"not-null="true"/>
</many-to-one>
<propertyname="Idiscount"type="java.lang.Double">
<columnname="Idiscount"not-null="true"/>
</property>
<propertyname="Inumgoods"type="java.lang.Integer">
<columnname="Inumgoods"not-null="true"/>
</property>
</class>
</hibernate-mapping>

```

4. 订单表

```

<?xmlversion="1.0"encoding="UTF-8"?>

```

```
<!DOCTYPEhibernate-mappingPUBLIC"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>

<classname="com.entity.Orders"table="T_Orders"catalog="projectdb">

<idname="Onumber"type="java.lang.Long">

<columnname="Onumber"/>

<generatorclass="native"/>

</id>

<many-to-onename="customers"class="com.entity.Customers">

<columnname="Cusername"not-null="true"/>

</many-to-one>

<many-to-onename="shops"class="com.entity.Shops">

<columnname="Susername"not-null="true"/>

</many-to-one>

<many-to-onename="books"class="com.entity.Books">

<columnname="Bnumber"not-null="true"/>

</many-to-one>

<propertyname="Olocal"type="java.lang.String">

<columnname="Sname"length="50"not-null="true"/>

</property>

<propertyname="Otime"type="java.sql.Timestamp">

<columnname="Otime"not-null="true"/>
```

```
</property>

<propertyname="Oquantity" type="java.lang.Integer">
  <columnname="Oquantity" not-null="true"/>
</property>

<propertyname="Osumprice" type="java.lang.Double">
  <columnname="Osumprice"/>
</property>

<propertyname="Oremark" type="java.lang.String">
  <columnname="Oremark" length="30"/>
</property>

<propertyname="Opay" type="java.lang.Boolean">
  <columnname="Opay" sql-type="Boolean"/>
</property>

<propertyname="Oaccept" type="java.lang.Boolean">
  <columnname="Oaccept" sql-type="Boolean"/>
</property>

<propertyname="Oreturn" type="java.lang.Boolean">
  <columnname="Oreturn" sql-type="Boolean"/>
</property>

<propertyname="Oreason" type="java.lang.String">
  <columnname="Oreason" length="30"/>
</property>
```

```

<propertyname="Ocancel" type="java.lang.Boolean">
<columnname="Ocancel" sql-type="Boolean"/>
</property>
<propertyname="Odiscount" type="java.lang.Boolean">
<columnname="Odiscount" sql-type="Boolean"/>
</property>
<propertyname="Ogetvalue" type="java.lang.Integer">
<columnname="Ogetvalue"/>
</property>
</class>
</hibernate-mapping>

```

5. 店员表

```

<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPEhibernate-mappingPUBLIC"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
<classname="com.entity.Shops" table="T_Shops" catalog="projectdb">
<idname="Susername" type="java.lang.String">
<columnname="Susername" length="20"/>
<generatorclass="assigned"/>
</id>
<propertyname="Sname" type="java.lang.String">

```

```
<columnname="Sname"length="20"not-null="true"/>
</property>
<propertyname="Spassword"type="java.lang.String">
<columnname="Spassword"length="20"default="123456"/>
</property>
<propertyname="Ssex"type="java.lang.String">
<columnname="Ssex"length="2"not-null="true"/>
</property>
<propertyname="Sphone"type="java.lang.String">
<columnname="Sphone"length="11"not-null="true"/>
</property>
<propertyname="Sshopname"type="java.lang.String">
<columnname="Sshopname"length="50"not-null="true"/>
</property>
<propertyname="Slocal"type="java.lang.String">
<columnname="Slocal"length="50"not-null="true"/>
</property>
</class>
</hibernate-mapping>
```