客户在线订书系统

软件设计说明书

团队编号：第10组

团队成员：王怡宁 周子渔 向鑫睿 陈骁霄 丛乐驰

完成日期：2023.6.24

目录

[1．引言 3](#_Toc43487745)

[1.1编写目的 3](#_Toc43487746)

[1.2项目背景 3](#_Toc43487747)

[1.3定义 3](#_Toc43487748)

[1.4参考资料 3](#_Toc43487749)

[2．概要设计 3](#_Toc43487750)

[2.1系统模块划分 3](#_Toc43487751)

[2.2系统结构设计 3](#_Toc43487752)

[2.3处理流程设计 3](#_Toc43487753)

[3．数据库设计 4](#_Toc43487754)

[4．接口设计 4](#_Toc43487755)

[4.1外部接口 4](#_Toc43487756)

[4.2内部接口 4](#_Toc43487757)

[5．模块详细设计 4](#_Toc43487758)

[6. 界面设计 4](#_Toc43487759)

[6.1界面样式设计 4](#_Toc43487760)

[6.2界面交互设计 4](#_Toc43487761)

[7．出错处理设计 5](#_Toc43487762)

[7.1错误类型及出错处理对策 5](#_Toc43487763)

# 1．引言

## 1.1编写目的

该文档在概要设计的基础上，进一步的细化系统结构，展示了软件结构的图标，物理设计、数据结构设计、及算法设计、详细的介绍了系统各个模块是如何实现的，包括涉及到的算法，逻辑流程等。

预期的读者：程序员

## 1.2项目背景

1. 待开发软件系统的名称：客户在线订书系统
2. 项目的任务提出者：北京工业大学计算机软件类综合性课程设计教学组
3. 项目的开发者：信息学部计算机学院计算机科学与技术专业软工小组第10组
4. 主管部门：信息学部计算机学院计算机科学与技术专业软工小组第10组
5. 项目的用户：其他小组的全体用户及老师
6. 运行该软件的设备：信息学部计算机学院计算机科学与技术专业软工小组第10组的全体硬件设备

## 1.3定义

系统结构：对系统整体布局的宏观的描述

算法：对于程序内部流程计算的逻辑表达方式。

SQL SERVER: 系统服务器所使用的数据库管理系统（DBMS）。

SQL: 一种用于访问查询数据库的语言

事务流：数据进入模块后可能有多种路径进行处理。

主键：数据库表中的关键域。值互不相同。

外部主键：数据库表中与其他表主键关联的域。

ROLLBACK: 数据库的错误恢复机制。

黑盒测试：通过测试来检验每个功能都是否能够正常使用。在测试时，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，测试人员进行直接测试，检查系统功能是否按照需求规格说明书的规定正常使用、是否能适当的接收输入数据而输出正确的结果等，检查相应的文档是否采用了正确的模板、是否满足规范需求。

## 1.4参考资料

列出有关的参考资料，如：

1. 郑人杰、《软件工程概论》、ISBN 978-7-111-64257-2、2021年6月、机械工业出版社
2. 《软件工程概述》 李存珠编著 南京大学计算机系出版 2001年8月
3. 《网上书店需求调查报告》

# 2．概要设计

## 2.1系统模块划分

用户管理模块：包括用户注册、登录、个人信息管理等功能。

商品管理模块：包括商品分类、商品信息管理、商品上下架等功能。

订单管理模块：包括订单生成、订单查询、订单支付等功能。

购物车管理模块：包括购物车添加、购物车删除、购物车结算等功能。

支付管理模块：包括支付方式选择、支付金额计算等功能。

以上模块与需求的对应关系如下：

| **模块** | **需求** |
| --- | --- |
| 1.用户管理模块 | 管理用户信息（1.1注册，1.2登录，1.3修改密码，1.4查询个人信息） |
| 2.商品管理模块 | 管理商品信息（2.1浏览书籍，2.2查询书籍，2.3在线购书） |
| 3.订单管理模块 | 管理订单信息（3.1查看订单信息） |
| 4.购物车管理模块 | 管理购物车信息（4.1加入购物车） |
| 5.支付管理模块 | 管理支付信息（5.1结算） |

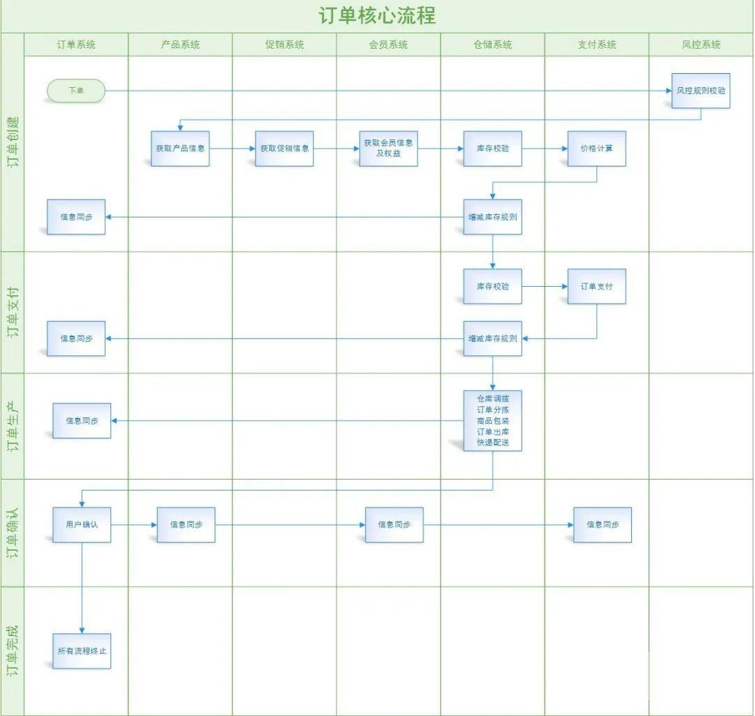
## 2.2系统结构设计

系统结构设计图如下：



## 2.3处理流程设计

描述各模块的主要操作流程（流程图）如下：



# 3．数据库设计

详细请见5.2数据库设计

# 4．接口设计

## 4.1外部接口

网站部分：本图书销售系统不包含与网络的交互联通功能。

客户端部分：简洁大方的图形用户界面，包含登陆窗口，以及登陆成功后显示的用户主界面窗口。主界面由各个系统功能子模块构成，要求清晰、美观。登陆界面包含不同角色的切换登陆功能，适用于管理员以及用户等不同身份权限的角色进行登陆，要求简洁、有序。

数据存储部分：本系统设计数据的存储和处理，通过ADO.NET与MySQL交互。

## 4.2内部接口

登录界面：用户名输入栏、密码输入栏、商家/管理员身份选择按钮、登陆按钮、注册按钮。

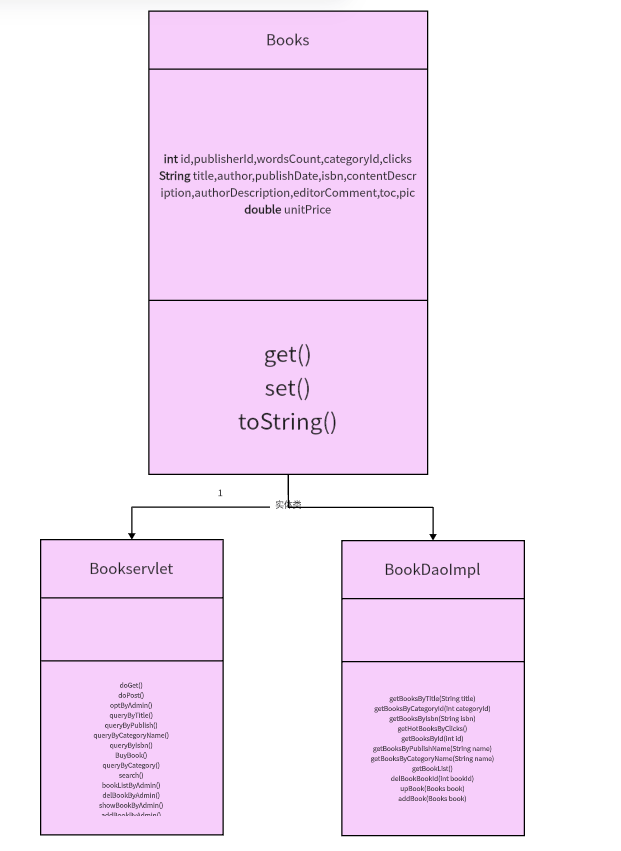
用户注册界面：用户名输入栏、密码输入栏、确认密码输入栏、确定注册按钮、返回按钮。

主界面：搜索按钮，分类按钮，图书按钮，购买按钮

# 5．模块详细设计

## 5.1 后端开发

**5.1.1 Book相关类：**



**Books：**

本类主要是图书的实体化类.其中定义了书籍的一切所需要的属性：名称，作者，价格等，并且有各个属性的get和set方法，最后设置toString将所有相关信息打印出来。

**Bookservlet：**

本层功能主要为请求和响应控制，从网页中接受相应的数据完成对books类的不同的操作，如查询书籍，购买书籍，搜索书籍等。具体的各个功能如类图所示

doGet()：重写doGet方法，将请求转发给doPost方法。

doPost():它的作用是处理客户端的POST请求，根据请求参数opt和role的不同值，执行不同的操作，如查询书籍，购买书籍，搜索书籍等。

optByAdmin():处理管理员的操作请求，根据请求参数opt的不同值，执行不同的操作，如显示书籍列表，删除书籍，编辑书籍，添加书籍等。

queryByTitle/queryByPublish/queryByCategoryName/queryByIsbn:

根据请求参数value的值，查询书籍的标题，然后调用BookDaoImpl类的getBooksByTitle/getBooksByPublishName/getBooksByCategoryName/getBooksByIsbn

方法，返回一个包含匹配书籍的列表。然后将这个列表封装成一个分页对象，存储在请求会话中，最后重定向到search.jsp页面。

BuyBook():根据图书Id获取图书,并且跳转到购买页面

queryByCategory():用于根据图书分类查询图书

search():用于根据分页信息显示查询结果

bookListByAdmin():用于显示所有图书的列表

delBookByAdmin():删除图书

showBookByAdmin():进入编辑图书页面

addBookByAdmin():编辑图书

**BookDaoImpl:**

这个BookDaoImpl类的功能是连接数据库，对数据库的数据进行增删改查的操作，将所得到的数据返回到一个list表中。具体实现手法为，首先创造一个list对象，用于储存查询结果，调用getCon方法连接到数据库，之后定义一个数据库数据，根据不同的函数进行不同的操作，创建一个数据库对象，将刚定义好的数据库数据当参数传入，之后执行数据库，返回结果集合。具体各个函数功能如下

getBooksByTitle(String title):根据图书标题模糊查询图书。

getBooksByCategoryId(int categoryId)：根据分类Id模糊查询图书

getBooksByIsbn(String isbn):根据Isbn获取图书

getHotBooksByClicks():按照clicks降序排列获取所有图书

getBooksById(int id):根据图书Id获取图书

getBooksByPublishName(String name):根据出版社获取图书

getBooksByCategoryName(String name):根据图书分类获取图书

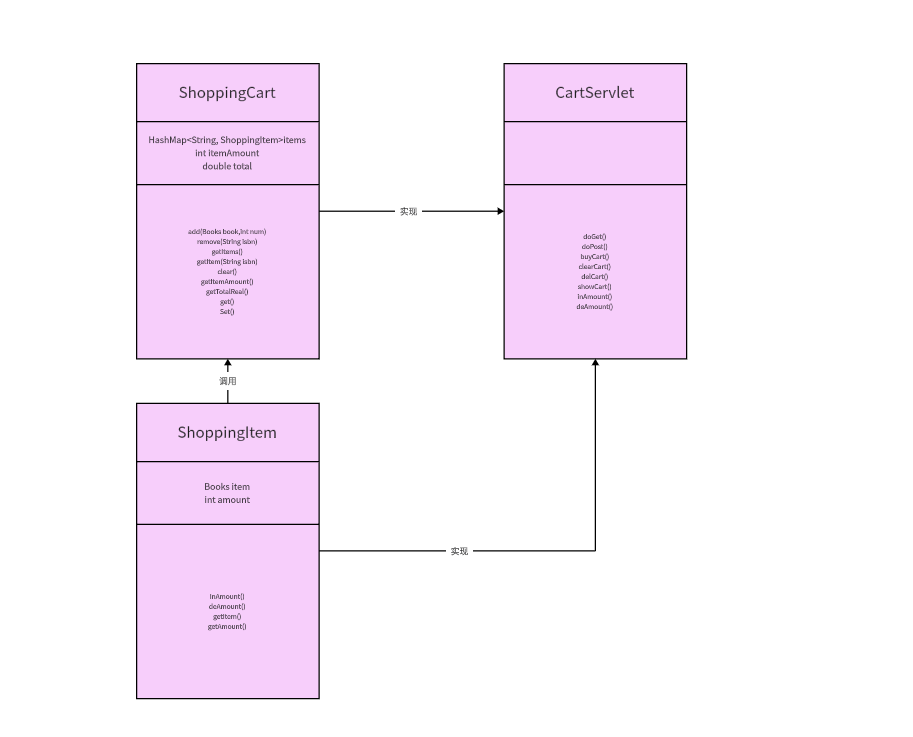
getBookList():根据图书分类获取图书

delBookBookId(int bookId):删除图书

upBook(Books book):修改图书信息

addBook(Books book)添加图书

**5.1.2 Cart相关类：**



**CartServlet：**

本层功能主要为请求和响应控制，从网页中接受相应的数据完成对购物车类的不同的操作，如添加购物车，清空购物车，删除购物车中的商品，显示购物车等。具体的各个功能如类图所示。

doGet():重写doGet方法，将请求转发给doPost方法。

doPost():重写doPost方法，根据请求参数opt的值执行不同的操作。

buyCart():将商品添加到购物车的方法。

clearCart():清空购物车。

delCart():删除购物车中的商品。

showCart():显示购物车中的商品。

inAmount():增加购物车中商品数量。

deAmount():减少购物车中商品数量。

**ShoppingCart：**

本类定义了购物车的一些重要的属性比如物品映射，价格和数量等，并且配备了相关的实现手法，比如增加书籍，删除书籍和清空购物车等，并且对于每个属性有着相关的get和set函数

add(Books book,int num):这是一个用于将商品添加到购物车。

remove(String isbn):删除购物车中的商品。

getItems():获取购物车中的商品列表。

getItem(String isbn):根据isbn获取购物车中对应的ShoppingItem。

clear():清空购物车。

getItemAmount():获取购物车中商品数量。

getTotalReal():计算商品总价。

get()

Set()

**ShoppingItem:**

本模块是购物车商品模块，定义了book类型的书籍商品，和书籍的数量。并且实现了初始化，数量的增减函数

Books item

int amount

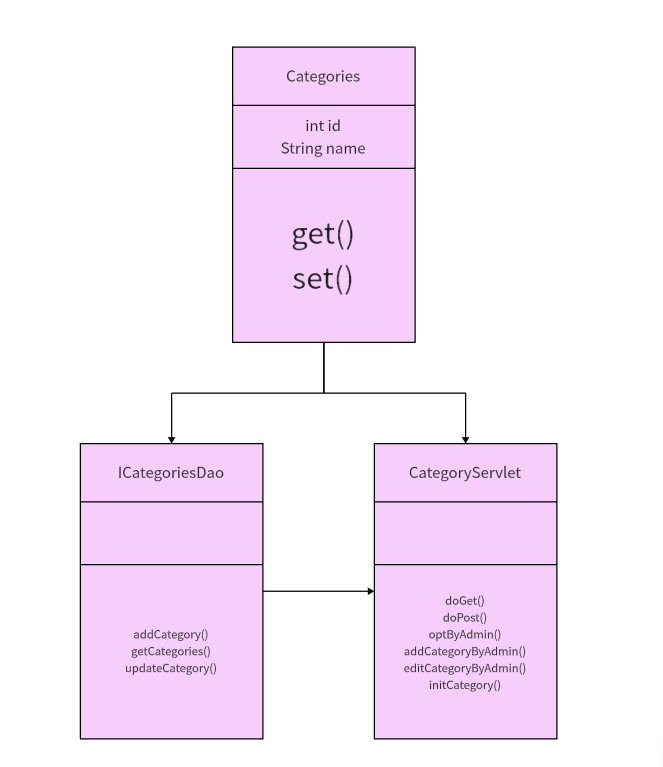
inAmount():数量增加

deAmount():数量减少

getItem():返回数量

getAmount():返回书籍

**5.1.3 Categories相关类：**



**Categories：**

本类主要是图书的实体化类.其中定义了书籍的一切所需要的属性：名称，作者，价格等，并且有各个属性的get和set方法，最后设置toString将所有相关信息打印出来。

**CategoryServlet：**

本层功能主要为请求和响应控制，从网页中接受相应的数据完成对书籍分类类的不同的操作，如增加分类，编辑分类和初始化等各个操作。具体的各个功能如类图所示。

doGet():重写doGet方法，将请求转发给doPost方法。

doPost():重写doPost方法，根据请求参数opt的值执行optByAdmin操作。

optByAdmin():根据请求参数opt的值来执行不同的操作，如添加分类或编辑分类。

addCategoryByAdmin():添加分类。

editCategoryByAdmin():编辑分类。

initCategory():初始化分类集合。

**ICategoriesDao:**

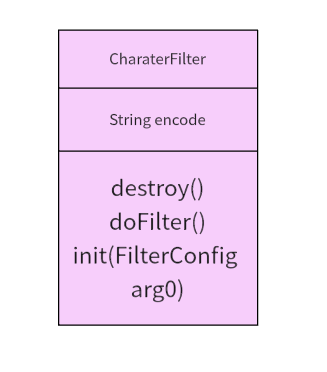
本类主要和数据库进行交互，通过对数据库的查找等各种操作实现，获取，添加和更新的一系列操作。

getCategories():获取分类,从数据库中查询所有的类别，并按照Id降序排列，然后返回一个包含Categories对象的列表。

addCategory():添加分类。向数据库中插入一个新的类别，并返回一个布尔值表示操作是否成功。

updateCategory():更新分类。类别的Id修改数据库中的类别名称，并返回一个布尔值表示操作是否成功。

**5.1.4 CharaterFilter过滤器类：**



过滤器，帮助我们过滤请求。过滤出需要要的请求，不需要的请求忽略掉。本函数设置的是匹配所有请求。其主要函数包括：destroy()、doFilter()、init(FilterConfig arg0)

**5.1.5 StringUti类：**

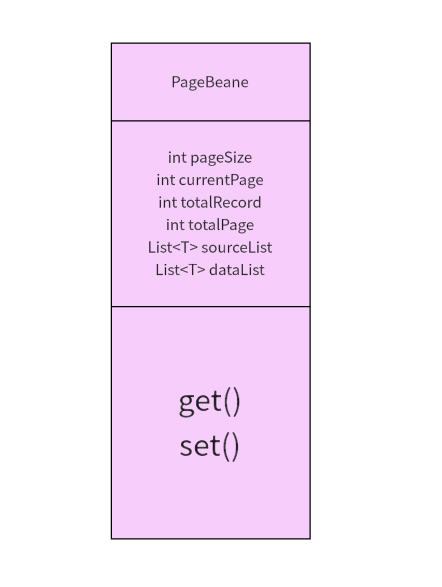
本函数是字符串处理工具类，主要定义了函数isNumber(String str)用来判断所给字符是否为正整数。

**5.1.6 InitServlet：**

这是一个初始化方法，用于在启动时加在一些数据主要通过调用init函数来完成

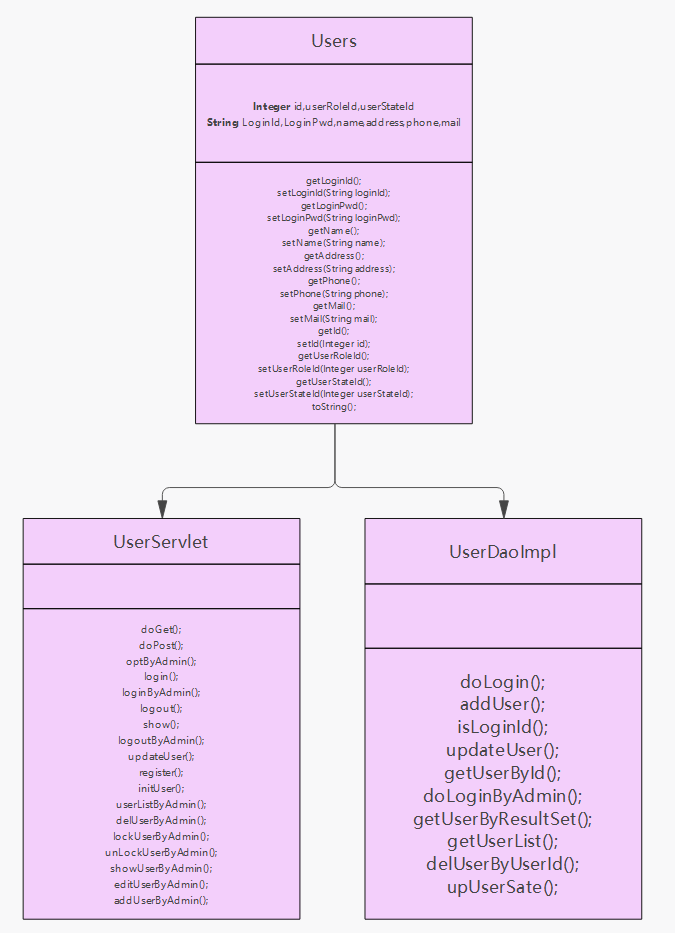
init(ServletConfig config)：调用此方法首先获取图书分类，然后获取行相应的获取出版社和热门图书信息，之后完成初始化。

**5.1.7 PageBeane实体类：**



封装每一页数据的实体类。其内部定义了int型的数据：每页多少的记录，当前第几页的数据，页数，记录数和list型的数据：要显示的数据和源数据。并且配置有相应的构造函数和每个变量的set和get函数

**5.1.8 User相关类：**

****

**Users：**

本类主要是用户的实体化对象类.其中定义了用户的一切所需要的属性：用户id，用户名，密码，姓名，地址，电话，邮箱，角色编号，用户状态，并且有各个属性的get和set方法，最后设置toString将所有相关信息打印出来。

**UserServlet：**

本层功能主要为请求和响应控制，从网页中接受相应的数据完成对Users类的不同的操作，如登录，注册，注销，修改信息等。具体的各个功能如类图所示

doGet()：重写doGet方法，将请求转发给doPost方法。

doPost():它的作用是处理客户端的POST请求，根据请求参数opt和role的不同值，执行不同的操作，如登录，注册，注销，修改信息等。

optByAdmin():处理管理员的操作请求，根据请求参数opt的不同值，执行不同的操作，如登录，获取用户列表，退出，删除用户，锁定用户，解锁用户等。

login():用户登录,调用dao层doLogin()方法进行登录验证。

loginByAdmin():管理员登录，调用dao层doLoginByAdmin()进行登录验证。

logout():用于用户注销退出

show():用于重定向到用户页面

logoutByAdmin():用于管理员注销退出

updateUser():用于修改用户信息，调用dao层updateUser()方法根据用户id找到用户信息并更新。

register():注册用户，获取用户输入的信息，调用dao层addUser()方法，将用户信息插入到数据库中。

initUser():初始化用户信息,调用dao层getUserById()方法,根据用户Id获取用户信息。（更新操作时使用）

userListByAdmin():用于管理员获取用户列表。

delUserByAdmin():用于管理员删除用户。

lockUserByAdmin():用于管理员锁定用户。

unLockUserByAdmin():用于管理员解锁用户。

showUserByAdmin():允许管理员进入编辑用户页面。

editUserByAdmin():用于管理员编辑用户。

addUserByAdmin():用于管理员添加用户。

**UserDaoImpl:**

UserDaoImpl类的功能是连接数据库，对数据库中的用户数据进行增删改查的操作。具体实现手法为，声明一个DB对象，用于获取数据库连接，调用getCon方法连接到数据库，创建预编译语句对象，用于执行对应的sql语句。之后定义一个数据库数据，根据不同的函数进行不同的操作，创建一个数据库对象，将刚定义好的数据库数据当参数传入，之后执行数据库，返回结果集合。具体各个函数功能如下

doLogin(String name, String pwd):用于登录验证，根据用户输入的用户名和密码在数据库中进行匹配。

addUser(Users u)：注册用户，在数据库中插入一条新的User数据。

isLoginId(String loginId):根据用户id判断用户是否存在。

updateUser(Users u):修改用户信息根据用户id找到用户信息并更新。

getUserById(int id):根据用户Id获取用户信息。

doLoginByAdmin(String name, String pwd):管理员后台登录。

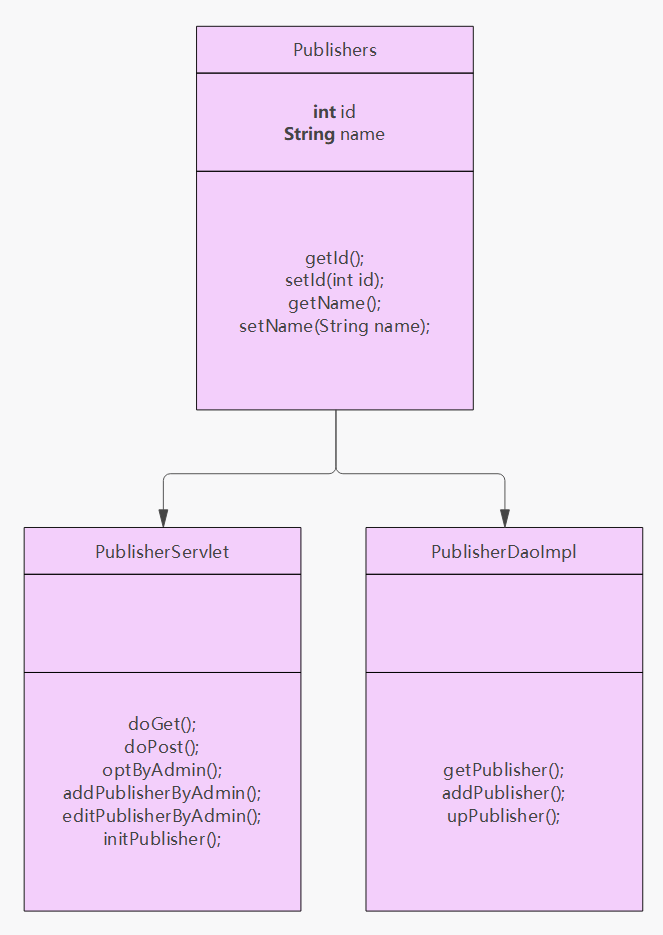
getUserByResultSet(ResultSet rs):封装User结果集。

getUserList():获取用户列表（不包括管理员）。

delUserByUserId(int userId):根据用户Id删除用户。

upUserSate(int userId,int state):更新用户状态，根据用户id查找用户并更改其状态。（锁定or解锁）

**5.1.9 Publisher相关类：**

****

**Publishers：**

本类主要是出版社的实体化对象类.其中定义了出版社的一切所需要的属性：出版社id，出版社名称，并且有各个属性的get和set方法。

**PublisherServlet：**

本层功能主要为请求和响应控制，从网页中接受相应的数据完成对Publishers类的不同的操作，如添加出版社，编辑出版社等。具体的各个功能如类图所示

doGet()：重写doGet方法，将请求转发给doPost方法。

doPost():它的作用是处理客户端的POST请求，根据请求参数opt和role的不同值，执行不同的操作，如登录，注册，注销，修改信息等。

optByAdmin():管理员操作，只允许管理员进行添加出版社，编辑出版社的操作。

addPublisherByAdmin ():添加出版社,调用dao层addPublisher ()方法在数据库中添加一条新的出版社信息。

editPublisherByAdmin ():编辑出版社，调用dao层upPublisher ()根据出版社id找到出版社信息并更新。

initPublisher ():初始化出版社集合。（用于添加出版社后的初始化）

**PublisherDaoImpl:**

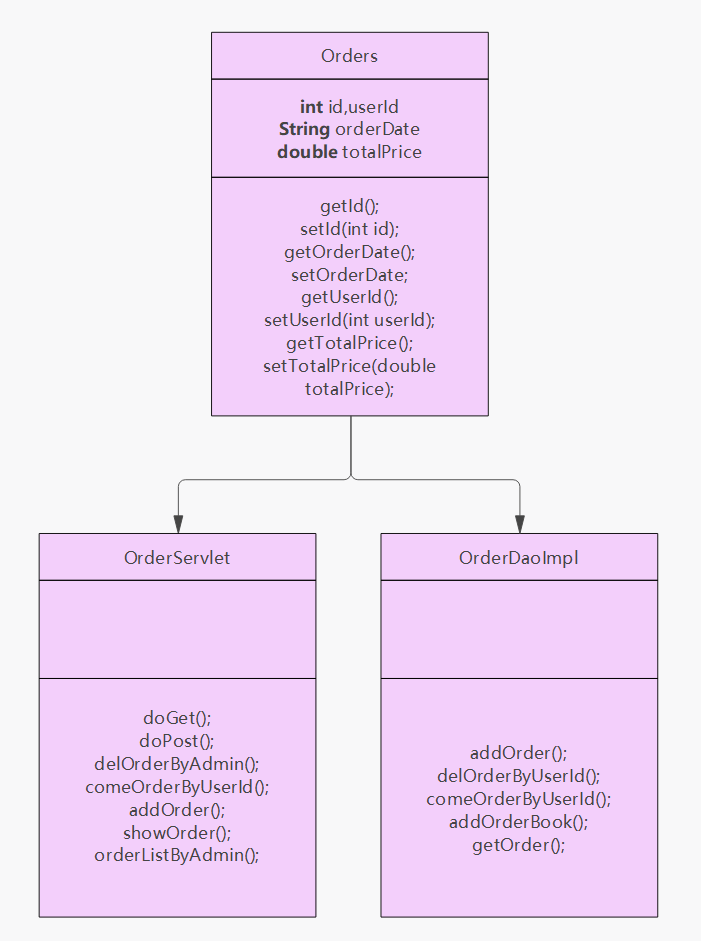
PublisherDaoImpl类的功能是连接数据库，对数据库中的用户数据进行增删改查的操作。具体实现手法为，声明一个DB对象，用于获取数据库连接，创建出版社list集合，调用getCon方法连接到数据库，创建预编译语句对象，用于执行对应的sql语句。之后定义一个数据库数据，根据不同的函数进行不同的操作，创建一个数据库对象，将刚定义好的数据库数据当参数传入，之后执行数据库，返回结果集合。具体各个函数功能如下：

getPublisher():用于获取所有出版社。

addPublisher(Publishers p)：添加出版社，在数据库中添加一条新的出版社信息。

upPublisher(Publishers p): 更新出版社，根据出版社id找到出版社信息并更新。

**5.1.10 Order相关类：**

****

**Orders：**

本类主要是订单的实体化对象类.其中定义了订单的一切所需要的属性：订单id，订单时间，用户id，订单总价，并且有各个属性的get和set方法。

**OrderServlet：**

本层功能主要为请求和响应控制，从网页中接受相应的数据完成对Orders类的不同的操作，如添加订单，删除订单，发货，查看订单，获取订单列表等。具体的各个功能如类图所示

doGet()：重写doGet方法，将请求转发给doPost方法。

doPost():它的作用是处理客户端的POST请求，根据请求参数opt和role的不同值，执行不同的操作，如添加订单，删除订单，查看订单，发货等。

delOrderByAdmin ():处理管理员删除订单的操作请求，调用dao层delOrderByUserId ()根据用户Id删除订单。

comeOrderByUserId ():发货，调用dao层comeOrderByUserId ()修改数据库中对应id下order的状态为1，表示发货。

addOrder ():添加订单，调用dao层addOrder ()，在数据库中插入一条新的Order数据。之后获取购物车中的所有图书种类，遍历每一项图书，将其封装成OrderBook对象并调用dao层addOrderBook()方法添加到明细表。随后清空购物车。

showOrder ():查看订单，调用dao层getOrder(int userId)方法，根据Order的id查看订单。

orderListByAdmin(): 获取订单列表，调用dao层getOrder ()方法获取订单列表。

**OrderDaoImpl:**

OrderDaoImpl类的功能是连接数据库，对数据库中的用户数据进行增删改查的操作。具体实现手法为，声明一个DB对象，用于获取数据库连接，调用getCon方法连接到数据库，创建预编译语句对象，用于执行对应的sql语句。之后定义一个数据库数据，根据不同的函数进行不同的操作，创建一个数据库对象，将刚定义好的数据库数据当参数传入，之后执行数据库，返回结果集合。具体各个函数功能如下

addOrder(Orders orders):用于添加订单，在数据库中插入一条新的Order数据。（用户编号、订单总价）

addOrderBook(OrderBook orderBook): 添加详细订单。（订单编号、图书编号、图书数量、图书价格）

comeOrderByUserId(int userId):发货，修改数据库中对应id下order的状态为1，表示发货。

delOrderByUserId(int userId): 根据用户Id删除订单。

getOrder(int userId): 根据当前登录用户Id获取对应的订单。

getOrder():获取订单列表。

## 5.2 数据库设计

**5.2.1 数据库建立**

创建和初始化一个MySQL数据库的脚本，用于创建一系列的表并插入一些初步的数据。

SET FOREIGN\_KEY\_CHECKS=0; 该命令是为了禁用外键检查，这样在创建或者删除具有外键关系的表时，不会因为外键约束而导致操作失败。

DROP TABLE IF EXISTS '<table>'; 该命令是为了删除（如果存在）名称为'<table>'的表。

CREATE TABLE '<table>' (...); 该命令用于创建一个名为'<table>'的新表，并指定它的列和数据类型。例如，表address有如下列：Id, UserId, Street, City, Province, PostalCode, Phone, Name。其中Id是主键，其数据类型是自增的整型。

INSERT INTO '<table>' VALUES (...); 该命令是用于向指定的表插入一行新的数据。例如，插入到表address的数据是：'1', '3', '温江', '成都市', '四川省', null, '13982111111', '测试'。

该数据库创建了以下表：

address：代表用户的地址，包括街道、城市、省份、邮编等信息。

books：代表书籍，包括书名、作者、出版商ID、出版日期、ISBN号、字数、单价等信息。

categories：代表书籍的分类，每个类别有一个名称。

orderbook：代表订单中的书籍，包括订单ID、书籍ID、数量和单价等信息。

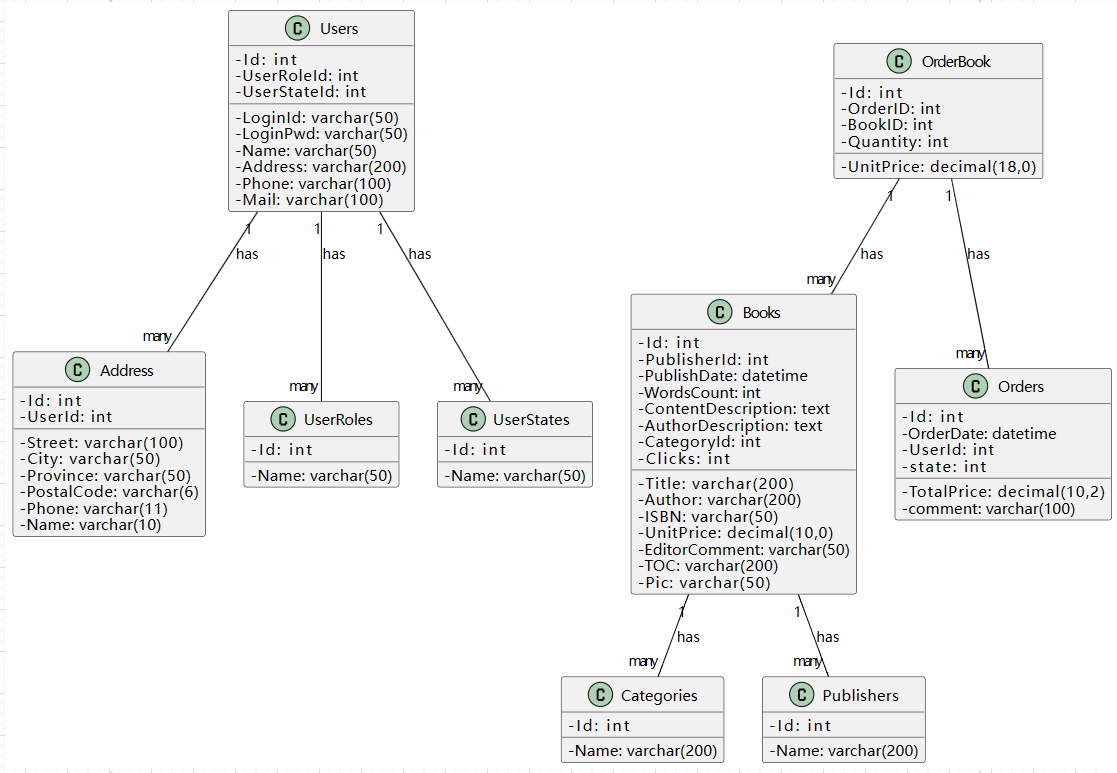
orders：代表订单，包括订单日期、用户ID、总价等信息。

publishers：代表出版商，每个出版商有一个名称。

userroles：代表用户角色，每个角色有一个名称。

users：代表用户，包括登录ID、密码、姓名、地址、电话、邮件、用户角色ID和用户状态ID等信息。

userstates：代表用户状态，每个状态有一个名称。



**5.2.2 数据库类定义**

className、url、user、password：这些变量保存了连接数据库所需的信息。这包括驱动类名(className)，数据库URL(url)，以及登录数据库的用户名(user)和密码(password)。

getCon()：这个方法用于获取数据库连接。它试图使用前面定义的信息来连接到数据库，并把得到的Connection对象保存在con变量中。如果连接失败，它会把con设为null。

doPstm(String sql, Object[] params)：这个方法执行SQL查询。它接收一个SQL语句字符串和一个对象数组作为参数，这个对象数组包含了SQL语句中需要的参数。如果params为null，那么将创建一个空数组。

方法首先尝试获取数据库连接。如果连接有效，它就创建一个PreparedStatement对象并设置所有参数，然后执行SQL语句。

getRs()：这个方法返回执行SQL查询后得到的ResultSet对象。

getEditRs()：这个方法返回由执行SQL更新语句（如INSERT、UPDATE或DELETE语句）生成的自动生成的键。

getCount()：这个方法返回由最后一次执行的SQL更新语句影响的行数。

closed()：这个方法关闭PreparedStatement和Connection对象，释放数据库资源。

## 5.3 前端开发

**5.3.1 登录界面**

登陆界面是一个用于用户登录的JSP（Java Server Pages）网页。具体内容如下：

该页面的**头部(<head>)**主要定义了页面的字符编码、标题、并引入了CSS样式表和JavaScript文件。CSS文件用于页面的样式布局，JavaScript文件用于页面的交互。

页面的**主体部分(<body>)**包括：

jsp:include page="header.jsp"：这行代码引入了一个名为"header.jsp"的文件，它可能包含导航栏或其他你希望在多个页面中重复使用的内容。页面的主体是一个包含登录表单的div。这个表单有一个用户名输入框和一个密码输入框，都附带了在用户输入失焦时调用的JavaScript函数以验证用户的输入。这个表单的action属性设置为"UserServlet?opt=login"，表示当用户点击登录按钮后，表单的数据将被发送到服务器上的UserServlet进行处理，opt参数值为"login"，标明是登录操作。提交按钮是一个input标签，类型为"submit"，这意味着当用户点击这个按钮时，表单的数据会被发送到在表单的action属性中指定的URL。提交表单时，会先执行onsubmit属性中的JavaScript函数check\_Login\_Submit()，只有该函数返回true，表单才会被提交。提供了一个复选框用于用户选择是否记住密码，以及一个链接用于找回密码。

最后，又引入了一个名为"footer.jsp"的文件，它可能包含页面底部的版权信息或其他内容。

在页面底部，引入了一个JavaScript文件"usercheck.js"，它可能包含用于处理页面交互的JavaScript代码，例如前面提到的用户输入验证函数。

**5.3.2 注册界面**

页面开头部分定义了页面的编程语言为Java、内容类型为text/html，字符编码为UTF-8。同时，引入了JSTL（JavaServer Pages Standard Tag Library）核心标签库，它提供了一套用于JSP页面的自定义标签，从而简化JSP页面的开发。

**<head>部分**设置了页面的字符编码为UTF-8，并定义了网页标题。

**<body>部分**首先引入了一个名为"header.jsp"的文件。然后是一个注册表单，用户需要填写各种信息（这部分内容在给出的代码中被省略了）。表单的action属性设置为"${ctxPath }/UserServlet?opt=register"，这表示当用户提交表单后，表单的数据将被发送到服务器上的UserServlet进行处理，"opt"参数值为"register"，表明这是一个注册请求。在表单提交前，onsubmit事件会调用JavaScript函数check\_Rg\_Submit()来进行表单验证。注册表单下方是一些注释，提供了用户名和密码的规则以及已有账号的用户可以直接登录的链接。页面底部引入了一个名为"footer.jsp"的文件，这可能是页面的底部信息，例如版权声明等。

最后，页面引入了一个JavaScript文件"usercheck.js"，它包含了一些处理页面交互的代码，例如前面提到的表单验证函数。另外，还有一个JavaScript代码片段用于处理注册结果的提示消息。如果注册结果消息为空，那么会隐藏提示消息并从session中移除注册结果消息。如果不为空，则会显示提示消息。

**5.3.3 登录、注册信息验证**

该部分主要进行了用户登录和注册信息的验证。以下是各部分的详细解释：

**首先定义了一些正则表达式变量来验证输入信息的格式：**

re\_LoginId：验证用户名的规则，应该以字母开始，然后是字母、数字或下划线，长度在6到20个字符之间。

re\_LoginPwd：验证密码的规则，应该是6到20个任何单词字符（字母、数字或下划线）。

re\_Email：验证电子邮件的规则，需要符合电子邮件的一般格式。

re\_Phone：验证手机号码的规则，需要符合中国手机号码的一般格式。

re\_Name：验证用户真实姓名的规则，应该是2到4个中文字符。

**然后为登录过程定义了一些验证函数：**

check\_Login\_Id(dom)：验证用户名。如果输入为空，将提示用户输入账号，并将边框变为红色。如果输入非空，将边框变为灰色。

check\_Login\_Pwd(dom)：验证密码。如果输入为空，将提示用户输入密码，并将边框变为红色。如果输入非空，将边框变为灰色。

check\_Login\_Submit()：提交前的验证，如果用户名或密码的验证状态为false，将在页面中的某处显示错误信息。

**接着为注册过程定义了一些验证函数**，它们的工作方式大致与登录验证类似，只是对更多的输入项进行了验证（例如确认密码、姓名、邮箱、手机号码和地址）。这些函数将在用户输入信息后进行验证，并根据验证结果更新相关的提示信息和状态标记。

**最后，注册过程还添加了对用户是否同意服务协议的验证**，只有在用户同意协议的情况下，才允许用户提交注册信息。

**5.3.4页眉界面**

这个页面是一个JSP页面，它的作用是实现一个图书搜索功能。这个页面包含了一些HTML代码和一些Java代码。HTML代码用于实现网页中静态内容的显示，Java代码用于实现网页中动态内容的显示。这个页面包含了一个搜索框，用户可以在这个搜索框中输入书名、分类、出版社等信息，然后选择搜索方式（书名、分类、ISBN、出版社），最后点击“搜索”按钮进行搜索。当用户点击“搜索”按钮时，这个页面会调用名为“BookServlet”的Servlet来处理用户的请求，并返回相应的结果。

这个页面使用了Bootstrap框架来实现响应式布局，并使用了FontAwesome图标库来实现图标显示。

**具体来说，这个页面包含了以下几个部分：**

1. HTML头部：包含了一些元数据，如页面的编码方式、样式表等。

2. 搜索框：包含了一个文本框和一个下拉框，用户可以在文本框中输入关键字，然后选择下拉框中的选项进行搜索。

3. 搜索按钮：当用户点击“搜索”按钮时，会触发一个JavaScript函数，该函数会将用户输入的关键字和选项传递给后台的Servlet。

1. 用户信息：包含了一些用户相关的信息，如登录状态、账户信息等。

**5.3.5页脚界面**

这段代码是一个JSP页面，它的作用是实现一个网页底部的显示。

**具体来说，这个页面包含了以下几个部分：**

1. <%@ page %>指令：用于指定页面的属性，如页面的编码方式、样式表等。

2. HTML头部：包含了一些元数据，如页面的编码方式、样式表等。

3. 页面主体：包含了一个div元素，用于显示网页底部的内容。

4. CSS样式表：用于定义网页底部的样式。

这个页面使用了CSS样式表来定义网页底部的样式，其中包括了字体、颜色、背景等属性。

**5.3.6索引主界面**

　　这是一个JSP文件，它生成了一个在线书店的主页HTML页面。该页面包含了图书分类列表、幻灯片轮播图和人气图书推荐。

　　html、head 和 body 元素定义了 HTML 页面的结构。meta 元素指定了页面的字符编码。title 元素设置了页面的标题。link 元素包含了幻灯片样式表。script 元素引入了 jQuery 库和幻灯片脚本，并在文档加载完成后调用 initSlide() 函数初始化幻灯片。

　　jsp:include 元素在页面中包含了 header.jsp 文件的内容。

　　带有 id="container" 的 div 元素定义了页面主要内容的容器。在此容器内，首先是一个带有类名 cateul 的无序列表，用于显示图书分类列表。列表的第一项是列表标题，其余项使用 JSTL 的 c:forEach 标记遍历应用程序范围内的分类列表，并为每个分类生成一个链接。

　　带有类名 content 的 div 元素，用于显示页面主要内容。其中包括一个带有 id="slide" 的 div 元素，用于显示幻灯片；以及一个带有类名 goodsbar 的 div 元素，用于显示人气图书推荐。人气图书推荐部分使用 JSTL 的 c:forEach 标记遍历人气图书列表，并为每本图书生成一个包含图片、标题、价格信息和点击量的块。

　　最后，另一个 jsp:include 元素在页面中包含了 footer.jsp 文件的内容。

**5.3.7图书细节界面**

　　生成了一个在线书店的图书详情 HTML 页面。该页面包含了图书图片、信息、数量选择、加入购物车按钮和详细信息。

　　带有 id="container" 和类名 buy 的 div 元素定义了页面主要内容的容器。在此容器内，首先是一个带有类名 panel panel-default 的 div 元素，用于**显示图书详情**面板。面板包括一个带有类名 panel-heading 的 div 元素，用于**显示面板标题**；以及一个带有类名 panel-body 的 div 元素，用于**显示面板内容**。

　　面板内容部分包括一个带有类名bookpic pull-left的div元素，用于**显示图书图片**；以及一个带有类名bookinfo pull-left的div元素，用于**显示图书信息**。

　　图书信息部分包括一个表单，用于**提交加入购物车请求**。表单中包括一个无序列表，用于显示图书标题、作者、价格和数量选择。数量选择部分包括一个带有类名input-group input-group-sm的div元素，用于**显示数量输入框和增减按钮**。增减按钮绑定了点击事件，调用JavaScript函数进行数量增减。表单最后还包括一个提交按钮，用于提交表单。

　　在图书信息部分下方，是一个带有类名introbar的div元素，用于**显示图书详细信息**。详细信息部分包括一个表格，用于显示图书作者、出版社、出版时间、ISBN、字数和销量等信息；以及若干个带有类名 introbar-content 的 div 元素，用于显示图书内容介绍、作者介绍和目录等信息。

**5.3.8购买与订单**

**addcartok：**

　　带有 id="container" 的 div 元素定义了页面主要内容的容器。在此容器内，首先是一个使用 JSTL 的 c:if 标记进行条件判断的语句，检查用户是否未登录。如果此条件为真，则使用 Bootstrap 的警告组件显示一条警告消息，提示用户登录。

　　在此条件语句下方，显示一条消息，指示商品已成功加入购物车，并附有一个结账按钮和一个继续购物的链接。最后，另一个 jsp:include 元素在页面中包含了 footer.jsp 文件的内容。

　　这段页面的作用是生成一个在线书店的购物车 HTML 页面，显示商品已成功加入购物车的消息，并提供结账或继续购物的选项。

**showcart：**

　　生成了一个在线书店的购物车 HTML 页面。该页面显示了购物车中的图书列表，包括图书标题、单价、数量、小计金额和删除操作。如果购物车不为空，则还显示总计金额、去结算按钮和清空购物车按钮。

　　带有 id="container" 的 div 元素定义了页面主要内容的容器。在此容器内，首先是一个使用 JSTL 的 c:if 标记进行条件判断的语句，检查**用户是否未登录**。如果此条件为真，则使用 Bootstrap 的警告组件显示一条警告消息，提示用户登录。

　　然后是一个带有类名 table table-hover 的 table 元素，用于**显示购物车列表**。表格包括一个表头，用于显示各列标题；以及一个表体，用于显示购物车中的图书列表。

　　表体部分包括一个使用 JSTL 的 c:if 标记进行条件判断的语句，检查**购物车是否不为空**。如果此条件为真，则使用 JSTL 的 c:forEach 标记遍历购物车中的图书列表，并为每本图书生成一行表格行。表格行包括若干个单元格，**分别显示**图书标题（并提供链接以便购买图书）、单价、数量（并提供增加和减少数量的链接）、小计金额和删除操作（提供删除图书的链接）。还包括另一个使用 JSTL 的 c:if 标记进行条件判断的语句，检查购物车是否为空。如果此条件为真，则显示一条提示信息，指示购物车为空，并提供继续选购的链接。

　　外层的最后一个条件判断的语句，检查购物车是否不为空。如果此条件为真，则显示总计金额、去结算按钮和清空购物车按钮。

**showorder：**

　　生成了一个在线书店的订单 HTML 页面。该页面显示了用户的订单列表，包括图书标题、购买数量、总价和下单时间。

　　在 body 元素内，有两个使用c:if标记进行条件判断的语句，分别检查 olist 变量是否为空或不为空。如果为空，则显示一条消息，指示用户没有订单。如果不为空，则显示一个标题和一条水平线，以及一个表格用于显示订单列表。表格包括一个表头，用于显示各列标题；以及若干个表格行，用于显示订单列表。

　　表格行部分使用 JSTL 的 c:forEach 标记遍历 olist 集合中的每个订单，并为每个订单生成一行表格行。表格行包括若干个单元格，分别显示订单的图书标题、购买数量、总价（数量乘以单价）和下单时间。

# 6. 界面设计

## 6.1界面样式设计

**6.1.1 初始化：**

打开集成开发环境INTELLIJ IDEA，运行BOOKSHOP，便可以进入初始界面：

界面顶部有图标、搜索栏、登录信息和跳转按钮。

主界面：界面左侧有图书分类的索引栏，界面中部有滚动菜单和人气图书。



## 6.2界面交互设计

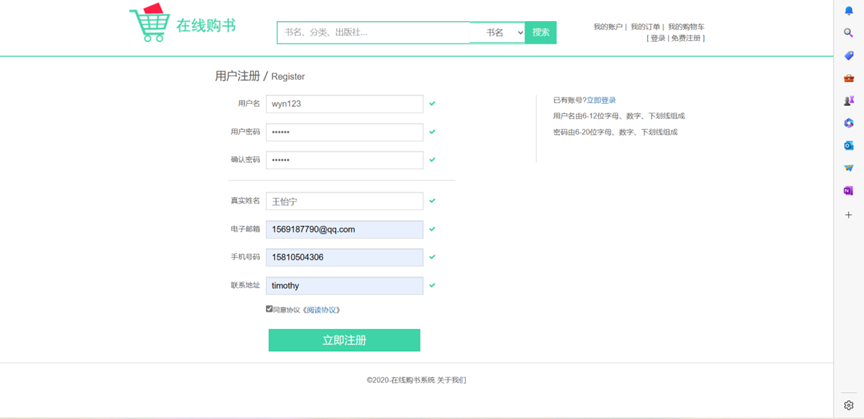
**6.2.1 登录：**

用户若想要进行购买等操作需要先登录，登录界面如下：



**6.2.2 注册：**

如果使用者没有账号，则用注册功能完成账号注册，具体界面如下：

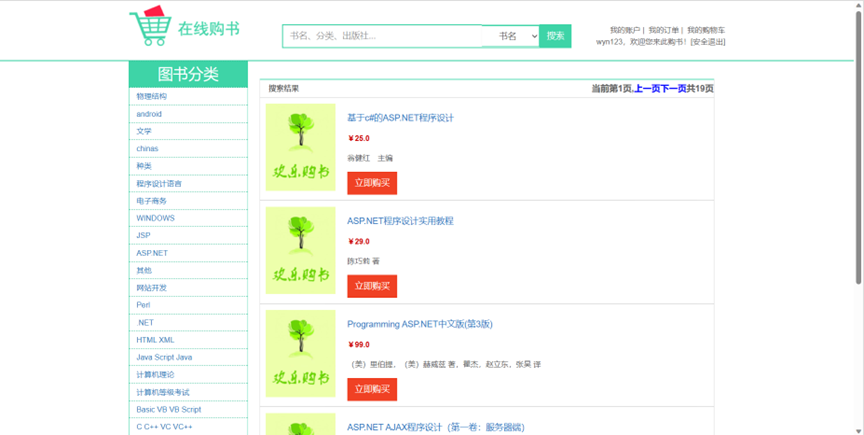


注册成功界面如下：

****

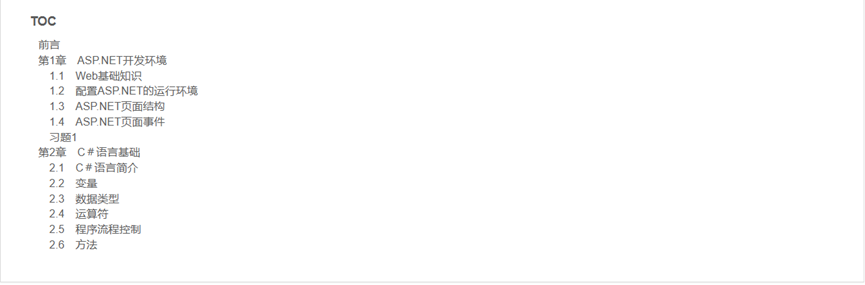
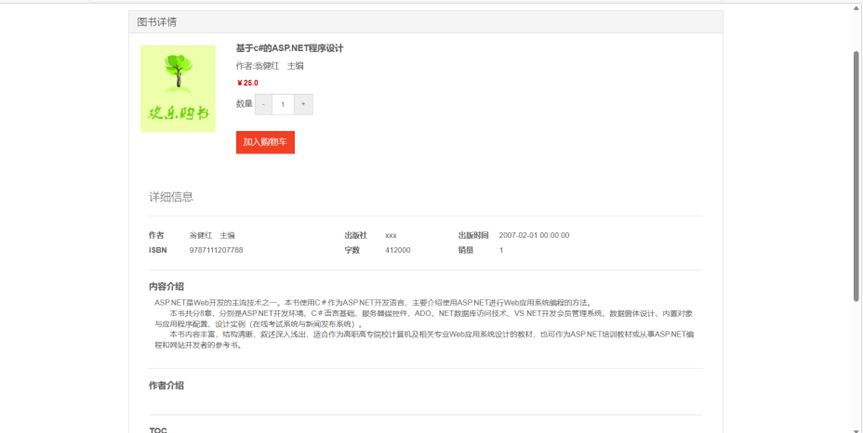
**6.2.3 查找：**

用户可以根据书名，分类，ISBN编号和出版社信息进行查找，查找后的结果将一一列举出来，效果如下所示。

还可以根据图书分类，浏览各个图书的内容。

**6.2.4 浏览：**

用户点击相应图书的，便可以浏览相应的图书。包括书的具体简介和基本信息，用户还可以在这里购买并且加入购物车。



**6.2.5 购买：**

用户浏览到喜欢的图书后可以加入购物车，并选择购买数量：

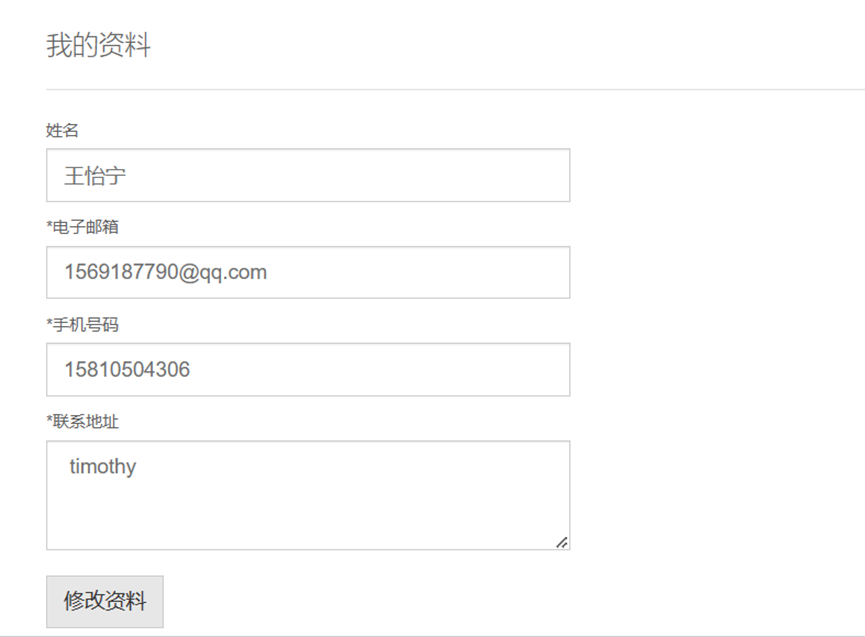
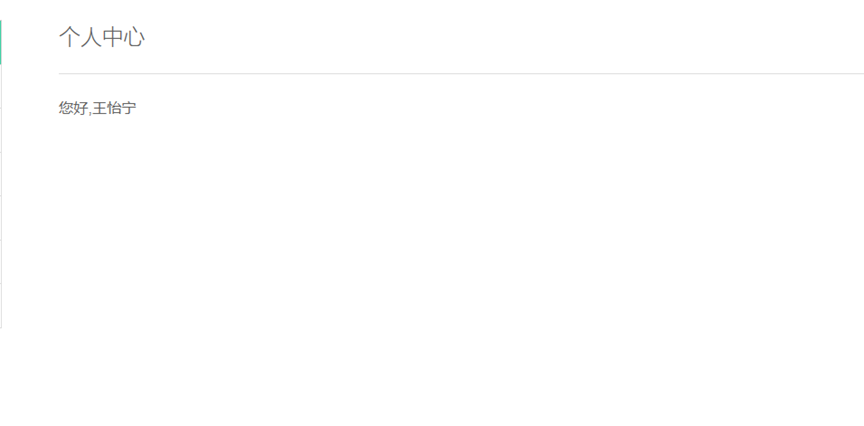
在购物车界面可以看到购物车的详细信息：包括有结算，清空购物车，删除商品

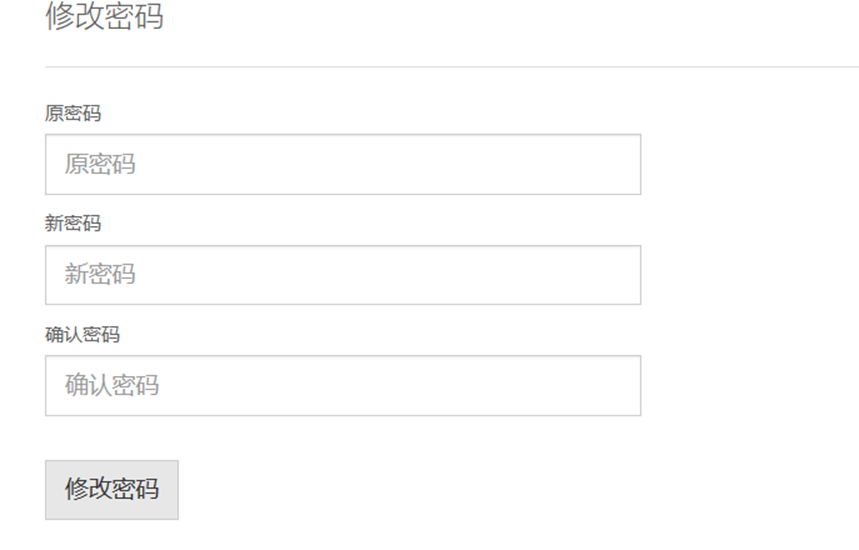


结算后进入订单界面，可以看到订单的相应信息：（包括总价，内容，时间等）

**6.2.6 个人信息：**

在个人信息界面可以查看个人信息和个人资料：

还可以修改资料，和修改密码。

之后可以安全退出。

# 7．出错处理设计

## 7.1错误类型及出错处理对策

若用户未能正常按要求进行操作，本系统怎会停止在当前这一步，并阻止下一步继续。