# 数据库实训项目报告

**组长：**

姓名：马熙弘 学号：201830662165 班级：软件工程4班

**成员：**

姓名：刘 柽 学号：201830662028 班级：软件工程4班

姓名：骆家森 学号：201830662134 班级：软件工程4班

姓名：温灿雄 学号：201830663124 班级：软件工程4班

目录

[数据库实训项目报告 1](#_Toc44625989)

[一、项目简介 3](#_Toc44625990)

[二、项目开发平台和工具 3](#_Toc44625991)

[三、页面整体架构 3](#_Toc44625992)

[四、页面交互功能 6](#_Toc44625993)

[五、系统业务功能与技术设计 8](#_Toc44625994)

[六、后端服务器搭建 17](#_Toc44625995)

[七、后端数据库设计 18](#_Toc44625996)

[八、系统功能测试 23](#_Toc44625997)

## 一、项目简介

我们小组所选项目为**校园二手交易平台**。

主要实现了：用户注册与登录、物品上架出售与分类展示、求物品区、购物车、个人信息展示与修改等功能。

## 二、项目开发平台和工具

1. 整体架构：整体采用B/S架构。
2. 前端：前端采用浏览器展示网页，采用的开发语言有：

HTML、CSS、JavaScript。

还使用了部分简单的Vue框架下的语句。

例如，简单的数据绑定{{user\_name}}，v-if条件渲染，v-for列表渲染，v-on:click点击事件绑定，v-model绑定表单数据。

1. 后端服务器部署：使用springboot框架、Java开发语言，使用JDBC连接数据库。
2. 数据库：使用mysql

## 三、页面整体架构

1. 可导航页面：

可导航页面，是每个页面顶部都有导航栏，可以点击直接跳转的页面。

有：首页，全部物品页，求物品区，个人中心页。

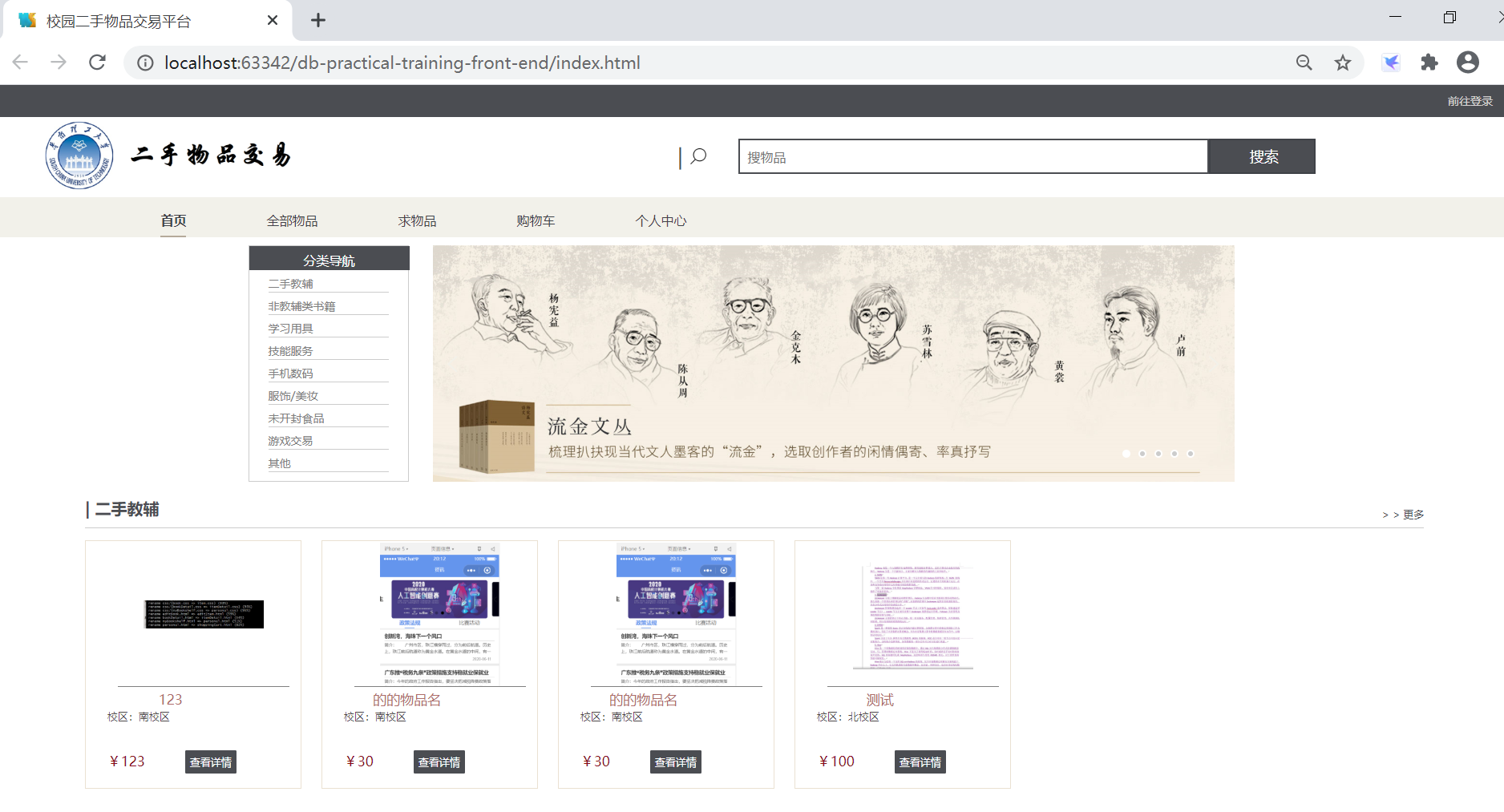


图1：首页



图2：全部物品页



图3：个人中心页

1. 只可跳转进入的页面：

只可跳转页面只能从可导航页点击特定位置跳转。

有：物品详情页（包括出售物品、请求物品），上传出售物品页，上传请求物品页。

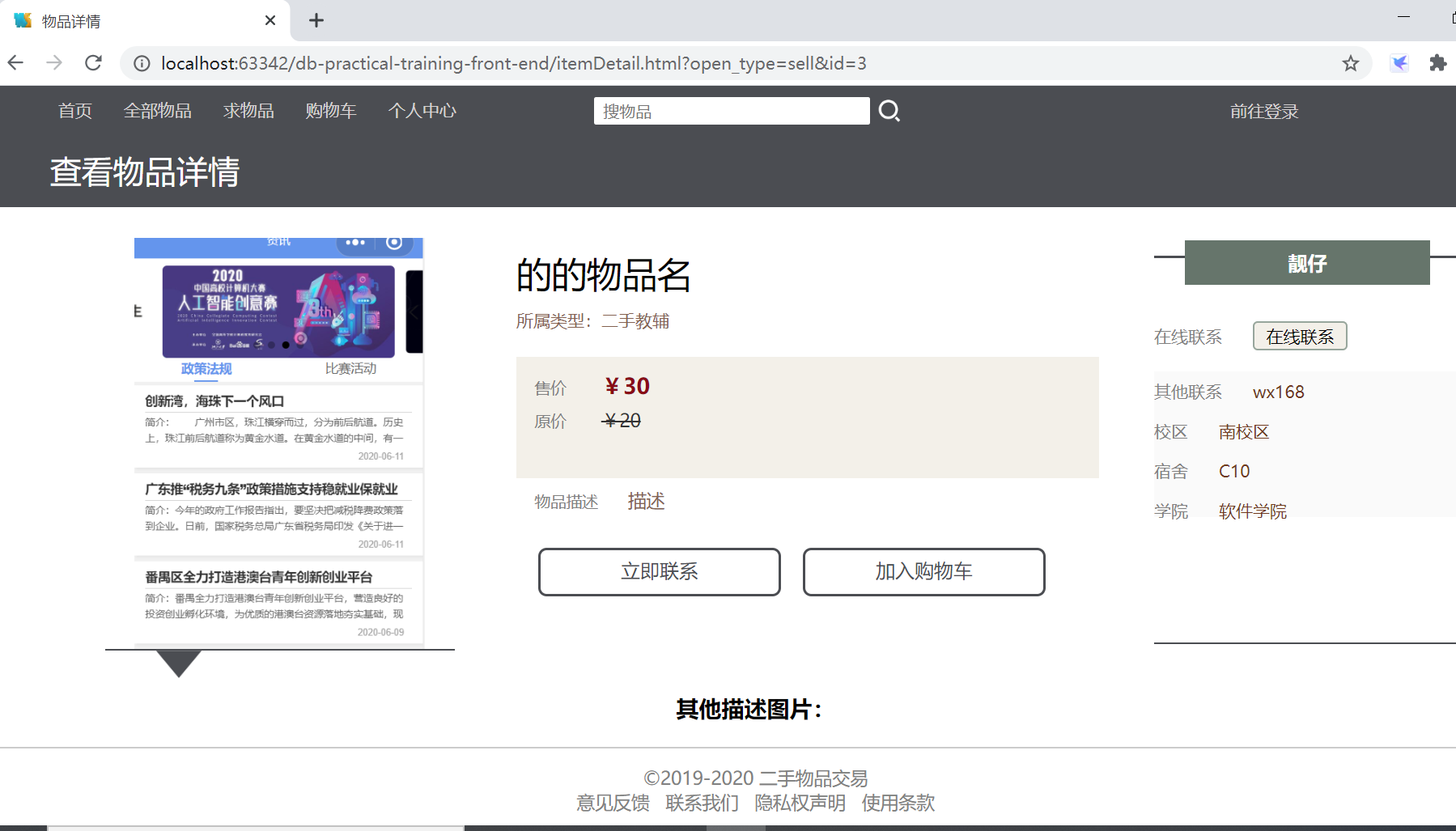


图4：物品详情页

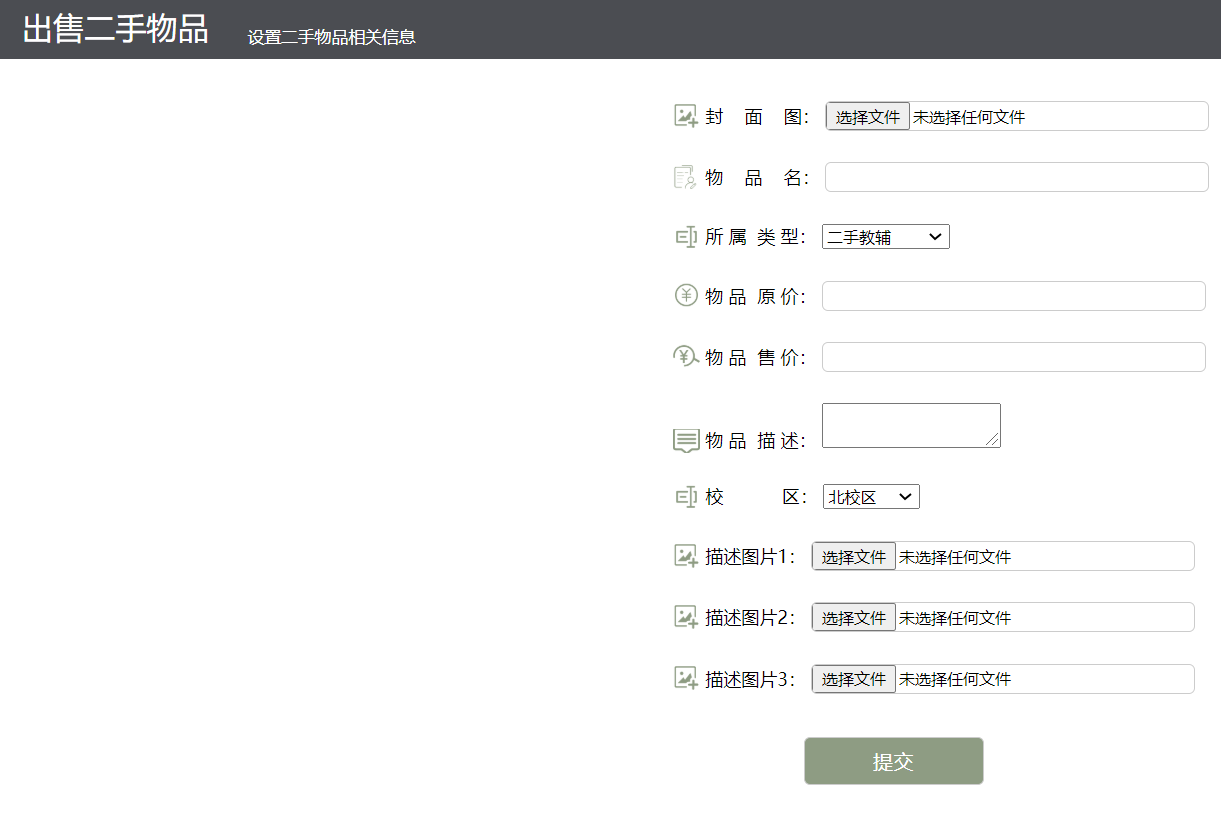


图5：出售物品上传页

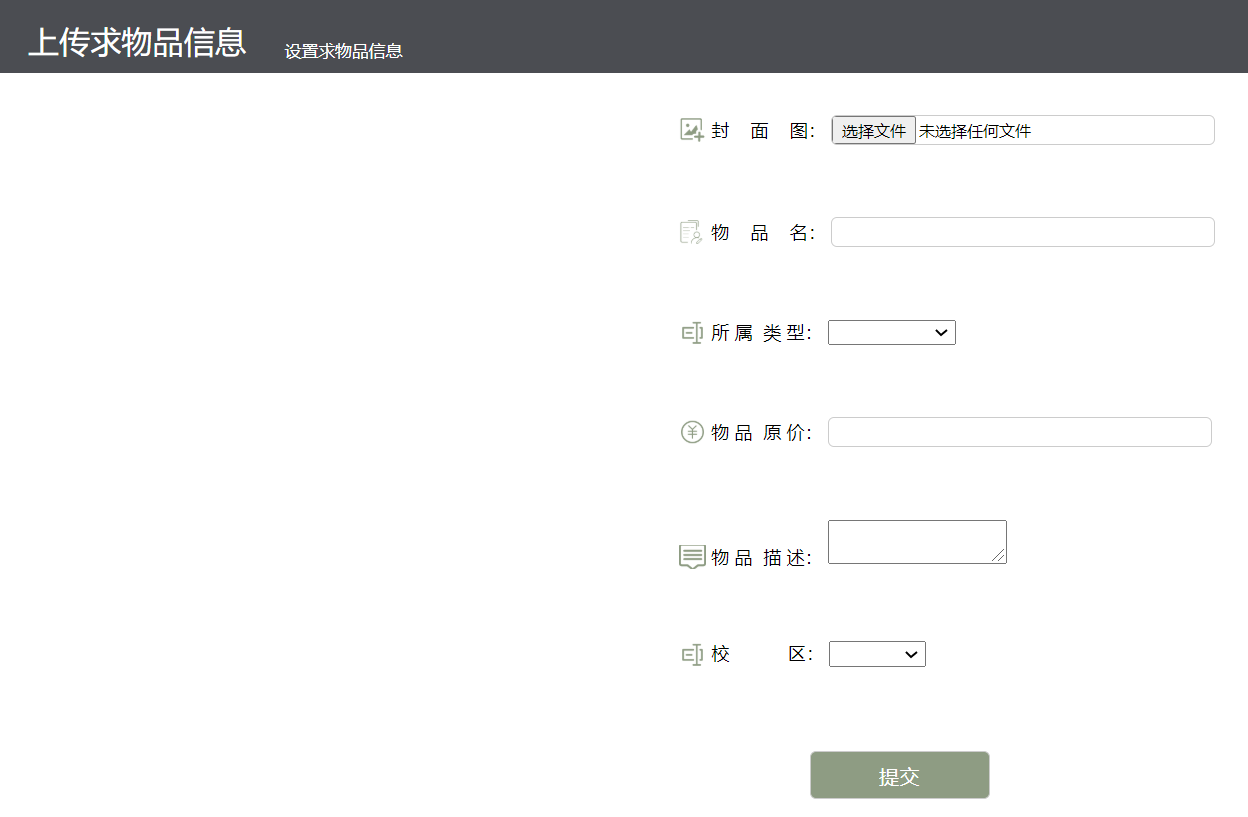


图6：请求物品上传页

1. 各个页面都包含的部件：

a.每个页面顶部的登录状态栏，用于判定用户的登录状态。

b.每个页面顶部的搜索框，用于搜索出售物品。目前仅支持搜索出售物品的信息，不支持搜索请求物品的信息。



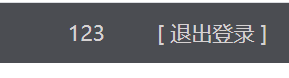


图7、8：登录状态



图9：搜索框

## 四、页面交互功能

网站有多个界面，具体内容如下：

1.首页：使用者前往网页首先看到的是首页。首页顶端导航栏显示用户的登录信息，如果已登录则显示用户名信息；如果没登录则有“前往登录”的按钮，点击后可以进入登录页面。往下左边是网站标题“二手物品交易”，右边是搜索栏，通过搜索栏，用户可以对数据库里已经有的商品名称和描述进行查询，数据库返回符合关键词的商品，显示在用户面前。往下左侧是分类导航，每个商品都有其分类属性。点击分类导航里的某个具体分类，数据库返回该分类属性的商品并显示给用户。其右是推广框，轮流随机展示几个商品供使用者查看，如果使用者看到心仪的商品，即可点击直接进入商品页面。再往下则是为每个商品分类单独分出一栏，显示部分该分类的商品。

2.登录界面：在首页点击前往登录后会进入登录界面。登录界面内容简单，画面中央有两个文本输入框，分别输入账号和密码，下方共有4个按钮，分别是“没有账号？前往注册“、“忘记密码？”、“登录”、“返回首页”。“没有账号？前往注册”可以进入注册界面；“忘记密码”按钮可以进入找回密码界面；“登录”按钮会向数据库发出查询请求，寻找是否存在相符的账号密码，有则登录成功，进入用该账户登录成功后的首页；“返回首页”按钮则是放弃登录，直接以未登录状态返回首页。

3.搜索界面：在首页搜索框输入商品名称，点击搜索按钮就可以进入搜索页面，网页向数据库查询商品名称，数据库返回目标商品的数据。然后在页面上显示出搜索到的具体商品图片、价格等简略信息。图片和商品名称是超链接，点击可以打开具体的商品页面。

4.商品页面：商品页面导航栏有“首页”、“全部物品”、“求物品”、“购物车”、“个人重心”等和搜索框以及搜索按钮。下方为商品具体描述：左为商品封面图；中为商品名称、商品所属分类、和文字物品描述；右为商品发布人的信息，包括练习方式、校区、宿舍、学院等。并有一个“立即联系”按钮。再下为其他描述图片。

5.所有商品页面：在首页或导航栏点击“所有商品”按钮可以打开所有商品页面。该页面直接展示所有还未出售的已发布二手商品，形式类似搜索页面。用户可以点击图片或商品名称进入商品页面。

6.求物品页面：在首页或导航栏点击“求物品”按钮可以打开求物品页面。页面将显示需要某种物品的同学所发布的请求，可以点击进入具体页面。

7。购物车：在首页或导航栏点击“购物车”按钮可以打开购物车页面。页面显示目前已加入购物车的商品图片和名称。如果不想买了也可以通过点击“移出购物车”按钮将物品移出购物车。购物车功能需登录才可以使用，否则会提示未登录。同样，可以通过点击商品图片或名称进入具体商品页面

8.个人页面：点击“个人页面”按钮可以进入个人页面。个人页面中可以编辑自己的个人信息，包括用户名、联系方式、校区、宿舍、学院等。同时可以查看自己当前正在出售的商品和求物品信息。通过点击下方的“我的出售”和“我的求物品”即可。在“我的出售”标签里，右方还有“上传出售物品”和“删除二手物品”两个按钮，上传出售物品可以在网站上张贴新的待出售物品，点击后进入出售物品上传页面；删除二手物品可以取消正在展示的物品。在“我的求物品”标签里，右方有“上传求物品信息”和“删除求物品”按钮，点击“上传求物品信息”可以进入求物品上传页面，点击“删除求物品按钮可以取消求物品。

9.出售物品上传页面：该页面有多个输入框，依提示分别为封面图、物品名、所属类型、物品原价、物品售价、物品描述、描述图片1、描述图片2、描述图片3。用户上传4张图片，并输入所有所需的信息后，便可点击“提交”按钮上传。上传后的商品可以被其他人浏览到，同时可以在个人页面里查看。

10.求物品上传页面：该页面类似出售物品上传页面，输入信息后可以提交上传。上传后的求物品可以被其他人浏览到，同时可以在个人页面查看。

11.物品编辑页面：已经上传的物品，需要修改信息时，可以在个人页面点击商品进入物品编辑页面。该页面与出售物品上传页面类似，修改信息后，按“提交”便可以更新已上传的物品的信息。

## 五、系统业务功能与技术设计

1. 登录，用户操作

用户要想进行个人用户操作（如加入购物车、创建物品等而非浏览），需要先注册。

注册时，服务器会先检查有无已存在的用户id，没有才能成功注册。

注册和登录用的http请求是post方法而不是get，以保证信息的安全。

代码如下：

*//登录，发送账号密码，返回操作用的code，和用户nickname*login:**function**(){  
 **let** param = **new** URLSearchParams();  
 param.append(**"user\_id"**,**this**.**userID**)  
 param.append(**"user\_password"**,**this**.**password**)  
 **let** self=**this** axios.post(**localStorage**.serverUrl+**'User/login'**, param)  
 .then(**function** (response) {  
 **console**.log(**'res:'**)  
 **console**.log(response);  
 *//下面设置用户登录态* **localStorage**.setItem(**'user\_id'**,response.data.map.**user\_id**);  
 **localStorage**.setItem(**'operation\_code'**,response.data.opcode);  
 **localStorage**.setItem(**'nickname'**,response.data.map.**user\_name**);  
 **localStorage**.setItem(**'userInfo'**,**JSON**.stringify(response.data.map));  
  
 **window**.**location**.**href**=**"index.html"**;  
 })  
 .catch(**function** (error) {  
 **console**.log(error);  
 });  
  
},

每次的用户登录，都会从服务器获取一个操作令牌，令牌在服务器也有一份。每次涉及用户操作、发送http请求时，在发送用户id的同时，也会发送这个操作令牌，令牌与服务器一致，才能进行用户操作。

这样设计的目的，是保证后端数据的安全，否则在前端只需要用用户id，就可以随意更改后端数据库的数据了。

代码如下：

*//发送用户id，获取购物车物品列表***var** self=**this**;  
axios.get(**localStorage**.serverUrl+**'shop/getUsersShopCartItems?user\_id='**+***login\_status***.**id**+**'&user\_login\_code='**+***login\_status***.**operation\_code**)  
 .then(**function** (response) {  
 **console**.log(response);  
 self.**itemList**=response.data;  
 })  
 .catch(**function** (error) {  
 **console**.log(error);  
 });

1. 商品的上架

在个人中心页，可跳转上传物品页。

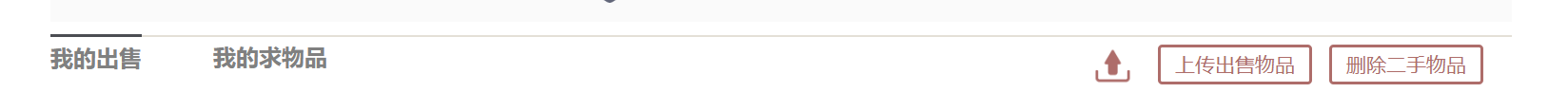


图10：上传出售物品按钮



图11：上传出售物品页

上传代码如下：  
upload1:**function**(){  
 **let** form = **new FormData**();  
 form.append(**'cover\_pic'**, **this**.**cover\_pic**,**this**.**cover\_pic**.name);  
 **if**(**this**.**desc\_img1**!=**''**) form.append(**'desc\_img1'**, **this**.**desc\_img1**, **this**.**desc\_img1**.name);  
 **if**(**this**.**desc\_img2**!=**''**) form.append(**'desc\_img2'**, **this**.**desc\_img2**, **this**.**desc\_img2**.name);  
 **if**(**this**.**desc\_img3**!=**''**)form.append(**'desc\_img3'**, **this**.**desc\_img3**, **this**.**desc\_img3**.name);  
 form.append(**'commodity\_name'**,**this**.**commodity\_name**)  
 form.append(**'commodity\_type'**,**this**.**commodity\_type**)  
 form.append(**'commodity\_original\_price'**,**this**.**commodity\_original\_price**)  
 form.append(**'commodity\_price'**,**this**.**commodity\_price**)  
 form.append(**'commodity\_description'**,**this**.**commodity\_description**)  
 form.append(**'area'**,**this**.**area**)  
 form.append(**'user\_id'**,**localStorage**.**user\_id**)  
 form.append(**'user\_login\_code'**,**localStorage**.**operation\_code**)  
 axios.post(**localStorage**.serverUrl+**'commodity/addCommodity'**, form, {**headers**: {**'Content-Type'**: **'multipart/form-data'**}})  
 .then(**function** (response) {  
 **if**(response.data==**'0'**){}  
 **else**{  
 ***modal***.**ifmodal** = **true** }  
 })  
 .catch(**function** (error) {  
 **console**.log(error);  
 });  
  
}

表单的标签如下：

<**div class="container"**>  
 <**form class="upload-form" id="formContain"**>  
 <**p**>  
 <**span class="pic-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 封**&nbsp;&nbsp;&nbsp;&nbsp;**面**&nbsp;&nbsp;&nbsp;&nbsp;**图：  
 <**input @change='add\_img' type="file"**>  
 *<!-- <img src="cover\_pic" id="cover\_pic"> -->* </**p**>  
  
 <**p**>  
 <**span class="author-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 物**&nbsp;&nbsp;&nbsp;&nbsp;**品**&nbsp;&nbsp;&nbsp;&nbsp;**名：  
 <**input type="text" v-model="commodity\_name" name="commodity\_name"**>  
 </**p**>  
 <**p**>  
 <**span class="name-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 所**&nbsp;**属**&nbsp;&nbsp;**类**&nbsp;**型：  
 <**select name="commodity\_type" v-model="commodity\_type"**>  
 <**option disabled value=""**>请选择</**option**>  
 <**option value="二手教辅"**>二手教辅</**option**>  
 <**option value="非教辅类书籍"**>非教辅类书籍</**option**>  
 <**option value="学习用具"**>学习用具</**option**>  
 <**option value="技能服务"** >技能服务</**option**>  
 <**option value="手机数码"**>手机数码</**option**>  
 <**option value="服饰/美妆"**>服饰/美妆</**option**>  
 <**option value="零食/饮品/水果"**>未开封食品</**option**>  
 <**option value="玩具/游戏交易"**>游戏交易</**option**>  
 <**option value="其他"**>其他</**option**>  
</**select**>  
 </**p**>  
  
 <**p**>  
 <**span class="orign-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 物**&nbsp;**品**&nbsp;&nbsp;**原**&nbsp;**价：  
 <**input type="text" v-model="commodity\_original\_price" name="commodity\_original\_price"**>  
 </**p**>  
 <**p**>  
 <**span class="price-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 物**&nbsp;**品**&nbsp;&nbsp;**售**&nbsp;**价：  
 <**input type="text" v-model="commodity\_price" name="commodity\_price"**>  
 </**p**>  
  
 <**p**>  
 <**span class="desc-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 物**&nbsp;**品**&nbsp;&nbsp;**描**&nbsp;**述：  
 <**textarea type="text" v-model="commodity\_description" name="commodity\_description"**></**textarea**>  
 </**p**>  
  
 <**p**>  
 <**span class="name-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 校**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**区：  
 <**select name="area" v-model="area"**>  
 <**option disabled value=""**>请选择</**option**>  
 <**option value="北校区"**>北校区</**option**>  
 <**option value="南校区"**>南校区</**option**>  
 <**option value="国际校区"**>国际校区</**option**>  
</**select**>  
 </**p**>  
  
 <**p**>  
 <**span class="pic-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 描述图片1： <**input type="file" name="desc\_img1" @change='add\_desc\_img'**>  
 *<!-- <img src="desc\_img1" id="desc\_img1"> -->* </**p**>  
 <**p**>  
 <**span class="pic-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 描述图片2： <**input type="file" name="desc\_img2" @change='add\_desc\_img'**>  
 *<!-- <img src="desc\_img2" id="desc\_img2"> -->* </**p**>  
 <**p**>  
 <**span class="pic-icon"**>**&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;**</**span**> 描述图片3： <**input type="file" name="desc\_img3" @change='add\_desc\_img'**>  
 *<!-- <img src="desc\_img3" id="desc\_img3"> -->* </**p**>  
  
  
 <**input type="submit" class="submit-btn" name="item-submit"**>  
 </**form**>  
</**div**>

1. 商品分类展示

首先，进入首页，左侧就是分类导航，左侧轮播图，底下是具体的商品。



图12：首页展示商品

全部物品页，左侧也有筛选菜单，



图13：全部物品页--筛选菜单及商品展示

点击，进入详情页，可查看详情



图14：商品详情页

首页的部分标签代码：

*<!-- 导航 -->*<**header class="navbar"**>  
 <**div class="nav-info" id="login\_status"**>  
 <**span v-if="id==''"**>  
 <**a href="login.html" class="username"**>前往登录</**a**>  
 </**span**>  
 <**span v-else**>  
 <**a href="personal.html" class="username"**>{{name}}</**a**>  
 <**a href="#" class="logout" v-on:click="logout()"**>[ 退出登录 ]</**a**>  
 </**span**>  
 </**div**> *<!-- nav-info -->  
 <!--搜索框-->* <**div class="nav-search" id="search"** >  
 <**a href="index.html"**><**img class="logo" src="img/logo3.png"**></**a**>  
 <**div class="search-form"**>  
 <**a href="#" class="search-logo"**>| **&nbsp;**<**img src="img/search.png"**></**a**>  
 <**input type="search" class="searchIn" placeholder="搜物品" v-model="keyWord"**>  
 <**button class="searchBtn" v-on:click="search"**>搜索</**button**>  
 </**div**>  
 </**div**>*<!-- nav-search -->* <**ul class="menu"**>  
 <**li**><**a class="active" href="index.html"**>首页</**a**></**li**>  
 <**li**><**a href="allItems.html"**>全部物品</**a**></**li**>  
 <**li**><**a href="askItem.html"**>求物品</**a**></**li**>  
 <**li**><**a href="shoppingCart.html"**>购物车</**a**></**li**>  
 <**li**><**a href="personal.html"**>个人中心</**a**></**li**>  
 </**ul**>   
</**header**>  
  
<**div id="index"** >  
*<!-- 轮播 -->*<**div class="carousel"**>  
 <**div class="item-guide"**>  
 <**h4**>分类导航</**h4**>  
 <**ul class="item-class"**>  
 <**li v-for="(item,index) in classify"**>  
 <**a href="#" v-on:click="change\_classify(index)"**>{{item}}</**a**>  
 </**li**>  
 </**ul**>  
 </**div**>*<!-- item-guide -->* <**div class="slider"**>  
 <**div class="slider-img"**>  
 <**ul class="slider-img-ul"**>  
 <**li**><**img src="img/carousel/5.jpg"**></**li**>  
 <**li**><**img src="img/carousel/1.jpg"**></**li**>  
 <**li**><**img src="img/carousel/2.jpg"**></**li**>  
 <**li**><**img src="img/carousel/3.jpg"**></**li**>  
 <**li**><**img src="img/carousel/4.jpg"**></**li**>  
 <**li**><**img src="img/carousel/5.jpg"**></**li**>  
 <**li**><**img src="img/carousel/1.jpg"**></**li**>  
 </**ul**>  
 </**div**>  
 </**div**>*<!-- slider -->*</**div**>*<!-- carousel -->  
………………………………….dwadsasadad大大大大大大达瓦伟大的哇  
<!-- 物品分类-->*<**div id="container"**>  
 <**div class="item-part" v-for="(type,index) in classify" v-if="showing==index"**>  
 <**h3 class="item-title"**><**a href="#"**>| {{type}}</**a**></**h3**>  
 <**a class="more" href="#" v-on:click="getMore(index)"** > > > 更多</**a**> *<!--跳转到全部物品页，传下标-->* <**ul class="item-lists"**>  
 <**li class="item-list" v-for="item in itemList" v-on:click="to\_detail(item.commodity\_id)"**>  
 <**a href="#" class="item-pic"**>  
 <**img :src="item.commodity\_cover\_pic\_url"**>  
 </**a**>  
 <**a href="#" class="item-info"**>  
 <**h5 class="item-name"**>{{item.commodity\_name}}</**h5**>  
 <**span class="item-detail"**>校区：{{item.area}}</**span**>  
 </**a**>  
 <**span class="item-price"**>￥{{item.commodity\_price}}  
 <**a href="#" class="item-buy"**>查看详情</**a**>  
 </**span**>  
 </**li**>  
  
  
  
  
 </**ul**>*<!-- item-list end -->* </**div**>*<!-- item-part end -->*</**div**> *<!-- container end-->*</**div**>

1. 交易流程

点击物品进入详情页，



图15：商品详情页

可加入购物车，

可根据右侧卖家的联系方式联系，进行线下交易。

交易完成后，在个人中心，自己出售的物品下点击完成交易按钮，则该商品下架，不再显示。

5.所有商品页面：在首页或导航栏点击“所有商品”按钮可以打开所有商品页面。该页面直接展示所有还未出售的已发布二手商品，形式类似搜索页面。用户可以点击图片或商品名称进入商品页面。

6.求物品页面：在首页或导航栏点击“求物品”按钮可以打开求物品页面。页面将显示需要某种物品的同学所发布的请求，可以点击进入具体页面。

7。购物车：在首页或导航栏点击“购物车”按钮可以打开购物车页面。页面显示目前已加入购物车的商品图片和名称。如果不想买了也可以通过点击“移出购物车”按钮将物品移出购物车。购物车功能需登录才可以使用，否则会提示未登录。同样，可以通过点击商品图片或名称进入具体商品页面

8.个人页面：点击“个人页面”按钮可以进入个人页面。个人页面中可以编辑自己的个人信息，包括用户名、联系方式、校区、宿舍、学院等。同时可以查看自己当前正在出售的商品和求物品信息。通过点击下方的“我的出售”和“我的求物品”即可。在“我的出售”标签里，右方还有“上传出售物品”和“删除二手物品”两

## 六、后端服务器搭建

1.框架：

基于Java的Springboot框架(MVC三层架构)：

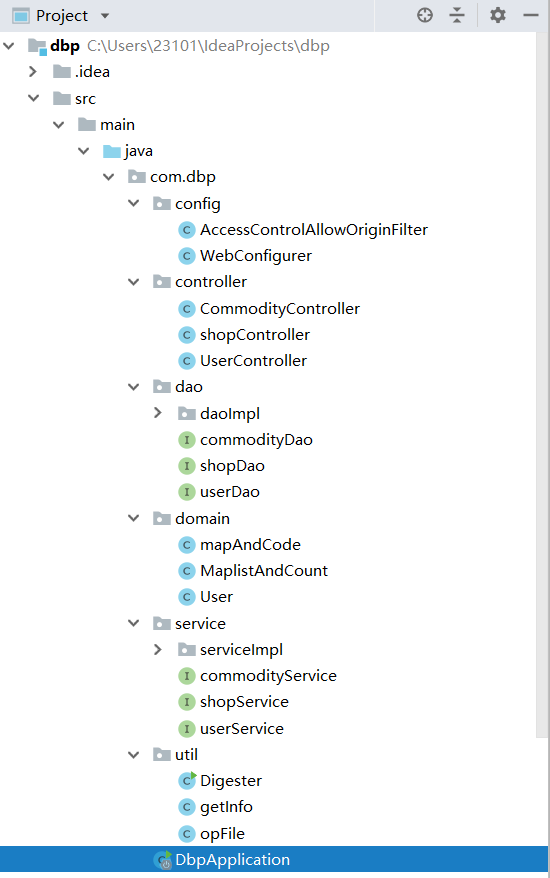


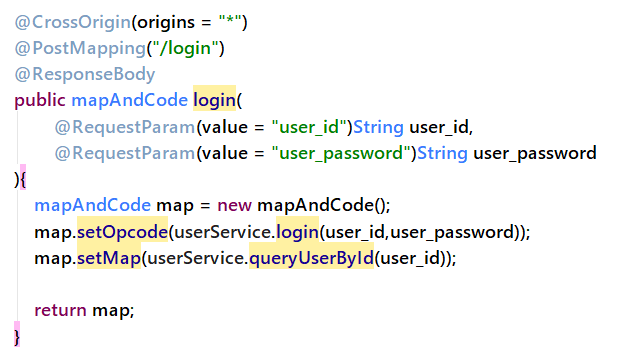
图16：后端工程目录结构

2.运行环境

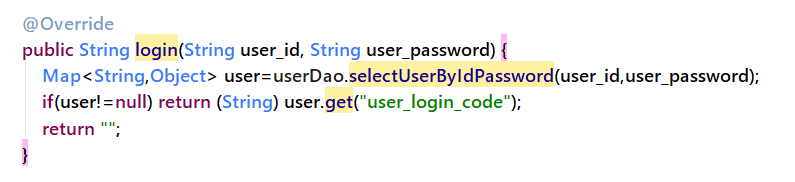
Centos7，Mysql,Tomcat

3.运行方式

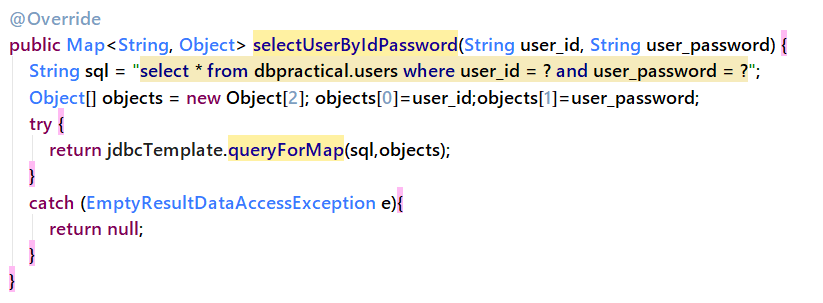
在Controller层接收请求，通过service层定义的服务来调用Dao层对应的数据库访问方法。接口示例：



UserController的登陆接口



UserService的登陆服务



Dao的数据库访问方法

使用Map<String,Object>而代替了Bean，减少了代码量。

## 七、后端数据库设计

1. 根据各个页面分析，得出所需数据

**出售物品信息：（\*号必填）**

物品名\*

所属类型\*

原价

售价\* （求物品没有价格）

描述\*

封面图片

描述图片1

描述图片2

描述图片3

物品状态：是否已完成交易

**需求物品信息：（\*号必填）**

物品名\*

所属类型\*

描述\*

封面图片

物品状态：是否已完成交易

**用户信息：**

账户，密码，每次操作用的code，

昵称，联系方式，校区，宿舍，学院

**用户连接信息：（一对多）**

加入购物车的物品id，购物车只有出售物品，没有请求物品

自己创建的出售物品id

自己创建的请求物品id

1. 画出E-R图

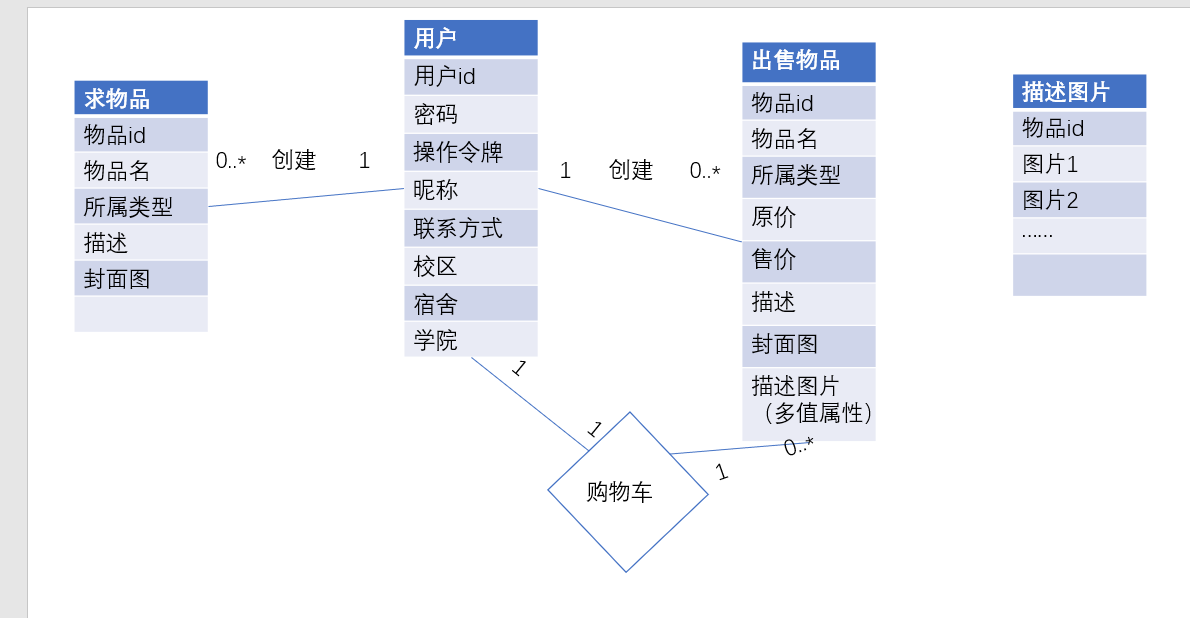


图17：数据库E-R图

1. 数据库DDL

create database dbpractical;

use dbpractical;

drop table commodity\_images;

drop table commodity;

drop table done\_commodity\_images;

drop table done\_commodity;

drop table required\_commodity;

drop table users;

create table users(

user\_id varchar(15) primary key,

user\_name varchar(255),

user\_password varchar(31),

user\_login\_code char(40),

tel char(11),

user\_area char(4),

dormitory varchar(7),

academy varchar(15),

check (user\_area in ('北校区','南校区','国际校区'))

);

create view seller\_commodity as(

select \* from users natural join commodity

);

create table commodity(

commodity\_id int unsigned auto\_increment primary key,

commodity\_name varchar(31),

commodity\_type varchar(15),

commodity\_original\_price int2,

commodity\_price int2,

commodity\_description varchar(1023),

commodity\_cover\_pic\_url varchar(2083),

publish\_time\_stamp char(16),

area char(4),

user\_id varchar(15),

foreign key(user\_id) references users(user\_id) on delete cascade,

check (area in ('北校区','南校区','国际校区')),

FULLTEXT (commodity\_name,commodity\_description,commodity\_type) WITH PARSER ngram,

FULLTEXT (commodity\_type) WITH PARSER ngram

);

create table commodity\_images(

commodity\_id int unsigned,

image\_url varchar(2083),

foreign key(commodity\_id) references commodity(commodity\_id) on delete cascade

);

create table done\_commodity(

commodity\_id int unsigned primary key,

commodity\_name varchar(31),

commodity\_type varchar(15),

commodity\_original\_price int2,

commodity\_price int2,

commodity\_description varchar(1023),

commodity\_cover\_pic\_url varchar(2083),

publish\_time\_stamp char(16),

area char(4),

user\_id varchar(15),

foreign key(user\_id) references users(user\_id) on delete cascade

);

create table done\_commodity\_images(

commodity\_id int unsigned,

image\_url varchar(2083),

foreign key(commodity\_id) references done\_commodity(commodity\_id) on delete cascade

);

create table required\_commodity(

commodity\_id int unsigned auto\_increment primary key,

commodity\_name varchar(31),

commodity\_type varchar(15),

commodity\_description varchar(1023),

commodity\_cover\_pic\_url varchar(2083),

publish\_time\_stamp char(16),

area char(4),

user\_id varchar(15),

foreign key(user\_id) references users(user\_id) on delete cascade,

check (area in ('北校区','南校区','国际校区'))

);

create table shopping\_cart(

user\_id varchar(15),

commodity\_id int unsigned,

primary key(user\_id,commodity\_id),

foreign key(user\_id) references users(user\_id) on delete cascade,

foreign key(commodity\_id) references commodity(commodity\_id) on delete cascade

);

1. 数据库索引

1)在用户经常访问的数据表上，建立了关于用户id的索引

alter table users add index user\_id\_index(user\_id);

alter table commodity add index commodity\_user\_id\_index(user\_id);

alter table done\_commodity add index done\_commodity\_user\_id\_index(user\_id);

alter table required\_commodity add index required\_commodity\_user\_id\_index(user\_id);

2)为了提供关键字搜索、类型筛选的功能。我们还在一些字段上建立了全文索引以提高效率：



图18：全文索引

1. 查询优化

1）在编写SQL语句访问数据库过程中，为了提高查询效率,我们极少直接使用Natural join。取而代之的是先筛选后链接，示例如下：

with com as

(select \* from dbpractical.commodity where commodity\_id = ?)

select *\** from com natural join users

2）分页

为了避免前端数据冗余，我们使用了分页机制，编写如下SQL语句：

select \* from dbpractical.commodity order by commodity\_id desc limit 0,10

从表中得到最新的10条数据，在接下来请求时，则有如下SQL语句：

select \* from dbpractical.commodity order by commodity\_id desc limit 10,10

select \* from dbpractical.commodity order by commodity\_id desc limit 20,10

select \* from dbpractical.commodity order by commodity\_id desc limit 30,10

………

以此类推，每次请求得到10条数据。

就算在分次请求间隔有新的数据插入表中，此过程也不会受影响：我们记录了第一次访问时表中的行数，使其后每次访问都带上此行数，通过此行数和请求时行数判断limit的第一个参数取值再执行即可。

1. 数据库安全

为了防止SQL注入，后端数据库的执行一律采用预编译模式执行，示例如下：

String sql=**"insert into required\_commodity(commodity\_name,commodity\_type,commodity\_description,commodity\_cover\_pic\_url,publish\_time\_stamp,area,user\_id) "** +  
 **"values(?,?,?,?,?,?,?)"**;  
Object[] objects = **new** Object[7];  
objects[0]=commodity\_name;  
objects[1]=commodity\_type;  
objects[2]=commodity\_description;  
objects[3]=commodity\_cover\_pic\_url;  
objects[4]=publish\_time\_stamp;  
objects[5]=area;  
objects[6]=user\_id;  
 **jdbcTemplate**.update(sql,objects); //执行SQL语句

1. 数据库规范

经检验，数据库符合BC范式，且在逻辑设计上有良好的设计

外码约束示例：



图19：commodity\_images的外码约束

当此商品完成交易时，描述图片的image\_url字段也自动删除。

## 八、系统功能测试

1. 登录，用户操作

我们首先注册一个账号，然后登录



图17：注册、登录

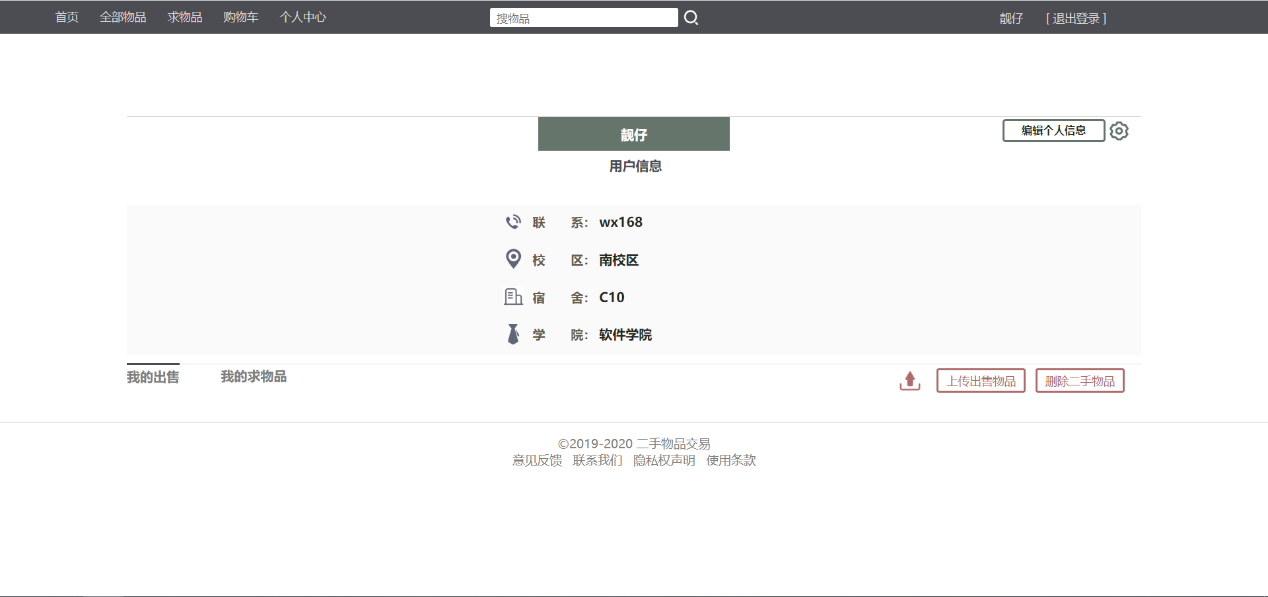
 

图18：个人信息页

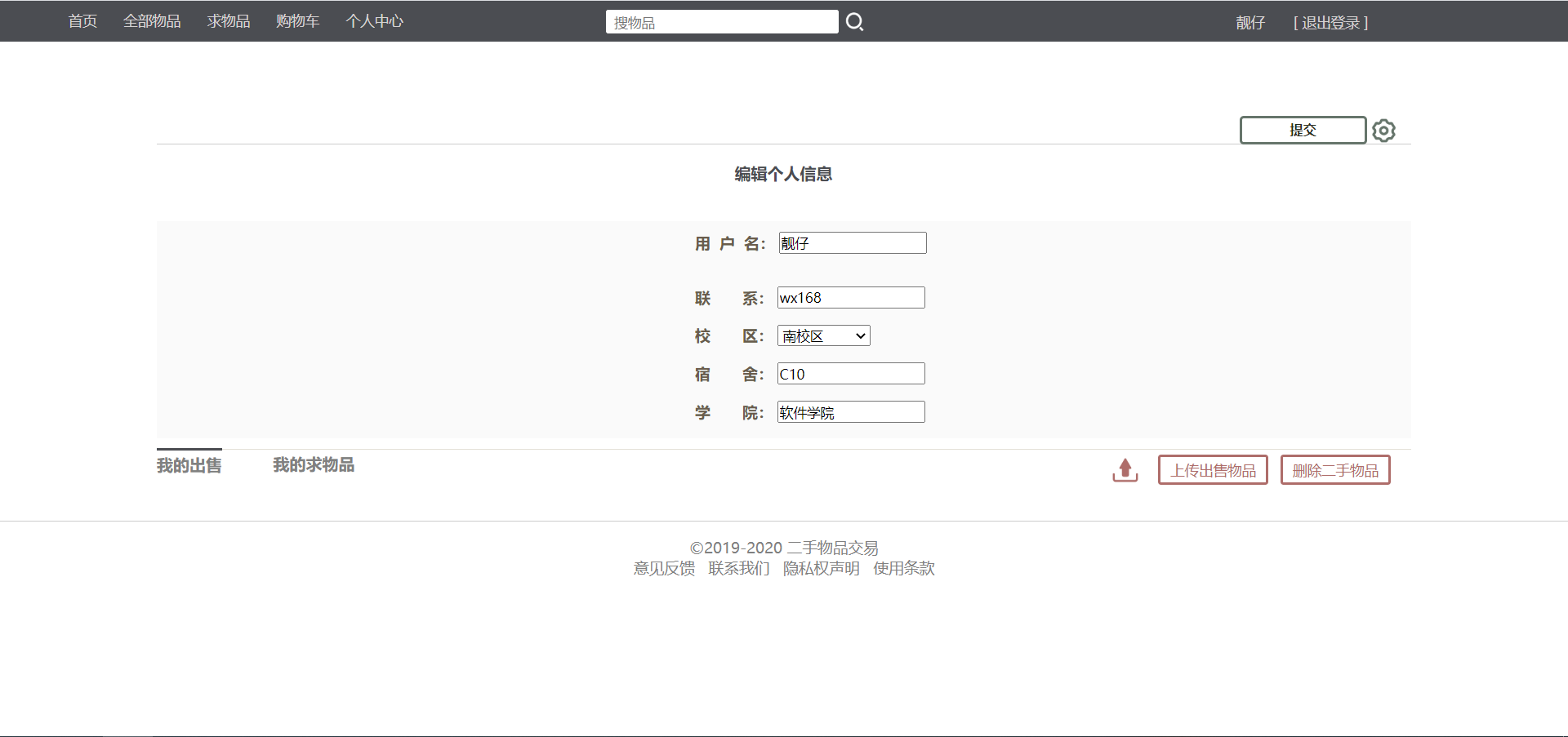
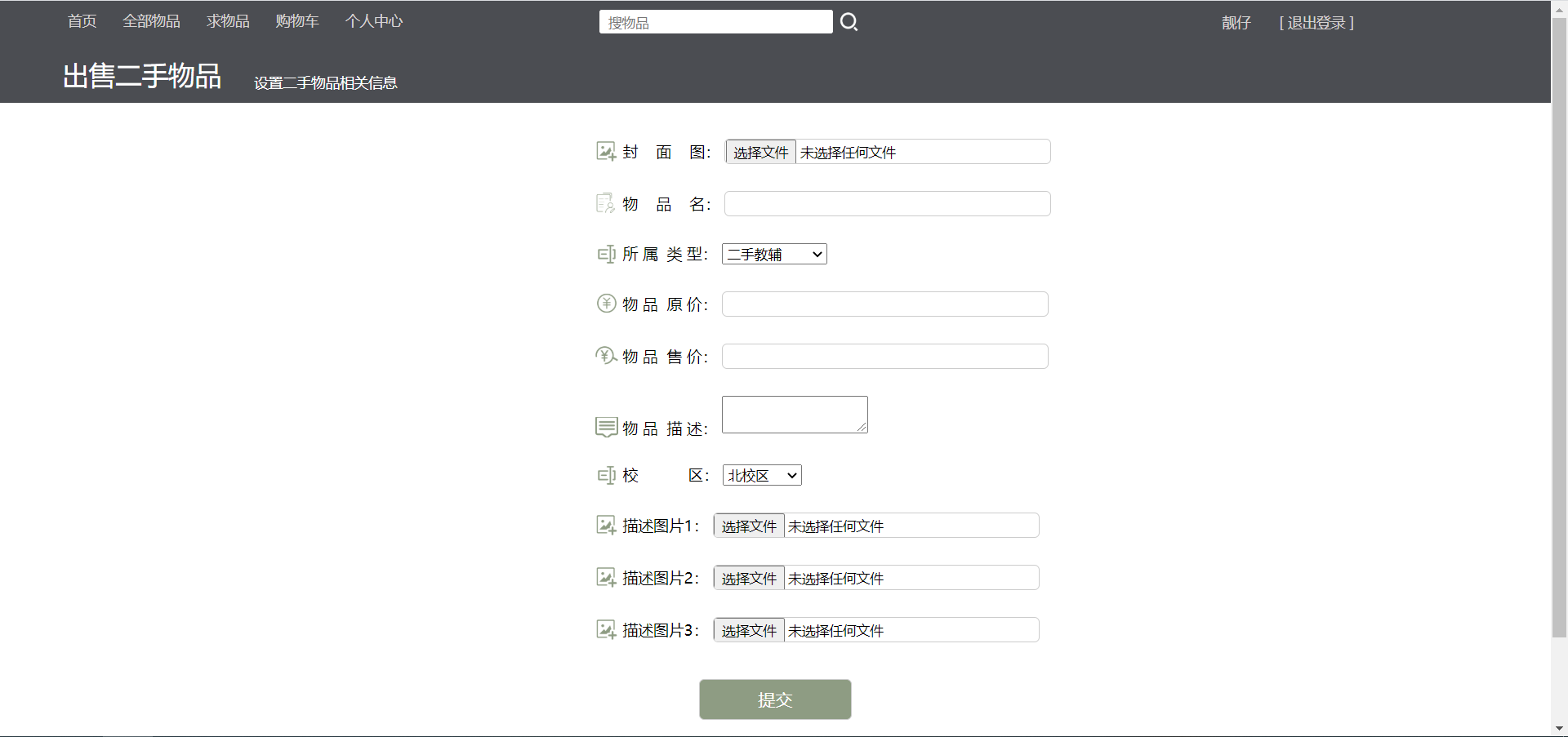


图19：修改个人信息

1. 商品的上架，分类展示



19：点击上传出售物品图

1. 交易流程

点击物品进入详情页



图20：物品详情

可加入购物车

可根据右侧卖家的联系方式联系，进行线下交易。

交易完成后，在个人中心，自己出售的物品下点击完成交易按钮，则该商品下架，不再显示。