# 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、二手商品上架与分类展示、交易过程中线上联系交流、求物区、购物车、个人信息展示与修改等功能。

项目已上线：项目地址：

**https://dbp.scutc10.club/index.html**

## 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。

还使用了少部分简单的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。

### 3.后端服务器部署。

使用springboot框架、Java语言开发，使用JDBC连接数据库。

### 4.数据库。

使用mySQL建立数据库。

## 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，全部二手商品页，求物区，购物车，个人中心页。

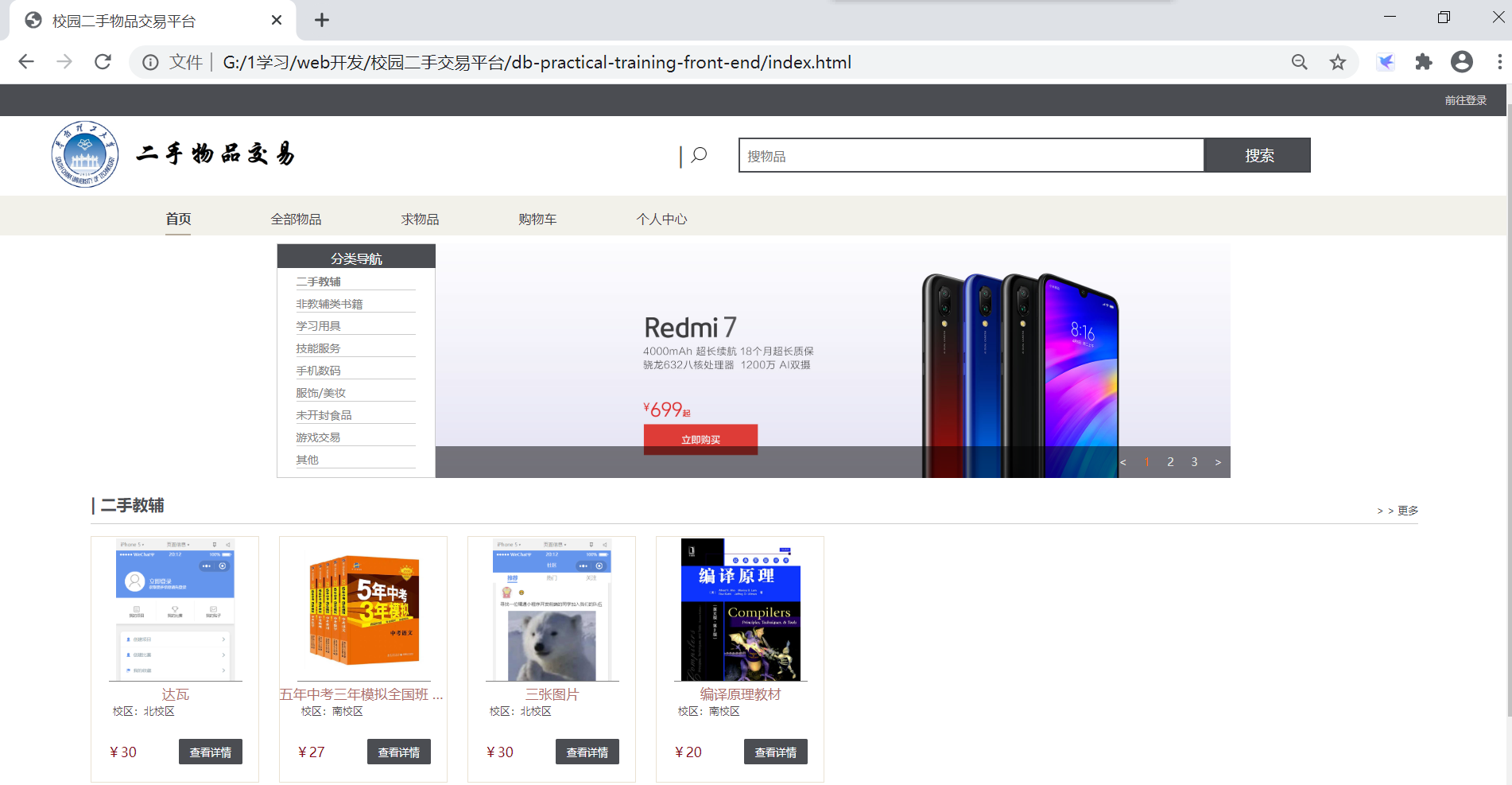


图1：首页

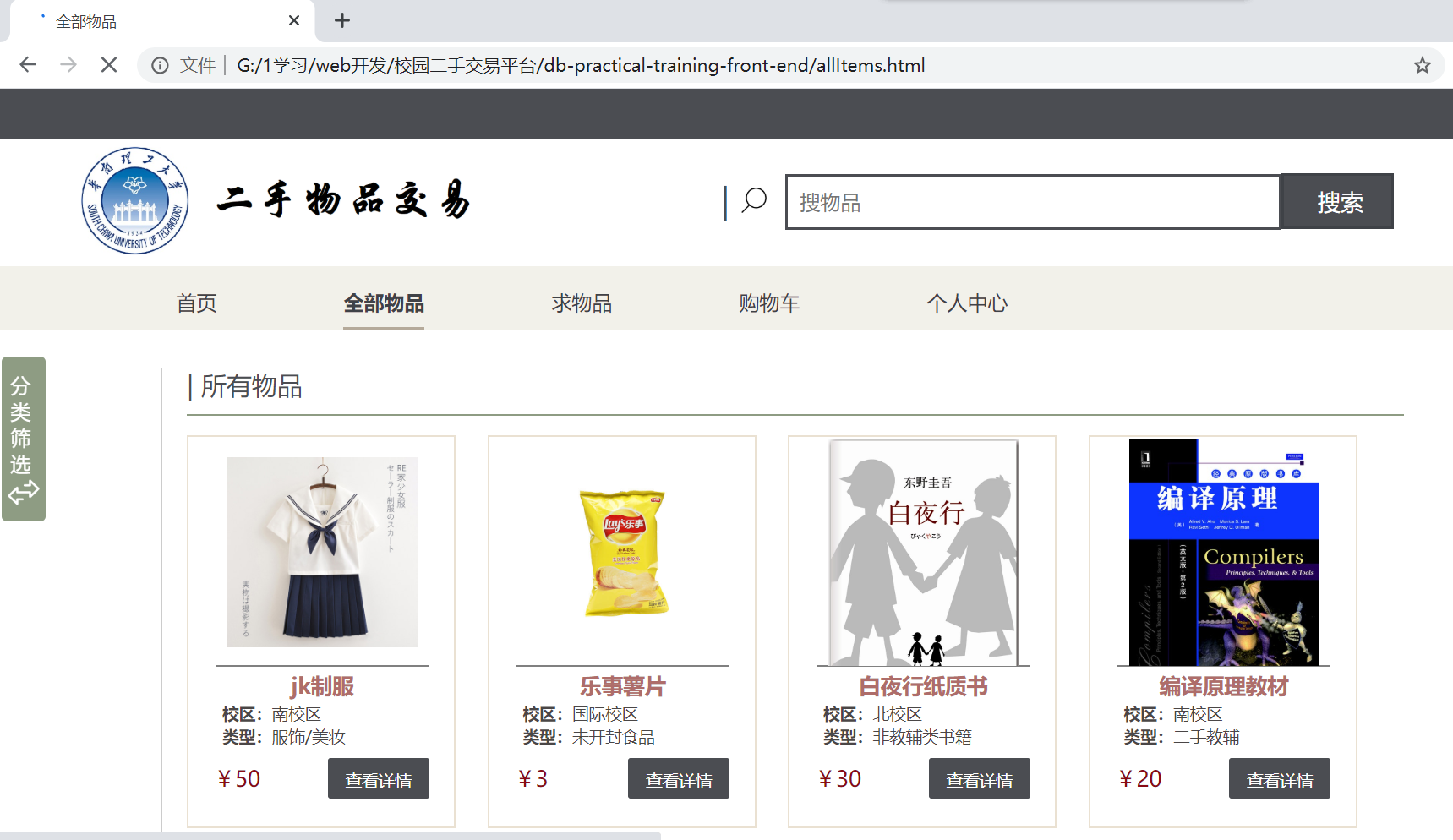


图2：全部二手商品页



图3：求物区

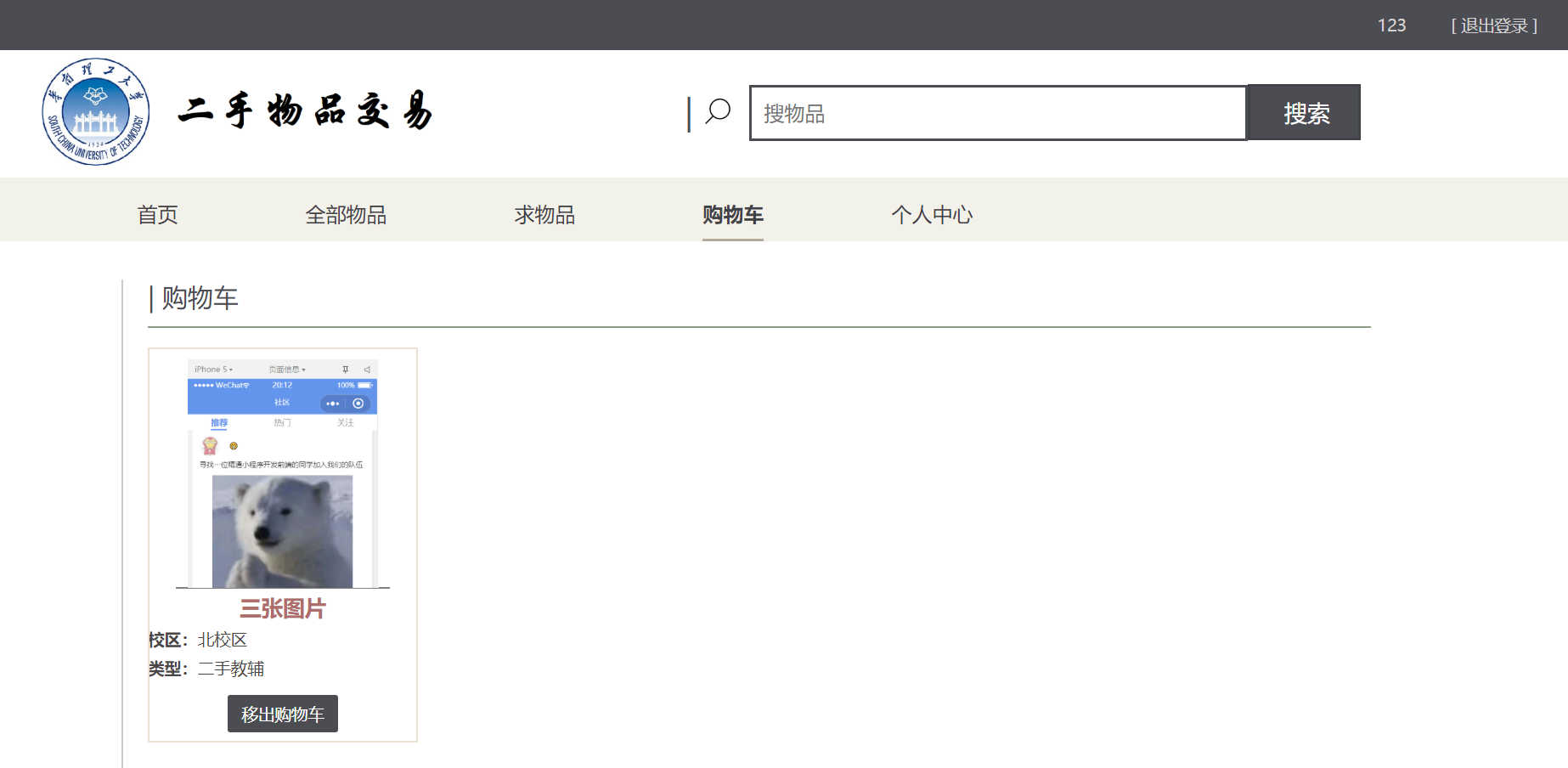


图4：购物车

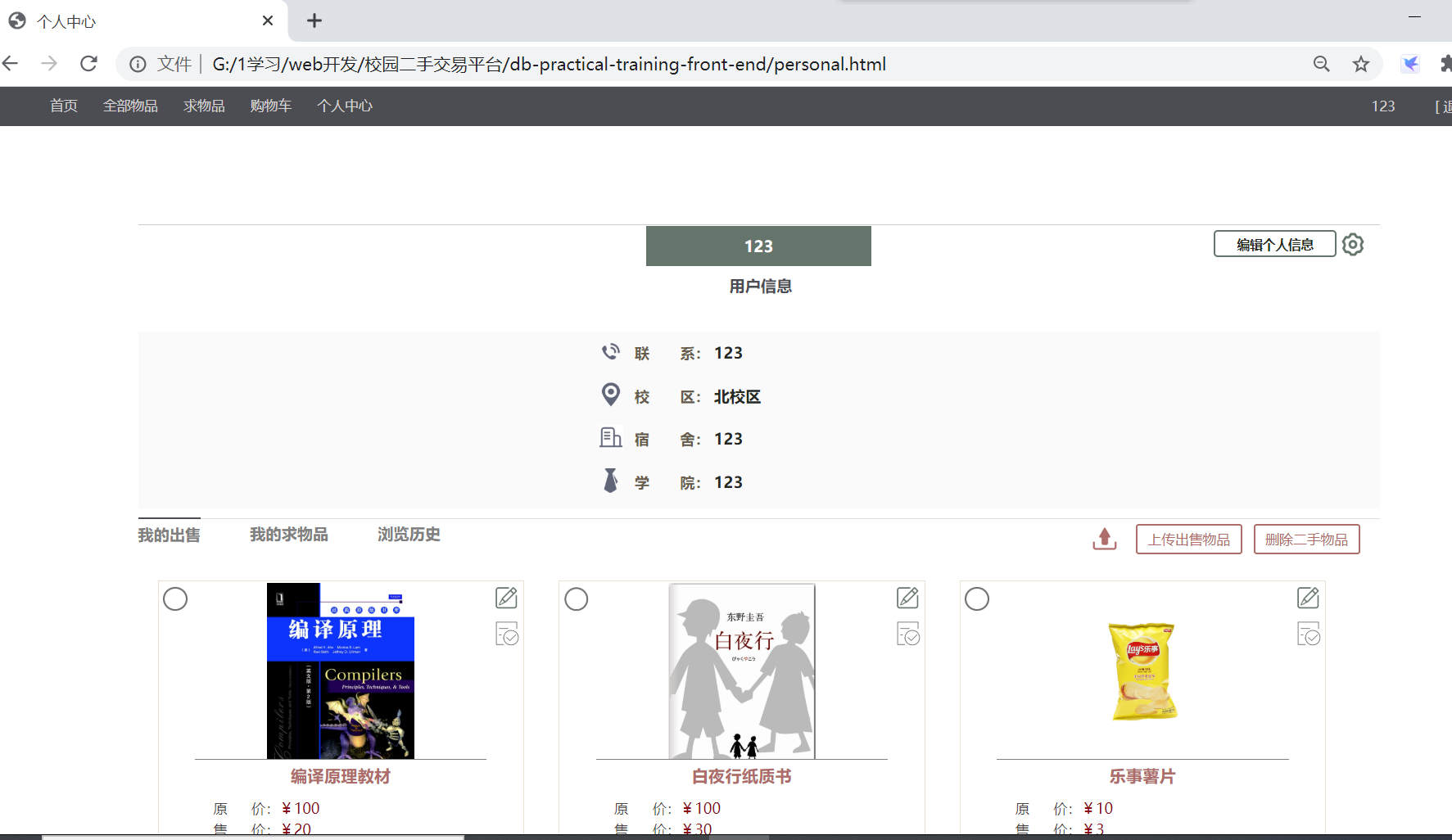


图4：个人中心

### 2.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：二手商品详情页（包括出售二手商品、求物），上传二手商品页，上传求物页。



图4：二手商品详情页

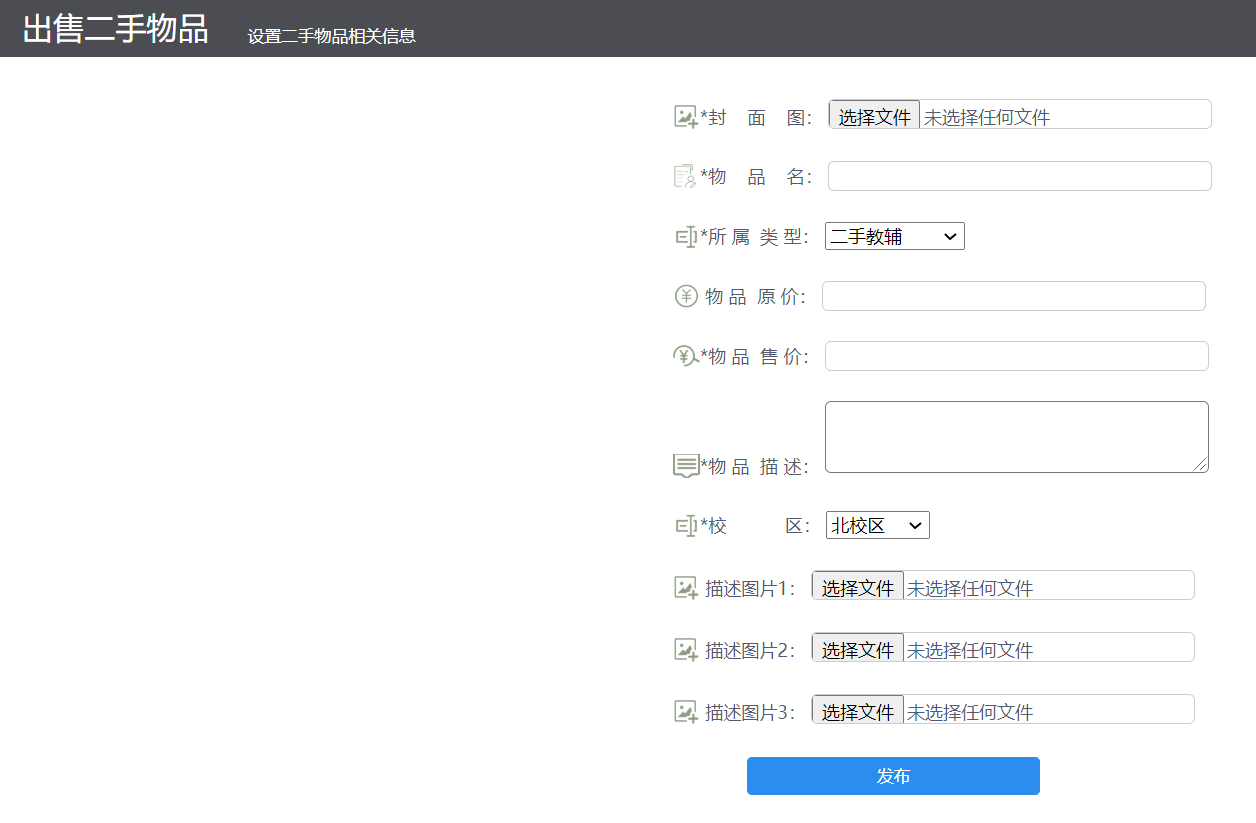


图5：二手商品上传页



图6：求物上传页

### 3.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，用于搜索二手商品。





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。

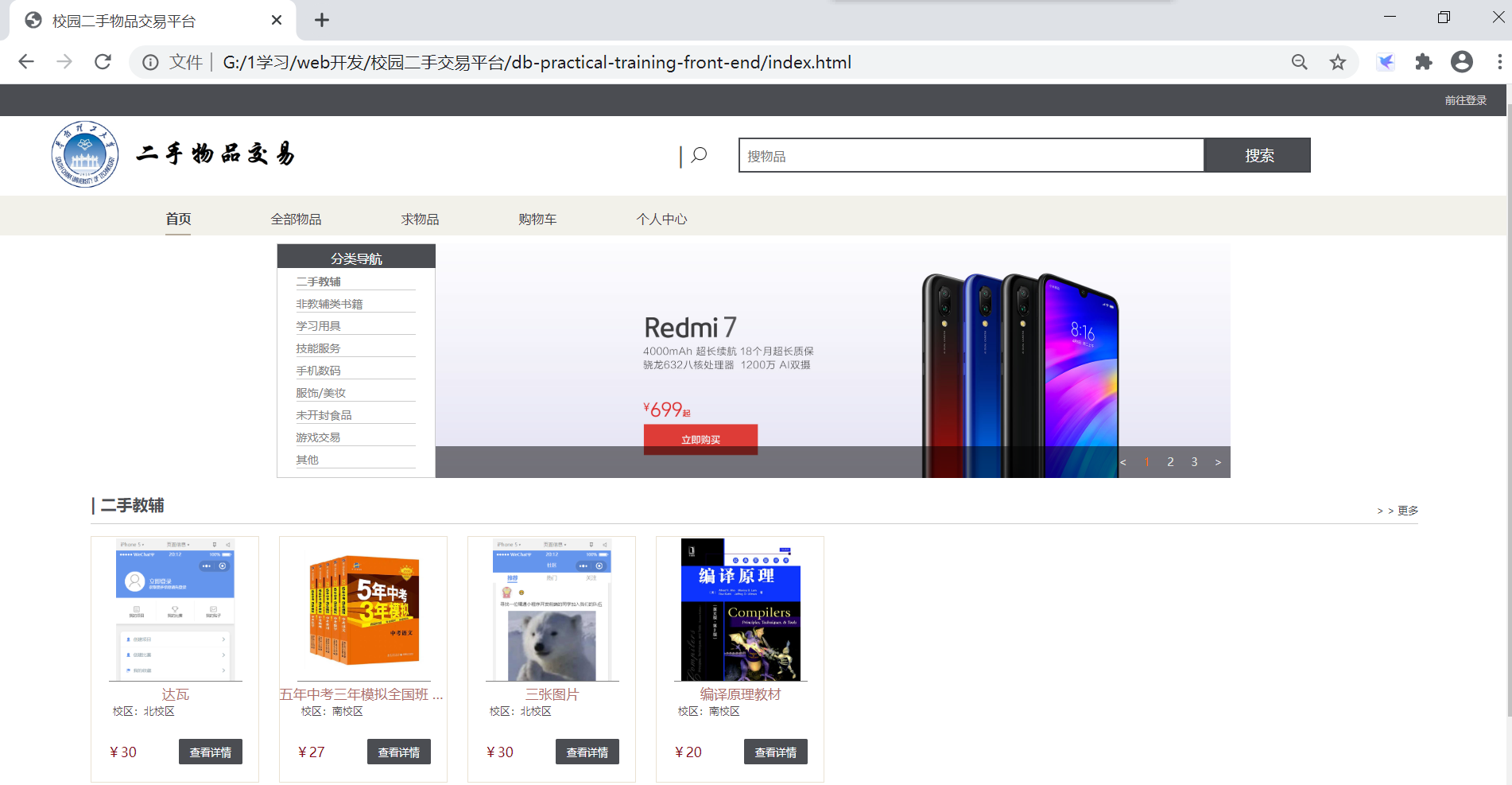


图10：首页

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。





图11：未登录、已登录状态

往下左边是网站标题“二手二手商品交易”，右边是搜索栏，通过搜索栏，用户可以对数据库里已经有的二手商品名称和描述进行查询，数据库返回符合关键词的二手商品，页面跳转到全部二手商品页显示。



图13：搜索框



图14：搜索功能

往下左侧是分类导航，每个二手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二手商品并显示给用户。

其右是轮播图，轮流随机展示几个二手商品供使用者查看，如果使用者看到心仪的二手商品，即可点击直接进入二手商品页面。

再往下则是为每个二手商品分类单独分出一栏，显示部分该分类的二手商品。点击右侧的“更多”按钮，跳转到全部二手商品页面，显示该分类的全部二手商品。

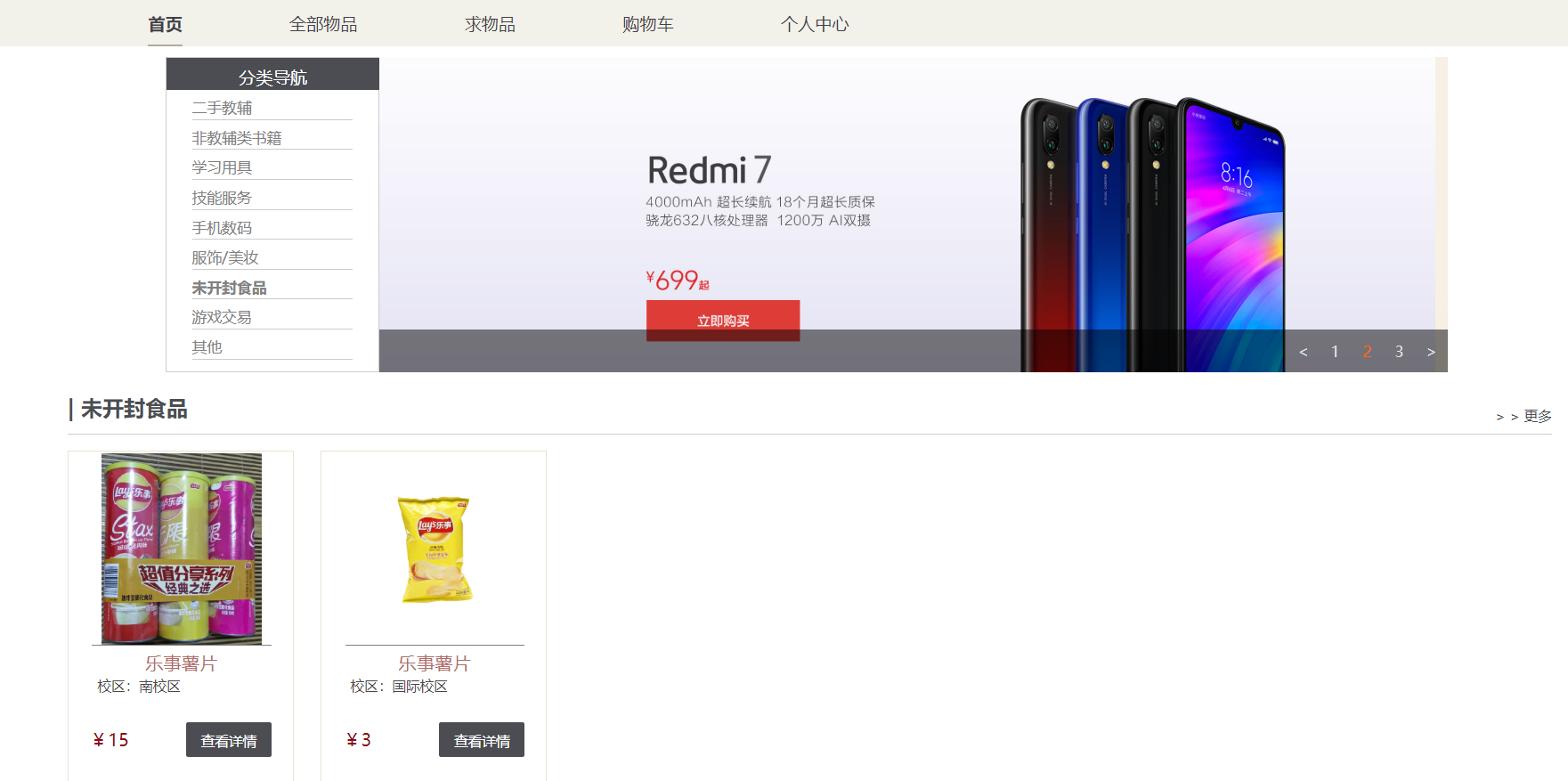


图15：首页的分类导航、轮播图

### 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，右上角有“返回首页”的按钮。

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3个按钮，分别是“没有账号？前往注册”、“忘记密码？”、“登录”。

“没有账号？前往注册”可以进入注册界面；

“忘记密码”按钮暂时没有功能；

“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。

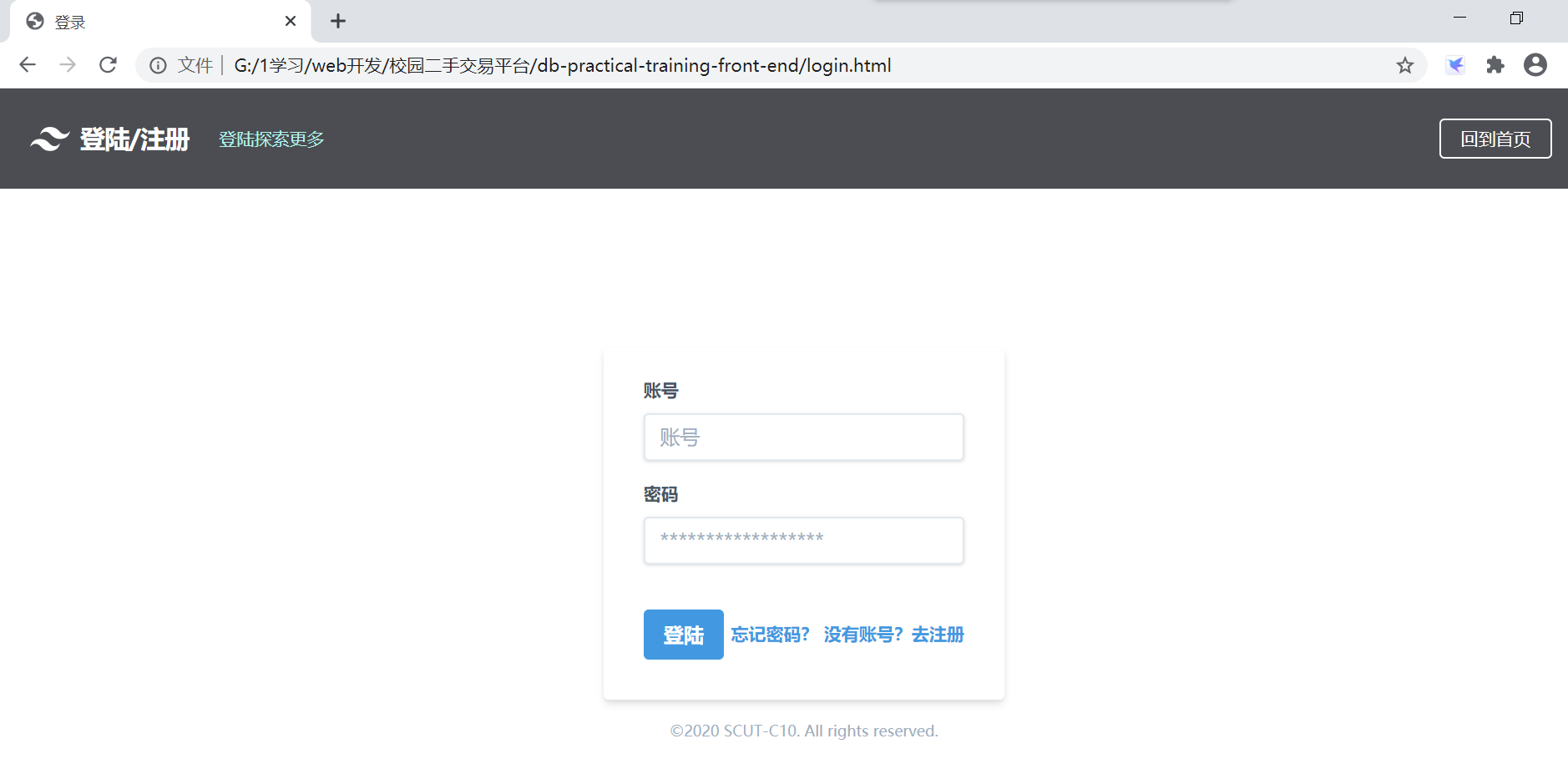


图16：登录页面



图17：注册页面

### 3.搜索界面。

在各个页面的搜索框输入二手商品名称，点击搜索按钮，网页向数据库查询二手商品名称，数据库返回目标二手商品的数据。然后跳转到全部二手商品页面上显示搜索到的具体二手商品图片、价格等简略信息。图片和二手商品名称是超链接，点击可以打开二手商品详情页面。



图14：搜索功能

### 4.二手商品详情页面。

从首页、全部二手商品页、求物页、购物车、个人中心的个人创建二手商品处，点击二手商品卡片，都能跳转到二手商品详情页。

二手商品详情页面也有导航栏：“首页”、“全部二手商品”、“求物”、“购物车”、“个人中心”和搜索框。

下方为二手商品具体描述：左为二手商品封面图；中为二手商品名称、二手商品所属分类、和文字二手商品描述，再往下是用户选填的3张二手商品描述图；右为二手商品发布人的信息，包括联系方式、校区、宿舍、学院等。

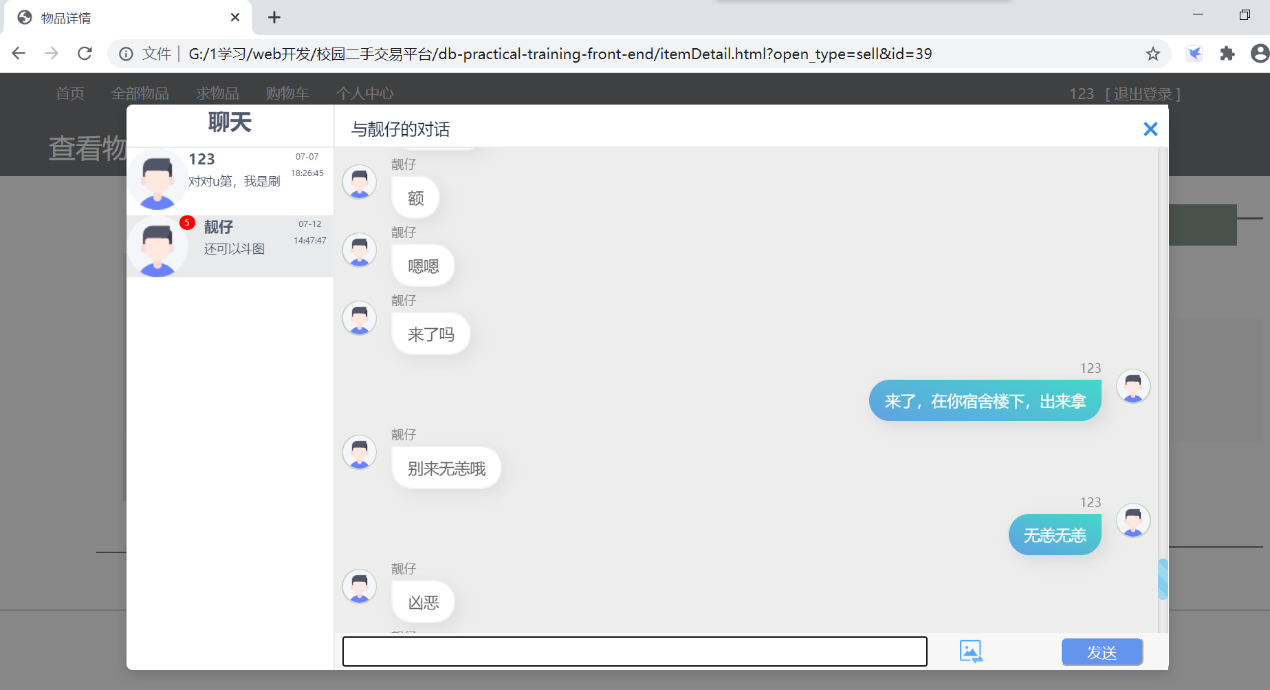


图15：二手商品详情页

加入购物车按钮，点击可将二手商品加入购物车。

有一个“立即联系”按钮。如果用户有意愿购买，可立即线上进行联系，也可根据右侧联系方式进行线下联系交易。

点击“立即联系”按钮，会弹出聊天框，用户可在线交流，聊天框功能强大，可发送文字、图片、聊天表情，聊天信息实时更新。



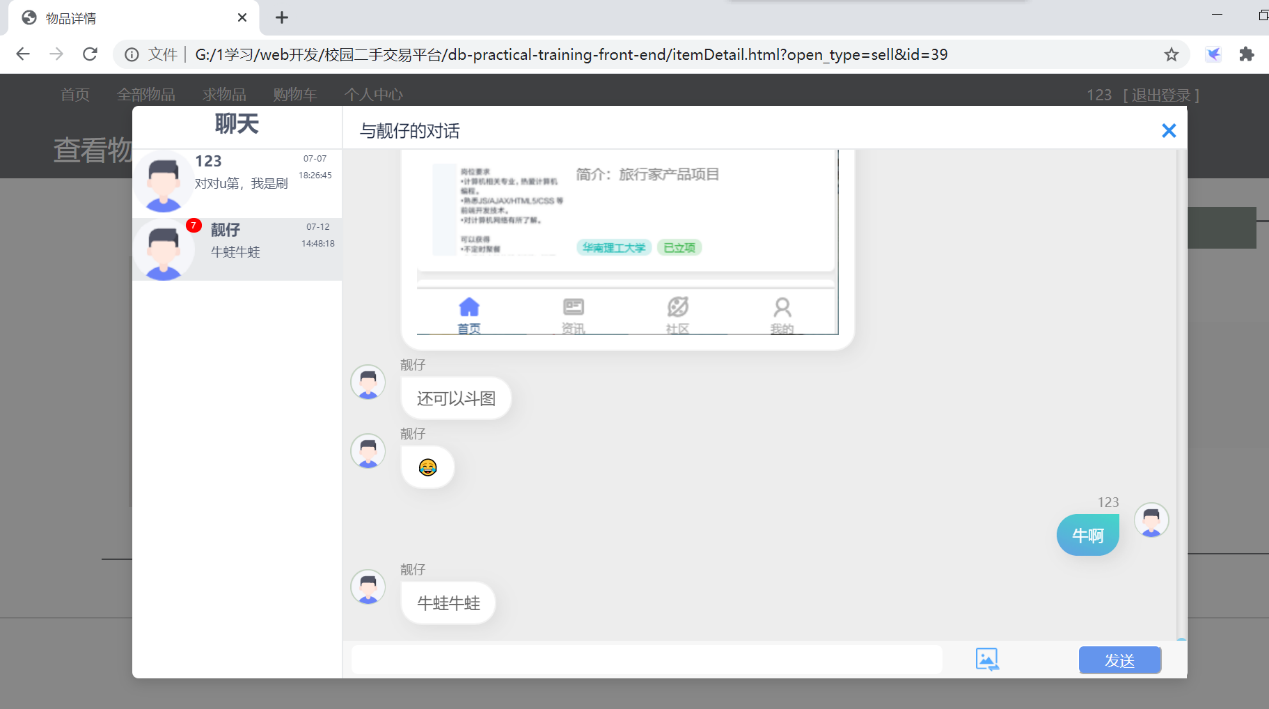




图16、17、18：在线交流功能

### 5.全部二手商品页面。

在各个页面导航栏点击“所有二手商品”按钮可以打开所有二手商品页面。

该页面直接展示所有还未出售的已发布二手商品。左侧是分类筛选菜单。用户可以点击图片或二手商品名称进入二手商品详情页面。

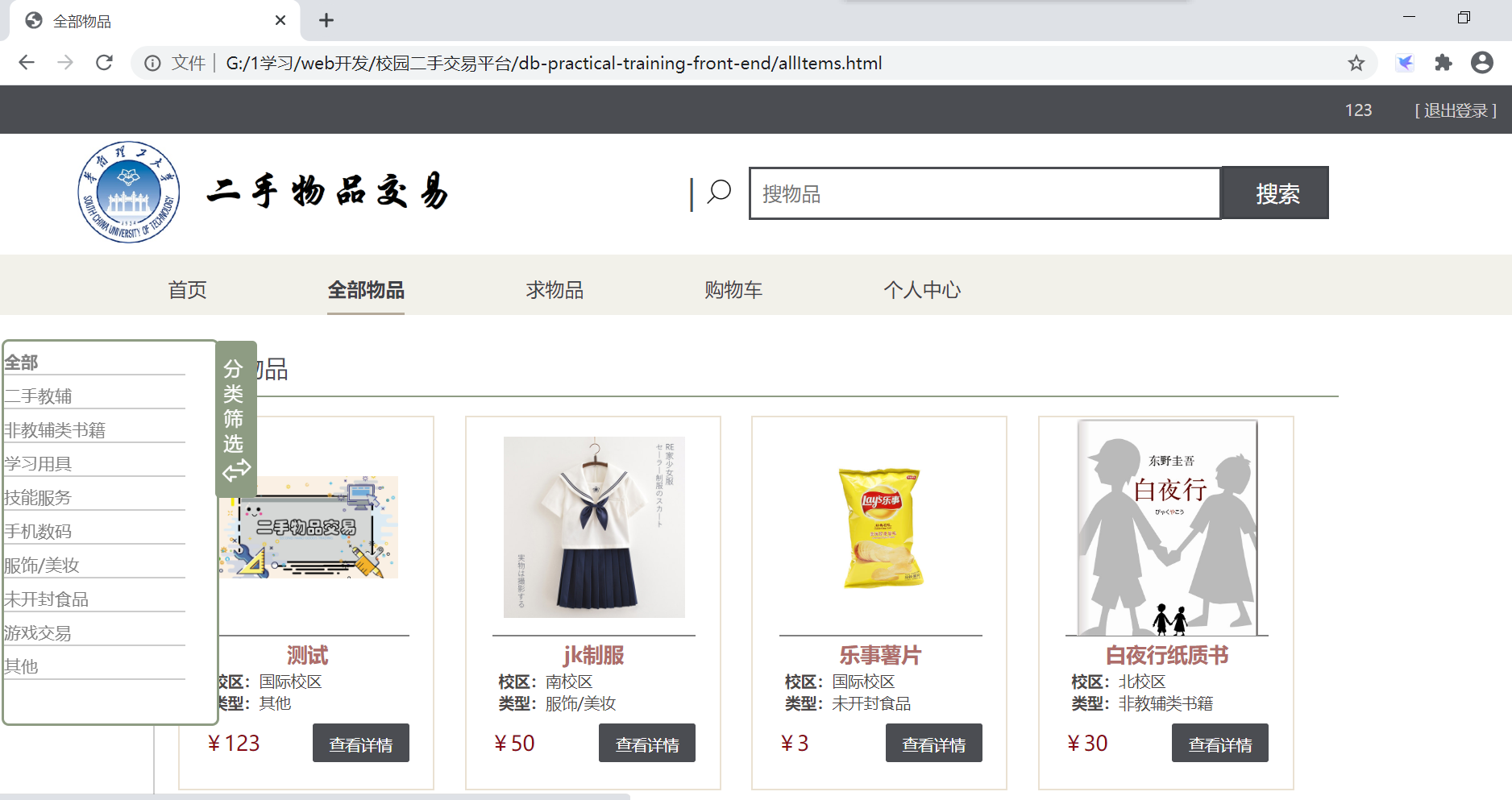


图19：全部二手商品页

### 6.求物页面。

在各个页面导航栏点击“求物”按钮可以打开求物页面。页面将显示需要某种二手商品的同学所发布的请求，可以点击进入具体页面。该页面没有分类功能。

请求的二手商品也可进入详情页。



图20：求物页



图21：求物详情页

### 7.购物车。

在各个页面导航栏点击“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二手商品图片和名称。如果不想买了也可以通过点击“移出购物车”按钮将二手商品移出购物车。

购物车功能需登录才可以使用，否则会提示未登录。同样，可以通过点击二手商品图片或名称进入具体二手商品页面

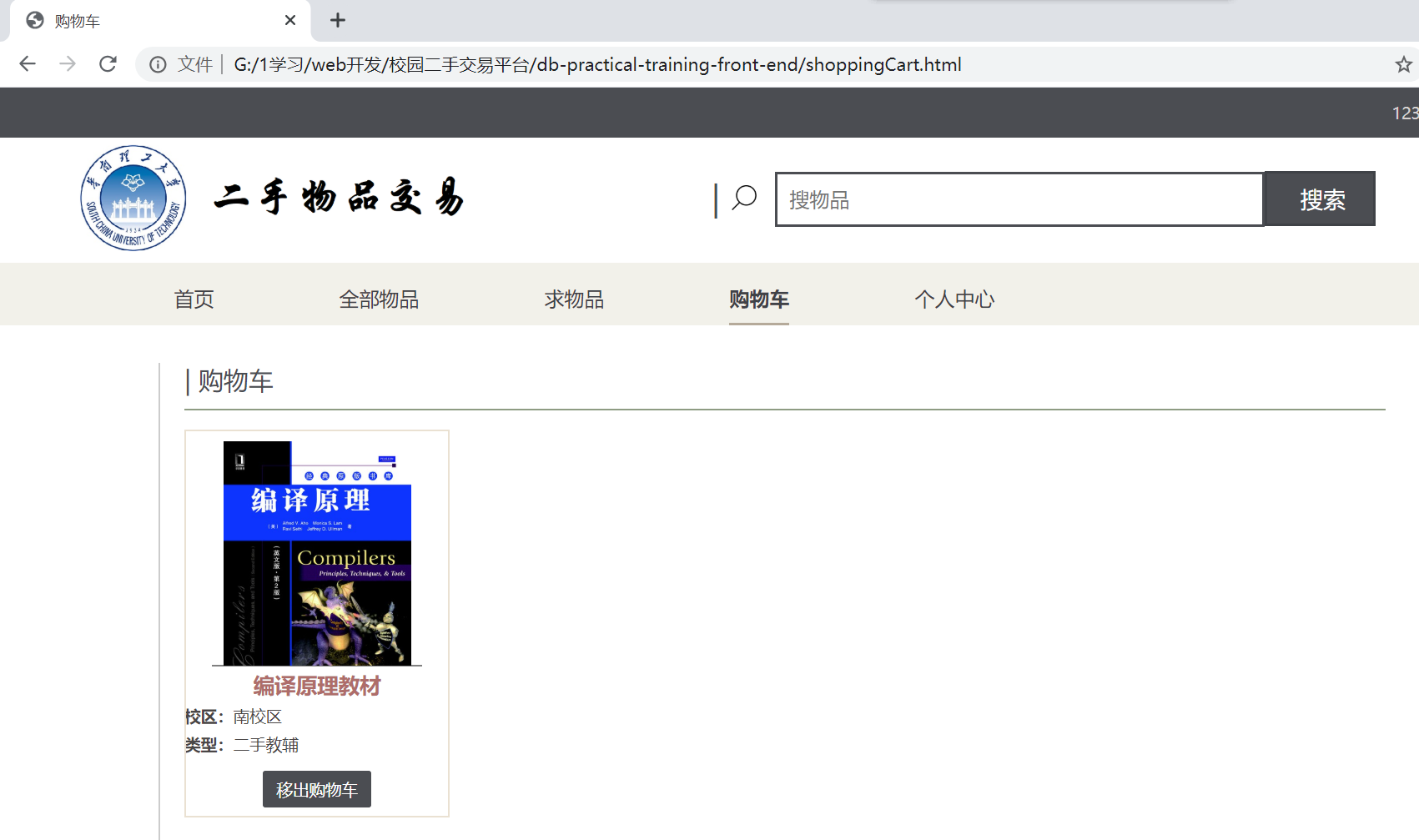


图22：购物车页

### 8.个人页面。

点击“个人页面”按钮可以进入个人页面。

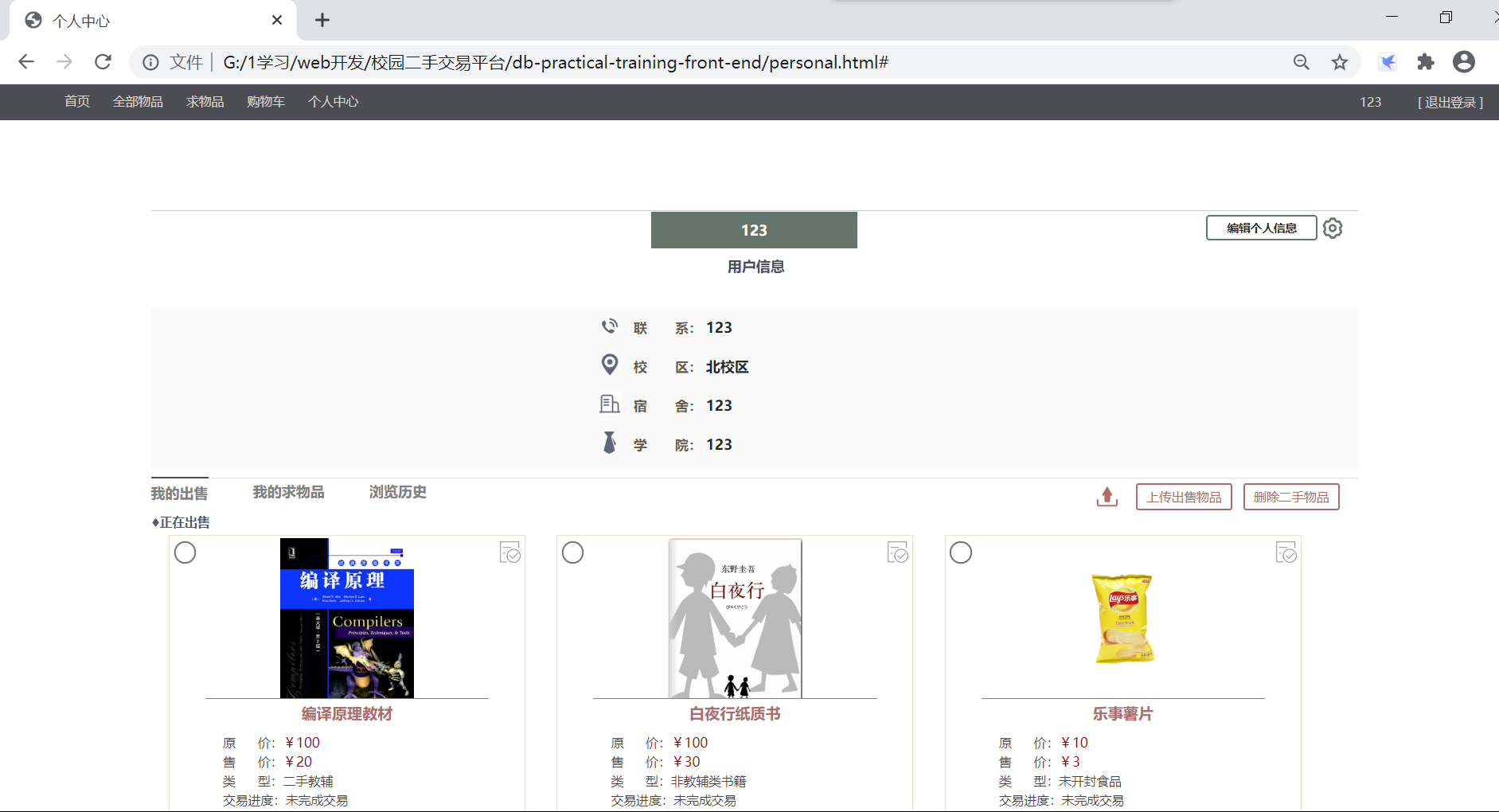


图23：个人中心页

个人页面显示个人信息，可以点击编辑，修改自己的个人信息，包括用户名、联系方式、校区、宿舍、学院。

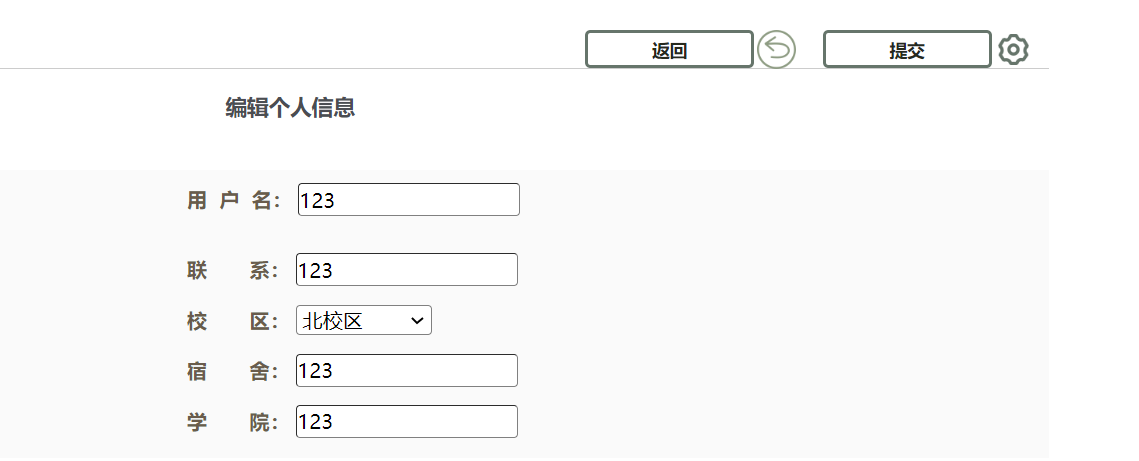


图24：修改个人信息

同时可以查看自己当前正在出售的二手商品、求物信息、浏览历史。通过点击下方的“我的出售”、“我的求物”、“浏览历史”即可。

在“我的出售”标签里，右方还有“上传二手商品”和“删除二手二手商品”两个按钮，上传二手商品可以上传其他二手商品，点击后进入二手商品上传页面；点击二手商品卡片左侧，可选择二手商品，点击删除按钮，可是删除二手商品，删除二手二手商品后二手商品将不再展示，创建者和其他用户都看不见，从数据库中彻底删除。

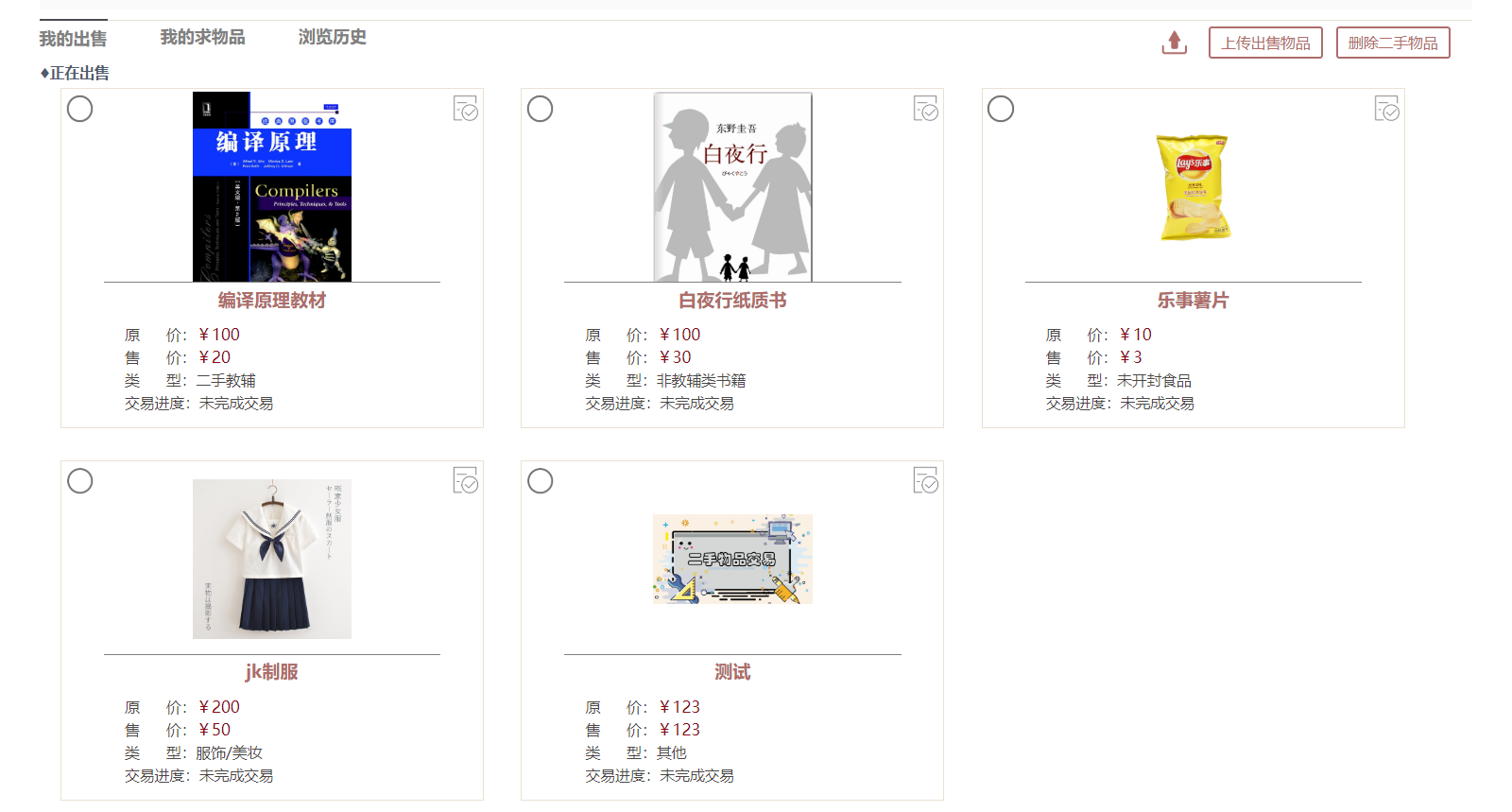


图25：我的二手商品



图26：选中二手商品

二手商品卡片右侧有完成交易按钮，点击后，二手商品进入完成交易列表，此后仅对创建者可见，不再在二手商品展示页面显示。

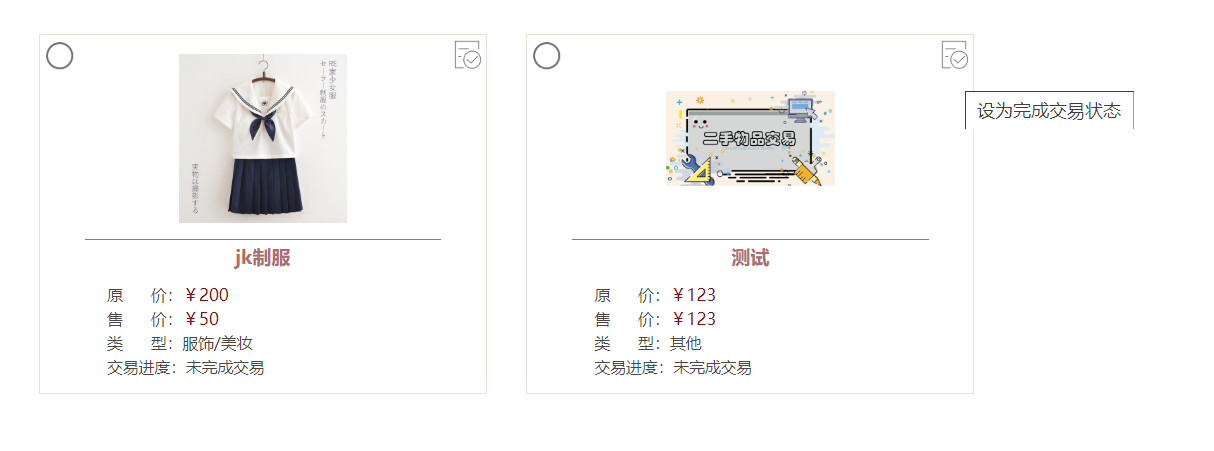


图27：二手商品完成交易按钮

往下，是已完成交易的二手商品，这些二手商品仅对创建者可见，不再在二手商品展示页面显示。

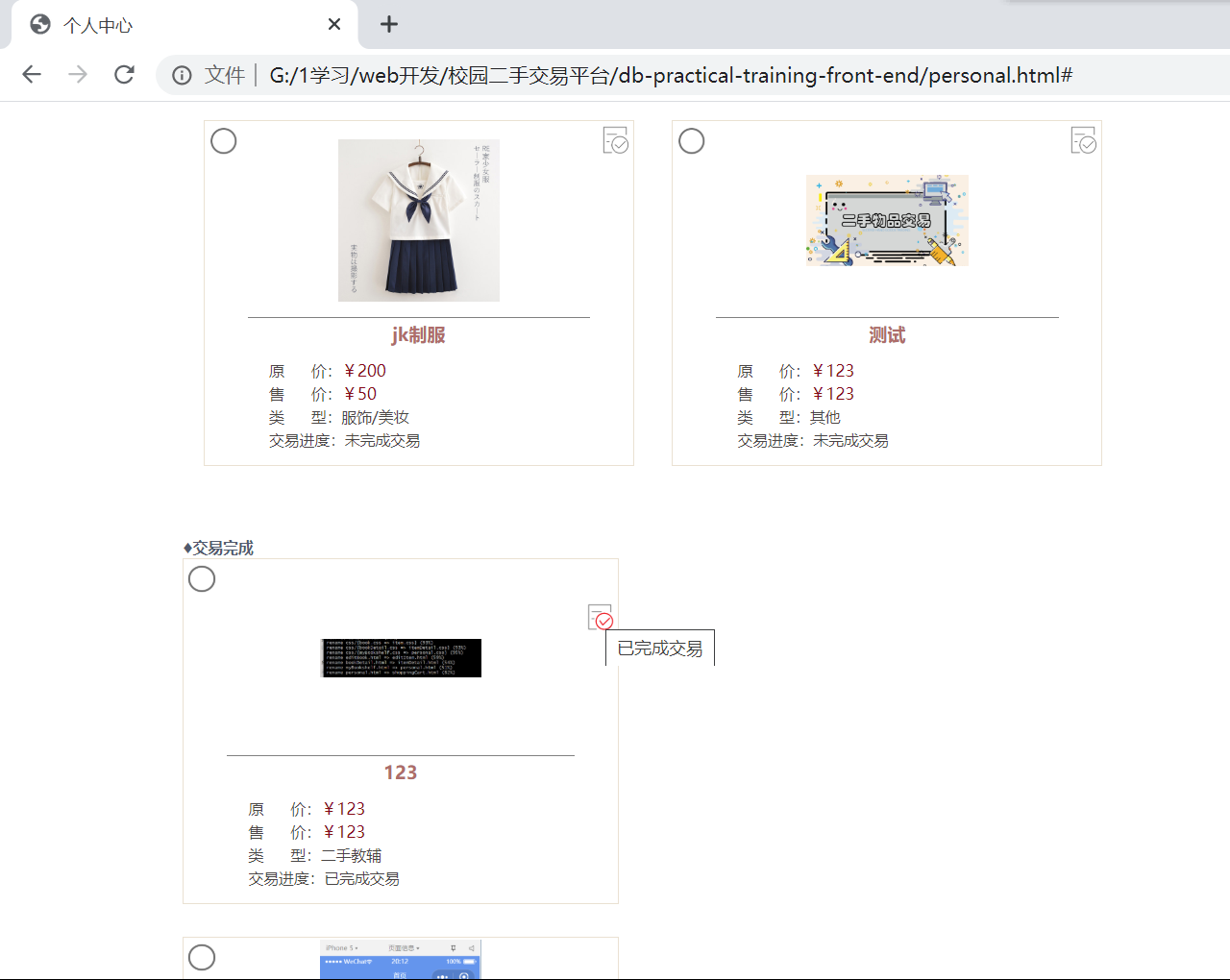


图28：完成交易二手商品

在“我的求物”标签里，右方有“上传求物信息”和“删除求物”按钮，点击“上传求物信息”可以进入求物上传页面，点击“删除求物按钮可以取消求物。



图29：我的求物

在“浏览历史”标签里，用户可看到自己浏览过的二手商品，该列表的二手商品也可删除。

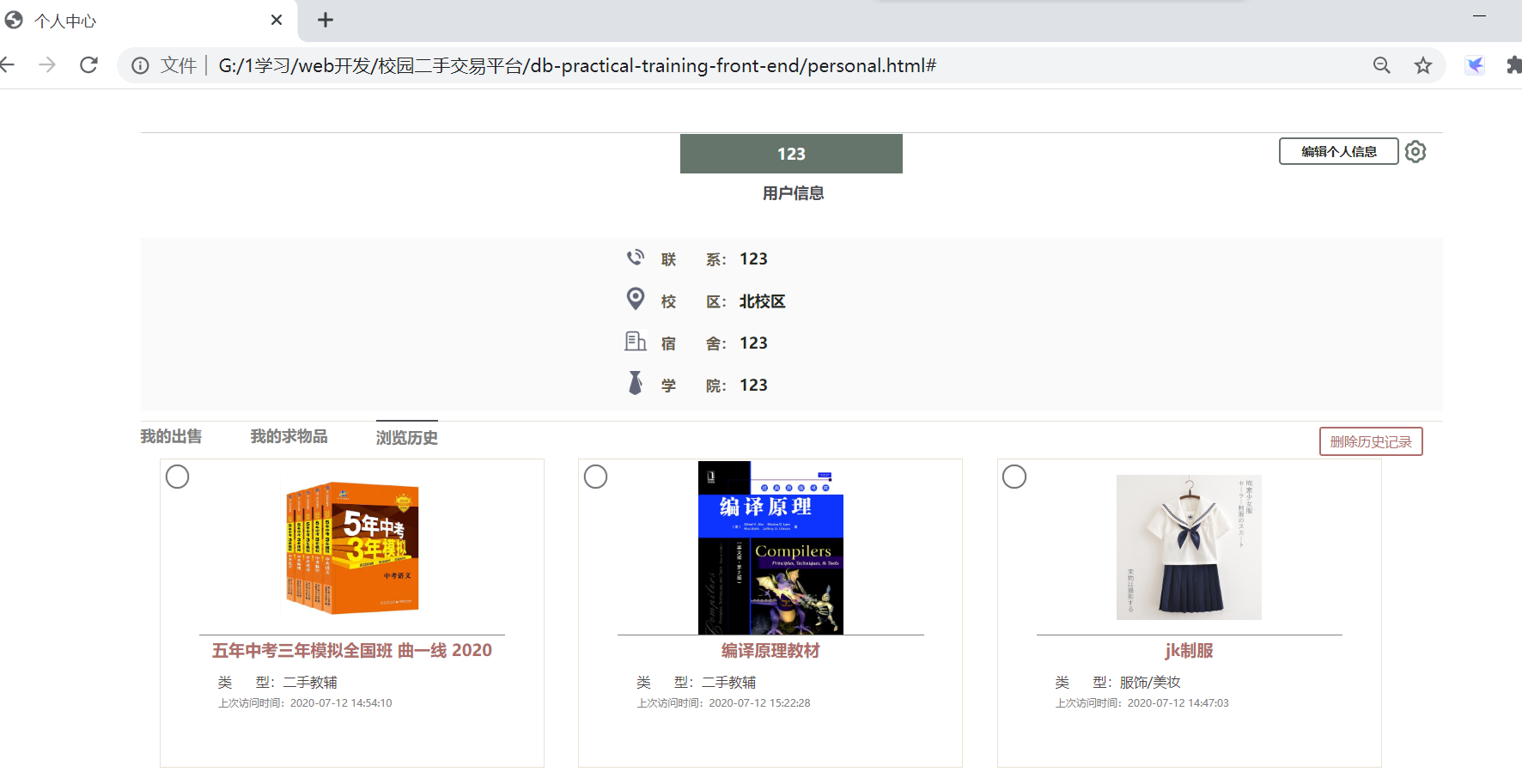


图30：我的求物

### 9.二手商品上传页面。

该页面有多个输入框，依提示分别为封面图、二手商品名、所属类型、二手商品原价、二手商品售价、二手商品描述、描述图片1、描述图片2、描述图片3。

用户上传封面图片，并输入所有必填的信息（打\*号）后，便可点击“提交”按钮上传。上传后的二手商品可以被其他人浏览到，同时可以在个人页面里查看。

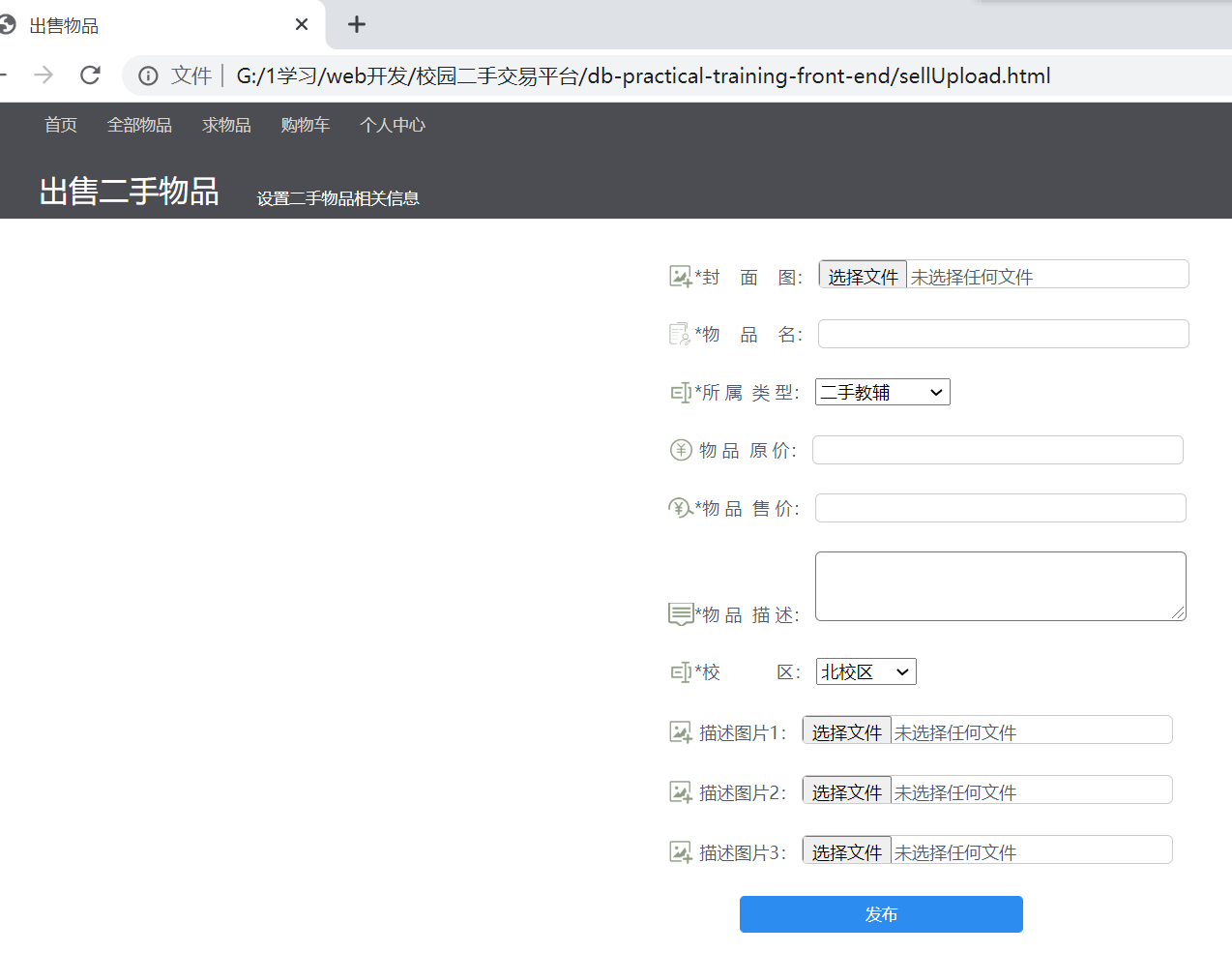


图31：二手商品上传页

### 10.求物上传页面。

该页面类似二手商品上传页面，输入信息后可以提交上传。上传后的求物可以被其他人浏览到，同时可以在个人页面查看。



图32：求物上传页

## 五、后端服务器搭建

### 1.框架：

基于Java的Springboot框架(MVC三层架构)：

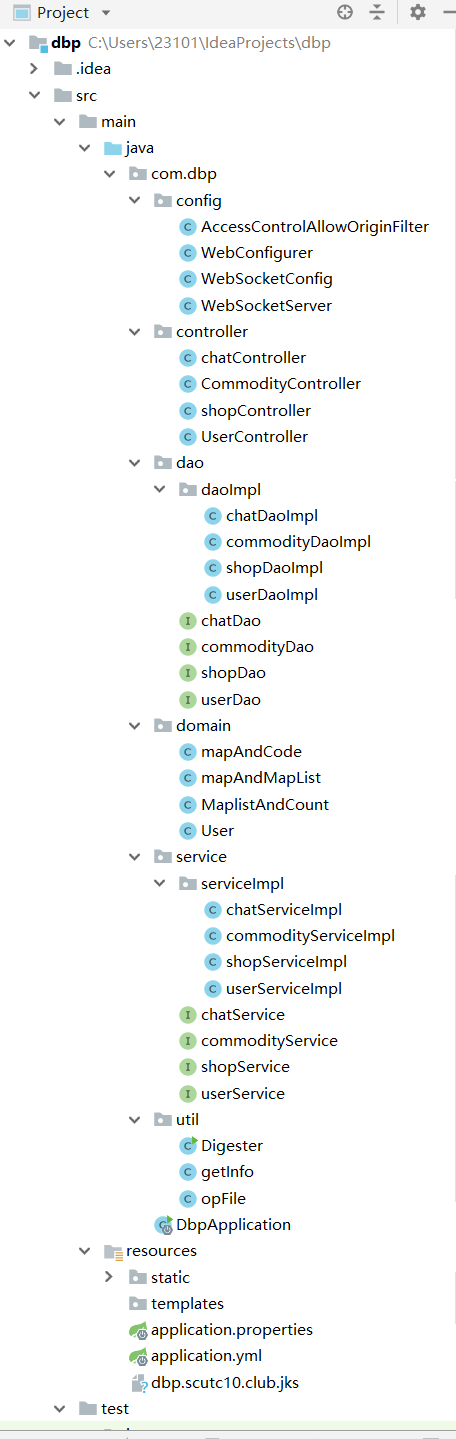


图16：后端工程目录结构

### 2.运行环境

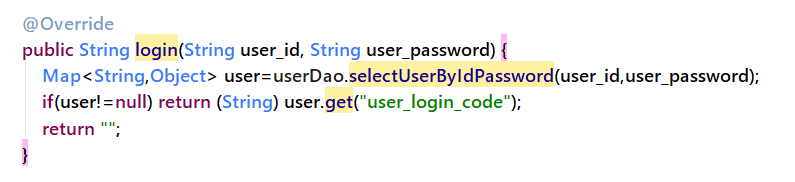
Centos7，Mysql,Tomcat

### 3.运行方式

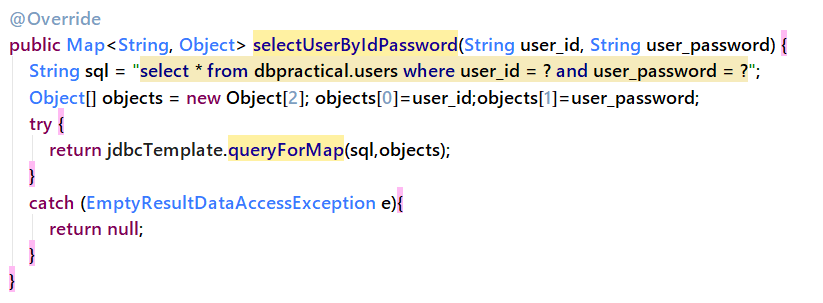
在Controller层接收请求，通过service层定义的服务来调用Dao层对应的数据库访问方法。接口示例：



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。

## 六、后端数据库设计

### 1.页面分析，得出数据

根据“四、页面交互功能”中的详细页面分析，我们得到所有所需记录信息如下：

**二手商品信息：（\*号必填）**

二手商品名\*

所属类型\*

原价

售价\* （求物没有价格）

描述\*

封面图片

描述图片1

描述图片2

描述图片3

二手商品状态：是否已完成交易

**求物信息：（\*号必填）**

二手商品名\*

所属类型\*

描述\*

封面图片

*注：求物没有价格、描述图片、完成交易状态。*

**用户信息：**

账户，密码，每次操作用的code令牌，

昵称，联系方式，校区，宿舍，学院

**用户连接的信息：（联系集）**

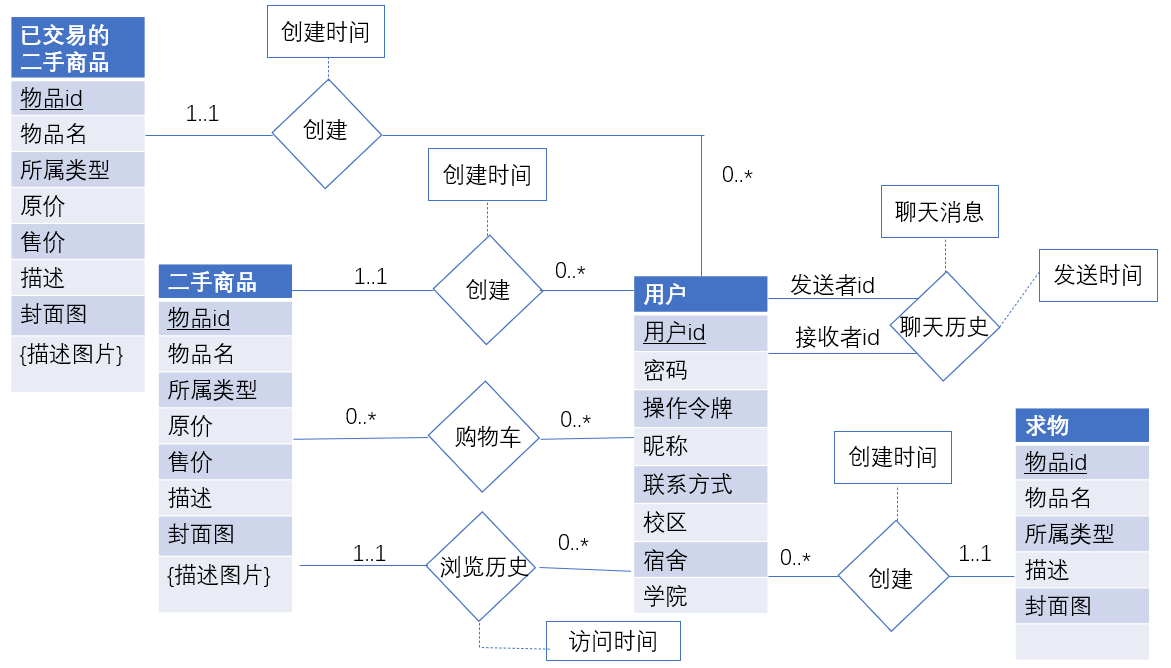
加入购物车的二手商品id，购物车只有二手商品，没有求物（多对多）

自己创建的二手商品id（一对多）

自己创建的求物id（一对多）

用户浏览历史：二手商品的id（一对多）

### 2.E-R图及分析



图：数据库E-R图

**实体集：**

用户，二手商品、已交易的二手商品、求物、浏览历史、聊天记录。

**联系集：**

用户创建二手商品、用户拥有已交易的商品、用户创建求物，用户浏览历史，用户将二手商品加入购物车，用户与其他用户聊天。

**E-R图设计分析：**

1. 二手商品、已交易的二手商品**实体集：**当二手二手商品完成交易后，用户将正在出售的二手商品设置为已完成交易状态，服务器自动将该商品从二手商品表移动到已交易二手商品表。
2. 二手商品、已交易的二手商品**实体集**：两表中的描述图片字段，是多值属性，转换为关系模式时，需要另建立一个表，存放二手商品id，图片地址。
3. 创建**联系集**：用户可以出售多个二手商品，可以拥有多个已交易的商品记录，也可以有多个求物需求，都是“一对多”，用户是“一”方，二手商品是“多”方，转换为关系模式时，将“一”方，即用户的主码和特有属性创建时间放进二手商品表中，从而表示此联系集。
4. 购物车**联系集**：用户和二手商品的加入购物车关系是“多对多”的关系，需要建立新的表，表中存有用户表的主码和二手商品表的主码。
5. 浏览历史**联系集**：保存用户浏览过的二手商品的ID及特有属性访问时间
6. 聊天历史**联系集：**保存用户与其他用户的聊天记录，在E-R图中表现为自我联系，是多对多的关系，每一条历史记录有发送者id，接收者id，记录时间戳，聊天内容。转换为关系模式时，需要另建立一个表，字段有聊天双方id、时间戳、聊天内容。

### 3.数据库DDL

具体的数据定义语句如下：

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;

注解：创建数据库，把原有的表删除，重新创建新的表。

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'))

);

注解：用户信息表。

信息有：账户id，用户昵称，账号密码，用户操作令牌，联系方式，校区，宿舍，学院。

其中，账户id是主码，校区限定3个：'北校区','南校区','国际校区'。

create view user\_info as(

select user\_id,user\_name,tel,user\_area,dormitory,academy from users

);

注解：创建视图。

用户对外展示的信息

（我们采用的数据库是mySQL，支持自然连接语句）

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

);

注解：二手商品表。

1.用户上传的待出售二手商品信息存于此表。

2.信息有：二手商品id，二手商品名，二手商品类型，二手商品原价，二手商品出价，二手商品描述，二手商品封面图，创建时间，校区，出售者用户id。

3.其中，主码是二手商品id，每次创建新二手商品，id自增；用户id是参照用户表上的外码，采用级联删除；

4.校区限定在：'北校区','南校区','国际校区'；

5.Fulltext：全文索引，使建立了此索引的字段支持快速的词语搜索。

具体使用方法为：

select \* from commodity

where MATCH(`commodity\_name`,`commodity\_type`,`commodity\_description`) AGAINST(‘java’ IN BOOLEAN MODE)

即可快速搜索出这三个字段中带有java关键字的二手商品（相比较于使用like来说）

6.由于用户创建二手商品信息，是一对多的联系集，对于联系集的表示，我们将“一”方的主码：用户id，放到“多”方的二手商品表中，从而表示出联系集。

create table commodity\_images(

commodity\_id int unsigned,

image\_url varchar(2083),

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

);

注解：二手商品描述图片表。

由于二手商品的描述图片属性是个多值属性，在转换为关系模式时，需要用另外一个表。

包含：二手商品id，多个图片的地址；

其中二手商品id是参照二手商品表的外码，级联删除。

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

);

注解：已完成交易的二手商品表。

交易流程设计中，若用户已完成线下交易，可将该二手商品设为已完成交易状态。

数据库实现采用分表的方式，将已完成交易的二手商品移至此表。

已完成交易的二手商品只对上传二手商品的用户可见，不再显示给其他用户。

表的设计与二手商品表相同。

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

);

注解：已完成交易二手商品的描述图片表。

由于二手商品的描述图片属性是个多值属性，在转换为关系模式时，需要用另外一个表。

包含：已完成二手商品id，多个图片的地址；

其中已完成二手商品id是参照已完成二手商品表的外码，级联删除。

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'))

);

注解：求物表。

我们设计了求物区。用户上传的求物信息存于此表。

信息有：求物id，每次创建新求物信息，id自增；求物信息名，求物信息类型，求物信息描述，求物信息封面图，发布时间，校区，创建用户id。

其中，主码是求物信息id，用户id是参照用户表的外码，校区限定在'北校区','南校区','国际校区'。

由于用户创建求物信息，是一对多的联系集，对于联系集的表示，我们将“一”方的主码：用户id，放到“多”方的求物表中，从而表示出联系集。

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

);

注解：购物车。

购物车是作为联系集存在的，用户可将二手商品加入到购物车中。

联系集是多对多，所以转换为关系模式时，建立一个表。

包含：用户id，二手商品id。

主码：用户id和二手商品id。用户id是参照用户表的外码，级联删除；二手商品id是参照二手商品表的外码，级联删除。

### 4.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：全文索引

### 5.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。

### 6.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

其中jdbcTemplate是封装好的jdbc

### 7.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

当此二手商品完成交易时，描述图片的image\_url字段也自动删除。

## 七、系统功能测试、数据库查询、技术实现

**(一) 系统功能测试与数据库查询**

### 1.注册与登录、用户操作令牌

用户要想进行个人用户操作（如加入购物车、创建二手商品等而非浏览），需要先登陆注册。

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。

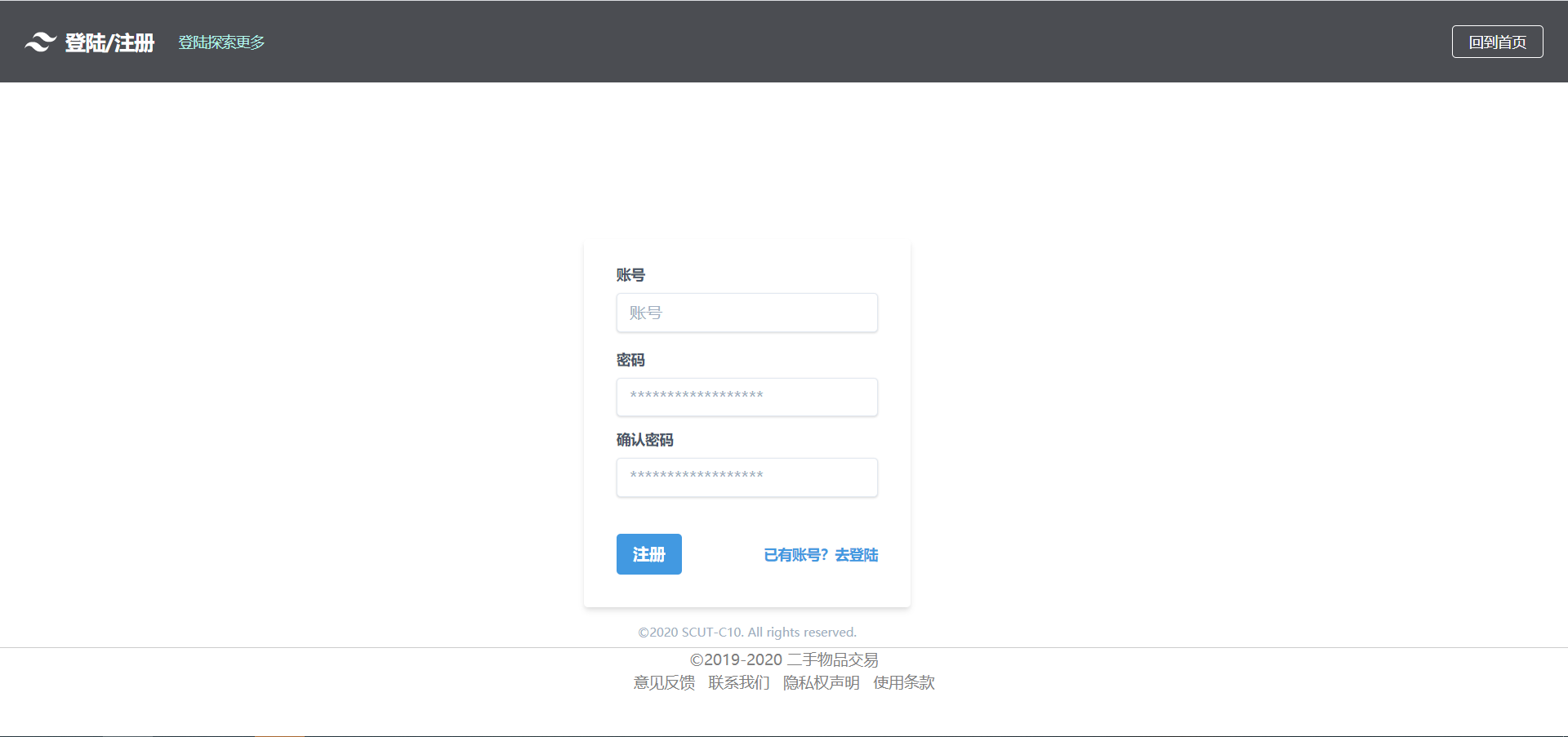


图 注册

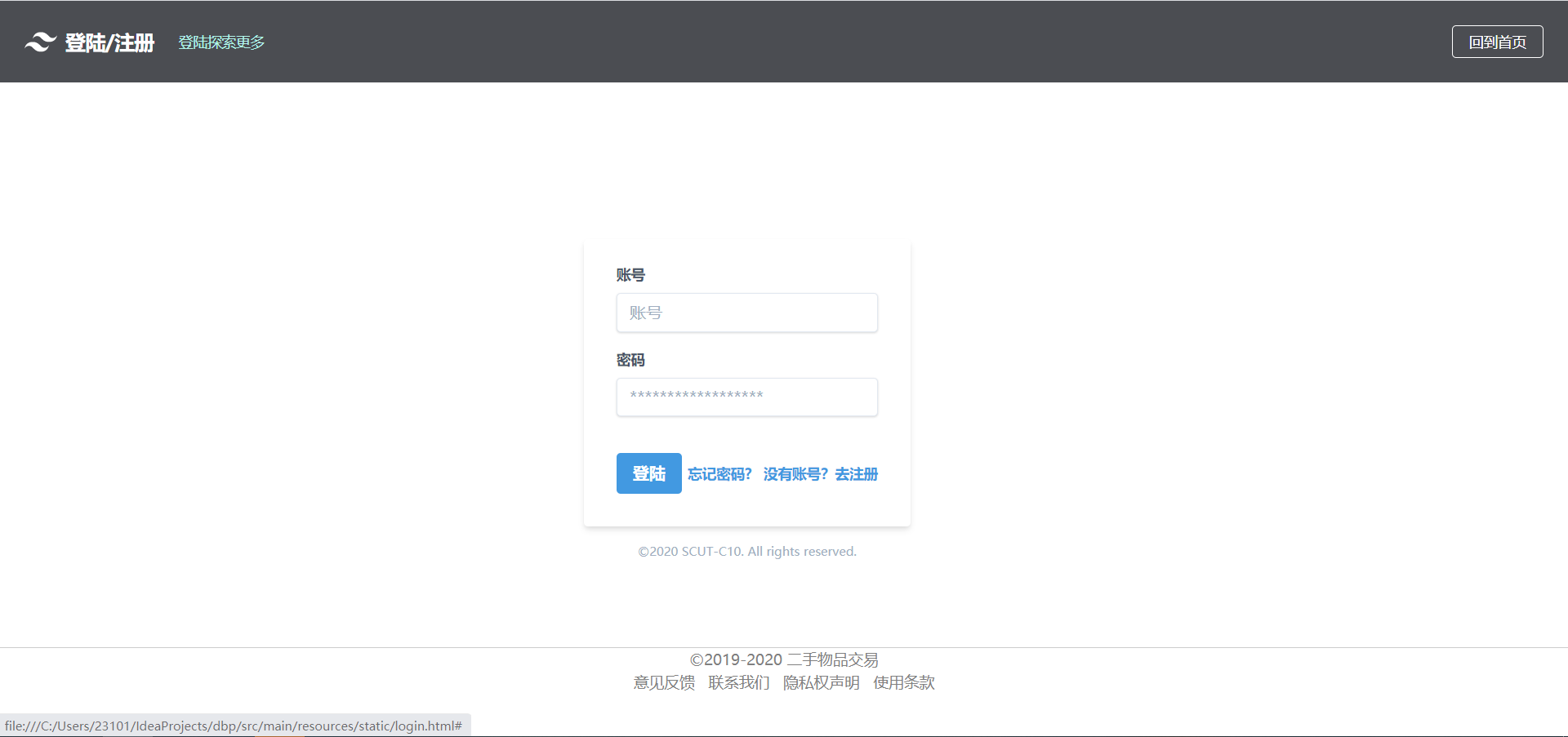


图 注册成功后，登陆

以下为登录的JavaScript代码：

*//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) {  
 *//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);  
 });  
}

**分析：**

以http的post方法发送用户id，密码，返回的operation\_code是后端利用SHA-1算法对密码及当前登陆时间（精确到毫秒）进行计算得到的160bit数据，作为用户本次登陆操作令牌，用于保障用户的信息安全。前端进行的一切用户级别的操作（发布二手商品信息、线上交流等），都需要此令牌，避免了每次操作都传输密码。



图 operation\_code

此外，后端服务器还安装了SSL证书，并基于HTTPS进行前后端的通信，保证了信息的安全。



图 网站的https连接

**数据库查询：**

SQL语句1：查询用户是否存在

select \* from dbpractical.users where user\_id = ?

SQL语句2：登陆操作，验证此用户密码是否正确

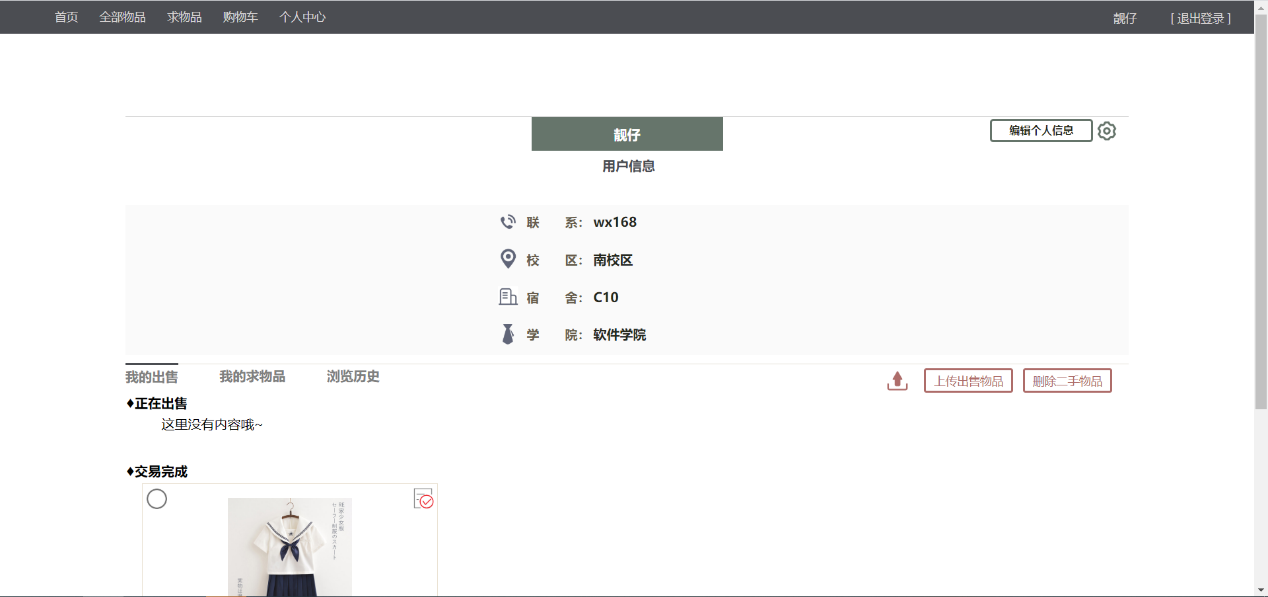
select \* from dbpractical.users where user\_id = ? and user\_password = ?

SQL语句3：用户级别操作，验证操作令牌

select \* from dbpractical.users where user\_id = ? and user\_login\_code = ?

注：采取的是预编译，接下来会用真实参数替代语句中的“?” 。

### 2.个人信息展示与修改



图：个人信息页



图：修改个人信息

**分析：**

修改时，只修改有改变的字段，把有变更的信息传给后端，后端再根据对应的字段拼接需要改变的字段修改

**数据库查询**

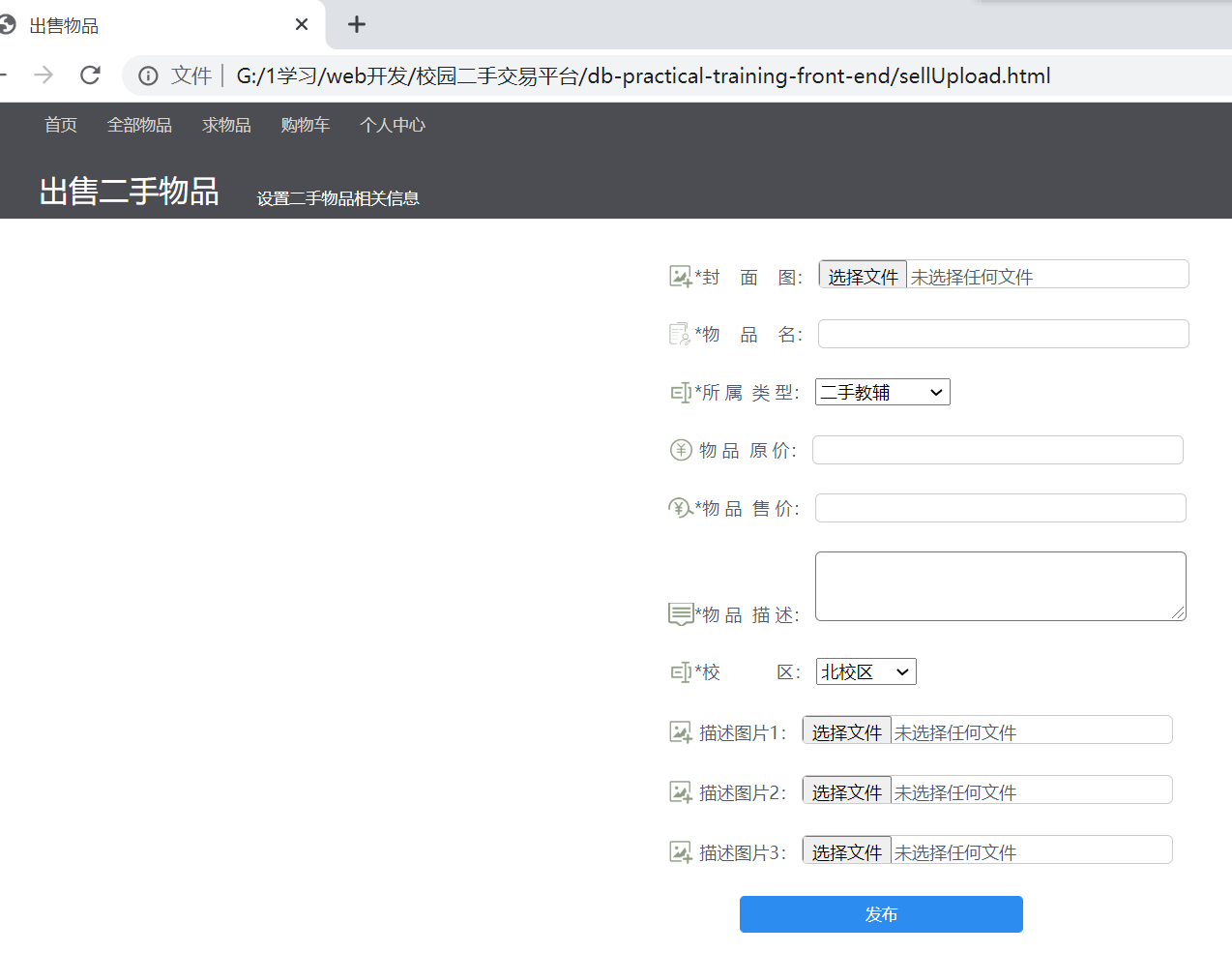
SQL语句：

update dbpractical.users set "+str+" where user\_id = ?;

其中的str则是由有变化的字段拼接而成

注：采取的是预编译，接下来会用真实参数替代语句中的“?” 。

### 3.商品的上架



19：点击上传二手商品图

**分析：**

对于图片，上传到服务器后，由数据库保存可访问的图片src，不直接保存图片。

**数据库查询**

SQL语句1： 增加到**二手商品表**中

insert into commodity(commodity\_name,commodity\_type,commodity\_original\_price,commodity\_price,commodity\_description,commodity\_cover\_pic\_url,publish\_time\_stamp,area,user\_id)

values(?,?,?,?,?,?,?,?,?);

SQL语句2： 增加到**二手商品的其余描述图片表**中

insert into commodity\_images(commodity\_id,image\_url) values(?,?);

注：采取的是预编译，接下来会用真实参数替代语句中的“?” 。

### 4.商品的分类展示



图：二手商品分类展示

分析：当用户点击某个分类，会向服务器发送此分类词，服务器利用全文索引(FULLTEXT)快速检索符合条件的二手商品

**数据库查询**

SQL语句： 从commodity表中检索此类商品

select commodity\_id,commodity\_name,commodity\_type,commodity\_price,commodity\_cover\_pic\_url,area

from dbpractical.commodity WHERE MATCH(`commodity\_type`) AGAINST(‘某类型’ IN BOOLEAN MODE)

### 5.交易流程

用户在商品列表页面，点击商品卡片即可进入详情页，

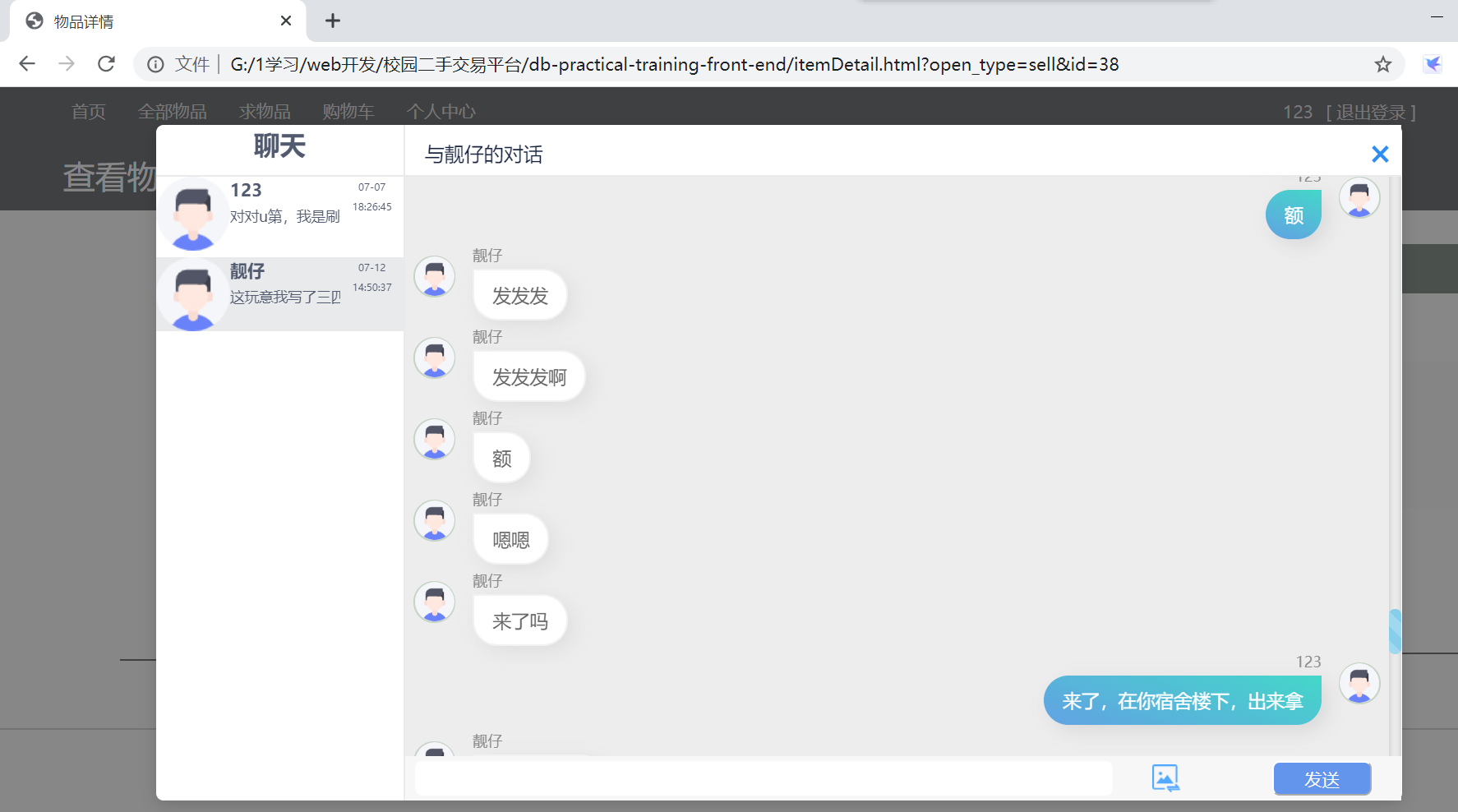


图：二手商品详情

·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。

·可点击“在线联系”按钮，进行在线联系。



图：在线联系

交易完成后，在个人中心，自己出售的二手商品下点击完成交易按钮【见下图】，则该二手商品下架，进入完成交易列表，此后仅对创建者可见，不再在二手商品展示页面显示。



图：二手商品完成交易按钮



图：已完成交易的二手商品

**（二）技术实现**

1.在线联系模块（聊天模块）：

技术框架：websocket + springboot + vue

技术实现：

websocket，类似于一种应用层协议，打开连接并与服务器保持连接，此时服务器上保存了此连接并由此知道发起连接者的在线状态。服务器相当于一个中转站，用户在前端页面向服务器发送消息对象，此对象包括发送方，接收方和消息。服务器根据你的接收方寻找对方，若对方在线则将此消息对象发给对方用户。最后将此消息对象存到数据库中保存。

springboot作为服务器端，在此模块的功能由上所述。

vue实现了前端实时渲染，在数据对象（消息）到来后实时渲染在界面上，实现实时通讯的功能。