****

**数据库系统原理课程设计**

**（设计报告）**

**图书管理系统**

姓 名：任思宇

学 号：1552179

所在院系：电子与信息工程学院

专 业：计算机科学与技术

指导教师：向阳

二〇一八年六月

**基于卷积神经网络的短文本分类研究**

摘要

本系统为一个二手物品发布平台，用户可以发布自己的二手物品，也可以浏览、评论他人发布的二手物品。

具体实现的功能包括：用户注册、用户登录、浏览已发布的二手物品、在某物品下评论、收藏/取消收藏某物品、发布自己的二手物品、关闭自己已发布的二手物品。

本系统采用Python语言编写，使用了Flask框架，数据库管理系统采用MySQL社区版

**关键词**：二手物品发布系统，Python，Flask，MySQL

目 录

[1 引言 1](#_Toc515560466)

[1.1 项目背景 1](#_Toc515560467)

[1.2 项目要求 1](#_Toc515560468)

[1.3 实现目标 1](#_Toc515560469)

[2 需求分析 2](#_Toc515560470)

[2.1 业务调查 2](#_Toc515560471)

[2.2原系统业务流程分析 2](#_Toc515560472)

[2.3 原系统数据流程分析 2](#_Toc515560473)

[2.4 新系统的目标 2](#_Toc515560474)

[2.4.1功能需求 2](#_Toc515560475)

[2.4.2性能需求 2](#_Toc515560476)

[2.4.3输入输出需求 2](#_Toc515560477)

[3 系统分析 3](#_Toc515560478)

[3.1新系统的子系统划分 3](#_Toc515560479)

[3.2新系统的业务流程 3](#_Toc515560480)

[3.3新系统的数据流程 3](#_Toc515560481)

[3.4新系统的数据字典 3](#_Toc515560482)

[4 系统设计 4](#_Toc515560483)

[4.1系统配置设计 4](#_Toc515560484)

[4.1.1硬件配置设计 4](#_Toc515560485)

[4.1.2软件配置设计 4](#_Toc515560486)

[4.1.3网络配置设计 4](#_Toc515560487)

[4.2系统结构设计 4](#_Toc515560488)

[4.2.1系统功能结构设计 4](#_Toc515560489)

[4.2.2系统网络结构设计 4](#_Toc515560490)

[4.3系统功能模块设计 4](#_Toc515560491)

[4.4编码设计 4](#_Toc515560492)

[4.5数据库设计 4](#_Toc515560493)

[4.5.1概念结构设计 4](#_Toc515560494)

[4.5.2逻辑结构设计 4](#_Toc515560495)

[4.5.3物理结构设计 4](#_Toc515560496)

[4.6输入输出界面设计 4](#_Toc515560497)

[4.6.1输入界面设计 4](#_Toc515560498)

[4.6.2输出界面设计 5](#_Toc515560499)

[4.7代码设计 5](#_Toc515560500)

[5 系统测试 6](#_Toc515560501)

[5.1 系统测试环境 6](#_Toc515560502)

[5.1.1测试的硬件环境 6](#_Toc515560503)

[5.1.2测试的软件环境 6](#_Toc515560504)

[5.1.3测试的网络环境 6](#_Toc515560505)

[5.2系统测试内容 6](#_Toc515560506)

[5.3系统测试方法 6](#_Toc515560507)

[5.4系统测试数据 6](#_Toc515560508)

[5.5系统测试结果分析 6](#_Toc515560509)

[6 课程设计总结 9](#_Toc515560510)

[参考文献 10](#_Toc515560511)

1 引言

1.1 项目背景

当今社会发展十分快速，物品淘汰、更新也十分的频繁。对于人们而言，很多使用了很久需要更新的物品舍不得直接扔掉，但又不知道该怎么处理，于是堆积成山。 如果能够将闲置的二手物品发布出来，有需要的其他人可以与二手物品的主人交流，通过线下或线上的渠道进行交易，那么就可以很好地解决这个问题。

于是我决定设计这样一个二手物品交易的Web系统，需要购买二手物品的用户可以在该平台上查找自己所需要的物品，而需要出手二手物品的用户也可以在该平台上发布相关信息。

1.2 项目要求

二手物品发布平台应该具有前台和后台功能，前台具备用户登录/注册、物品搜索、查看物品、收藏物品、用户评论、发布物品等功能；后台则应该具备管理员管理评论信息、已发布的二手物品的功能。使得该系统既能满足用户的需求，也能便于管理员进行统一、有效的管理。

1.3 实现目标

实现目标为一个可供用户访问的Web网站。对于用户，该网站的功能包括：用户注册、用户登录、浏览已发布的二手物品、在某物品下评论、收藏/取消收藏某物品、发布自己的二手物品、关闭自己已发布的二手物品。对于管理员，功能包括管理已发布的物品、管理物品下的用户评论。

2 需求分析

2.1 业务调查

现有的校园二手物品交易大多需要中介结构或店铺。比如开设在校园内的各种诸如“爱回收”这样的中介机构，或者其他店铺。如果想要将自己的二手物品出手掉，必须托管在中介机构处，等到中介机构出帮你卖出后，中介机构将交易额交给客户。这样的流程十分的繁琐，等待周期可能会十分漫长，并且有可能遭到中介机构的克扣、贱卖等情况出现。这样的交易环境是非常不利于在校园内进行二手物品交易的。

2.2原系统业务流程分析

原系统业务流程如下图所示：

开始

寻找中介机构

托管二手物品

交易额返还用户

结束

图2.2 原系统业务流程图

2.3 原系统数据流程分析

原系统的数据流程如下：

二手物品持有者

二手物品

中介机构

售卖情况

二手物品信息

购买意愿及出价

二手物品信息

二手物品购买者

图2.3 原系统数据流程图

2.4 新系统的目标

2.4.1功能需求

该二手物品交易系统针对用户群体和系统管理员。

用户的功能需求如下图所示：

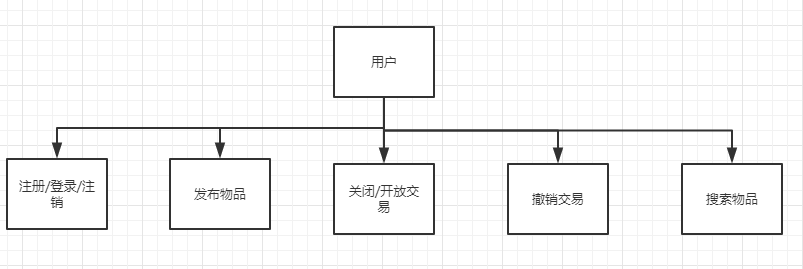


图2.4.1 用户功能需求

管理员的功能需求包括管理所有已发布的物品、管理所有用户发布的评论，如下图所示：

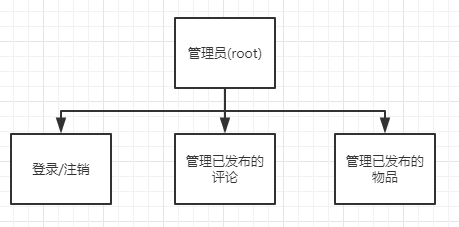


图2.4.2 管理员功能需求

2.4.2性能需求

系统的性能需求主要表现在数据库中的各个表需要频繁地被插入、删除以及更新。对于用户来说，系统地相应时间不宜太长，否则会降低用户体验，为此需要建立良好的表结构，加上足够的存储空间以及硬件性能。

2.4.3输入输出需求

系统的输入应满足最小量原则、简单性原则以及及早检验原则，即在满足处理要求的前提下应使输入量尽量小，系统能够计算出的数据不要由用户输入；输入过程应尽量简单；对输入数据的检验尽量接近原始数据发生点，使错误能及时得到改正。

系统的输出内容应简洁明了，使得用户一目了然；本系统中输出设备选择在屏幕上输出。

3 系统分析

3.1新系统的子系统划分

二手物品发布平台系统分为3个子系统，如下图所示：

二手物品发布系统

评论管理子系统

物品管理子系统

用户管理子系统

图3.1.1 子系统划分

用户管理子系统的功能包括用户注册、修改用户信息等。

物品管理子系统的功能包括二手物品的发布、关闭、浏览、收藏等。

评论管理子系统的功能包括添加评论、删除评论等。

3.2新系统的业务流程

开始

登陆

发布或浏览物品

结束

图3.2.1 新系统的业务流程

3.3新系统的数据流程

物品数据项

物品数据表

二手物品信息

搜索结果

用户

物品数据项集合

关键词

评论内容

评论数据项

评论数据表

用户信息

用户数据项

用户数据表

收藏物品信息

用户收藏物品数据表

收藏物品数据项

图3.3.1 新系统用户数据流程图

物品数据项

物品数据表

物品信息

管理员

评论数据项

评论数据表

评论信息

图3.3.2 新系统管理员数据流程图

3.4新系统的数据字典

发布二手物品：处理过程，将用户输入的欲发布的二手物品的信息进行处理。输入数据流为二手物品信息，输出数据流为物品数据项。

搜索二手物品：处理过程，使用用户输入的关键词查找相关物品。输入数据流为关键词，输出数据流为物品数据项集合。

评论：处理过程，在某一物品下添加用户的评论。输入数据流为评论内容，输出数据流为评论数据项。

修改用户信息：处理过程，更新用户的个人信息。输入数据流为用户信息，输出数据流为用户数据项。

收藏物品：处理过程，更新用户的收藏物品列表。输入数据流为收藏物品信息，输出数据流为收藏物品数据项。

管理物品：处理过程，管理所有发布的物品。输入数据流为物品信息，输出数据流为物品数据项。

管理评论：处理过程，管理所有物品下的所有评论。输入数据流为评论信息，输出数据流为评论数据项。

物品数据表：数据存储，存储所有物品的信息。输入数据流为物品数据项和物品数据项集合，输出数据流为搜索结果。

评论数据表：数据存储，存储所有评论的信息。输入数据流为评论数据项。

用户数据表：数据存储，存储所有用户的信息。输入数据流为用户数据项。

用户收藏物品数据表：数据存储，存储所有用户的收藏物品信息。输入数据流为收藏物品数据项。

用户：数据流源点。输入数据流为搜索结果，输出数据流包括二手物品信息、关键词、评论内容、用户信息、物品信息。

管理员：数据流源点。输出数据流包括物品信息和评论信息。

二手物品信息：数据流。数据流组成为：物品名称+物品价格+物品详细描述+物品图片

关键词：数据流。数据流组成为：中文字符串

评论内容：数据流。数据流组成为：中文字符串

用户信息：数据流。数据流组成为：邮箱+密码

收藏物品信息：数据流。数据流组成为：物品ID

搜索结果：数据流。数据流组成为：{(物品名称+物品价格+物品详细描述+物品图片)}

物品数据项：数据流。数据流组成为：物品ID+物品名称+物品价格+物品详细描述+物品图片

评论数据项：数据流。数据流组成为：评论ID+评论内容+评论时间+评论所属用户ID+评论所属物品ID

用户数据项：数据流。数据流组成为：用户ID+用户名+用户密码+用户邮箱+用户权限

收藏物品数据项：数据流。数据流组成为：用户ID+物品ID

4 系统设计

4.1系统配置设计

4.1.1硬件配置设计

本系统所需的硬件配置为：服务器1台。

4.1.2软件配置设计

本系统所需的软件配置如下

操作系统：Windows、Linux、MacOS

数据库管理系统：MySQL

编程语言：Python

web框架：Flask

4.1.3网络配置设计

本系统的网络配置为可远程访问，系统架设在腾讯云服务器上，服务器操作系统为Ubuntu 16.04 LTS。采用的服务器软件为nginx和gunicorn。Nginx作为反向代理服务器，gunicorn则作为接口负责服务器与flask框架的交互。用户可以通过访问IP来使用本系统。

4.2系统结构设计

4.2.1系统功能结构设计

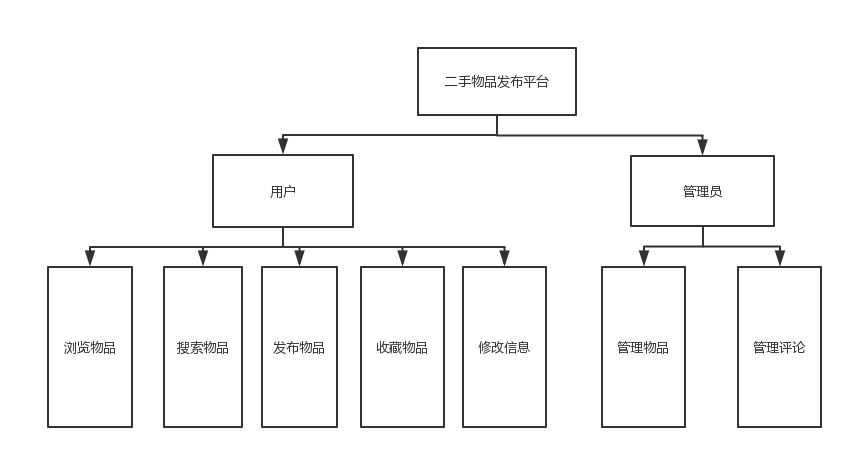


图4.2.1.1 系统功能结构图

4.2.2系统网络结构设计

系统的网络结构分为三部分：客户端、Web服务器和Web框架。如下图所示：

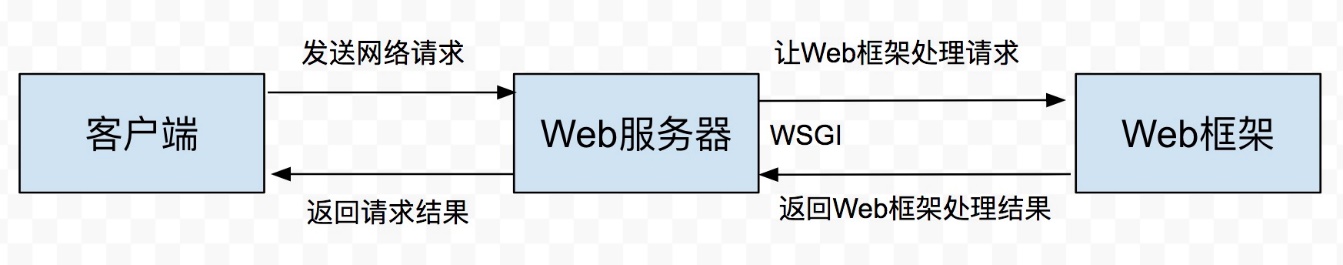


图4.2.2.1 系统网络结构图

客户端一般指浏览器或者移动端。

Web服务器的作用主要是接收客户端请求并返回处理结果，常用的Web服务器有Nginx、Apache等，Web服务器的挑战是数据的并发、并行和吞吐量。

Web框架主要是对接收到的客户端请求进行处理，常用的Web框架有flask、Django等。Web框架的挑战是易于业务逻辑的开发。

4.3系统功能模块设计

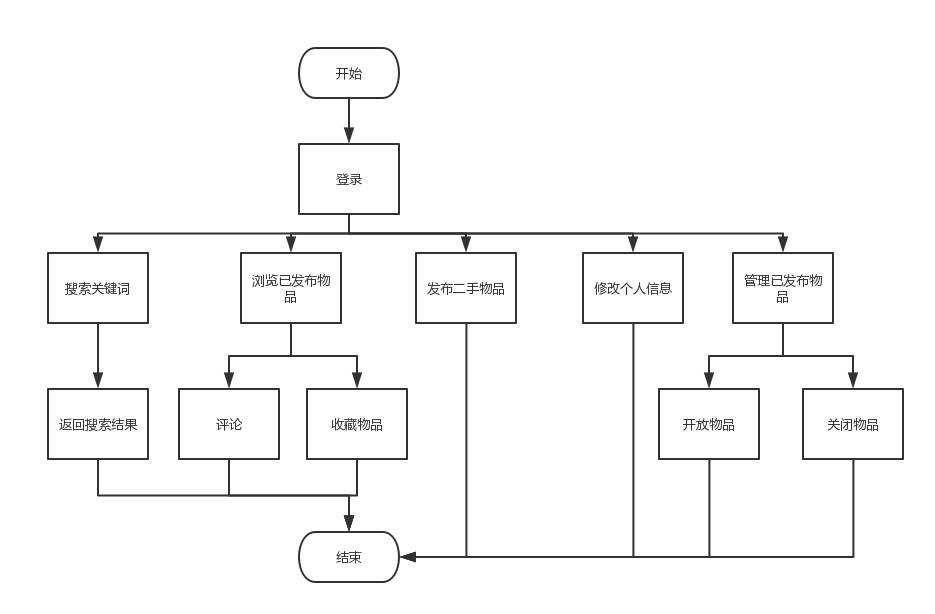


图4.3.1 用户功能程序流程图

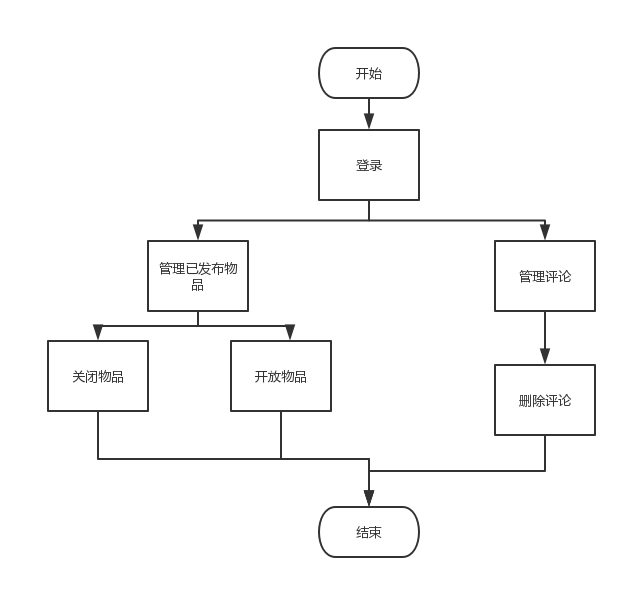


图4.3.2 管理员功能程序流程图

4.4编码设计

本系统中的编码设计包括：用户编码、物品编码、评论编码、兴趣编码。

用户编码从0开始计数，依次增加1。

物品编码从0开始计数，依次增加1。

评论编码从0开始计数，依次增加1。

兴趣编码从0开始计数，依次增加1。

4.5数据库设计

4.5.1概念结构设计

本二手物品交易平台中一共有用户、管理员、物品、评论这三个基本的模型(实体)。一个用户可以发布多个物品，而一个物品只能对应于一个用户。一个用户可以发布多条评论，而一条评论只能对应于一个用户。一个物品交易下可以有多条评论，但一条评论只能对应于一个物品。ER图如下：

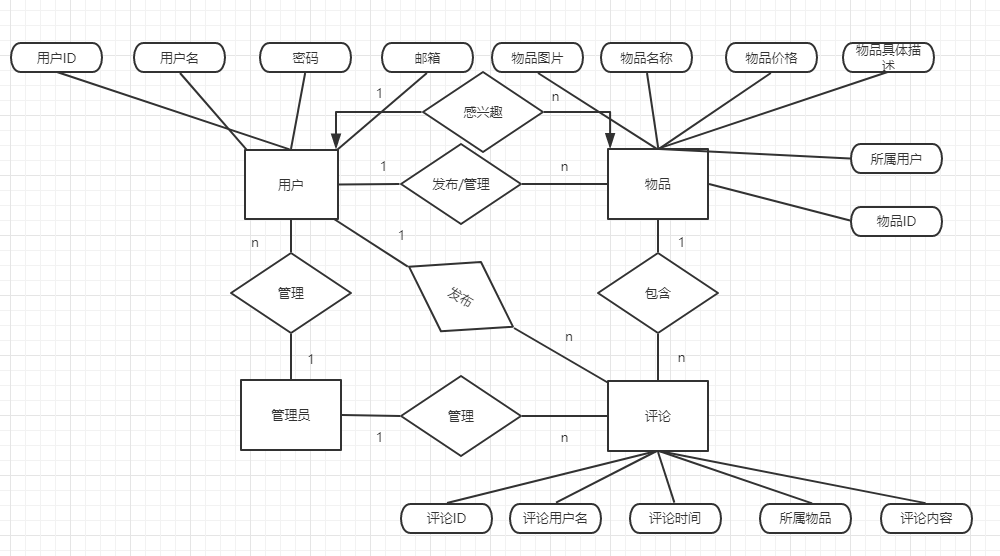


图4.5.1.1 二手物品发布平台ER图

用户(非管理员):用户ID、用户名、用户密码、用户邮箱。

物品：物品ID、物品名称、物品价格、物品详细描述、物品图片

评论：评论ID、评论内容、评论时间、评论所属物品、评论所属用户

感兴趣：用户ID、物品ID

管理员的属性和用户的属性基本相同，但管理员拥有管理所有发布的物品和发布的评论的功能，且管理员不能发布物品和评论。管理员是唯一的。

4.5.2逻辑结构设计

将概念结构设计中的各个模型转化为DBMS支持的表结构，同时保持不会出现插入异常、删除异常和修改异常。表结构应该做到符合第三范式。在本系统中，一共有三个实体，实体转化为数据库模型为如下所示：

(1)实体转化为关系模式：

用户(用户ID、用户名、用户密码、用户邮箱、用户权限)

物品(物品ID、物品名称、物品价格、物品详细描述、物品图片)

评论(评论ID、评论内容、评论时间、评论所属用户ID、评论所属物品ID)

(2)一个一对多联系转化为一个关系模式

用户-物品(用户ID,物品ID)

用户-评论(用户ID,评论ID)

物品-评论(物品ID，评论ID)

兴趣(用户ID、物品ID)

(3)利用以上关系模式得到的所有数据表如下所示：

表4-1 用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 数据长度 | 可否为空 | 备注 |
| Id | integer | 4 | Not null | 用户ID（主键） |
| Username | varchar | 20 | Not null | 用户名 |
| Password | varchar | 100 | Not null | 用户密码 |
| Email | varchar | 30 | Not null | 用户邮箱 |
| Authority | boolean | 1 | Not null | 用户权限 |

表4-2 物品表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 数据长度 | 可否为空 | 备注 |
| Id | integer | 4 | Not null | 物品ID（主键） |
| Name | varchar | 30 | Not null | 物品名 |
| Description | text | 不定 | Null | 物品详细描述 |
| Price | flaot | 4 | Not null | 物品价格 |
| Imgpath | varchar | 300 | Null | 物品图片URL |
| Create\_time | datetime | 8 | Not null | 物品发布时间 |
| Is\_closed | integer | 4 | Not null | 交易是否被关闭 |
| Ownerid | Integer | 4 | Not null | 所属用户ID |

表4-3 评论表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 数据长度 | 可否为空 | 备注 |
| Id | integer | 4 | Not null | 评论ID（主键） |
| Content | text | 不定 | Not null | 评论内容 |
| Itemid | integer | 4 | Not null | 评论物品ID |
| Ownerid | integer | 4 | Not null | 评论所属用户ID |
| Create\_time | datetime | 8 | Not null | 评论创建时间 |

表4-4 兴趣表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 数据长度 | 可否为空 | 备注 |
| Id | integer | 4 | Not null | 兴趣ID（主键） |
| Userid | integer | 4 | Not null | 用户ID |
| Itemid | Integet | 4 | Not null | 物品ID |

其中除了兴趣之外的一对多关系模式在系统的具体实现中并没有建立实体的表结构，在使用flask框架中有相关的函数实现。

4.5.3物理结构设计

1、选择关系模式的存取方式

对数据库逻辑结构设计中建立的表结构，用户表的用户ID属性唯一决定每一个用户元组，所以对用户表建立以用户ID为主关键字的索引。同理，对物品关系模式、评论关系模式也采用类似的索引存取方法。

2、数据表存储结构设计

本系统的所有数据表均存放在物理磁盘中。用户表、物品表和评论表的结构是相对稳定的，表中的已有记录是要长期保存的，在此基础上系统会相应用户的操作对数据表进行增、删、查、改等操作。

4.6输入输出界面设计

4.6.1输入界面设计

输入界面设计包括用户登录界面、注册界面、搜索界面、评论界面、发布物品界面、修改个人信息界面等，如下图所示：

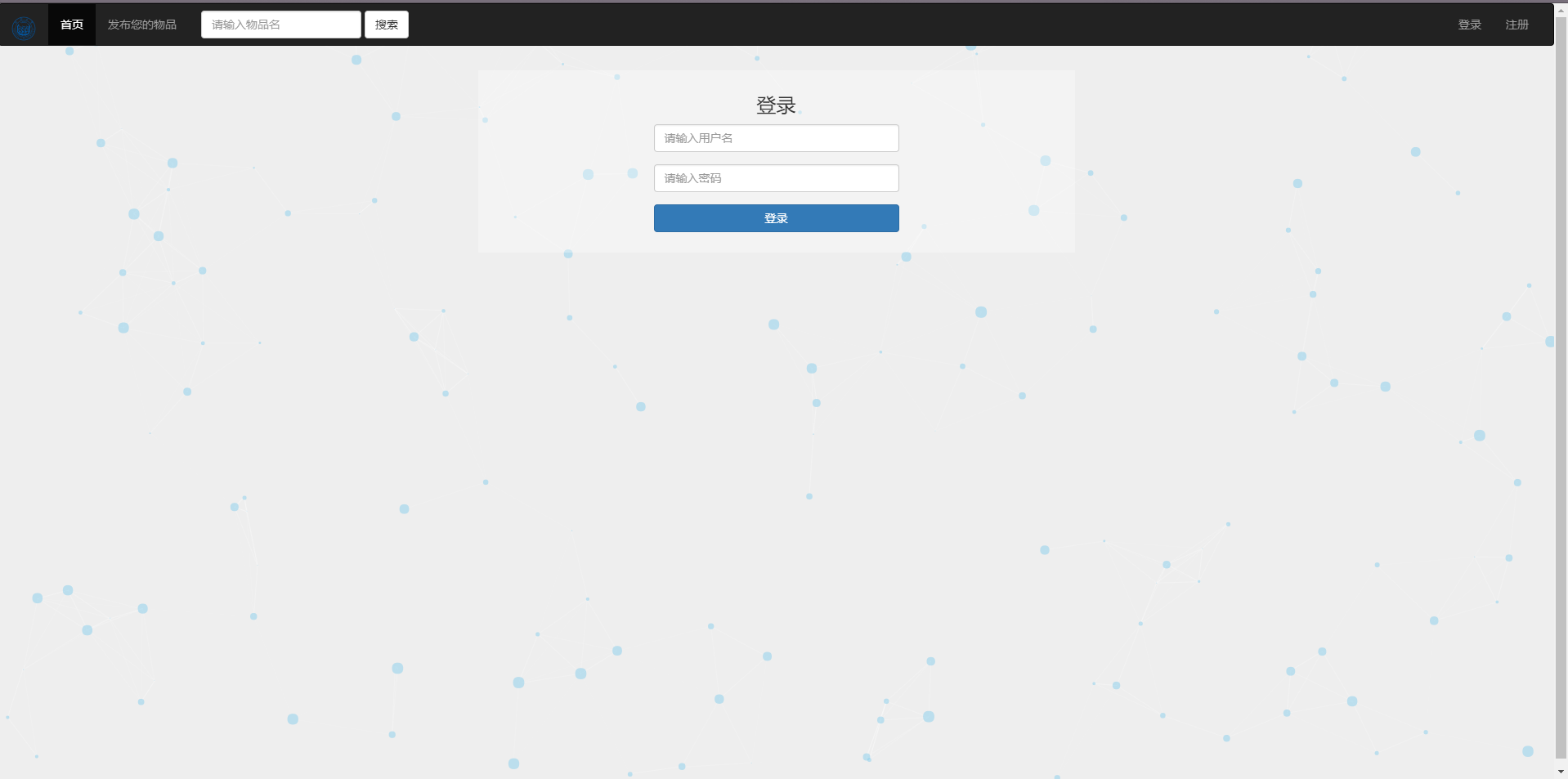


图4.6.1.1 登录界面

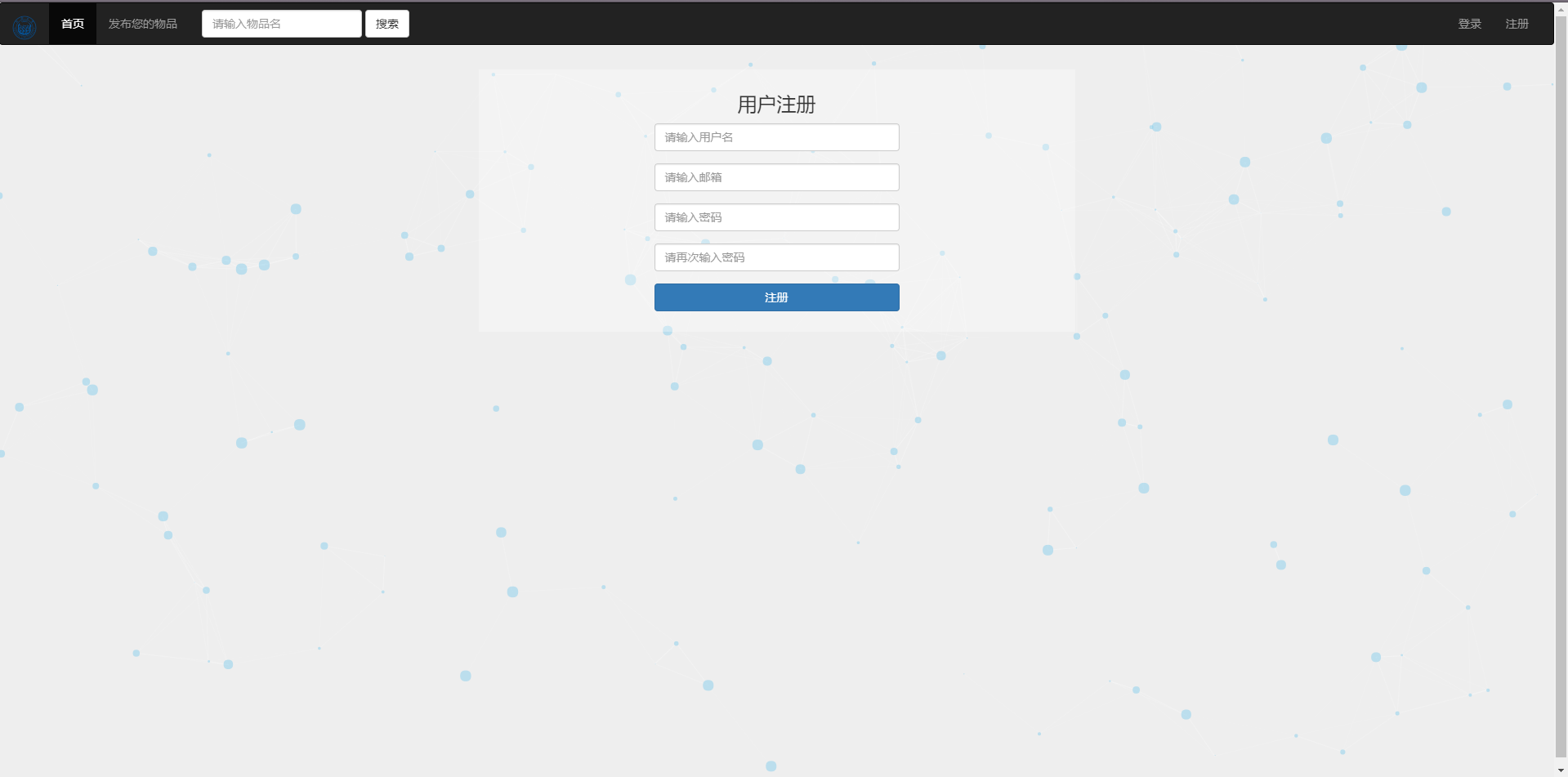


图4.6.1.2 注册界面



图4.6.1.3 搜索框界面

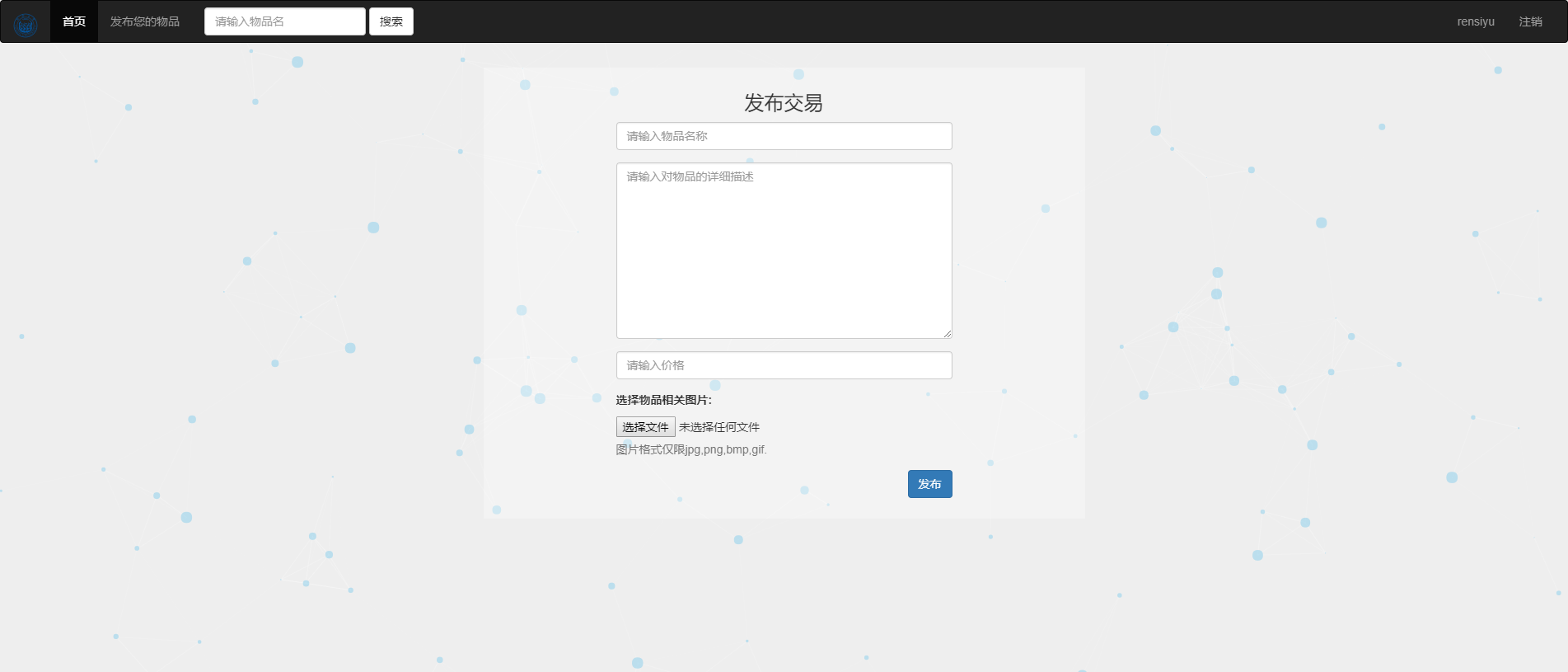


图4.6.1.4 发布物品界面

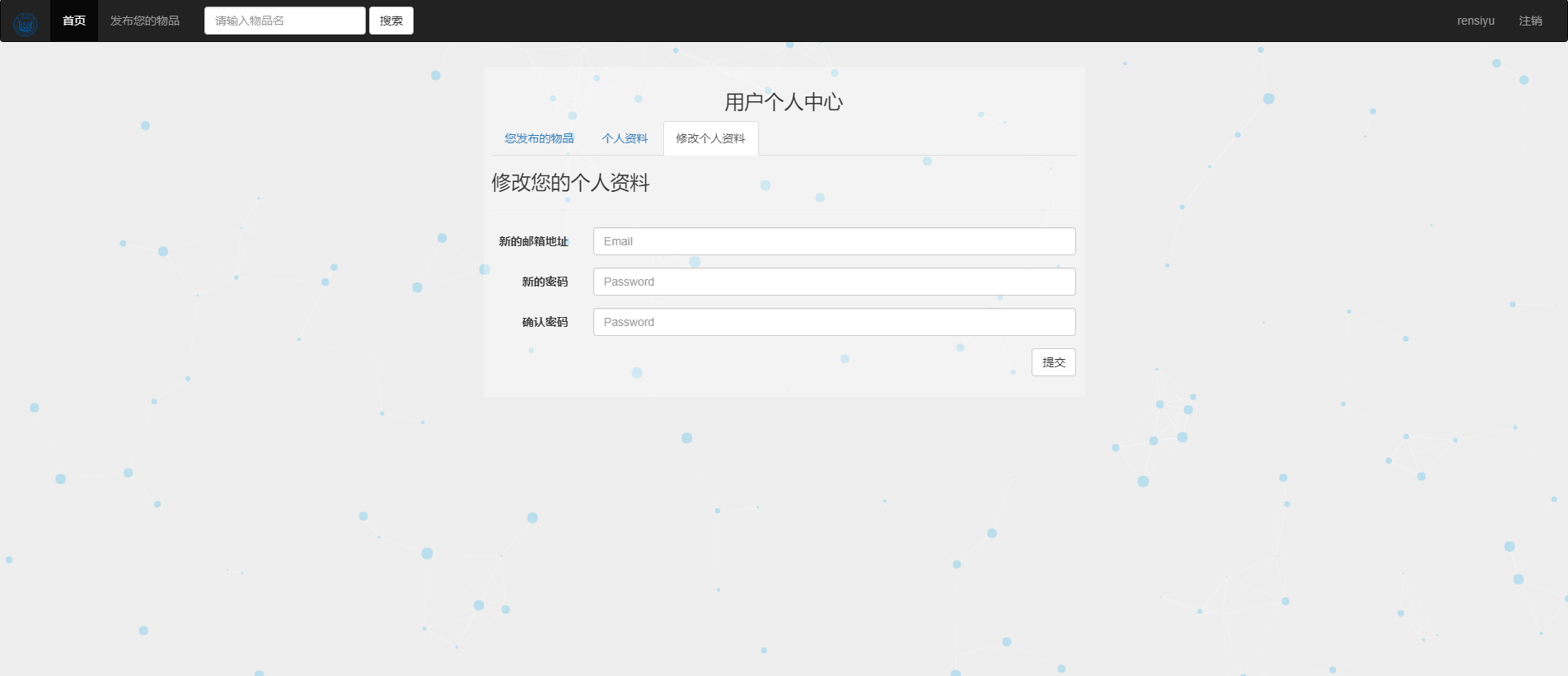


图4.6.1.5 修改个人资料界面

4.6.2输出界面设计

输出界面设计包括：首页界面、物品详情界面、用户个人中心界面等，如下图所示：



图4.6.2.1 首页界面



图4.6.2.2 物品详情界面

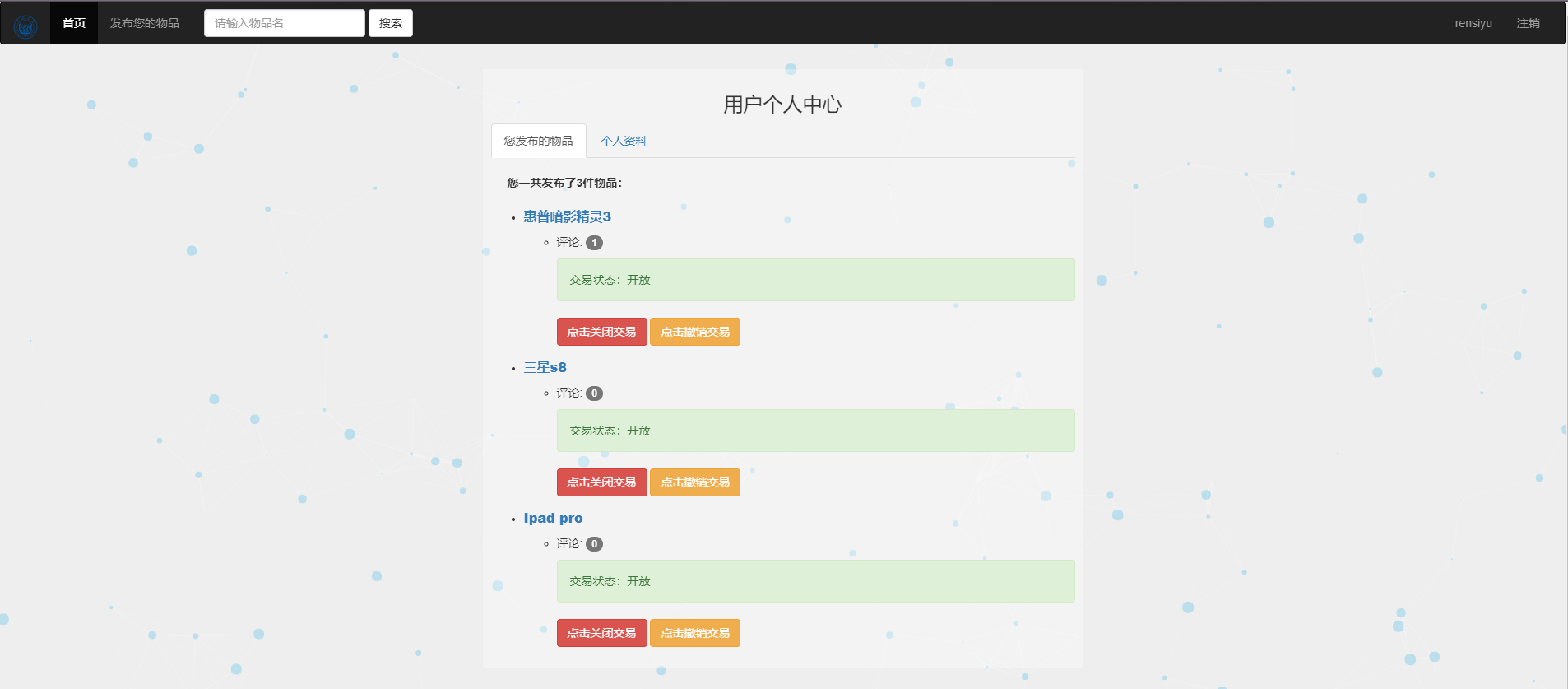


图4.6.2.3 用户个人中心界面之一



图4.6.2.4 用户个人中心界面之二

4.7代码设计

数据库课程设计正文（主要代码片段，不超过三页，要有描述）

|  |
| --- |
| **from extensions import db**  **from datetime import datetime**  **from werkzeug.security import generate\_password\_hash,check\_password\_hash**  **##用户模型**  **class User(db.Model):**  **\_\_tablename\_\_='user'**  **id = db.Column(db.Integer,primary\_key=True,autoincrement=True)**  **username = db.Column(db.String(20),nullable=False)**  **password = db.Column(db.String(100),nullable=False)**  **email = db.Column(db.String(30),nullable=False)**  **authority = db.Column(db.Boolean,nullable=False,default=False)**  **def \_\_init\_\_(self,\*args,\*\*kwargs):**  **email = kwargs.get('email')**  **username = kwargs.get('username')**  **password = kwargs.get('password')**  **authority = kwargs.get('authority')**  **self.email = email**  **self.username = username**  **self.authority = authority**  **self.password = generate\_password\_hash(password)**  **def checkPassWord(self,input\_password):**  **return check\_password\_hash(self.password,input\_password)**  **##二手物品模型**  **class Item(db.Model):**  **\_\_tablename\_\_ = 'item'**  **id = db.Column(db.Integer,primary\_key=True,autoincrement=True) ##物品ID**  **name = db.Column(db.String(30),nullable=False) ##物品名称**  **description = db.Column(db.Text,nullable=True) ##物品详细描述**  **price = db.Column(db.Float,nullable=False) ##物品价格**  **imgpath = db.Column(db.String(300),nullable=True) ##物品图片url**  **ownerid = db.Column(db.Integer,db.ForeignKey('user.id')) ##物品发布者id**  **owner = db.relationship('User',backref=db.backref('items')) ##物品发布者**  **create\_time = db.Column(db.DateTime,default=datetime.now) ##物品发布时间**  **isclosed = db.Column(db.Integer,nullable=False,default=0) ##物品交易是否被关闭，1为关闭**  **##评论模型**  **class Comment(db.Model):**  **\_\_tablename\_\_ = 'comment'**  **id = db.Column(db.Integer,primary\_key=True,autoincrement=True) ##评论ID**  **content = db.Column(db.Text,nullable=False) ##评论内容**  **itemid = db.Column(db.Integer,db.ForeignKey('item.id')) ##评论物品的ID**  **ownerid = db.Column(db.Integer,db.ForeignKey('user.id')) ##评论发布者ID**  **create\_time = db.Column(db.DateTime,default=datetime.now) ##评论发布时间**  **item = db.relationship('Item',backref=db.backref('comments',order\_by=id.desc())) ##评论物品**  **owner = db.relationship('User',backref=db.backref('comments')) ##评论发布者**  **##用户--物品 感兴趣的模型**  **class Interest(db.Model):**  **\_\_tablename\_\_ = 'interest'**  **id = db.Column(db.Integer, primary\_key=True, autoincrement=True)**  **userid = db.Column(db.Integer,db.ForeignKey('user.id')) ##用户ID**  **itemid = db.Column(db.Integer,db.ForeignKey('item.id')) ##物品ID**  **from functools import wraps**  **from flask import session,redirect,url\_for**  **##登录限制装饰器**  **def login\_required(f):**  **@wraps(f)**  **def wrapper(\*args,\*\*kwargs):**  **if session.get('user\_id'):**  **return f(\*args,\*\*kwargs)**  **else:**  **return redirect(url\_for('login'))**  **return wrapper** |

5 系统测试

5.1 系统测试环境

5.1.1测试的硬件环境

测试时的硬件环境如下所示：

PC：个人笔记本电脑，CPU为AMD Ryzen2代低压U，内存为8G，硬盘为256G固态硬盘。

服务器：阿里云主机

5.1.2测试的软件环境

测试时的软件环境如下所示：

操作系统：Windows10家庭版

浏览器: Chrome

5.1.3测试的网络环境

测试时的网络环境为20M带宽的校园无线局域网。

5.2系统测试内容

系统的测试内容主要为功能性测试，即测试本系统得而各项功能是否能够正常工作。功能测试包括用户登录功能测试、注册功能测试、发布物品功能测试、搜索物品功能测试、修改个人信息功能测试、收藏物品功能测试、管理员删除评论功能测试、管理员删除物品功能测试。

5.3系统测试方法

系统采用的测试方法为黑盒测试。黑盒测试又叫功能测试或数据驱动测试。黑盒测试是在程序接口上进行测试，主要是为了发现以下错误：是否有不正确或者是遗漏了的功能，在接口上，输入能否正确的接受，能否输出正确的结果；是否有数据结构错误或者外部信息访问错误；性能上是否满足要求；是否有初始化或终止性错误；用黑盒测试发现程序中的错误，必须在所有可能的输入条件和输出条件中确定测试数据，来检查程序是否都能产生正确的结果。

5.4系统测试数据

系统测试数据如下：

1.使用用户名为rensiyu，密码为68700695a的账户(该账户已存在在数据库中)登录系统，测试登录功能是否正常。

2.使用系统的注册功能，注册用户名为test\_user，邮箱为610141528@qq.com，密码为123456的账户，然后使用该账户登录系统，测试注册功能是否正常。

3.使用系统的搜索功能，输入“惠普”关键字(数据库中存在相关物品)，测试系统的搜索功能是否正常。

4.使用系统的发布物品功能，发布物品名为测试物品，详细描述为zzzz,价格为666，图片为空的物品，测试系统的发布物品功能是否正常。

5.使用系统的修改个人信息功能，将rensiyu账号的密码修改为68700695，邮箱改为666666666@qq.com测试系统的修改个人信息功能是否正常。

6.使用系统的收藏物品功能，将“惠普暗影精灵3”收藏，测试系统的收藏功能是否正常。

7.使用系统的管理员删除评论功能，将“惠普暗影精灵3”物品下的第一条评论删除，测试系统的删除评论功能是否正常。

8.使用系统的管理员删除物品功能，将“惠普暗影精灵3”物品删除，测试系统的删除物品功能是否正常。

5.5系统测试结果分析

测试1结果如下图，测试结果正确：

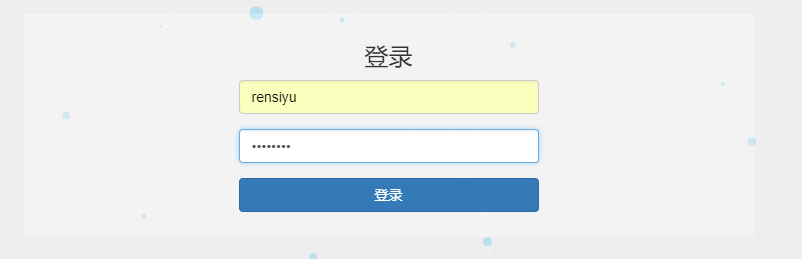


图5.5.1 登录之一，输入用户名和密码



图5.5.2 登录之二，登录成功

测试2结果如下，测试结果正确：



图5.5.3 注册之一，输入注册信息



图5.5.6 注册之二，登录成功

测试3结果如下，测试结果正确：



图5.5.7 搜索之一，输入关键字



图5.5.8 搜索之二，搜索结果

测试4结果如下，测试结果正确：

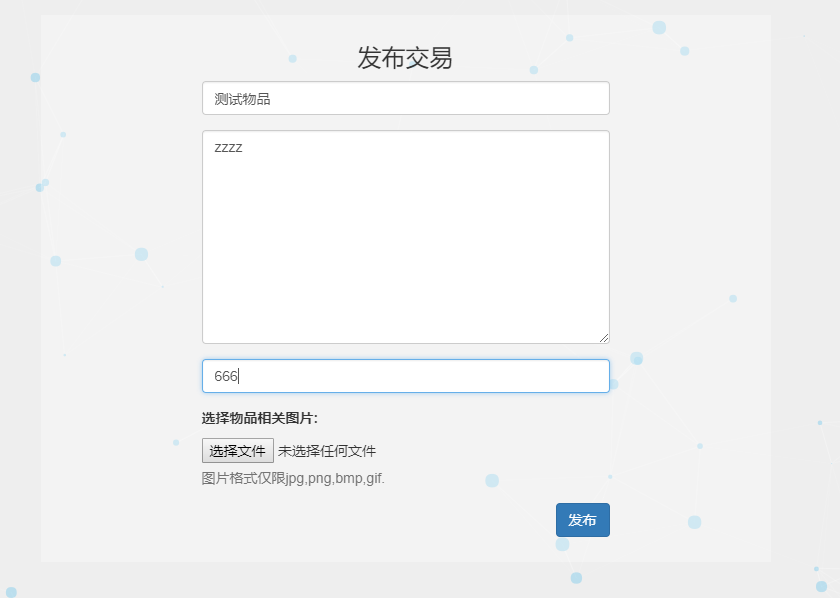


图5.5.9 发布物品之一，输入信息

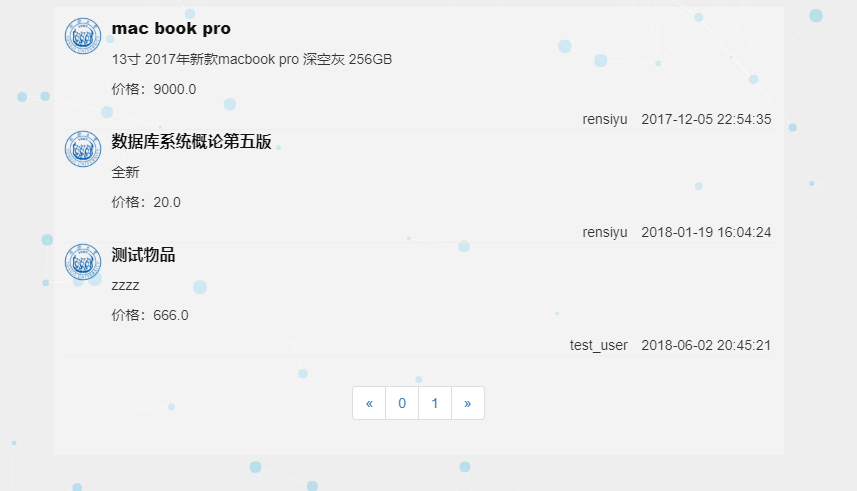


图5.5.10 发布物品之二

测试5结果如下，测试结果正确：



图5.5.11 修改个人资料之一



图5.5.12 修改个人资料之二

测试6结果如下，测试结果正确：



图5.5.13 收藏物品之一



图5.5.14 收藏物品之二



图5.5.15 收藏物品之三

测试7结果如下，测试结果正确：



图5.5.16 管理员删除评论之一

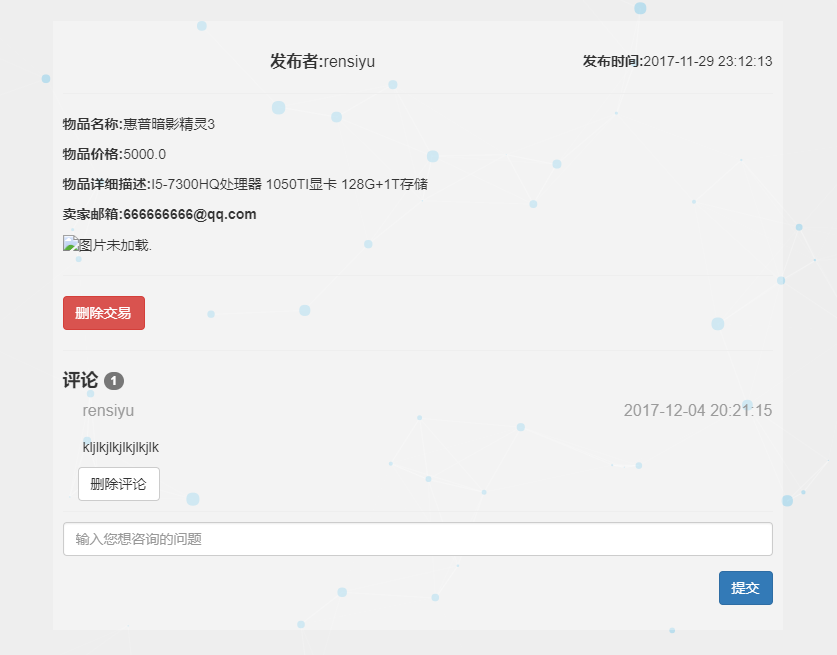


图5.5.17 管理员删除评论之二

测试8结果如下，测试结果正确：

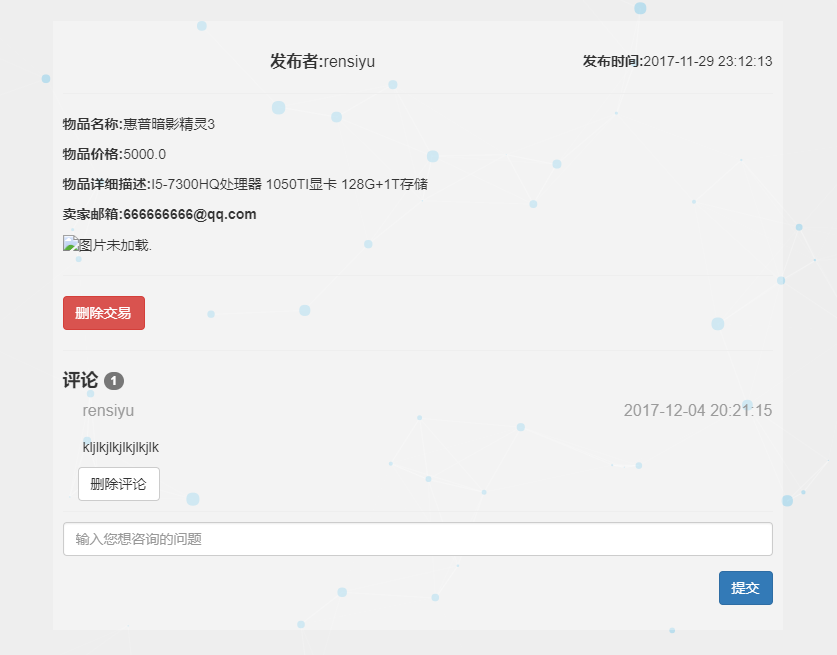


图5.6.18 管理员删除物品之一，点击删除交易



图5.6.19 管理员删除物品之二，查看结果

6 课程设计总结

这次数据库课程大作业是一次非常有趣的实践过程。通过把老师在课上讲的内容运用到实战中去，不仅能够提高对数据库的理解，也同时锻炼了我们关于WEB开发相关的技能，如html、css、javascript等代码的编写能力。除此之外也让我更加深入地了解了Python语言的flask框架，实际运用了ORM模型来实现操作类对象和数据库交互。

设计一个二手物品发布平台需要进行需求分析，可行性分析等一系列研究。分析用户在使用我的系统时需要哪些功能，怎样设计WEB页面能够让用户简单快捷地进行想要的操作。在数据库中建立所需要的数据表，并且最大限度地保证没有冗余，且查询等操作的速度较快。有一点经验是在编写代码的过程中需求可能是变化的，比如某个时刻可能需要给一个表添加新的属性，那么通过flask\_script和flask\_manage插件可以完成数据库的migrate和upgrade，更新实际表的内容。

系统实现的功能包括了基本需求，注册、登录、发布、评论、个人中心等等。但没有实现类似淘宝、京东那样的购物网站一样的交易功能，本系统的交易功能是以提供卖家的联系方式让用户和卖家通过第三方渠道交易实现的，相当于提供了一个类似预定的功能。

在完成课程设计的过程中，主要遇到了以下几个问题及难点。第一个问题是由于数据库中存在其他表的foreign key是其他表的primary key，所以在实现一些功能比如删除物品的时候，由于物品下属有对于该物品的评论，如果简单地删除物品表中对应的项，会造成严重的错误。经过排查之后，在相关的视图函数中进行了修改。第二个问题是系统的界面设计，一开始系统的界面设计完全是自己手写HTML、CSS来实现每一个页面，但这样实现起来太过繁琐。后来使用了bootstrap,是推特推出的一个用于前端开发的开源工具包，使用它来实现界面就方便快捷了许多。第三个问题是搭建服务器。一开始由于没有相关的经验，完全不知道如何开始下手。后来经过在网络上查询相关资料，在腾讯云上租用了一台云服务器。在云服务器上配置好了跟本地类似的编程环境和数据库。到此一切顺利，接下来配置好gunicorn和nginx后，已经可以在浏览器中访问服务器公网IP登录网站，但是却出现了用户登录后session丢失的情况。针对这种情况，再查阅资料后，发现解决方法是把Flask框架默认的客户端session改为服务端sesison，在服务器上存储session。为此在云服务器上安装了redis数据库用以存放Sesison。改为服务端Session后，问题得以解决。

总的来说，这次的课程设计是一次艰难而充满收获的实践。

参考文献

[1] 《信息系统分析与设计》.向阳 .2013