密级：公开

**基于Node.js供求信息网站的设计与实现**

**Design and Implementation of Supply and Demand Information Web Site Based on Node.js**

学 院：信息科学与工程学院

专 业 班 级：计算机科学与技术1402班

学 号：

学 生 姓 名：

指 导 教 师：王溪波（教授）

2018 年 6 月

# 摘 要

现今随着信息化社会的快速发展，人们对计算机的依赖程度也有增无减。在互联网这个大世界，人们利用信息的共享性，分享自己的信息。利用互联网这个平台对这些用户的个人需求信息加以传播，在传播的同时，通过一定的技术手段为用户提供平台，用户根据自己的需求会自发的相互沟通，促进了信息社会的发展，不仅具有便捷性还很直观，而且也极大的节约了人们的物力财力，最重要的是节约了人们的时间。

本文简单的介绍了课题的背景和意义，并列举了课题相关的主要技术，并对研究课题进行了详细的需求分析。本文研究课题为供求信息网站，该系统主要分为数据接口端与网站端这两个子模块。数据接口端主要为系统后端提供所需的数据，并且进行必要的数据交互。网站端主要功能是调用数据端接口并且为前端提供必要的数据、进行必要的数据交互，分为用户登录注册、浏览、查找、具体详情页的沟通、留言等功能模块。而且本文较为详细的描述了系统每个功能模块的具体实现过程，并通过图片和代码等相关技术对每个功能模块进行了详尽的分析。

本系统运行使用的是Windows10系统。主要开发平台是Webstorm。系统的前端是HTML/CSS和JavaScript编写。后端运用了Node.js的经典框架Express，并且前后端分离，方便管理。数据存储主要是MongoDB数据库。对数据管理使用的是Robo数据可视化管理工具。

关键字:供求信息网；Express框架；Node.js；

# 

# Abstract

With the rapid development of the information-based society, people’s dependence on computers has also increased. In the big world of the Internet, people use information sharing to share their own information. Using the platform of the Internet to disseminate these users' personal needs information, at the same time, through the use of certain technical means to provide users with a platform, users will spontaneously communicate with each other according to their needs, and promote the development of the information society, not only convenient Sex is also very intuitive, and it also greatly saves people's material and financial resources. The most important thing is saving people's time.

This article briefly introduces the background and significance of the project, and lists the main technologies related to the project, and conducts a detailed analysis of the research needs. The research topic of this paper is the supply and demand information website. The system is mainly divided into two submodules: data interface and website. The data interface end provides the required data for the system backend and performs the necessary data interaction. The main function of the web site is to invoke the data interface and provide the necessary data for the front end and perform necessary data interaction. It is divided into functional modules such as user login registration, browsing, search, detailed details page communication, and message. Moreover, this article describes the specific implementation process of each functional module of the system in detail, and carries out a detailed analysis of each functional module through related technologies such as pictures and codes.

This system is running using the Windows 10 system. The main development platform is Webstorm. The front end of the system is written in HTML/CSS and JavaScript. The back end uses Node.js's classic framework Express, and the front and back ends are separated for easy management. The data storage is mainly the MongoDB database. The data management uses Robo data visualization management tools.

**Keywords: Supply and demand information website; Express; Node.js;**

目 录

[摘 要 I](#_Toc327651524)

[Abstract II](#_Toc327651525)

[第1章 绪论 1](#_Toc327651526)

[1.1课题的背景及意义 1](#_Toc327651527)

[1.2课题研究内容 3](#_Toc327651529)

[第2章 系统相关技术 4](#_Toc327651530)

[2.1 Struts框架技术 4](#_Toc327651531)

[2.1.1 Struts框架工作原理 4](#_Toc327651533)

[2.1.2 Struts框架优点 5](#_Toc327651534)

[2.2 Ajax技术 5](#_Toc327651535)

[2.2.1 Ajax技术开发模式 6](#_Toc327651537)

[2.2.2 Ajax技术优点 6](#_Toc327651538)

[2.3 FreeMarker技术 7](#_Toc327651535)

[2.4 系统环境 7](#_Toc327651535)

[第3章 系统需求分析 8](#_Toc327651539)

[3.1系统功能需求概述 8](#_Toc327651540)

[3.2 系统可行性分析 8](#_Toc327651544)

[3.3 用例模型 9](#_Toc327651545)

[3.3.1 员工用例 9](#_Toc327651537)

[3.3.2 管理员用例 1](#_Toc327651538)0

[3.4 类对象模型 1](#_Toc327651546)1

[3.5 交互模型 1](#_Toc327651547)2

[3.6 用户界面 1](#_Toc327651548)3

[第4章 系统总体设计 1](#_Toc327651549)5

[4.1 系统流程概述 1](#_Toc327651550)5

[4.2 系统总体模块功能设计 1](#_Toc327651554)5

[4.3 系统主要E-R图 1](#_Toc327651555)7

[4.4 系统数据库设计 1](#_Toc327651557)9

[4.4.1 数据库简介 1](#_Toc327651558)9

[4.4.2 物理结构设计 2](#_Toc327651559)0

[第5章 系统功能实现与测试 2](#_Toc327651561)5

[5.1 员工管理系统首页的设计与实现 2](#_Toc327651562)5

[5.2 系统前端功能的设计与实现 2](#_Toc327651563)7

[5.2.1 员工个人信息模块 2](#_Toc327651564)7

[5.2.2 员工意见反馈模块 2](#_Toc327651565)7

[5.2.3 员工信息管理模块 2](#_Toc327651565)8

[5.2.4 员工社区新闻模块 2](#_Toc327651565)9

[5.3 系统后端功能的设计与实现 3](#_Toc327651563)0

[5.3.1 管理员信息审批模块 30](#_Toc327651565)

[5.3.2 管理员新闻发布模块 31](#_Toc327651565)

[5.3.3 管理员意见管理模块 31](#_Toc327651565)

[5.3.4 管理员员工管理模块 32](#_Toc327651565)

[5.4 系统测试 38](#_Toc327651568)

[第6章 结论 30](#_Toc327651569)

[参考文献 41](#_Toc327651570)

[致 谢 42](#_Toc327651571)

# 第1章 绪论

## 1.1课题的背景及意义

随着现今社会信息技术的提高，计算机被广泛的应用于当今社会的各个领域，而它已经悄然之间成为了当今社会发展的首要技术动力。现今人们生活的各个方面都离不开互联网，互联网的方便性为人们提供了有力的条件。

从古至今在生活中人们都离不开两个字“管理”，管理层次有高有低，管理范围有大有小。对于学校应有良好的管理模式才会提高教育模式，提高学生的水平。而对于公司应该有针对性的管理模式，才会得到较高的效益与利润。对于传统的管理模式而言，需求的岗位过多，而且人与人之间的交接也会过于的繁琐，有时候反倒会增加了一些工作量，而且交接时也会出现一些不必要的错误。

本课题的研究内容为供求信息网，愿景就是提供一个信息交换的平台，让用户通过这个平台来发布自己的需求，旨在通过此系统来展示用户上传的所需的信息，并且对这些信息加以管理，让用户可以实时的查看这些信息，并且对这些信息进行筛选，找到自己心仪的并与信息发布者沟通，也可以在下方进行评论，评论后发布者会实时的收到这些回复并且为这些咨询者答疑解惑，也可以单独进行沟通，沟通方式是通过邮件进行，保证了信息的安全性和私密性，方便了用户的使用，该平台功能简单易用，用户也能很快上手来满足自己的需求。

通过本员工管理系统可从多样化的角度了解用户的状态，并且对用户对某一条信息点击量进行排序，让用户可以查看所有用户当前的需求，并且针对这些需求做出相应的改变，来为更多的用户服务。

## 1.2课题研究内容

本系统为供求信息网，整个系统分为数据接口层与前端业务层。一个系统最基础的就是应该具备一定的完整性，整体的流程要相对明晰，无论是前期开发还是后期维护都是至关重要的。剖析该系统，首先对于未注册的用户可以浏览这个平台的供求信息，查找自己所需的信息详情，但是无法上传自己的供求信息、无法与信息发布者进行交流、无法在信息详情页下方进行评论。但是如果用户在平台注册了个人信息，那么就可以使用邮箱和自己设置的密码来登录本平台，登录成功后用户可管理自己个人的相关信息，可以发布、管理自己的供求信息，并且也可以与信息发布者进行沟通、在具体的信息详情页下面进行评论。用户也可以及时的看到自己的评论在什么时间被回复，让用户的信息具有实时性和高效性。该平台为所有的用户提供了一个相对公平的环境，首页的热门商品全是根据点击量为用户实时推荐，坚持为用户提供最优的选择，查询也是根据用户所需展示最匹配的信息，一旦消息失效，信息发布者可以管理及时的下架该信息，十分的快捷方便,尽最大的努力为用户提供最好的服务体验。第2章 系统相关技术

## 2.1 Node.js相关技术

在近些年来各种编程语言层出不穷，开发不同的场景会根据场景的需求特性来选择合适的语言。而Node.js这门语言的特性就十分符合该平台的开发，Node.js语言简洁，是运行在服务器端的JavaScript，由于自身的异步特性，能够轻易的承担一些高并发场景，而且，Node.js有自己的包管理系统NPM，该系统能够方便的利用一些组件包来快速完成一些功能的开发，因此Node.js能够承担该项目的开发。

### 2.1.1 Node.js优点

Node.js 使用事件驱动模型，当web server接收到请求，就把它关闭然后进行处理，然后去服务下一个web请求。当这个请求完成，它被放回处理队列，当到达队列开头，这个结果被返回给用户。这个模型非常高效可扩展性非常强，因为webserver一直接受请求而不等待任何读写操作。（这也被称之为非阻塞式IO或者事件驱动IO）在事件驱动模型中，会生成一个主循环来监听事件，当检测到事件时触发回调函数。



1. 采用事件驱动、异步编程，为网络服务而设计。其实Javascript的匿名函数和闭包特性非常适合事件驱动、异步编程。而且JavaScript也简单易学，普及度很高，前端设计人员可以很快上手做后端设计。
2. Node.js非阻塞模式的IO处理给Node.js带来在相对低系统资源耗用下的高性能与出众的负载能力，非常适合用作依赖其它IO资源的中间层服务。
3. Node.js轻量高效，可以认为是数据密集型分布式部署环境下的实时应用系统的完美解决方案。Node非常适合如下情况：在响应客户端之前，预计可能有很高的流量，但所需的服务器端逻辑和处理不一定很多。

综合以上的原因，考虑采用该技术进行相关系统的开发。

### 2.1.2 Express框架的优点

Express是基于Node.js的 平台，快速、开放、极简的 web 开发框架。该框架把路由和中间件完美融合，通过中间件形式把业务逻辑细分，简化，一个请求进来经过一系列中间件处理后再响应给用户，将复杂的业务转变成线性的，清晰明了，并且提供一系列强大的特性，可以协助开发者创建各种 Web 和移动设备应用。丰富的 HTTP 快捷方法和任意排列组合的 Connect 中间件，使得当开发者创建健壮、友好的 API 时变得既快速又简单。该框架是在Node.js上扩展了 Web 应用所需的基本功能。

## 2.2 Ajax技术

Ajax 指异步 JavaScript 及 XML**，**Web应用的交互如Flickr,Backpack和Google就在该方向已经获得巨大的进步。在基于数据的使用过程中，使用者所需的数据例如联系人列表，能够单独的被填写到网页中而与本身的网页各自独立 ，使缓慢的Web应用起来给人一种类似于桌面软件的感觉。

Ajax的主要内容为XmlHtpRequest。该名词第一次于Internet Explroer 5中被提出，这是能够实现几个命令不同步进行的技术。简单来说，XmlHtpRequest能够让使用者对通过JavaScript向终端提出命令得到相应，同时又不影响其他指令。

### 2.2.1 Ajax技术开发模式

大多数关键的技术与Ajax的研发方式能够从已有的技术中取得。比如，当一个指令被送到服务端的过程中，率先考虑的是指令的运送顺序，哪个指令最先被送达，如果指令在规定时间内没有送达的处理方式以及该指令不正确的情况下的处理方式。以上这些条件有很多被Web服务涵盖。随着技术的不断深入发展更多的部分将会被完善。

与普通的计算机科学开发相比Ajax开发有着巨大的区别。这些区别导致了一系列难题的出现，其中最主要的难题在于该技术是否普遍适用于与大多数情况。因为该技术依靠于浏览器，因此该技术能否在浏览器中正常使用成为该系统在开发过程中最根本的难题。

根据每种不同的技术与关联程度极强的用户端的环境，Ajax也出现了一种独创的开发模式。Ajax的技术开发者首先要理解传统的MVC的结构，这样就导致应用层次之间的边界被限制了，与其同时，开发者们还应考虑CS环境的外部以及利用Ajax技术去控制MVC边界。Ajax开发者在考虑Web应用时必须要将其设想为单个页面去开发绝对不能用页面集合的方式去思考。

### 2.2.2 Ajax技术的优点

Ajax技术有更快的反应速度。在固有的Web应用中使用者可以写入表格，填写的表格被提交后就向该终端发送该指令。终端收到指令后对指令进行处理，表格被处理后便跳入一个新的页面。但是因为提交表格的两个页面的前后，很多的html代码都是相同的，因此使用该方法浪费的较多的流量。因为每次这种指令的传递都向终端发出指令，指令的响应速度就等同于终端的响应速度，造成了该应用的响应速度照比与本地应用的速度迟缓。

和传统的Web应有不同，Ajax应用在向终端发送命令后仅仅需要得到重要的数据，它利用SOAP与另外的使用XML的Web service接口，且在客户端利用JavaSroipt来处理终端发来的响应。而且大多数的处理都能够于输送该指令的客户端的设备上进行，因此这大大增快了Web终端的响应时间。

Ajax技术相比于其他技术主要的优点在于，在对数据进行维护的时候，可以确保所有的页面都不会被刷新。因为页面不会被反复更新，所以Web上的应用程序能够对用户发出的指令实现快速的响应，同时对于网络上已经存在而且没有被变更的数据也不需要再次的上传了。

## 2.4 系统环境

硬件环境:

服务器 CPU：PIII 500以上 ，内存：512M以上。

客户机 CPU：P200MMX以上，内存：32M以上。

软件环境：

客户端： Windows XP 以上，Internet Explorer(IE)等。

服务器端：Windows NT/Windows2000 以上，Tomcat 5.0,JDK 1.5及其

以上版本，IE等；

数据库：采用MongoDB，运行于服务器端

# 第3章 系统需求分析

需求分析在软件开发流程起着至关重要的作用。而对于软件程序起着决定性的指导也正是需求分析。同时它也是软件开发进度的保证。进行需求分析时，如果分析的人没有准确的意识到顾客的真实所需，那么开发者把软件做的如何完善，顾客也是不会接受的。而或没有合理的需求分析，在软件的完成时间上也会有大大的折扣，因此可见需求分析的重要性。一般需求分析会用到用例模型、类对象模型、交互模型进行程序分析。

## 3.1系统功能需求概述

本系统为供求信息网，实际上也就是对用户提供的供求信息加以管理并展示，而对于信息管理系统则适用于信息与事物的管理，其中包括信息处理等多项技术。一个稍微完整的系统则需要具有增加、删除、更新、查询等基本操作，也需具有一定的规则性、安全性和完整性等。经过调研，基本系统应具有以下几方面：

1. 要以用户的要求为前提形成操作便易，视觉美观的人机界面。
2. 要时刻保证使用者的信息安全，配备完整的权限管理。
3. 为避免人工数据计算出现不必要的偏差，系统应将本部分自动完成。
4. 能支持多种条件对数据进行修改，查询，删除等，而且还要使操作过程

简单，具有一定的稳定性。

## 3.2 系统可行性分析

可对于系统的可行性分析，可行性研究过程中的首要任务是首先要清楚的明确下来客户的要求，并要紧密结合现实的环境与状况。与其同时要在社会、经济还有技术等这三个主要方面验证和研究本系统的可行性，还要编写出可行性研究报告，制定初步项目开发计划，而可行性分析的目的则是判断一个系统是否值得去开发，如果可开发怎样以最少的人力物力来实现。

技术可行性分析：本系统为基于Node.js的供求信息网站的实现，开发使用的主要是Node.js语言，而且本系统使用了Express框架，使用的技术中包含了JQuery和Ajax等，数据的存储坏境为MongoDB数据库。前后端分离，减轻了服务器端的压力，也让前端充分发挥自己的功能，能让职责清晰、也方便协作，会大大提高开发效率。开发的坏境为Webstorm，前端与后端开发都可以基于该平台进行，简单高效。其中也采用了模块化的设计方法，不仅方便组合和修改系统的各个功能，也便利了后台开发的技术人员对系统进行二次的补充与维护。

社会可行性分析：对于现今科技化的社会，互联网技术日益强大，计算机的使用也越来越广泛，对于现今大部分的工作，计算机的使用以及计算机知识的掌握都是必要的，而本系统的整体设计简单直观、操作便易。很容易被人们接受。并且使用本系统的初衷就是减少用户满足需求这个过程的时间，在最短的时间尽最大努力为用户提供最优质的服务，因此本系统在社会方面是完全可行的。

经济可行性分析：该平台与日常获取信息的方式相比，大大节约了用户的时间，来让用户有充分的时间去忙其他的事情，对社会对企业都是有极大用处的，因此在构建这样一个供求信息网站在经济方面是完全可行的。

## 3.3 用例模型

用例模型能够使开发人员可以以一种最直观的用图形的方式来了解软件程序。以这种方式可以使开发人员对整个程序系统有更加清晰和多元化的了解，更加方便开发人员对整个系统的业务流程进行解析。图形与文字起到一个互补的作用，两者之间的结合描述的更加清楚，明晰的将系统的开发任务显现出来。从宏观的角度对整个系统有了更加透彻和分明的认知。

### 3.3.1 用户用例

(1) 参与者：用户

(2) 用例描述

注册：用户可以前往注册页面注册该平台账号，以便使用该平台下的功能。

登录：用户可以使用在该平台注册的账号来登录该平台，享受该平台的所有服务。

浏览信息：凡是访问该网站的人都可以浏览该网站的供求信息。

查找信息：凡是访问该网站的人都可以前往分类页面查找自己所需的供求信息。

个人信息：包括个人信息的更改、上传个人信息、删除个人信息，其中个人信息更改可以更改用户在该平台账号的登录密码，上传个人信息是指用户可以根据自己的所需上传供求信息，删除个人信息是指对自己上传的供求信息加以管理，在这些信息失效时及时下架，保证平台信息的有效性、真实性。

沟通用户：包括留言评论和邮件通知，当用户浏览信息详情时，可以在页面下方的留言区写下自己想说的话，这些留言也会被其他用户看到，其他用户也可以对这些留言进行评论，信息提醒是当其他用户给你留言或者回复你的留言时会把这些消息及时的让用户知道，邮件通知是指用户在浏览信息时，想要与信息发布者单独沟通，这时系统会协助信息发布者与用户建立联系。

## 

## 图3-1员工用例图

## 3.4 用户界面

本系统面向的群体主要是平时使用互联网的用户，考虑到这些人日常时间比较紧迫，所以在页面设计上加入了导航功能，能让用户迅速到达自己想访问的页面，页面风格也比较简单大方，能让用户迅速上手使用并且了解平台的作用，总的来说，页面功能罗列清楚，方便用户使用。

# 第4章 系统总体设计

## 4.1 系统流程概述

对于普通的游客，在进入该平台后，可以浏览该平台所有的供求信息，也可以根据自己的需求查找具体的信息，游客也可以前去注册页面注册该平台的账号来使用该平台，已有账号的用户可以登录来使用该平台，登录通过后可以查看自己的个人信息，更改自己的登录密码，也可以根据需求来发布自己的供求信息，发布成功后其他用户可以浏览并留言，当用户希望与你深入交流时系统会发送邮件协助你们建立联系，用户也可以管理自己发布的供求信息，及时下架那些失效的信息，同样，用户可以在信息详情页下与感兴趣的信息的发布人联系，或者在页面下方留言，与发布人建立沟通。

## 4.2系统总体模块功能设计

本系统为供求信息网，整个平台整体上分为四大功能模块。分为信息查看模块、登录注册模块、信息管理模块、用户交流模块，这四个大模块下还各分为几个小模块，每个模块之间看似相互独立，但是在数据库的内部却又是紧密相连的。根据以上所述，具体每个小模块的实际功能如下所示：

1. 信息查看模块
2. 信息浏览

用户可以查看该平台的所有供求信息以及信息详情。

1. 信息查找

用户可以根据自己的需求查询自己想要的信息。

1. 登录注册模块
2. 用户注册

用户可以前往指定页面注册该平台的账户，以便享受该平台的所有服务。

1. 用户登录

用户可以根据自己在平台注册的账号来登录该平台，来享受该平台的一系列服务。

1. 信息管理模块
2. 个人信息模块

用户在登录该平台后管理自己的账号登录密码，进行重置，来保证一定的安全性。

1. 供求信息发布

用户可以根据自己的需求发布具体的信息,发布后信息将会被放到平台，供其他人浏览。

1. 供求信息管理

用户可以管理自己发布的供求信息，根据需要及时下架无效的信息，保证信息的真实性、有效性。

1. 用户交流模块
2. 信息详情页留言

已经登录的用户可以在信息详情页下方，输入自己的态度、感受或者是问题，来让其他用户看到、分享这些内容。

1. 信息提醒

信息提醒就是已经登录的用户会在个人账户下的与我相关的页面下看到其他用户的留言或者回复。

1. 信息沟通

当信息得到用户的青睐时，可以与信息发布者进行沟通，这时系统会发送邮件提醒信息发布者，协助用户之间的沟通。

系统模块结构总图如图4-1所示。

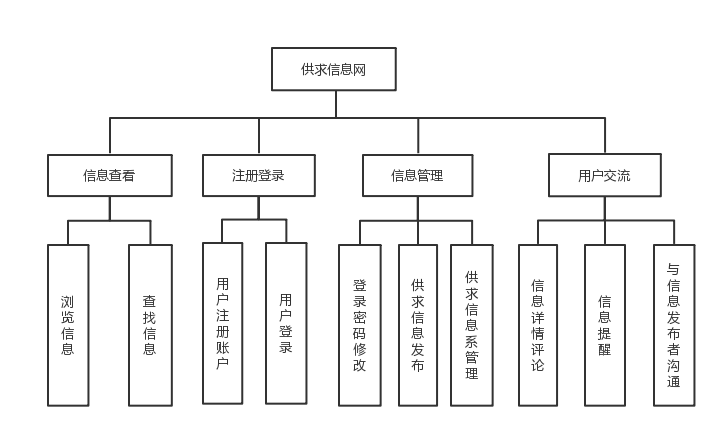


图4-1 系统功能模块图

## 4.3 系统主要E-R图

E-R图又称为实体联系图，以一种概念模型的方式来描绘现实世界。E-R图也能够作为一种简单的关系模型。其中的每一个实体就相当于一张表，每一个实体或者是一张表都可能包含几个实体间的关系，使用主键和外键来关联关系表以及实体表。客观存在并且可以相互之间区别的事物称为实体，实体具有一个以上的属性，而实体与实体之间的联系又称为关系，清晰的了解了以上三点才能够准确描绘E-R图

该平台信息是整个平台的关键，也相当于线索贯穿该平台。

该平台信息设计的E-R模型如图4-2所示。

## 

图4-2 供求信息模型

## 4.4 系统数据库设计

### 4.4.1 数据库简介

本系统采用的是MongoDB数据库，对于MongoDB具有可用性强、可扩展性强、实用性强、高效易用的优点，具体特点如下：

(1) MongoDB是一个面向文档的数据库，它并不是关系型数据库，直接存取BSON，这意味着MongoDB更加灵活，因为可以在文档中直接插入数组之类的复杂数据类型，并且文档的key和value不是固定的数据类型和大小，所以开发者在使用MongoDB时无须预定义关系型数据库中的”表”等数据库对象，设计数据库将变得非常方便，可以大大地提升开发进度。 (2) MongoDB在高可用和读负载均衡上的实现非常简洁和友好，MongoDB自带了副本集的概念，通过设计适合自己业务的副本集和驱动程序，可以非常有效和方便地实现高可用，读负载均衡。而在其他数据库产品中想实现以上功能，往往需要额外安装复杂的中间件，大大提升了系统复杂度，故障排查难度和运维成本。

(3) 在扩展性方面，假设应用数据增长非常迅猛的话，通过不断地添加磁盘容量和内存容量往往是不现实的，而手工的分库分表又会带来非常繁重的工作量和技术复杂度。在扩展性上，MongoDB有非常有效的，现成的解决方案。通过自带的Mongos集群，只需要在适当的时候继续添加Mongo分片，就可以实现程序段自动水平扩展和路由，一方面缓解单个节点的读写压力，另外一方面可有效地均衡磁盘容量的使用情况。整个mongos集群对应用层完全透明，并可完美地做到各个Mongos集群组件的高可用性。

### 4.4.2 物理结构设计

本系统使用的是MongoDB数据库，在本数据库包含了三张员工管理系统的表，主要数据库相关表如下：

1. 用户信息表(USER\_TABLE)：本表保存了用户的相关信息，如表4-1所示。

表4-1 用户信息表

|  |  |  |
| --- | --- | --- |
| 字段名 | 备注 | 类型 |
| User.name | 用户账号名称 | String |
| User.\_password | 用户账号的密码 | String |

1. 信息详情表(INFORMATION\_TABLE)：本表保存了供求信息的详细细节，如表4-2所示。

表4-2 信息详情表

|  |  |  |
| --- | --- | --- |
| 字段名 | 备注 | 类型 |
| Information.name | 信息详情名称 | String |
| Information.email | 信息联系邮箱 | String |
| Information.txtarea | 信息详情说明 | String |
| Information.user | 信息发布者 | String |
| Information.imgsrc | 信息详情图片地址 | String |
| Information.hot | 信息热度 | Number |
| Information.type | 信息类型 | String |
| Information.contentType | 信息内容类型 | String |

1. 信息详情评论表(COMMENT\_RECREATION\_TABLE)：本表保存了具体信息下方评论的相关信息，其中留言的id就是信息详情的ID，如表4-3所示。

表4-3信息详情评论表

|  |  |  |
| --- | --- | --- |
| 字段名 | 备注 | 类型 |
| Comment.id | 留言所属的页面ID | String |
| Comment.comment | 留言信息 | Array |

1. 用户消息表(NEWS\_TABLE)：本表保存了具体消息的相关信息，如表4-3所示。

表4-4信息详情评论表

|  |  |  |
| --- | --- | --- |
| 字段名 | 备注 | 类型 |
| news. customer | 消息的发出人 | String |
| news.content | 消息的内容 | String |
| news.time | 消息发出的时间 | String |
| news.goodId | 消息对应的信息ID | String |
| news.user | 消息的接收人 | String |

**第5章 系统功能实现与测试**

## 5.1系统首页的设计与实现

本界面为网站平台首页，因为网站面向的群体是喜欢互联网的年轻人，所以风格比较随意、简单。首页作为平台的门面，一般十分重要，在设计风格上比较简单大方，目的就是让使用者能够快速使用。该平台首页首先用了banner图，告诉访问者这是一个怎样的平台，可以用来干什么，还有欢迎的小表情在期待用户的加入。首页中包含了一般系统的须有基础功能：用户登录、注册。首页中间是平台根据点击量选出的热门信息，更容易让用户了解到其他用户的需求。最后是一个导航，可以让熟悉该平台的人快速到达想去的页面，也有网站平台负责人的信息，平台出现问题用户可以随时沟通。最后的尾部该平台的每个页面都有，方便用户随时访问，为用户提供最快捷的服务。

首页界面如图5-1所示。

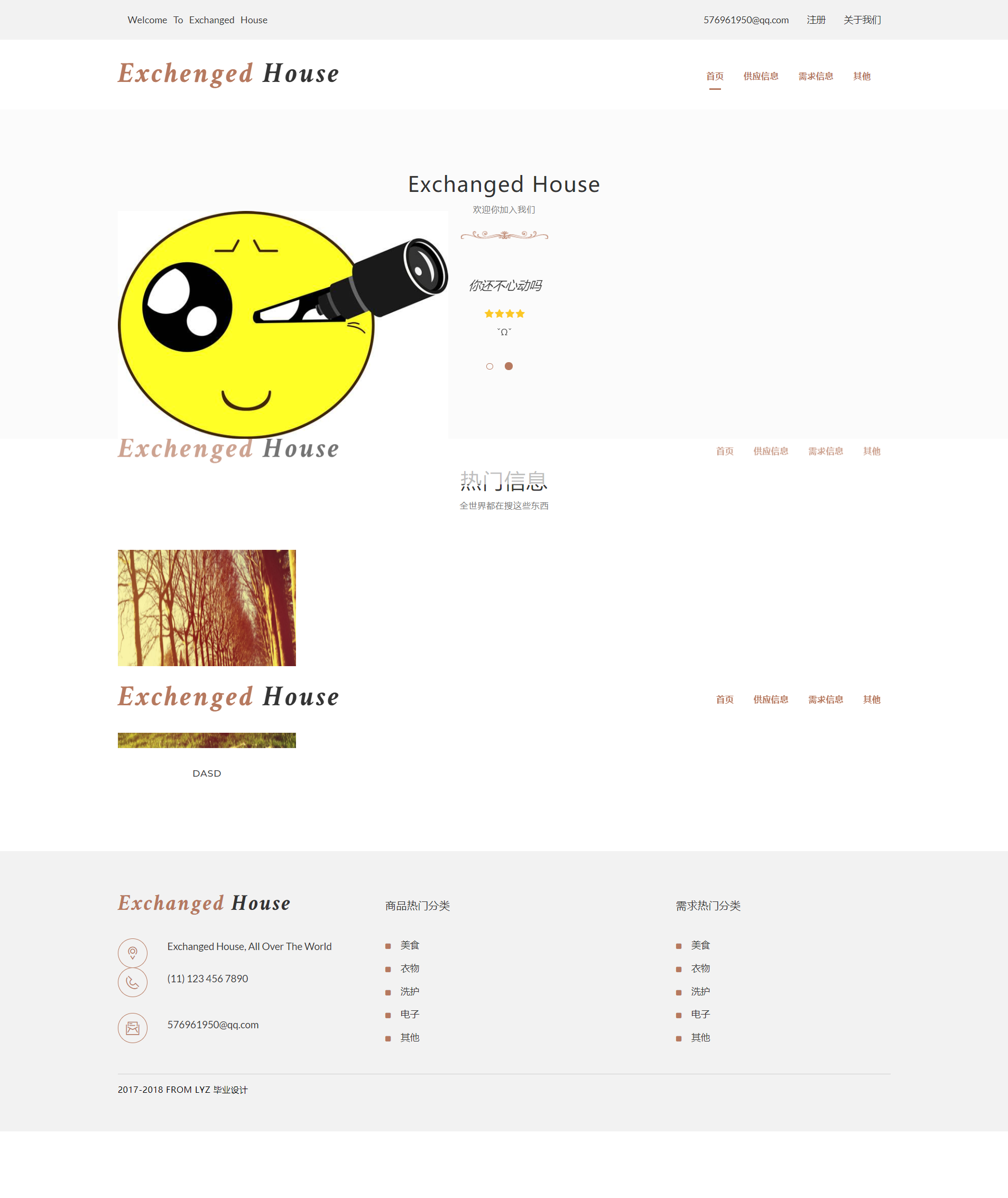


图5-1 系统首页

## 5.2 注册登录功能的设计与实现

对于第一次进入该平台的人来说，用户需要在该平台注册才可以享受到该平台的所有服务，前端注册使用了Ajax技术来检查用户输入的用户名是否被重复注册。同样，注册成功后登录经过验证才能访问其他功能，登录也是用了该技术来进行自动登录，将账户信息加密存储在用户本地的Cookie中，当下次访问该平台时，自动获取Cookie并验证来实现自动登录。登录、注册相关页面如图5-2,5-3所示。



图5-2 登录页面

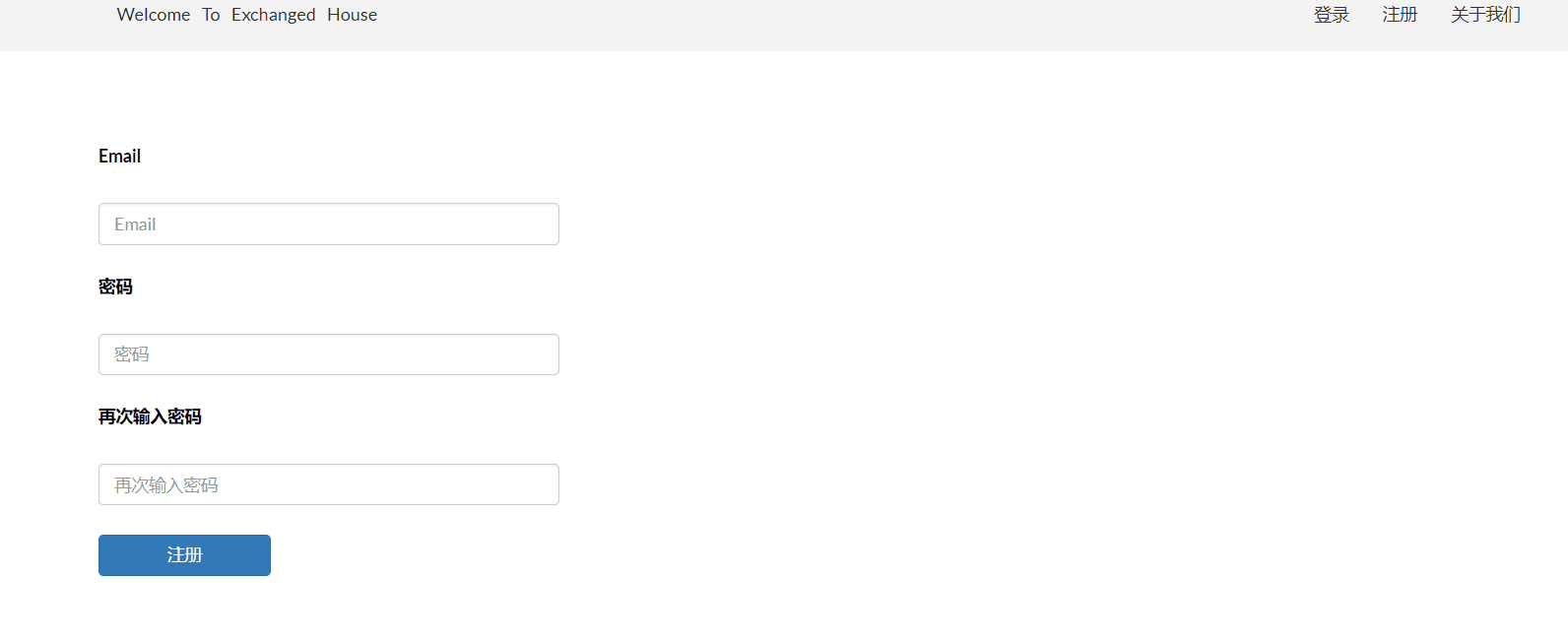


图5-4注册页面

注册页面主要代码：

|  |
| --- |
|  |
|  | function () { |
|  | $.get("http://127.0.0.1:9000/usercheck/"+email.value,{},function (data) { |
|  | if (data.length>0){ |
|  | alert("邮箱已被注册"); |
|  | email.value="" |
|  | } |
|  | }) |
|  | }  自动登录实现代码：  function() {   |  |  | | --- | --- | |  | var arr = document.cookie.split("; "); | |  | var user,password; | |  | for(var i=0,len=arr.length;i<len;i++){ | |  | var item = arr[i].split("="); | |  | if(item[0]=="user"){ | |  | user=item[1]; | |  | } | |  | if(item[0]=="password"){ | |  | password=item[1]; | |  | } | |  | } | |  | if(user && password){ | |  | $.post("http://127.0.0.1:9000/login",{name:user,password:password},function (data) { | |  | if(data.length==0){ | |  | alert("账号密码错误"); | |  | }else { | |  | document.getElementById("modal-756894").href="/user/"+data[0].\_id; | |  | document.getElementById("modal-756894").innerHTML=data[0].name; | |  | document.cookie = 'user='+user; | |  | document.cookie = 'password='+password; | |  | } | |  | }) | |  | } | |  |  | |  | } | |

## 5.2 个人信息管理功能的设计与实现

5.2.1修改个人信息模块

该模块是针对已经注册的用户，用户在登录该平台后可以查看管理个人信息，页面设计以表单为主，左侧是导航条，用户可以通过导航信息进入自己想访问的页面，页面内容交互是用的是ajax技术，并且充分利用Cookie中存储的用户信息来进行验证。用户可以在表单中输入内容来与系统进行交互。综合来说，页面设计应该清晰明朗，让用户更快的了解该页面的功能以及使用，页面风格也简单大方。用户修改个人信息页面如图5-2-1所示。

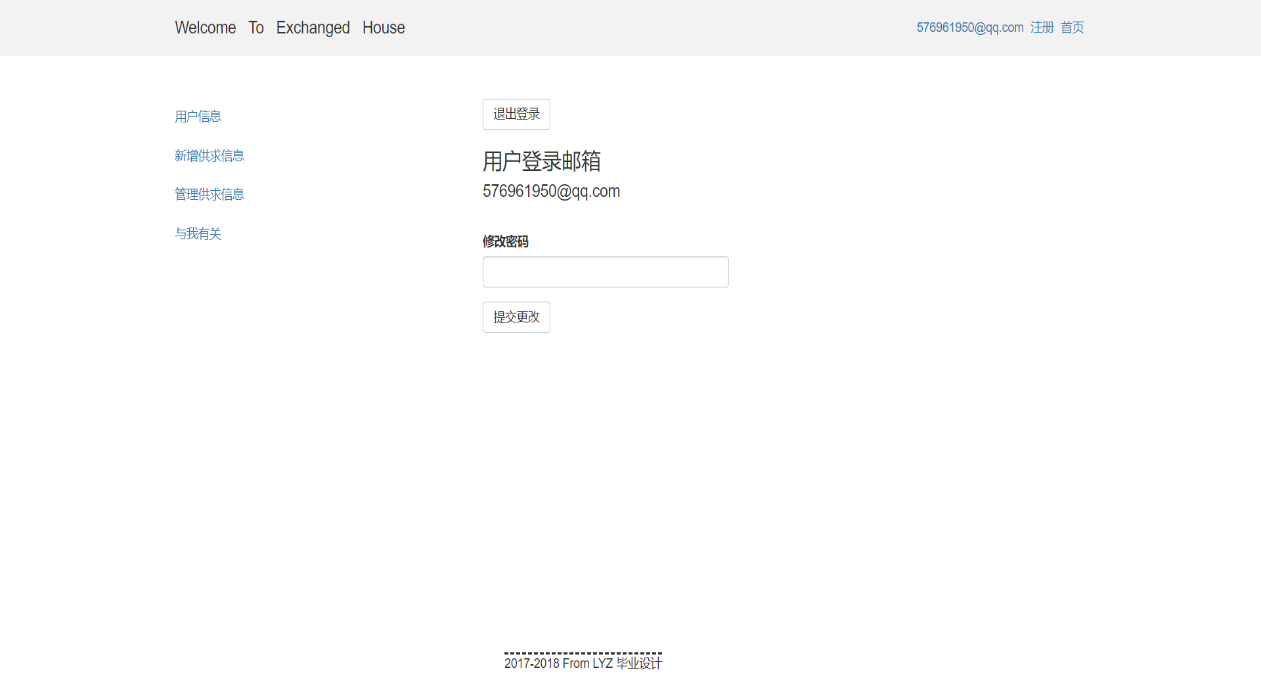


图5-2-1 个人信息页面

页面主要代码：

var name=document.getElementById("modal-756894").innerHTML;

var password=document.getElementById("exampleInputPassword1").value;

var id=document.getElementById("viewnone").innerHTML;

$.post("http://127.0.0.1:9000/user/"+id,{name:name,password:hex\_md5(password)},function (data) {

if(data.ok){

alert("密码修改成功");

};

})

5.2.2 用户上传供求信息模块

在本模块中，用户可以上传自己的供求信息，把信息的图片上传到服务器，并且选择信息的分类，把信息完整的上传到该平台，来向该平台的其他用户展示。

文件上传的实现方式引用了Node的formidable包，并且限制了文件大小是10M，来限制用户的上传大小，上传是把这些图片上传到服务器端的一个附属文件夹，当上传成功后会返回存放图片的地址，服务器端收到地址后会把用户的上传信息和地址放在一起，命名方式是获取此时此刻用户上传的时间加上四位随机数，时间精确到秒，来保证用户上传图片地址的唯一性。页面显示如图5-6所示。

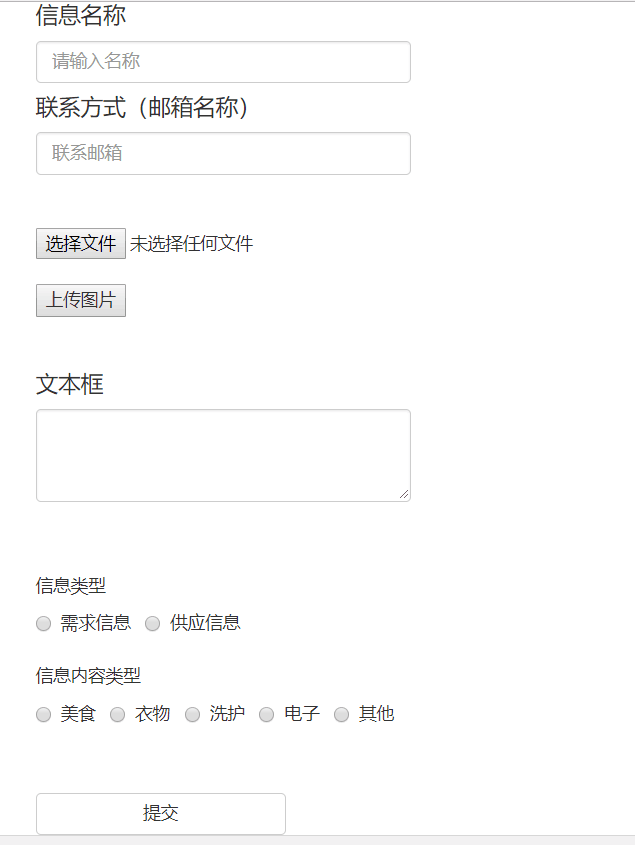


图5-2-2 用户上传供求信息页面

上传供求信息页面主要代码：

exports.uploadImg =function (req,res) {  
 var form = new formidable.IncomingForm(); //创建上传表单  
 form.encoding = 'utf-8'; //设置编辑  
 form.keepExtensions = true; //保留后缀  
 form.uploadDir=path.resolve(\_\_dirname,'../public');  
 form.maxFieldsSize = 2 \* 1024 \* 1024; //文件大小  
 let newname=moment().format("MM-DD-h-mm-ss");  
 let extname = path.extname(req.files.files.name);  
 var newPath =form.uploadDir +'/img/'+newname+extname;  
 console.log(newPath);  
 let readable = fs.createReadStream(req.files.files.path);  
 let writable = fs.createWriteStream(newPath);  
 readable.pipe(writable);  
 res.send({  
 status:1,  
 path:newPath  
 });  
}

5.2.3 用户管理供求信息模块

在该模块中，用户可以管理自己上传的供求信息，可以实时的把信息进行下架，保证了信息的真实性、有效性。页面显示用户上传的供求信息的名称、联系人等，方便用户管理自己的上传信息。管理供求信息模块页面如图5-7、5-8所示。



图5-2-1 用户管理信息页面

用户管理信息页面主要代码：

<table class="table">  
 <caption>管理我的信息</caption>  
 <thead>  
 <tr>  
 <th>商品名称</th>  
 <th>联系邮箱</th>  
 <th>操作</th>  
 </tr>  
 </thead>  
 <tbody>  
 <% for(var i=0;i<information.length;i++){ %>  
 <tr>  
 <td><%= information[i].name%></td>  
 <td><%= information[i].user%></td>  
 <td><a target="\_blank" class="playitbtn tryitbtnsyntax" id="http://127.0.0.1:9000/userDelinformation/<%= information[i].\_id%>" onclick="torefresh(this)">信息下架</a></td>  
 </tr>  
 <% } %>  
   
 </tbody>  
</table>

**5.3 信息分类和详情显示功能的设计与实现**

5.3.1 信息分类模块

在此模块中，服务端根据用户上传的供求信息的内容进行分类，然后根据所分的类别进行展示，方便用户能在简短的时间内找到自己所需的信息，内容以分页的形式展示，每页展示固定的九个供求信息，其它依次向后分页，方便用户浏览和查询。查询采用的是模糊查询，只要用户输入的内容大致符合，就会从数据库中筛选出来，方便了用户的使用。如图5-3所示。

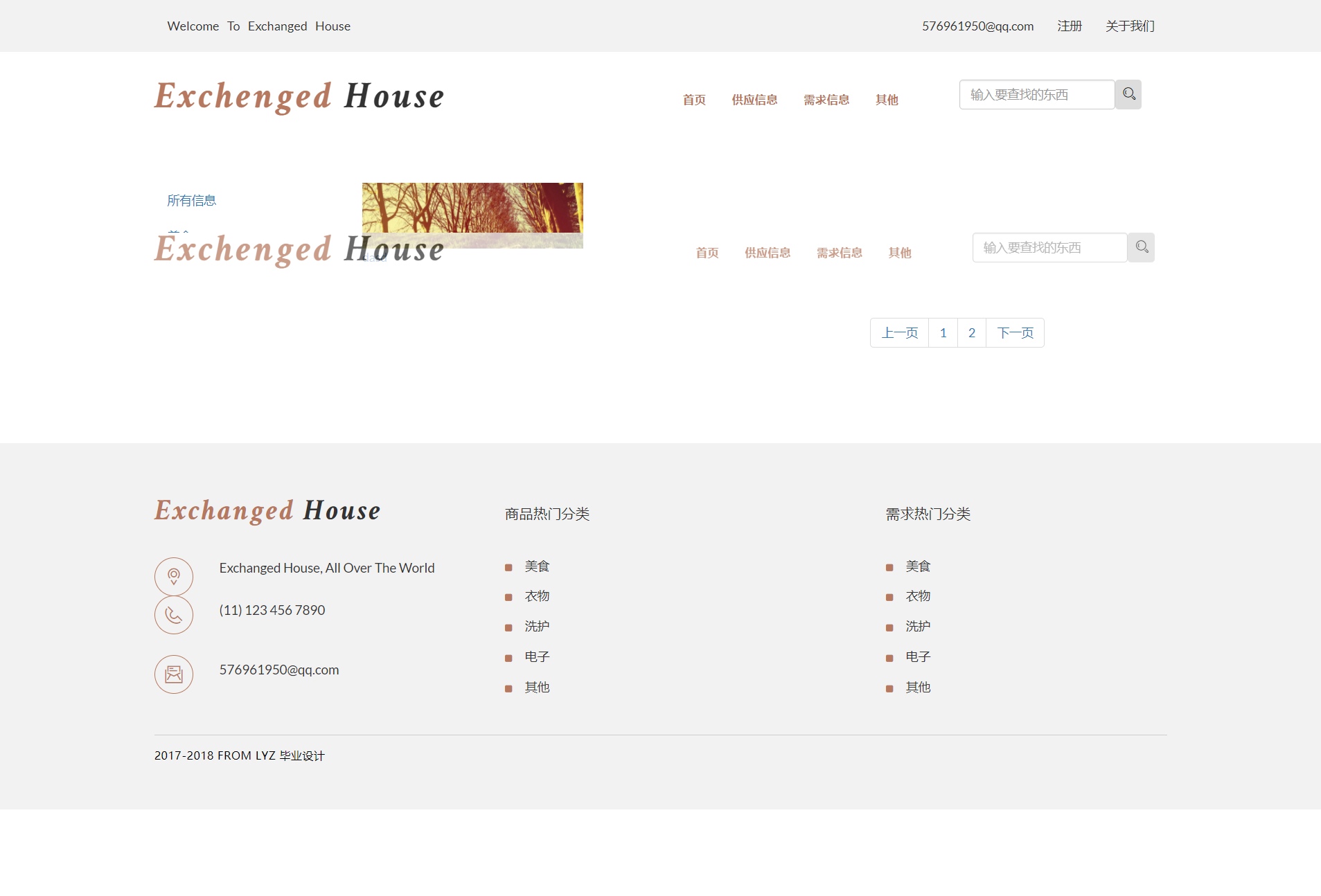


图5-10 人员审批页面

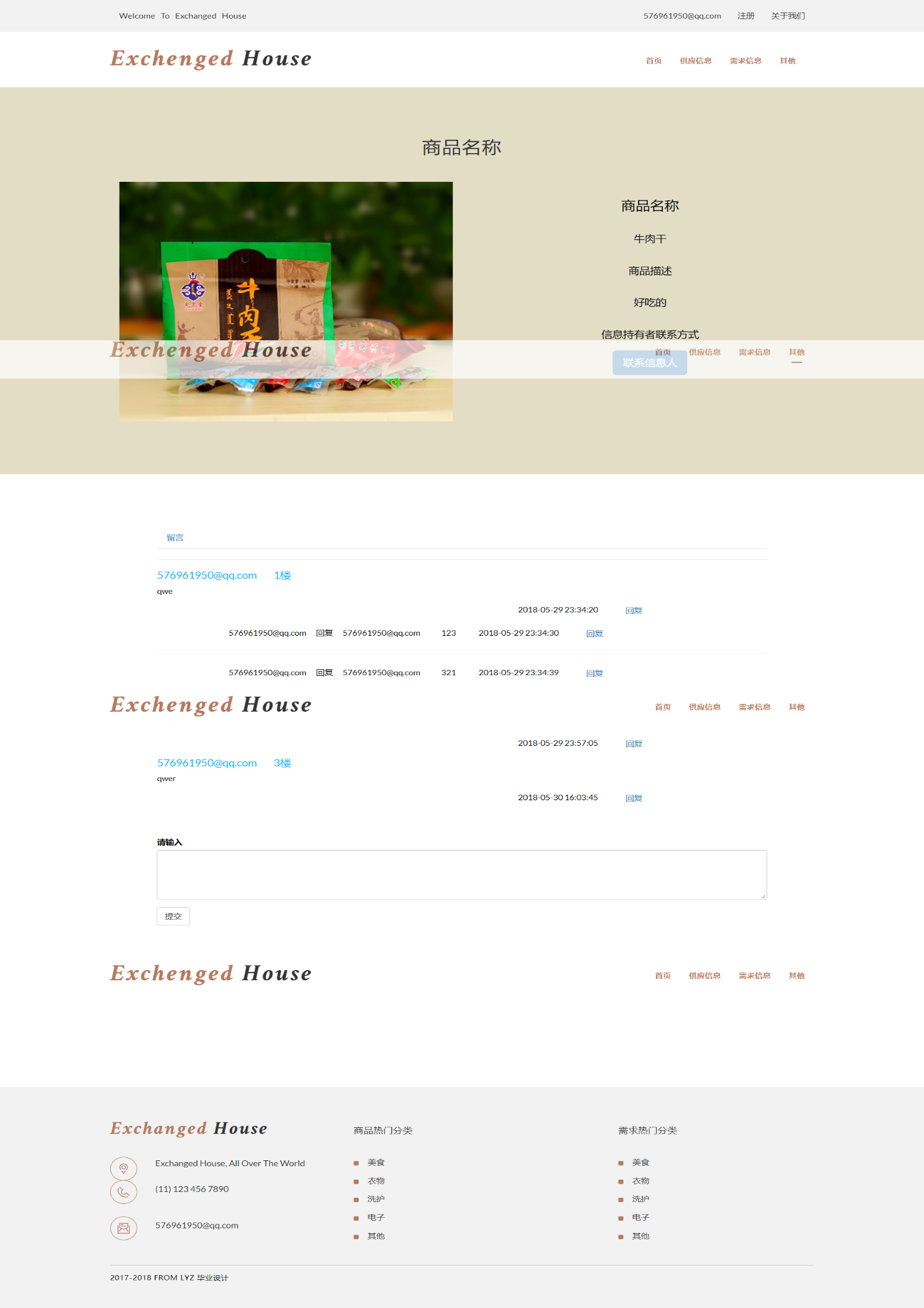
信息分类模块主要代码：

<ul class="row best-room\_ul" style="list-style: none">  
 <% for(let i=0;i<information.length;i++){%>  
 <li class="col-lg-4 col-md-4 col-sm-6 col-xs-12 best-room\_li">  
 <div class="best-room\_img">  
 <a href="#"><img src="<%= information[i].imgsrc%>" style="width: 255px;height: 177px;"></a>  
 </div>  
 <div class="best-room-info">  
 <div class="best-room\_t"><a href="/goodsId/<%=information[i].\_id %>"><%=information[i].name %></a></div>  
 <div class="best-room\_desc"><%= information[i].txetara%>  
 <p>&nbsp;</p>  
 </div>  
 </div>  
 </li>  
 <% }%>  
</ul >

5.3.2 信息详情显示模块

在本模块中，该平台把用户上传的信息实时的展示出来，在信息详情页，用户会看到有关该信息的所有相关信息，包括发布人、联系方式、以及发布的详细内容，在该页面的下方有用户评论区，用户可以看到其他用户在这的评论，用户也可以在下方发表自己的评论或者给信息发布人留言。

实现方式充分利用了Node.js的异步特性，async.parallel()该函数把放入里面执行的函数并行的执行并返回有关信息，大大减少的了串行调用所花费的时间。首先详情页展示的有用户上传的供求信息详情和有关该信息下方用户的评论详情，这两部分信息存储在不同的数据库表中，当获取到用户点击的供求信息ID后，这两部分就会调用相应的后端接口，获取到数据后返回给前端，来完成页面的展示。信息详情显示模块如图所示



信息详情显示模块关键代码：

async.parallel({  
 information: (callback) => {  
 Http.get("http://127.0.0.1:9000/information/" + goodsId, {}, (err, result) => {  
 if (result == null || result.length == 0) {  
 callback(null,null)  
 } else {  
 result[0].imgsrc = result[0].imgsrc.replace("G:\\graduate\\gradu\\API\\", "http://localhost:9000/");  
 callback(null,result[0])  
 }  
  
 })  
 },  
 comment1: (callback) => {  
 Http.get("http://127.0.0.1:9000/comment/" + goodsId, {}, (err, result) => {  
 if (result == null || result.length == 0) {  
 } else {  
 if(result ){  
 result = result.comment  
 }else {  
 result = []  
 }  
 }  
 callback(null,result)  
 })  
 },  
}, function(err, results) {  
 console.log(results)  
 res.render('specific', { result:results});  
});

## 5.4用户沟通功能的设计与实现

5.4.1 评论以及回复模块

在该模块中，用户可以在信息详情页下方的评论区进行评论或者回复某一条回复，所发表的言论对其他用户是可见的，下方评论中会显示该评论的内容、回复的人、被回复的人以及评论发布的时间。方便了用户之间的沟通，保证了信息的实时传递。

评论功能的实现方式是获取到cookie中当前的用户名以及页面展示的供求信息、信息ID、信息发布人等，然后把这些信息以及用户的评论一起传递到后端至数据处理层，数据处理层会把评论存放到一个数组中，每当有新的评论时就会向数组中依次存入一条新的评论。回复功能是当用户回复某条评论后，前端会收集所需的数据传递到后端，后端会根据回复的具体位置在其中插入这条评论。

评论功能主要代码：

function rere(user,i,j){  
 document.getElementById('reReplypeople').innerHTML = user;  
 document.getElementById('temp').innerHTML = i;  
 document.getElementById('temp2').innerHTML = j;  
  
}  
function rereend() {  
 var user=document.getElementById('reReplypeople').innerHTML;  
 var from=document.cookie;  
 var username=document.cookie.split(";"); //.split("=")[1]  
 for(var i=0 ;i<username.length;i++){  
 var temp=username[i].split("=");  
 if(temp[0]==' user'){  
 from=temp[1]  
 }  
 }  
 var replycontent = document.getElementById('rerecontent').value;  
 var high = document.getElementById('temp').innerHTML;  
 var high2 = document.getElementById('temp2').innerHTML;  
 var goodsId = document.getElementById("goodsId").innerHTML;  
  
 $.post("http://127.0.0.1:9000/updatecomment",{goodsId:goodsId,from:from,to:user,content:replycontent,high:high,high2:high2,big:1},function (data) {  
 console.log(data)  
 })  
}

var comment = document.getElementById("comment").value;  
 var to=document.getElementById("toEmail").innerHTML;  
 var from=document.cookie;  
 var username=document.cookie.split(";"); //.split("=")[1]  
 for(var i=0 ;i<username.length;i++){  
 var temp=username[i].split("=");  
 if(temp[0]==' user'){  
 from=temp[1]  
 }  
 }  
 if (from == ""){  
 alert("请登录")  
}  
var goodsId = document.getElementById("goodsId").innerHTML;  
 $.post("http://127.0.0.1:9000/comment",{goodsId:goodsId,user:from,comment:comment,havepeople:to},function (data) {  
  
 })  
 window.location.href="/goodsId/"+goodsId  
 }

服务端代码：

add: async (req, res, next) => {  
 try {  
 let comment = req.body;  
 if(comment.user == ""){  
 comment.user = "匿名"  
 }  
 let result = await commentDao.get\_by\_id(comment.goodsId);  
 var time = moment().format("YYYY-MM-DD HH:mm:ss");  
 if (result && result.comment.length >0) {  
 result.comment.push({  
 user : comment.user,  
 comments : comment.comment,  
 time : time,  
 others : []  
 });  
 let result1 = await commentDao.update(result);  
 let temp = await newsDao.add({  
 customer : comment.user,  
 content : comment.comment,  
 time : time ,  
 goodsId : comment.goodsId,  
 user:comment.havepeople,  
 });  
 res.json(result1)  
 }else {  
 let result1 = await commentDao.add({goodsId:comment.goodsId,comment:[{  
 user:comment.user,comments:comment.comment ,time:time, others : []  
 }]});  
 let temp = await newsDao.add({  
 customer : comment.user,  
 content : comment.comment,  
 time : time ,  
 goodsId : comment.goodsId,  
 user:comment.havepeople ,  
 });  
 res.json(result1)  
 }  
 } catch (err) {  
 next(err)  
 }  
},

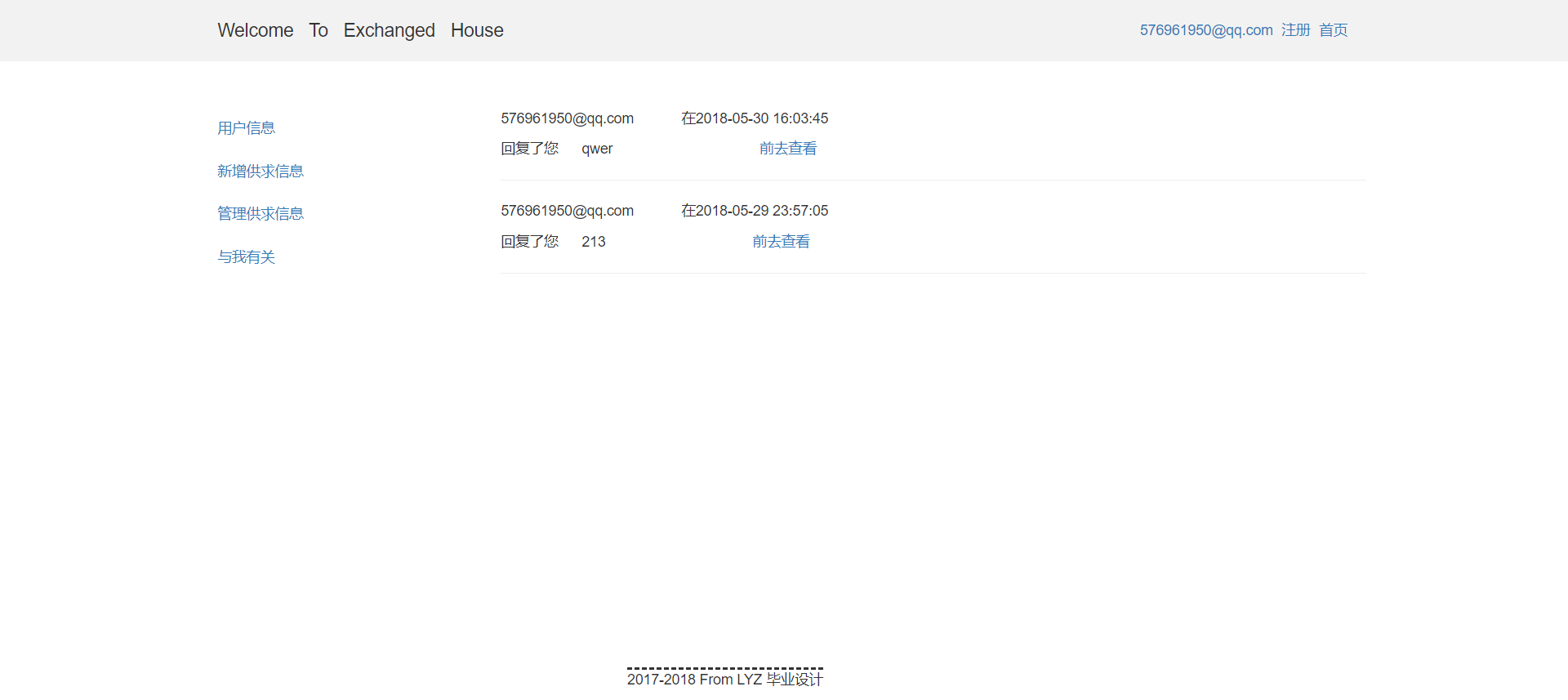
update: async (req, res, next) => {  
 try {  
 var comment = req.body;  
 let result = await commentDao.get\_by\_id(comment.goodsId);  
 console.log(comment)  
 var time = moment().format("YYYY-MM-DD HH:mm:ss");  
 if(comment.big==0){  
 result.comment[comment.high].others.push({from:comment.from,to:comment.to,content:comment.content,time:time})  
 var cols = commentDao.updateNews(result);  
 let temp = await newsDao.add({  
 customer : comment.from,  
 content : comment.content,  
 time : time ,  
 goodsId : comment.goodsId,  
 user:comment.to ,  
 });  
 res.json(cols)  
 }else {  
 var len = result.comment[comment.high].others.length-1;  
 var temp1 = result.comment[comment.high].others.slice(0, Number(comment.high2)+1);  
 var temp2 = result.comment[comment.high].others.slice(Number(comment.high2)+1, len);  
 temp1.push({from:comment.from,to:comment.to,content:comment.content,time:time})  
 result.comment[comment.high].others = temp1.concat(temp2);  
 var cols = commentDao.updateNews(result);  
 let temp = await newsDao.add({  
 customer : comment.from,  
 content : comment.content,  
 time : time ,  
 goodsId : comment.goodsId,  
 user:comment.to ,  
 });  
 res.json(cols)  
 }  
  
 }catch (err) {  
 next(err)  
 }  
},

5.4.2 用户提醒模块

该模块是把与用户所有有关的信息都收集起来，当用户进入到用户管理界面点击与我有关的页面时，会查看与当前用户有关的所有信息，并且点击查看详情可以跳转到具体的页面，页面中显示具体的时间其他用户回复的内容，方便用户了解以及查看，方便了用户之间的沟通，保证了用户能够实时收到消息并且及时给予回复。

提醒模块的实现方式是当用户在详情页下方留言或者回复评论时，服务器端会受到这些评论留言信息，当修改数据表中的评论时会同时把一些必要的信息提取出来同时存储到消息数据表中，这样用户访问该页面会根据用户进行查询，返回相关的必要数据。

用户提醒模块页面如图所示：



用户提醒模块关键代码：

<div class="col-md-9 column">  
 <% **for**(**var** i =0;i< information.length ;i++){ %>  
 <p><%= information[i].customer%>;在<%= information[i].time%></p>  
 <p>回复了您<a href="/goodsId/<%= information[i].goodsId %>"> 前去查看</a></p>  
 <hr>  
 <% } %>  
</div>

search: async (req, res, next) => {  
 try {  
 var username= req.body.username;  
 var results =await newsDao.select(username);  
 res.json(results)  
 }catch (err) {  
 next(err)  
 }  
}

5.4.3 邮件提醒模块

该模块是系统发送邮件提醒信息发布者有用户想进行私下沟通，该功能是用户与用户两者进行沟通，加上用邮件的方式进行数据传递，保证了信息的安全性与私密性。

该功能的实现方式利用了Node的'nodemailer'包，该包封装了用户发送邮件的实现方式只暴露了一些函数供用户使用。与此同时注册了163邮箱，开启了POP3/SMTP服务，配置好相关内容后就可以根据需要向用户发送消息。相关页面展示是信息详情页与用户联系的按钮，点击后即可发送邮件，建立与信息发布者之间的联系。

邮件发送功能主要代码实现：

var nodemailer = require('nodemailer');  
var transporter = nodemailer.createTransport({  
 service: '163',  
 auth: {  
 user: 'liyanzhevip@163.com',  
 pass: xxxxxxx' //授权码,通过QQ获取  
  
 }  
});  
var mailOptions = {  
 from: 'liyanzhevip@163.com', // 发送者  
 to: '576961950@qq.com', // 接受者,可以同时发送多个,以逗号隔开  
 subject: 'Exchanged-House', // 标题  
 html: "<h4>您发布的信息被关注，赶紧去联系用户xxxxx@qq.com吧</h4>", // text：文本  
 // html: "你好"  
};  
  
function sendEmail(from,to){  
 mailOptions.to=to;  
 mailOptions.html="<h4>您发布的信息被关注，赶紧去联系用户"+from+"吧</h4>";  
 let message =transporter.sendMail(mailOptions, function (err) {  
 if (err) {  
 return "发送失败"  
 }else {  
 return"发送成功"  
 }  
 });  
 return message;  
}   
module.exports.sendEmail = sendEmail;

## 5.4 系统测试

系统测试的目的就是让软件缺陷尽早的被发现和改正，从而达到需求的要求，减少后期维护成本。提高系统的质量与可靠性。系统测试是软件开发周期的最后一环，也是最重要的一环。是软件能否顺利运行的基础。

系统测试主要是测试系统的功能是否都正确实现了。与软件需求分析是否一致。是否符合开发需要。对于本系统而言主要测试的是：

1. 编码格式是否符合要求，是否简单明了。
2. 对于需求分析的功能是否都正确实现。
3. 是否存在中文乱码的问题。
4. 对数据进行正确性验证，输入的数据是否符合实际要求。

主要的解决的方法为：

(1) 对于软件的编码格式主要采用mvc设计方式，使程序层次化，使软件编码更鲜明、简单、明了。

(2) 软件开发中主要部分为需求分析的描述，是否能顺利的开发系统的基础。对于需求分析的描述必须完全正确的实现。无任何纰漏。

(3) 中文乱码一直是软件开发人员首先要考虑的问题。在本系统中对于中文乱码的问题主要采取了统一的编码格式。Utf-8编码格式是主要的编码格式。

(4) 本系统主要是用户在该平台发布供求信息，平台对这些信息加以管理。所以应保证信息的安全性和稳定性，来确保程序的正确运行。使使用人员能清楚的了解系统的功能。

## 5.5 系统修复

主要问题及修复方法

1. **退出登录失败**

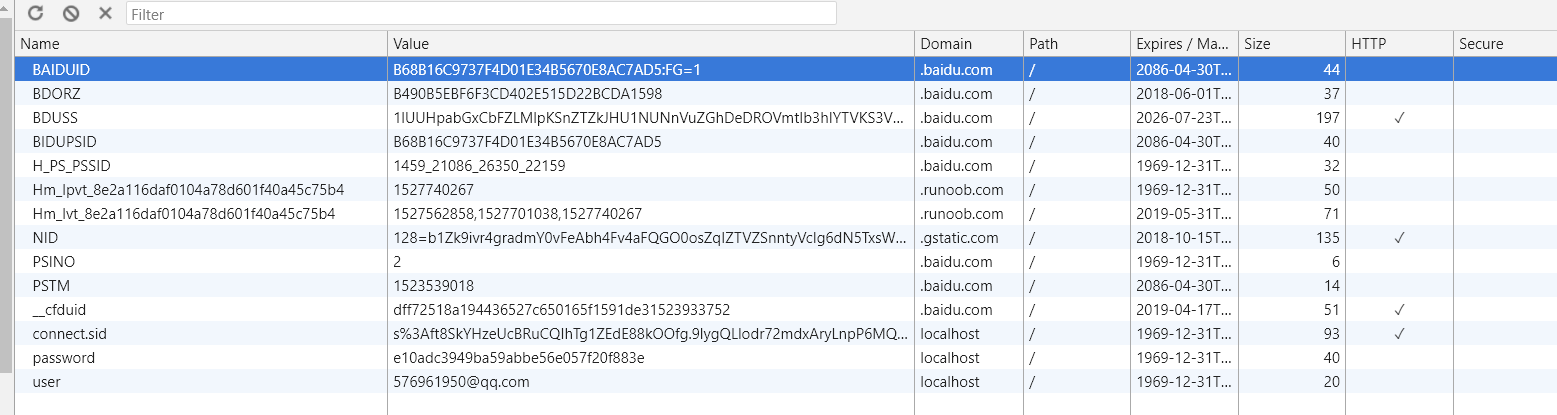
问题描述：

当用户在用户管理界面退出登录时，页面正常跳转到首页，但是

返回首页是仍显示登录状态，点击仍可以进入到用户管理页面，退出登录无效。

解决方法：

首先查看当前域名下的cookie都存在那些信息，如图5-4-1所示：



这是当前域名下的所有cookie的值，会发现自动登录的user和password属性依然存在，此时查阅与cookie相关的资料。

首先Cookie是由HTTP服务器设置的，保存在浏览器中，但HTTP协议是一种无状态协议，在数据交换完毕后，服务器端和客户端的链接就会关闭，每次交换数据都需要建立新的链接，cookie机制则是在客户端保持状态的方案，该系统采用的是持久性的cookie，cookie会一直存储在浏览器缓存中，在默认时间后才会销毁或者用户手动销毁。

Cookie的属性详解：

name字段：为一个cookie的名称。

value字段：为一个cookie的值。

domain字段：为可以访问此cookie的域名。

非顶级域名，如二级域名或者三级域名，设置的cookie的domain只能为顶级域名或者二级域名或者三级域名本身，不能设置其他二级域名的cookie，否则cookie无法生成。

顶级域名只能设置domain为顶级域名，不能设置为二级域名或者三级域名，否则cookie无法生成。

二级域名能读取设置了domain为顶级域名或者自身的cookie，不能读取其他二级域名domain的cookie。所以要想cookie在多个二级域名中共享，需要设置domain为顶级域名，这样就可以在所有二级域名里面或者到这个cookie的值了。  
顶级域名只能获取到domain设置为顶级域名的cookie，其他domain设置为二级域名的无法获取。

path字段：为可以访问此cookie的页面路径。 比如domain是abc.com,path是/test，那么只有/test路径下的页面可以读取此cookie。

expires/Max-Age 字段为此cookie超时时间。若设置其值为一个时间，那么当到达此时间后，此cookie失效。不设置的话默认值是Session，意思是cookie会和session一起失效。当浏览器关闭(不是浏览器标签页，而是整个浏览器) 后，此cookie失效。

Size字段：此cookie大小。

http字段：cookie的httponly属性。若此属性为true，则只有在http请求头中会带有此cookie的信息，而不能通过document.cookie来访问此cookie。

secure字段：设置是否只能通过https来传递此条cookie。这时会发现cookie设置的path存在问题。重新设置域名下的cookie即可。

简单cookie的使用：document.cookie="key=value;expires=失效时间;path=路径;domain=域名;secure;(secure表安全级别）

相关修复代码：

document.cookie = 'user='+ ";path=/";  
document.cookie = 'password='+ ";path=/";

1. **页面刷新失效**

**第6章 结论**

本文介绍的是一个供求信息网站，是针对互联网的一款方便快捷、高效率的员工管理系统。本文中对系统的整体架构、功能模块、逻辑结构等有着详细的解释。开发中包含着数据库中表的建立，数据的存储等。系统后台代码主要已mvc设计模式，条例清晰。对于以后的系统维护很方便。系统主要分为三部分。第一部分是客户端的实现，前端使用JS对页面进行的渲染，页面效果有了很大的提高，也能与数据端进行交互,优化了用户的体验，客户端主要是收集用户前端的数据，并把数据发送给后端。第二部分是后台代码的实现，包括对前端数据进行逻辑业务层的处理，实现了数据之间的相互转换，以及数据的存储形态并将前端需要的数据及时返回给前端。第三部分主要系统的数据库的建立，表之间的联系和管理等。页面与后台通过Express框架作为桥梁，实现了页面的数据与后台之间的传递。

在开发系统前期通过需求分析对系统整体功能模块和逻辑结构等进行了仔细的研究与分析。解决了很多的设计上出现的问题，为以后代码的编写打下了很多的基础。首先熟悉了系统的整个开发流程，并且提高了对需求的理解与认识能力，并且能够将需求加以实现，完成预期的目标。

该供求信息网站前端使用的语言是HTML/CSS和JavaScript，结合了前端BootStrap框架，努力为用户提供一个最佳的视觉体验，但是在页面的细节上还有待提高，前端也有很大的优化空间。后台主要是mvc结构设计模式，其中前后端分离，一定程度上保证了数据的安全，也让整个开发流程更加清晰，方便管理员对系统的管理、维护。本次课题的基本功能都以实现了，但存在着很多的问题，对于功能的细化，逻辑的细化、页面的美化等。这次供求信息网站的实现同时也使自己受益匪浅，对动手能力、查找和解决问题的能力都有着很大的提高。在今后的工作和学习中应不断努力学习来充实自己。

参考文献

1. 吴中骅, 雷宗民, 赵坤. Node.js实战. 第2季[M]. 电子工业出版社, 2015.
2. 朴灵. 深入浅出node. JS[M]. 人民邮电出版社, 2013.
3. Ethan Brown. Node与Express开发[M]. 人民邮电出版社, 2015.
4. Kristina Chodorow, 邓强, 王明辉. MongoDB权威指南[M]. 人民邮电出版社, 2014．
5. 扎卡斯. JavaScript高级程序设计[M]. 人民邮电出版社, 2010.
6. Loiane Groner. 学习JavaScript数据结构与算法[M]. 人民邮电出版社, 2015.
7. 陈瑶. 基于Node.js高并发web系统的研究与应用[D]. 电子科技大学, 2014.
8. 朱建兵. 基于Node.JS高并发网络应用架构的研究与实现[D]. 北京邮电大学, 2014.
9. 罗宾逊, E. ), 弗里曼,等. 深入浅出HTML与CSS= Head First HTML and CSS : 第2版 : 英文[M]. 东南大学出版社, 2013.
10. 李轶. 基于Node.js的JavaScript并发控制流框架[J]. 江汉大学学报(自然科学版), 2015(2):170-176.
11. 高飞, 何利力, 高金标. 基于Node.JS内存缓存的Web服务性能研究[J]. 工业控制计算机, 2015(11):109-110.
12. 王金龙, 宋斌, 丁锐. Node.js:一种新的Web应用构建技术[J]. 现代电子技术, 2015, v.38;No.437(6):70-73.
13. Schulz W L, Nelson B G, Felker D K, et al. Evaluation of relational and NoSQL database architectures to manage genomic annotations.[J]. Journal of Biomedical Informatics, 2016, 64:288.
14. Kudo T, Ishino M, Saotome K, et al. A Proposal of Transaction Processing Method for MongoDB[J]. Procedia Computer Science, 2016, 96:801-810.
15. Gackenheimer C. Networking with Node.js[J]. 2013.

致 谢

毕业设计和论文能够顺利的完成，首先要感谢王溪波导师。

老师在毕业设计和论文期间从选题、论证、开题、设计以及论文审核上，都帮我准确地把握课题发展方向和进度、保证课题质量。正是在老师的严格要求和细心指导下，课题得以顺利完成。

其次要感谢毕业设计审查组的各位老师，指出了我的课题和论文中不足之处，使设计和论文更加完善，同时给我提出宝贵意见，增加了我的论文撰写经验，我受益匪浅。