

毕业论文

基于文本分类的社交系统的设计与实现

|  |  |
| --- | --- |
| 院 别 | 管理学院 |
| 专业名称 | 信息管理与信息系统 |
| 班级学号 | 180319 |
| 学生姓名 | 房圆浩 |
| 指导教师 | 袁静波 |

2022年4月29日

基于文本分类的社交系统的设计与实现

摘 要

移动互联网技术的发展，让人们的沟通和交流变得越来越简单，各种社交APP软件成为了人们日常生活中非常重要的一项工具，很多人都通过社交APP丰富了自己的朋友圈,也认识了更多志同道合的朋友。社交APP在社会中扮演的角色变得越来越重要。

用户通过社交APP会生成大量的生活数据,其中海量的生活资讯具有很重要的分析价值。而随着移动网络科技的蓬勃发展,更多的网友也能够随时的发布自身生活的动向以及对资讯事物的看法,用户可以以短文本或多媒体信息的方式,在网络平台上进行即时的生活资讯共享和互动。用户提供的文本时间虽短,但长期累积下来的文章数量中蕴含着大量的关于使用者的个性化特征等信息。为筛选对于使用者最有价值的信息,推荐算法就变得十分必要。本文通过将文本分类的方式,，收集用户产生的信息提取标签生成用户画像，据此向用户推荐感兴趣的人和兴趣圈。

本文设计与实现了一个基于文本分类的社交系统，采用mvvm架构，使用Springboot、Vue3、Mybatis作为主要开发框架、采取了前后端分离的开发方式，结合TF-IDF关键词提取算法、jieba分词、HanLP框架作为文本分析工具构建推荐算法向用户推荐可能感兴趣的人或圈子，本系统主要利用IDEA和git工具进行代码实现和代码托管，系统的主要模块有兴趣圈管理、个人信息管理、帖子管理、用户交流和个性化推荐，实现了基于文本分类的社交系统的系统目标。

关键词：社交系统；Springboot；Vue；文本分类

**Design and implementation of a social system based on text classification**

Author: Fang YuanHao

Tutor: Yuan Jingbo

Abstract

The development of mobile Internet technology makes people's communication and exchange easier and easier, and various social APP software has become a very important tool in people's daily life, and many people have enriched their circle of friends through social APP, and met more like-minded friends. The role of social app in the society is becoming more and more important.

Users will generate a lot of life data through social app, among which a large amount of life information is very important to analyze. With the booming development of mobile network technology, more users are able to post their life trends and views on information at any time, and users can share and interact with life information instantly on the network platform in the form of short text or multimedia information. Although the text provided by users is short, the number of articles accumulated over a long period of time contains a large amount of information about the personal characteristics of users. In order to filter the most valuable information for users, recommendation algorithms become necessary. In this paper, by classifying text, we collect user-generated information and extract tags to generate user profiles, and recommend users to people and circles of interest accordingly.

In this paper, we design and implement a social system based on text classification, using mvvm architecture, Springboot, Vue3, Mybatis as the main development framework, adopting the development method of front and back-end separation, combining TF-IDF keyword extraction algorithm, jieba word separation, HanLP framework as a text analysis tool to build recommendation algorithm to recommend users The system mainly uses IDEA and git tools for code implementation and code hosting, and the main modules of the system are interest circle management, personal information management, post management, user communication and personalized recommendation, which achieves the system goal of a text-based classification social system.

**Key words:** social system; springboot; vue; text classification

目 录

[1 绪论 5](#_Toc102740704)

[1.1 研究背景和研究意义 5](#_Toc102740705)

[1.2 国内外研究现状 5](#_Toc102740706)

[1.3 论文的设计方法与内容 6](#_Toc102740707)

[1.3 论文的组织结构 7](#_Toc102740708)

[2 相关技术介绍 8](#_Toc102740709)

[2.1 前后端分离开发模式和系统架构 8](#_Toc102740710)

[2.1.1 前后端分离的开发模式 8](#_Toc102740711)

[2.1.2 系统架构 8](#_Toc102740712)

[2.2 主要开发技术 9](#_Toc102740713)

[2.2.1 Spring boot简介 9](#_Toc102740714)

[2.2.2 Vue简介 10](#_Toc102740715)

[2.2.3 Mybatis简介 11](#_Toc102740716)

[2.3 运行方式 12](#_Toc102740717)

[2.4 开发工具 13](#_Toc102740718)

[2.4.1开发工具IDEA 13](#_Toc102740719)

[2.4.2版本控制工具Git 13](#_Toc102740720)

[2.5 本章小结 14](#_Toc102740721)

[3 系统分析 15](#_Toc102740722)

[3.1 系统目标 15](#_Toc102740723)

[3.2 功能需求分析 15](#_Toc102740724)

[3.3 用例图 16](#_Toc102740725)

[3.4 系统非功能性需求分析 19](#_Toc102740726)

[3.5 可行性分析 20](#_Toc102740727)

[3.6 本章小结 20](#_Toc102740728)

[4 系统设计 21](#_Toc102740729)

[4.1 系统整体架构设计 21](#_Toc102740730)

[4.1 功能结构图 21](#_Toc102740731)

[4.2 数据库设计 22](#_Toc102740732)

[4.2.1 数据库设计原则 23](#_Toc102740733)

[4.2.2 主题数据库 23](#_Toc102740734)

[4.2.3 数据库概念设计 24](#_Toc102740735)

[4.2.4 数据库逻辑结构设计 27](#_Toc102740736)

[4.3 本章小结 31](#_Toc102740737)

[5 系统实现与测试 32](#_Toc102740738)

[5.1 基于文本分类的社交系统的实现 32](#_Toc102740739)

[5.1.1 登录注册界面 32](#_Toc102740740)

[5.1.3 发帖页面 34](#_Toc102740741)

[5.1.4 广场页面 34](#_Toc102740742)

[5.1.5 个人信息页面 35](#_Toc102740743)

[5.2 基于文本分类的社交系统的测试 35](#_Toc102740744)

[5.2.1 测试目的 35](#_Toc102740745)

[5.2.2 测试环境 36](#_Toc102740746)

[5.3 核心代码 36](#_Toc102740747)

[5.4 本章小结 36](#_Toc102740748)

[结 论 37](#_Toc102740749)

[致 谢 38](#_Toc102740750)

[参考文献 39](#_Toc102740751)

[附 录 40](#_Toc102740752)

# 1 绪论

## 1.1 研究背景和研究意义

社交产品的出现无疑在很大程度上便利了人们的交往，时间、空间和地域不再是社交难以跨越的红狗，人们可以随时随地与想交流的人建立联系，而不必花费巨大的交流成本。当今世界，随着互联网的发展，人们可以越来越便捷的在网上与自己的好友联系，也能够在各类app上结识新的朋友，互联网以一种新的方式满足了人们的社交需求，不同的人在寻找合适的网友时有着不同的要求，多数人们更喜欢找跟自己有相同爱好的网友，或者是加入一个兴趣圈，从而能够交流爱好，培养感情。随着网民基数越来越大，人们面对的选择太多，怎样找到有相同爱好的网友成为一个问题，尤其在当前疫情的环境下，多数人选择不外出、少外出，在这种情况下，人们把大量的时间花在网络上，网络也满足了很多人的社交需求。

在近十年来，随着智能手机和其他智能设备的兴起，全球网络中的在线人数急剧增加，人们在网络上的各种操作都会产生信息，网络上产生的数据量越来越大，这些海量的数据有一定的分析价值，带动了大数据技术和云计算的兴起，如今技术更新迭代很快，在应用开发方面，从页面构建到服务器的处理再到数据存储各种技术层出不穷，在推荐算法方面，自然语言处理和计算机视觉技术也在飞速发展，各大公司训练自己的AI模型并将推荐算法应用到各行各业，例如电子商务、社交平台、短视频平台。推荐算法在一定程度上为用户解决了信息过载的问题，利用推荐算法向用户展示了感兴趣的或是最需要的，也能够发掘用户的潜在需求。

## 1.2 国内外研究现状

1.2.1社交系统研究现状

近几年，社交系统在国内外发展迅速，在国内，2009年新浪微博的推出，拉开了中国微信息社交时代的序幕，微博根据用户的价值取向、兴趣取向等维度进行划分，用户可通过推介及自行搜索等方式构建起自己的小圈子。微信将线下的熟人社交转移到移动互联网端，但它为聊天方式带来了语音聊天的新模式，凭借这一功能，微信迎来了用户的井喷，并最终奠定下其移动社交巨头的地位。而有一些产品另辟蹊径，从切实解决年轻人的社交需求出发，主打社交元宇宙，典型代表就是2016年上线的Soul。Soul的产品核心是移除用户的表达压力和通过算法帮助年轻用户找到志同道合的人，获得高质量的反馈，如今Soul成为了基于兴趣图谱和社交画像，以算法为驱动的社交元宇宙，Soul展现出了不小的势能，并验证了自身探索方向的可行性。在国外，Facebook、Twitter和Instagram是大多数国家的主流社交媒体，发展多年已相对成熟，Twitter类比于国内的微博，有着大量的用户和活跃的社群，自2018年以来，TikTok迅速崛起，成为社交媒体世界的新星，目前仍保持着较高的增长率。

1.2.2 推荐算法研究现状

随着大数据时代的到来，推荐算法是迅速发展的一项新技术，推荐系统在国内外迅速发展，并在电商、音乐视频等领域得到广泛地运用，具有代表性的有 Amazon、淘宝网、京东商城、豆瓣等。推荐算法作为推荐系统的核心，在实际应用中仍存在着一些问题和局限性，影响了推荐结果。如基于内容的推荐算法存在用户冷启动问题， 基于用户的协同过滤推荐算法无法避免数据稀疏性问题等等。为解决现有推荐算法所存在的局限，很多国内外学者提出了各种新型推荐算法来解决这些问题。目前推荐系统普遍面临着数据量大以及推荐准确度不高的问题，而 Badr 等人则提出了基于 Spark 的分布式群组推荐系统，并运用降维技术和监督学习来降维、发掘群组用户以克服以上缺点。李等人针对电子商务，提出了基于网络表示学习的推荐算法。张等人采取了一种新的混合推荐算法，融入了用户上下文情感分析和多视图推荐算法，大大提升了推荐准确度。但单单提高推荐准确度会导致推荐给用户的内容都是非常类似的，在推荐结果多样性方面存在不足，因此，孟等人提出的推荐算法为用户推荐与其历史偏好相对应的但推荐内容之间又存在一定多样性的物品，此种方法不仅能提供给用户其感兴趣的结果，同时丰富了推荐内容的多元性。

## 1.3 论文的设计方法与内容

1、设计方法

本文依据软件工程方法进行研究设计，完成系统需求分析、系统设计、系统实现和系统测试等内容。通过阅读相关文献及实际调研，对社交系统进行业务流程分析及功能需求分析，并采用面向对象的方法进行用例图设计。在需求分析的基础上，进行系统功能结构设计及数据库设计，并基于Springboot + Vue框架，使用IDEA开发工具和Git版本控制工具，采用Mysql数据库，以Mybatis作为ORM框架，采用JAVA，TypeScript编程语言，采用MVVM架构对系统进行设计及编程实现。

2、研究内容

基于文本分类的社交系统的主要功能模块有帖子管理、兴趣圈管理、个人信息管理、好友管理、用户交流和个性化推荐，每个功能模块根据需求提供了最基本的查询、更新、删除和添加功能。在帖子管理界面，用户可以管理自己的帖子和当前帖子下的评论。在兴趣圈管理页面，用户可以管理兴趣圈的成员和帖子，确保兴趣圈气氛良好。在个性化推荐模块，系统利用jieba中文分词工具和HanLP开源库进行分词和关键词提取以及语义相似度计算，从而得到对用户最佳的推荐结果，提高用户的满意度。

## 1.3 论文的组织结构

本论文共有五章，各章内容如下所述：

第1章绪论。主要介绍了基于文本分类的社交系统的开发背景以及意义然后介绍了论文的主要研究内容、研究方法，最后是文章的组织结构。

第2章相关技术简介。介绍了Springboot + Vue + Mybatis的框架简介、开发的相关工具、开发方式以及开发环境。

第3章系统分析。具体是对系统目标说明；功能需求分析,并对系统各个模块做了详细的介绍；性能需求分析，从各个角度对系统作了评估，还进行了可行性分析，分析系统的可行性。然后给出了本系统的运行平台，最后是对本章内容的总结。。

第4章系统设计。主要就本系统的开发模式进行说明，接着进行功能设计和数据库设计，最后是本章小结。

第5章基于文本分类的社交系统的实现与测试，还简单介绍了部分关键代码。主要是对系统界面的实现和对系统功能关键编码的实现部分进行了详细的介绍，还对系统测试作了介绍，最后对本章作了总结。

# 2 相关技术介绍

## 2.1 前后端分离开发模式和系统架构

本文使用IDEA开发软件和Git版本控制工具，基于Springboot + Vue + Mybatis框架，进行系统设计与实现。下面对所需要的相关技术进行介绍。

### 2.1.1 前后端分离的开发模式

不同于以往的服务端渲染的传统web开发模式，目前的web应用程序大多使用前后端分离的开发模式，服务端渲染的开发模式是指：服务器发送给客户端HTML页面，是在服务器通过字符串的拼接动态生成的，因此客户端不需要使用Ajax这样的技术额外请求页面的数据，然而服务端渲染的缺点也显而易见，一是占用服务器端的资源，服务器端需要完成HTML页面内容的拼接，如果请求较多，会对服务器造成一定的访问的压力，不利于项目高效开发；二是不利于前后端分离，开发效率低，使用服务器端渲染，则无法进行分工合作，尤其对于前端复杂度高的项目，不利于高效开发。随着Ajax技术的广泛应用，出现了前后端分离的开发模式，简而言之，前后端分离的开发模式，就是后端只负责提供API接口，前端使用Ajax调用接口的开发模式，前后端分离的优点是：开发体验好，前端专注于UI页面的开发，后端专注于API的开发，且前端有更多的选择性，同一套后端API可以用到移动端和PC端；用户体验好，利用Ajax技术，极大的提高了用户体验，可以轻松实现页面的局部刷新；减轻了服务器端的渲染压力，页面最终是在每个用户的浏览器中生成的。

### 2.1.2 系统架构

本系统采用MVVM架构进行设计，MVVM是指：Model数据层、View视图层和ViewModel层，VM是MVVM思想的核心。在MVVM中，为了避免Controller层太过臃肿，Controller不再像MVC架构负责解析Model完成业务逻辑和页面逻辑，而是将业务逻辑和页面逻辑剥离出来放到ViewModel层，由ViewModel负责解析Model，Model负责对数据进行取和存，各部分的通信都是双向的，MVVM是将页面与数据模式分离的模式。

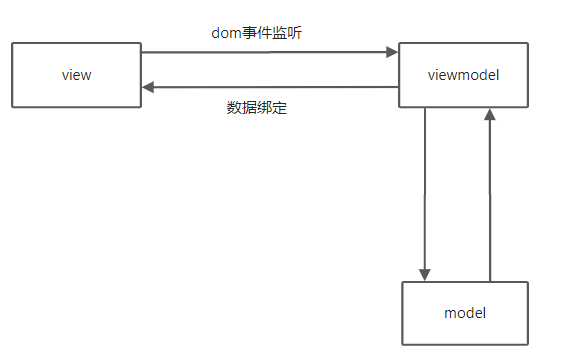


图2.1 MVVM架构图

Html页面作为View层，使用Vue框架实现ViewModel层，使用Springboot完成Model层，Mybais作为持久层框架，WebSocket作为实时通信框架。

本系统主要基于Springboot框架搭建并进行后端核心功能的开发。Springboot框架集成了大量的基础框架，同时内嵌了Tomcat容器，与传统的需要大量繁琐配置工作的J2EE框架相比，Springbbot框架的开发速度更快捷，效率更高。

在前端页面的设计中，借助了ant-design-vue组件库的大量组件，完成了快速开发和美化页面，并且借助github开源的自然语言处理库HanLP和jieba完成了推荐算法，下面对用到的主要技术进行介绍。

## 2.2 主要开发技术

### 2.2.1 Spring boot简介

Springboot起源于Spring框架，设计目的是简化Spring应用的初始搭建以及开发过程，其默认配置了很多框架的使用方式，只需要非常少的几个配置就可以搭建起来一个web项目。

Spring框架是一个开放源代码的J2EE应用程序框架，由Rod Johnson发起，是针对bean的生命周期进行管理的轻量级容器。Spring解决了开发者在J2EE开发中遇到的许多常见的问题，提供了功能强大IOC、AOP及Web MVC等功能。Spring可以单独应用于构筑应用程序，也可以和Struts、Webwork、Tapestry等众多Web框架组合使用，并且可以与 Swing等桌面应用程序AP组合。因此， Spring不仅仅能应用于JEE应用程序之中，也可以应用于桌面应用程序以及小应用程序之中。

Spring框架具有面向切面编程（AOP）框架，SpringAOP框架基于代理模式，同时运行时可配置；AOP框架主要针对模块之间的交叉关注点进行模块化。Spring框架下的事务管理、远程访问等功能均可以通过使用SpringAOP技术实现。Spring的事务管理框架为Java平台带来了一种抽象机制，使本地和全局事务以及嵌套事务能够与保存点一起工作，并且几乎可以在Java平台的任何环境中工作。Spring集成多种事务模板，系统可以通过事务模板、XML或Java注解进行事务配置，并且事务框架集成了消息传递和缓存等功能。

SpringBoot框架中还有两个非常重要的策略：开箱即用和约定优于配置。开箱即用，是指在开发过程中，通过在MAVEN项目的pom文件中添加相关依赖包，然后使用对应注解来代替繁琐的XML配置文件以管理对象的生命周期。这个特点使得开发人员摆脱了复杂的配置工作以及依赖的管理工作，更加专注于业务逻辑。约定优于配置，是一种由SpringBoot本身来配置目标结构，由开发者在结构中添加信息的软件设计范式。这一特点虽降低了部分灵活性，增加了BUG定位的复杂性，但减少了开发人员需要做出决定的数量，同时减少了大量的XML配置，并且可以将代码编译、测试和打包等工作自动化。

### 2.2.2 Vue简介

  Vue是一套用于构建用户界面的渐进式的js框架，发布于2014年2月。与其它大型框架不同的是，Vue被设计为可以自底向上逐层应用。Vue的核心库只关注视图层，不仅易于上手，还便于与第三方库如vue-router，vuex或既有项目整合，Vue框架是MVVM模式的实现者，提供了双向数据绑定模式，在Vue.js中Model是指模型层，View是视图层，在这里表示DOM（HTML操作的元素）；ViewModel是连接视图和数据的中间件，Vue.js就是MVVM中的ViewModel层的实现者。在MVVM架构中，是不允许数据和视图直接通信的，只能通过ViewModel来通信，而ViewModel就是定义了一个Observer观察者，ViewModel能够观察到数据的变化，并对视图下对应的内容进行更新，ViewModel能够监听到视图的变化，并能够通知数据发生改变，Vue的核心就是实现了DOM监听与数据绑定。

Vue的优点是：1. 轻量级，体积小是一个重要指标，Vue压缩后只有20多kb。2. 易上手，学习曲线平稳，文档齐全。3. 吸取了Angular（模块化）和React（虚拟DOM）的长处，并拥有自己独特的功能，如计算属性4. 作为一款开源框架，社区活跃度高。

Vue有两大核心要素，一是数据驱动，当数据发生变化时，vue自动更新视图，它的原理是利用了 Object.definedProperty 中的setter/getter 代理数据，监控对数据的操作。二是组件化，Vue将页面上每个独立的可交互的区域视为一个组件，每个组件对应一个工程目录，组件所需的各种资源在这个目录下就近维护，页面作为组件的容器，组件可以嵌套自由组合（复用）形成完整的页面。

### 2.2.3 Mybatis简介

MyBatis 是支持定制化 SQL、存储过程以及高级映射的优秀的持久层框架。MyBatis 避免了几乎所有的 JDBC 代码和手工设置参数以及抽取结果集。MyBatis 使用简单的 XML 或注解来配置和映射基本体，将接口和 Java 的 POJO(普通的 Java对象)映射成数据库中的记录。它内部封装了jdbc，使开发者只需要关注sql语句本身，而不需要花费精力去处理加载驱动、创建连接、创建statement等繁杂的过程。mybatis通过xml或注解的方式将要执行的各种 statement配置起来，并通过java对象和statement中sql的动态参数进行映射生成最终执行的sql语句。最后mybatis框架执行sql并将结果映射为java对象并返回。采用ORM思想解决了实体和数据库映射的问题，对jdbc 进行了封装，屏蔽了jdbc api 底层访问细节，使我们不用与jdbc api 打交道，就可以完成对数据库的持久化操作。

Mybatis的优点是：1. 简单轻量，mybatis本身就很小且简单。没有任何第三方依赖，最简单安装只要两个jar文件+配置几个sql映射文件易于学习，易于使用，通过文档和源代码，可以比较完全的掌握它的设计思路和实现。2. 灵活，mybatis不会对应用程序或者数据库的现有设计强加任何影响。 sql写在xml里，便于统一管理和优化。通过sql基本上可以实现我们不使用数据访问框架可以实现的所有功能，或许更多。3、解除sql与程序代码的耦合，将业务逻辑和数据访问逻辑分离，使系统的设计更清晰，更易维护，更易单元测试。sql和代码的分离，提高了可维护性。4、提供映射标签，支持对象与数据库的orm字段关系映射5、提供对象关系映射标签，支持对象关系组建维护6、提供xml标签，支持编写动态sql。

## 2.3 运行方式

下面来详细介绍Springboot+Vue的运行方式。由于采取的是前后端分离的开发方式，降低了前端和后端的耦合度，Springboot后端代码和Vue前端代码的启动互不影响。

首先用户面对的是表现层，也就是前端页面，用户通过页面完成一系列操作，Vue会监听用户的操作，根据用户的行为向后端发送请求或者是改变DOM，当Vue使用axios框架向后端发送请求时，后端的controller层接收到前端发来的请求，调用相应的service层，service层进而调用dao层，也就是数据持久层，dao层设计为一个个的接口，其中每个dao接口对应一个mapper，以xml文件存在，xml文件中每个标签内有sql语句，实现了对应的方法，这里会依赖ORM框架Mybatis的对象映射，当service调用dao的对象时，会寻找相应的mapper，调用相应的方法，完成数据库表的增删改查功能，操作完成之后mapper返回数据给service层，service根据请求对数据进行包装处理后返回给controller层，controller最后将response返回给前端页面，Vue将得到的response进行解析和处理，得到只对用户展示的数据，并将其通过数据绑定展示在页面上，用户观察到结果，这样就完成了从用户的操作到视图的更新整个流程。

系统架构图如图2.1所示。

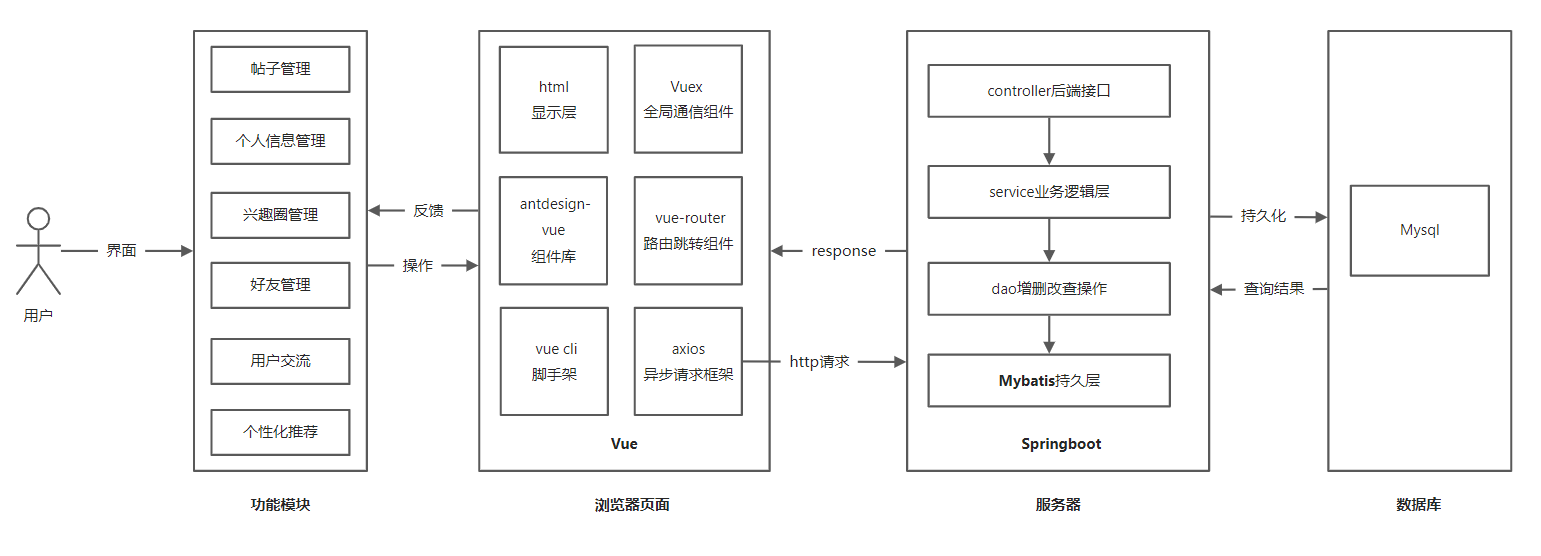


图2.2 系统架构图

## 2.4 开发工具

本次系统开发所使用的工具包括开发工具IDEA和版本控制工具git，下面将一一介绍。

### 2.4.1开发工具IDEA

IDEA 全称IntelliJ IDEA，是用于java语言开发的集成环境（也可用于其他语言），IntelliJ在业界被公认为最好的java开发工具之一，尤其在智能代码助手、代码自动提示、重构、J2EE支持、Ant、JUnit、CVS整合、代码审查、 创新的GUI设计等方面的功能可以说是超常的。

IDEA有着丰富的支持开发的功能，比如编码辅助，一键生成getter/setter、构造方法和ToString方法；快捷键智能补全代码；良好的版本控制支持，集成了市面上常见的所有版本控制工具插件，包括git、svn，让开发人员在编程的工程中直接在idea里就能完成代码的提交、检出、解决冲突、查看版本控制服务器内容等等。例如git；对重构的优越支持；动态语法检测和代码检查，任何不符合java规范、自己预定义的规范、累赘都将在页面中加亮显示；预制模板，预置模板可以让开发人员把经常用到的方法编辑进模板，使用时你只用输入简单的几个字母就可以完成全部代码的编写。在本系统开发中就多次使用预制模板功能完成打印日志代码。总的来说，IDEA是一款智能、高效、全面和人性化的开发集成环境。

### 2.4.2版本控制工具Git

Git是目前世界上最先进的分布式版本控制系统。Git 是一个开源的分布式版本控制软件，用以有效、高速的处理从很小到非常大的项目版本管理。 Git 是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件。

Git的优点有：1、更方便的 Merge： 分布式管理必然导致大量的 Branch 和 Merge 操作。因此分布式版本控制系统都特别注意这方面。在传统的 CVS 里面制作 Branch 和 Merge 需要很大的工作量，而Git的Merge操作很方便。2、更方便的管理：传统的版本控制系统使用中央仓库，一些仓库相关的管理就只能在仓库上进行。赋予开发团队每一个人中央仓库的管理权限是非常不好的。但是有时候确实会比较不方便的地方。3、更健壮的系统：分布式系统一般情况下总是比单服务端的系统要健壮，因为当服务端一旦挂掉了整个系统就不能运行了。然而分布式系统通常不会因为一两个节点而受到影响。4、对网络的依赖性更低：虽然现在网络非常普及，但是并不是随时随地都有高速网络，甚至有时候根本没有网络可以访问。而Git可以支持离线操作，将commit记录存放在本地。5、更少的“仓库污染”：有时候开发人员要做一个模块，它不是太大，所以没有必要为它新建一个 branch ，但是它又不是那么小，不可能一次提交就做好。于是便会提交一些不完整的代码到仓库，有时候会导致整个程序无法运行，严重影响团队里其他人的开发。大多数人在这种情况下的解决办法都是写完之后再提交。如果是分布式系统的话就不会存在这样的问题，因为本地仓库的修改不会影响到别人的仓库。当开发人员完成并测试以后，其它开发人员就可以从这里 pull 代码了。

总结来说，Git的优点是：1. 适合分布式开发，强调个体。2. 公共服务器压力和数据量都不会太大。3. 速度快、灵活。4. 任意两个开发者之间可以很容易的解决冲突。5. 离线工作。

## 2.5 本章小结

本章主要介绍了基于文本分类的社交系统系统的相关技术，包括Springboot、Vue和Mybaits框架简介，介绍了前后端分离的系统的运行方式，最后介绍了IDEA开发工具和Git版本控制工具。

# 3 系统分析

## 3.1 系统目标

进行需求分析是开发软件系统的前提，文章主要对基于文本分类的社交系统的系统需求进行对于设计基于文本分类的社交系统的目的是希望用户通过在此系统上发帖和评论找到和自己有相同爱好的人。本系统希望通过分析用户的行为来判断用户的喜好，为不同的用户建立用户画像，并向其推荐不同的好友。

本系统完成了基本的社交系统的功能，并将系统分为以下五个模块，分别是帖子管理、兴趣圈管理、个人信息管理、用户交流和个性化推荐。

本次设计希望通过系统可以达到以下目标：

（1）界面美观简洁。简洁美观的页面给人带来轻松的感觉。

（2）系统能够完成社交系统的基本功能。系统能够满足基本的社交需求，用户的分享能够让其它用户看到，有良好的交互性。

（3）用户交流方便快捷。

（4）系统能够有效分析数据，采用合适的模型和算法对数据进行有效分析，使得用户个性化推荐模块有良好的效果。

## 3.2 功能需求分析

1、帖子管理

帖子作为用户输出信息的主要载体，有着全面的增删改查功能，首先是自我帖子管理，用户可以在发帖页面发布帖子，也可以在我的帖子页面删除帖子，可以重新编辑帖子也可以查看用户发过的帖子，还有他人帖子管理，用户可以对自己感兴趣的帖子收藏起来，也可以在自己的收藏夹找到自己收藏的帖子，并且可以取消收藏。

2、个人信息管理

个人信息管理包括查看个人信息和修改个人信息，个人信息包括：昵称、头像、地址、年龄、性别、自我介绍和个人标签。

3、兴趣圈管理

用户可以对所在兴趣圈进行管理，作为兴趣圈成员，用户可以搜索感兴趣的兴趣圈，并且可以加入兴趣圈或者退出兴趣圈，作为兴趣圈管理员，管理员应当可以对兴趣圈进行信息管理，成员管理和兴趣圈内的帖子管理，确保所管理的兴趣圈内氛围良好。

4、好友管理

好友管理是社交系统的一个重要模块，用户通过点击其它用户头像进入其他用户的个人信息界面可以关注好友。在用户的好友列表里进行好友管理，用户在该页面可以取消关注好友。

5、用户交流

用户交流模块包含多种方式，首先是用户之间的私信，这是用户之间一对一的交流，再者是在用户的帖子上进行的交流，包括评论、点赞和点踩，通过对喜欢的用户评论或者点赞能够获得发帖人的关注，营造良好的社区氛围。

6、个性化推荐

个性化推荐模块是基于文本分类的社交系统的核心模块，该模块通过自然语言处理中的关键词提取算法和文本相似度算法将用户的喜好提取出来，生成相应的用户画像，并据此向用户推荐可能感兴趣的人或兴趣圈。

## 3.3 用例图

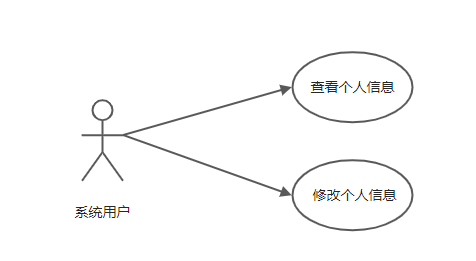
业务流程图采用的表达方式是图形，可以很好地表达业务功能之间的关系和流程，但是对功能的描述相对薄弱。下面将系统功能部分转换为用例图，详细说明了，人与业务之间的关系。

1. 帖子管理用例图

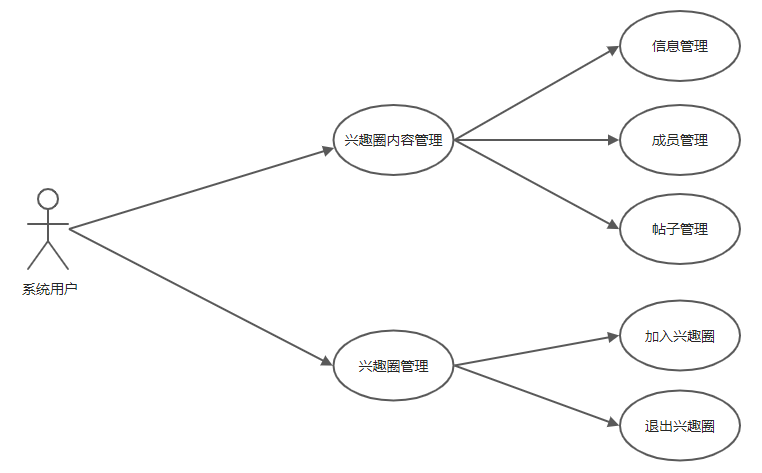
帖子管理主要分为自我帖子管理和他人帖子管理两大方面。



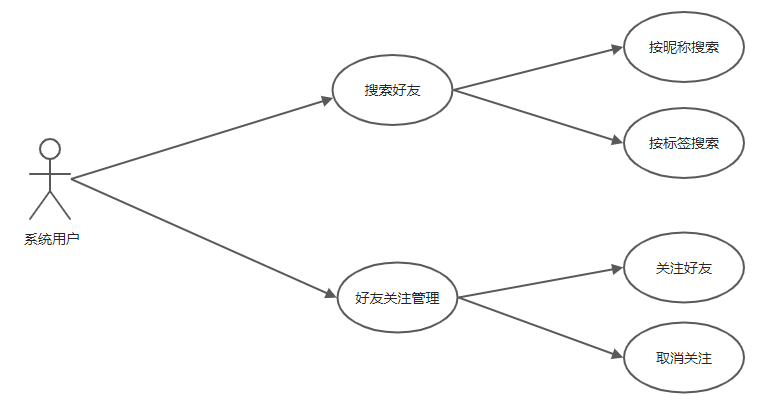
2、“个人信息管理”用例图



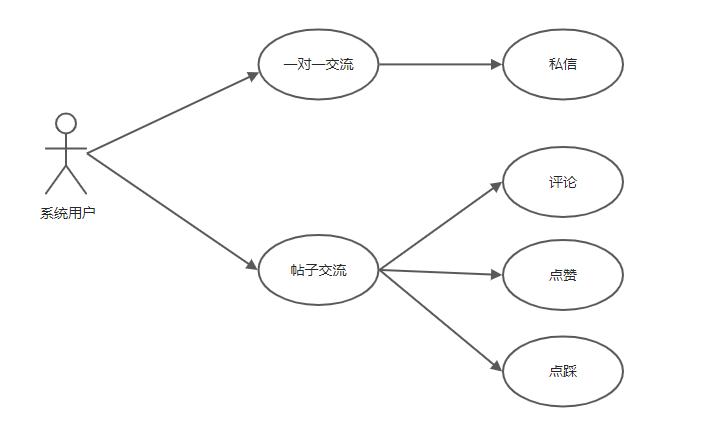
3、“兴趣圈管理”用例图



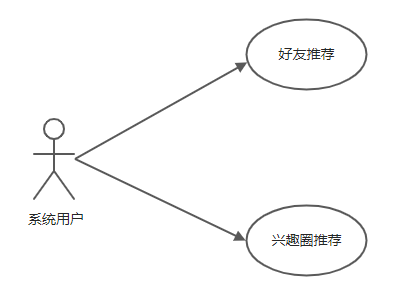
4、“好友管理”用例图



5、“用户交流”用例图



6、“个性化推荐”用例图



## 3.4 系统非功能性需求分析

设计并实现一个好的系统，不仅需要完成功能需求，也需要完成一些非功能性的需求。基于文本分类的社交系统是以社交为主的面向广大群众的系统，本系统从实际出发，不仅要实现帖子管理、个人信息管理、兴趣圈管理、好友管理、用户交流和个性化推荐等功能，也需要满足可扩展性、可靠性、易用性、可移植性等非功能性需求。

1. 可扩展性

本系统采用前后端分离的开发模式，前端页面使用组件化开发，可扩展性高，并且后端开发时将每种request和response封装成一个类，采用controller、service和dao层三层设计，耦合度低，便于进行扩展。

1. 可靠性

本系统应该安全可靠，容错性强，保证数据存储安全和具有较高的异常处理能力。同时系统故障率保持在较低水平，不发生或者少发生故障。即使发生故障后，经过运维人员的处理也能很快恢复系统的正常运行。

1. 易用性

本系统应该简洁易用、界面美观友好，方便用户对系统的学习和操作，用户不需要花费很多时间，在较短时间内就能轻松学会使用该系统进行相关社交网络数据的分析工作。用户在使用系统时，系统应该及时将响应反馈给用户，带给用户良好的使用体验。

1. 可移植性

本系统具有可移植性，目前使用较多的系统是Windows和Linux，本系统在这两个操作系统中都能正常部署和运行，能够满足多数人的需求。

## 3.5 可行性分析

## 3.6 本章小结

本章首先介绍了系统的设计目标，进而确定了系统要实现的功能范围。接着详细分析了基于文本分类的社交系统管理的业务流程，画出了的业务流程图。第三，又作了功能需求分析，对系统各个管理模块作了详细的介绍。第四，画出了用例图，详细说明了用户与功能之间的关系。第五，对系统性能作了非功能性需求分析，从可扩展性、可靠性、易用性、可移植性，考虑了系统的性能，并对系统的运行平台对了介绍。通过本章的分析，明确了系统开发的实现目标、业务流程、功能需求、性能需求等，为下一章系统设计作了准备。

# 4 系统设计

## 4.1 系统整体架构设计

本系统采用B/S架构，即(Brower/Server,浏览器/服务器)架构，与C/S(Client/Server,客户端/服务器)架构相对应。这种模式不需要安装客户端，客户端就是WEB浏览器，简化了系统的开发、维护和使用。由于WEB应用程序的体系结构非常复杂，使用传统的WEB开发技术如Servlet和Jsp,在开发和后期维护的过程中，耦合度较高，添加和修改模块工作量很大，开发效率很低，而且系统不易维护和扩展。为了解决这一问题，经过综合考虑，系统使用了SpringBoot框架，SpringBoot是一个开源的微服务框架，它是对Spring框架的一个封装，提供很多默认配置和自定义的配置，并且内置了Tomcat容器，使用该框架搭建系统和开发系统非常简单。

## 4.1 功能结构图

用户的需求具体体现在各种信息的提供、保存、更新和查询方面,这就要求数据库结构能充分满足各种信息的输出和输入。根据需求分析，得到基于文本分类的社交系统的功能结构图，如图4.1所示：

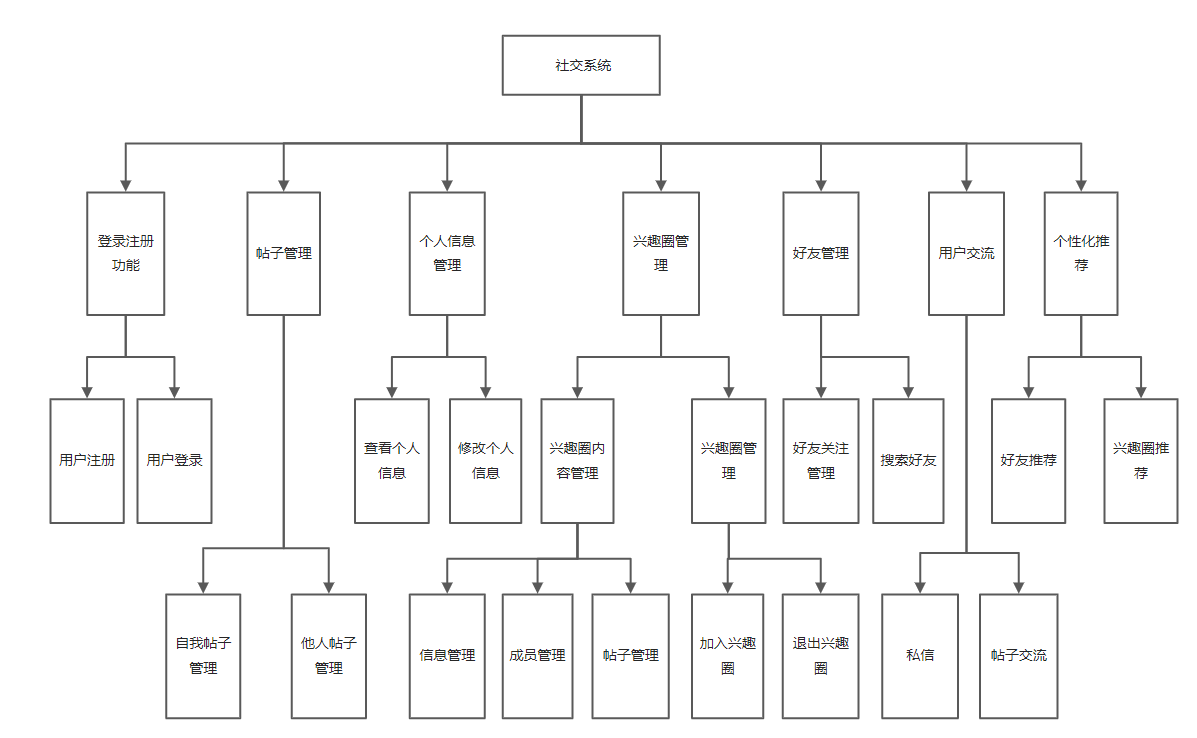


图4.1 社交系统的功能结构图

（1）登录注册功能，包括用户注册和用户登录。

（2）帖子管理功能包括自我帖子管理和他人帖子管理，自我帖子管理包括发帖、删除帖子、编辑帖子和查看我的帖子，他人帖子管理包括的功能有收藏他人帖子，取消收藏帖子和查看收藏的帖子。

（3）个人信息管理包括查看个人信息和修改个人信息功能。

（4）兴趣圈管理包括兴趣圈内容管理和兴趣圈管理两部分，兴趣圈内容管理是作为管理员行使的功能，包括信息管理、成员管理和帖子管理，另一部分是兴趣圈管理，兴趣圈管理包括加入兴趣圈和退出兴趣圈。

（5）好友管理包括好友关注管理和搜索好友，在好友关注管理模块用户可以关注好友或者取消关注，在搜索好友模块用户可以针对用户昵称搜索或者根据用户标签搜索。

（6）用户交流包括一对一交流和在帖子上进行的交流，一对一交流的方式是私信，在帖子上交流的方式是评论、点赞和点踩，用户可以在他人发表的帖子上发表自己的观点。

（7）个性化推荐模块是本系统的重点模块，该模块在用户发的帖子中提取关键字，并生成用户画像，为用户推荐可能感兴趣的人或兴趣圈。

## 4.2 数据库设计

数据库是一种完成数据存储和操作的工具。由于基于文本分类的社交系统需要采集、存储及处理大量的数据信息，所以对于数据的管理就尤为重要，一个系统的成功很大部分上取决于数据库是否合理的设计，如何设计出稳定、高效的数据库是每个系统设计人员需要着重考虑的，只有好的数据库设计才能使后期维护顺利进行。数据库设计是信息管理系统中的主要部分，数据库是存放数据的仓库。数据库设计是指对于一个固定的系统环境，尽量设计出最优的、最合理的数据库结构，并且不会对固定的应用环境有更高的要求，达到合理有效的存储和管理数据信息，使用户满意。

数据库的设计要满足数据库的信息录入、信息更新、信息删除等方面的需求。当用户想要查询信息时，会通过对系统某个模块的操作，对数据库内的数据进行筛选，以达到对数据的使用。

### 4.2.1 数据库设计原则

1、表中的记录有唯一的标识符

在设计数据库表的时候，应该有一个良好的习惯。用一个ID号唯一标识行记录，即ID作为标的主键，尽量不要通过名字、编号等字段对数据进行区分。每个表都应该有一个ID列，任何两条记录都不能共享同一个ID值，另外ID在值应该有数据库自动管理，而不能交给前台应用程序，否则的话很容易产生ID值不同意的状况。在本次设计中每个表都有ID列，作为主键，并以雪花算法生成。

2、数据库对象要有统一的前缀名

一个复杂的数据库设计，其对应的数据库表往往会以千记，而分清每张表的的作用或意义的难度会很大。这样，会使设计人员在数据库设计时，造成很大的困扰，甚至会减缓项目进度。本次毕业设计，虽然需要的表不多，但是仍然采用前缀名设计，以养成良好的习惯。每个管理模块的前缀名都要尽量不一样,尽量使设计人员看到表名，就理解其意义，即属于哪个管理模块的哪张表。

3、变量名的命名规则

（1）变量名采用小写英文字母

（2）变量命名尽量可以见词知意。

（3）变量名长度不超过30个字符。

### 4.2.2 主题数据库

主题数据库表如下所示：

**表4.1 主题数据库表**

|  |  |  |
| --- | --- | --- |
| 主题数据库 | 数据表 | 数据表的主要内容 |
| 用户信息管理 | 账号信息表 | 用户ID、用户名、昵称、密码等 |
| 个人信息表 | 用户ID、年龄、头像、性别、个人介绍、个人标签、在线状态等 |
| 聊天信息管理 | 聊天信息表 | 发送人ID、接收人ID、发送时间、发送内容等 |
| 兴趣圈管理 | 兴趣圈信息表 | 兴趣圈ID、兴趣圈名字、兴趣圈介绍、管理员ID、创建时间、兴趣圈标签等 |
| 用户-兴趣圈表 | 兴趣圈ID、用户ID、加入时间等 |
| 帖子管理 | 帖子信息表 | 帖子ID、发帖时间、帖子内容、点赞数、点踩数 |
| 评论表 | 评论ID、帖子ID、评论者ID、评论内容、评论时间 |
| 点赞/点踩表 | ID、评价人ID、被评价帖子ID、是否点赞、评论时间 |
| 收藏表 | ID、收藏人ID、被收藏帖子ID、收藏时间 |

### 4.2.3 数据库概念设计

根据对系统的需求分析和系统设计，需要找出本系统的相关实体。所谓实体，即客观存在并可以相互区别的实物，实体可以是具体的人、事、物，也可以是抽象的概念或联系。本系统的实体有用户实体、帖子实体、兴趣圈实体等等实体。

由于实体属性较多，故不再列出，实体属性可以参照论文下面的数据库逻辑结构设计部分。

1. 用户实体E-R图

用户实体主要包括用户的基本信息，由于使用频率的差异，在本系统中将用户实体分为两个表来存储，一个表是用户登录信息表，其中只存储了用户ID、用户名、用户密码、昵称和头像url这些最基本的登录信息，而更详细的信息被存储在用户详细信息表中，包括用户年龄、性别、自我介绍、个性标签和在线状态。

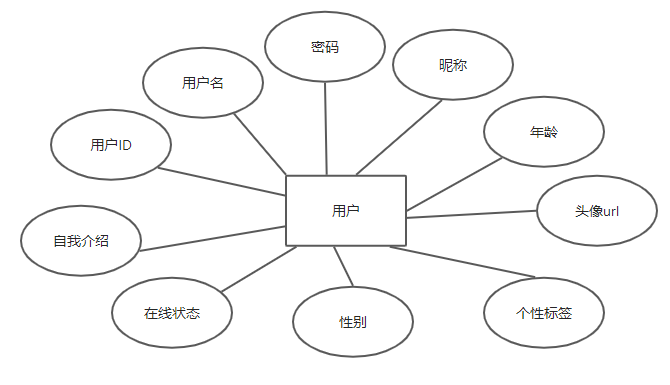


图4.2 用户实体E-R图

1. 帖子实体E-R图

帖子实体E-R图如图所示，帖子是用户进行信息展示的主要载体，帖子表存储的信息有：帖子ID、发帖人ID、发帖内容、发帖时间、点赞数、点踩数和评论数。

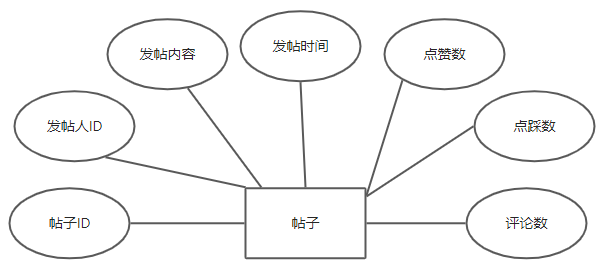


图4.2 帖子实体E-R图

3、评论实体E-R图

评论实体E-R图，如图4.3所示。评论表包括：评论ID、评论者ID、评论帖子ID、评论时间和评论内容，一篇帖子与它的评论结合起来形成一个帖子组件展示给用户。

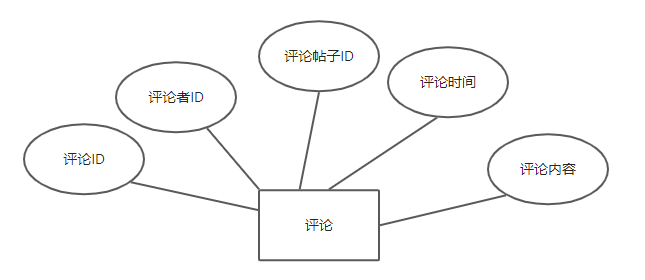


图4.3 “出库”实体-联系图

4、兴趣圈实体E-R图

兴趣圈实体E-R图，如图4.4所示。兴趣圈里聚集着有着相同兴趣的用户，用户可以单独在兴趣圈里发布帖子，认识好友，兴趣圈实体包括兴趣圈ID、兴趣圈名字、兴趣圈介绍、管理者ID、兴趣圈标签、创建时间和兴趣圈人数。

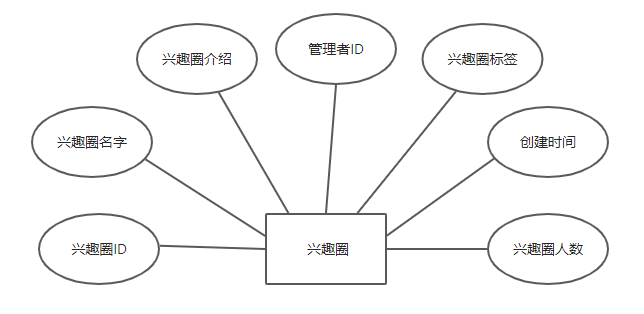


图4.4 “销售”实体-联系图

4、消息实体E-R图

消息实体-联系图，如图4.5所示。包括：消息ID、消息发送者ID、消息接收者ID、发送时间、消息内容

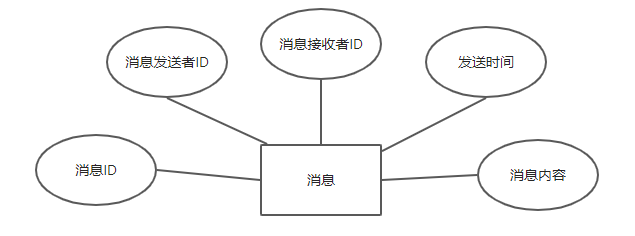
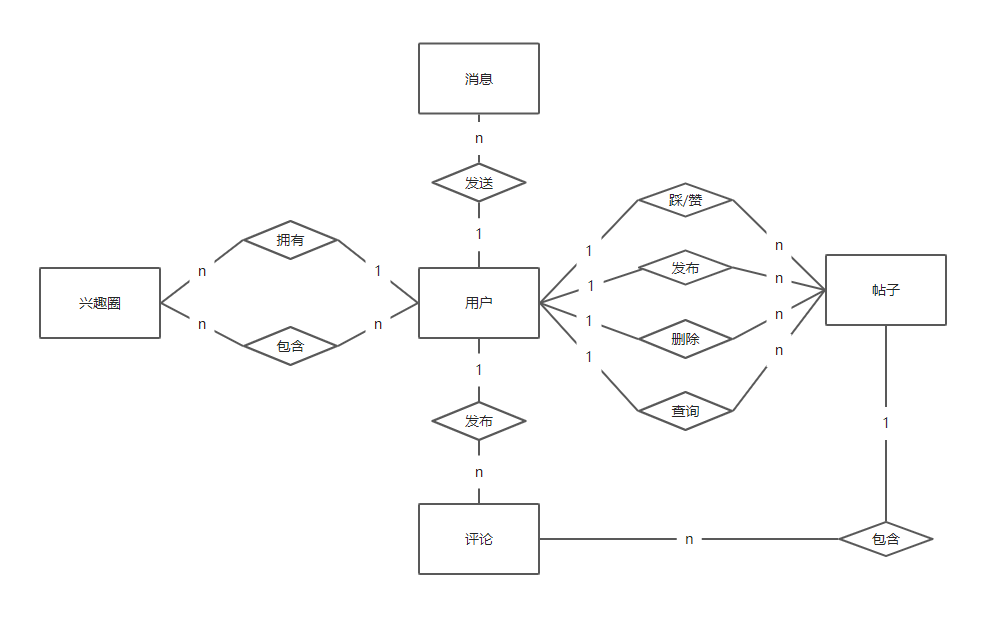


图4.5 消息实体-联系图

上述实体关系之间、每个实体存在着不同的对应关系，例如可能存在一对一、一对多和多对多的关系，为了清晰反映出各个实体之间的关系，总体E-R图如下图所示：



### 4.2.4 数据库逻辑结构设计

数据库的概念结构设计是根据社交系统的需求，需要按照某种排列对数据进行存储，概念模型设计阶段主要是根据需求分析抽象出事物的数据模型，然后再进一步通过E-R图来描述。概念结构是对具有相同特性事务的一种抽象，它独立于任何数据模型结构。而逻辑结构设计的主要目的是把概念结构阶段设计出来的E-R图转换为关系表、通过分析每个实体对象具有的属性和不同实体之间存在的关联关系，将概念模型转换成满足一定范式要求的关系模型。本社交平台关系数据库的设计遵循了一致性、完整性、可伸缩性、规范性、安全性的原则，同时采用了一些冗余字段提高数据库查询效率。总而言之，数据库逻辑结构设计，反映了数据库中所有表的详细结构和属性内容，即是对数据库表的设计。

下面根据以上的E-R图和实体分析，进行数据库表的详细分析。

1. 用户账号表

用户账号表主要定义了用户账号的相关信息，该表中的信息被使用的频率很高，因此将用户表分成用户账号表和用户详细信息表。

**表4.2 用户账号表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| username | varchar |  | 否 |
| name | varchar |  | 否 |
| password | varchar |  | 否 |
| avatar | varchar |  | 否 |

**表4.3 用户个人信息表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| self\_intro | varchar |  | 是 |
| age | int |  | 是 |
| gender | varchar |  | 是 |
| circles | mediumtext |  | 是 |
| managed\_circles | mediumtext |  | 是 |
| tags | mediumtext |  | 是 |
| status | bool |  | 是 |

**表4.4 兴趣圈信息表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| circle\_name | varchar(100) |  | 否 |
| intro | mediumtext |  | 否 |
| manager\_id | bigint | 外键 | 否 |
| created\_time | datetime |  | 否 |
| tags | mediumtext |  | 是 |

**4.5 用户-兴趣圈表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| circle\_id | bigint | 外键 | 否 |
| user\_id | bigint | 外键 | 否 |
| manager\_id | bigint | 外键 | 否 |
| join\_time | datetime |  | 是 |

**表4.6 关注表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| like\_id | bigint | 外键 | 否 |
| beliked\_id | bigint | 外键 | 否 |
| follow\_time | datetime |  | 否 |

**表4.7 帖子表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| author\_id | bigint | 外键 | 否 |
| publish\_time | datetime |  | 否 |
| content | mediumtext |  | 否 |
| vote\_num | int |  | 否 |
| oppose\_num | int |  | 否 |
| comment\_num | int |  | 否 |

**表4.8 评论表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| blog\_id | bigint | 外键 | 否 |
| commentator\_id | bigint | 外键 | 否 |
| content | varchar(1000) |  | 否 |
| comment\_time | datetime |  | 否 |

**表4.9 点赞/踩表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| commentator\_id | bigint | 外键 | 否 |
| blog\_id | bigint | 外键 | 否 |
| is\_like | bool |  | 否 |
| comment\_time | datetime |  | 否 |

**表4.10 收藏表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| collector\_id | bigint | 外键 | 否 |
| blog\_id | bigint | 外键 | 否 |
| collect\_time | datetime |  | 否 |

**表4.11 消息表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 主键或外键 | 是否为空 |
| id | bigint | 主键 | 否 |
| sender\_id | bigint | 外键 | 否 |
| receiver\_id | bigint | 外键 | 否 |
| time | datetime |  | 否 |
| content | varchar(1000) |  | 否 |

## 4.3 本章小结

本章主要是系统设计部分，首先画出了系统的功能结构图，介绍了商品销售管理系统主要功能。然后是数据库设计，说明了数据库设计原则，主题数据库、概念结构设计、逻辑结构设计。

# 5 系统实现与测试

## 5.1 基于文本分类的社交系统的实现

1、开发环境

操作系统：Windows10

系统位数：64位

Web处理器：Tomcat8.0

处理器：Intel Core i7

系统内存：8GB

数据库：MySQL8.0

1. 开发工具

开发语言：java(jdk8)、python(3.8)、javascript(ES6)

开发工具：Intellij IDEA、Git、MobaXterm、Pycharm

前端开发框架：Vue3、ant-design vue(Vue3)

后端开发框架：Springboot(2.6)、Mybatis(3.5)、WebSocket、maven

数据库：MySQL(8.0)、Redis

插件：Lombok、PageHelper、axios、vuex、vue-router、vuecli4

## 5.1.1 项目工程搭建

本系统后端主要使用Springboot搭建的Maven项目工程，在application.properties 文件中配置启动端口、日志级别、数据库连接和Redis配置。

使用maven的优点是可以在pom.xml文件中通过<dependency>标签引入项目所需要的各种依赖，maven会自动下载需要的jar包，不需要手动去下载jar包，使得开发引入外部依赖的过程变的简单便捷。

下图为后端项目工程的结构，其中doc文件夹下是文档文件，http文件夹下是进行后端接口测试的http文件，src文件夹下是项目代码文件，在stars包下是以功能分类的包，其中config包存放的是配置类，包括解决跨域问题的配置、Jackson工具类JSON解析的配置和拦截器配置等，controller包中存放的是后端接口controller类，domain存放的是实体类，exception包存放的是错误信息类，interceptor存放的是拦截器，例如登录状态拦截器，未登陆的用户不得访问未授权的功能，mapper包中存放的是dao类，用于向数据库中进行增删改查操作，req包中存放的是请求类，当前端向后端发送Post请求时，前端的参数会被封装成相应的请求类，resp包中存放的是回应类，用户发起的请求经过前端向后端发起请求，后端处理请求之后需要向前端返回相应的信息，将信息封装在resp回应类中，返回给前端，service包中存放的是业务逻辑类，后端的调用关系为controller根据前端的请求调用service，service再调用mapper从而完成相应的业务逻辑，这是典型的后端三层架构，util包中存放的是工具类，包括雪花算法等等，雪花算法用于生成id，雪花算法生成的id保证是唯一的，最后StarsApplication是该项目的启动类，在启动类中也可以进行一些配置，比如扫描包的设置，通过@ComponentScan注解即可设置能扫描到的包。

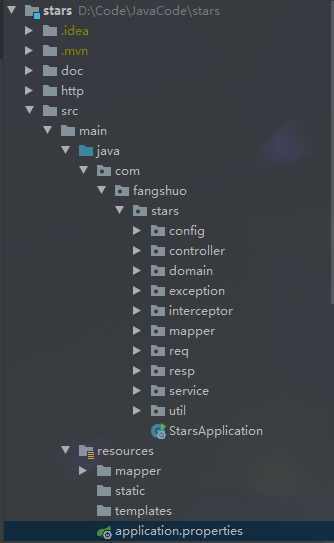


图5.1 项目后端结构

基于文本分类的社交系统首页，如图5.1所示。

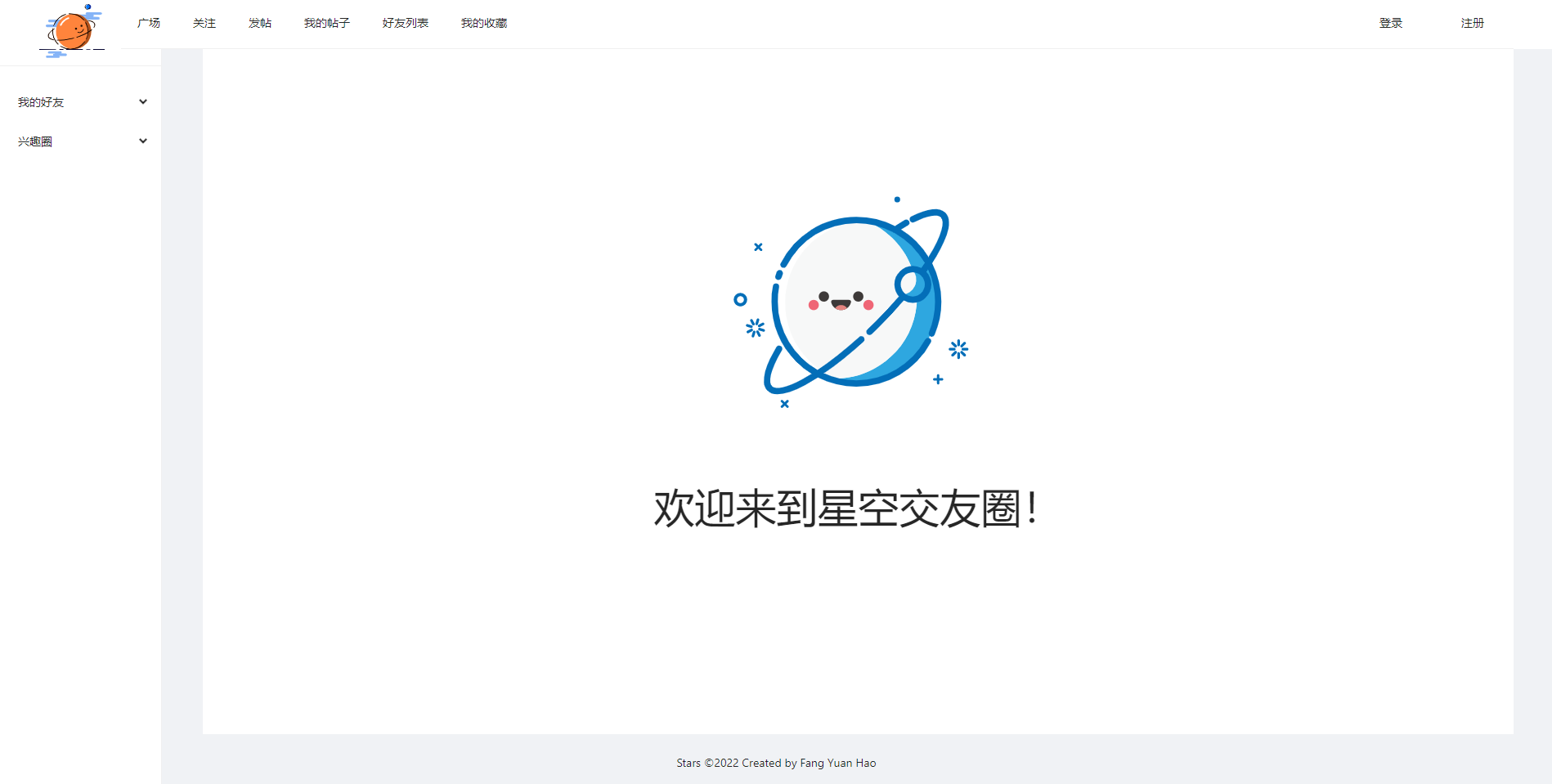


图5.1 社交系统首页

### 5.1.1登录注册界面

注册页面对于用户输入表单应当有一定的校验规则，该表单的校验规则为：用户名不能超过20个字符，密码不能少于6位且不能多于30位，确认密码与密码保持一致。

注册页面：

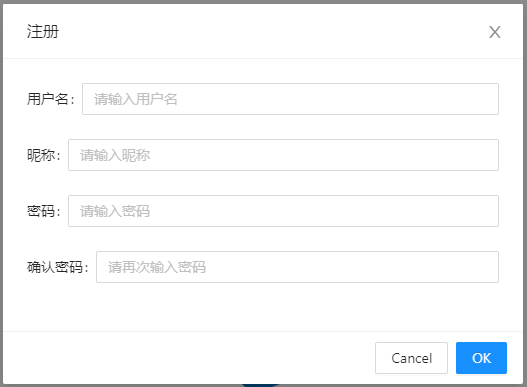


图5.2 用户注册页面

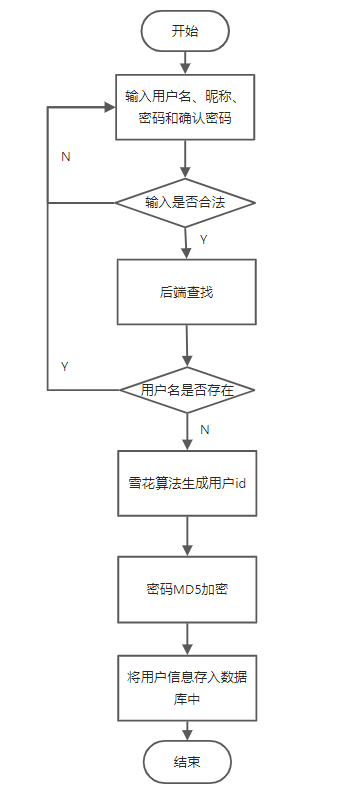


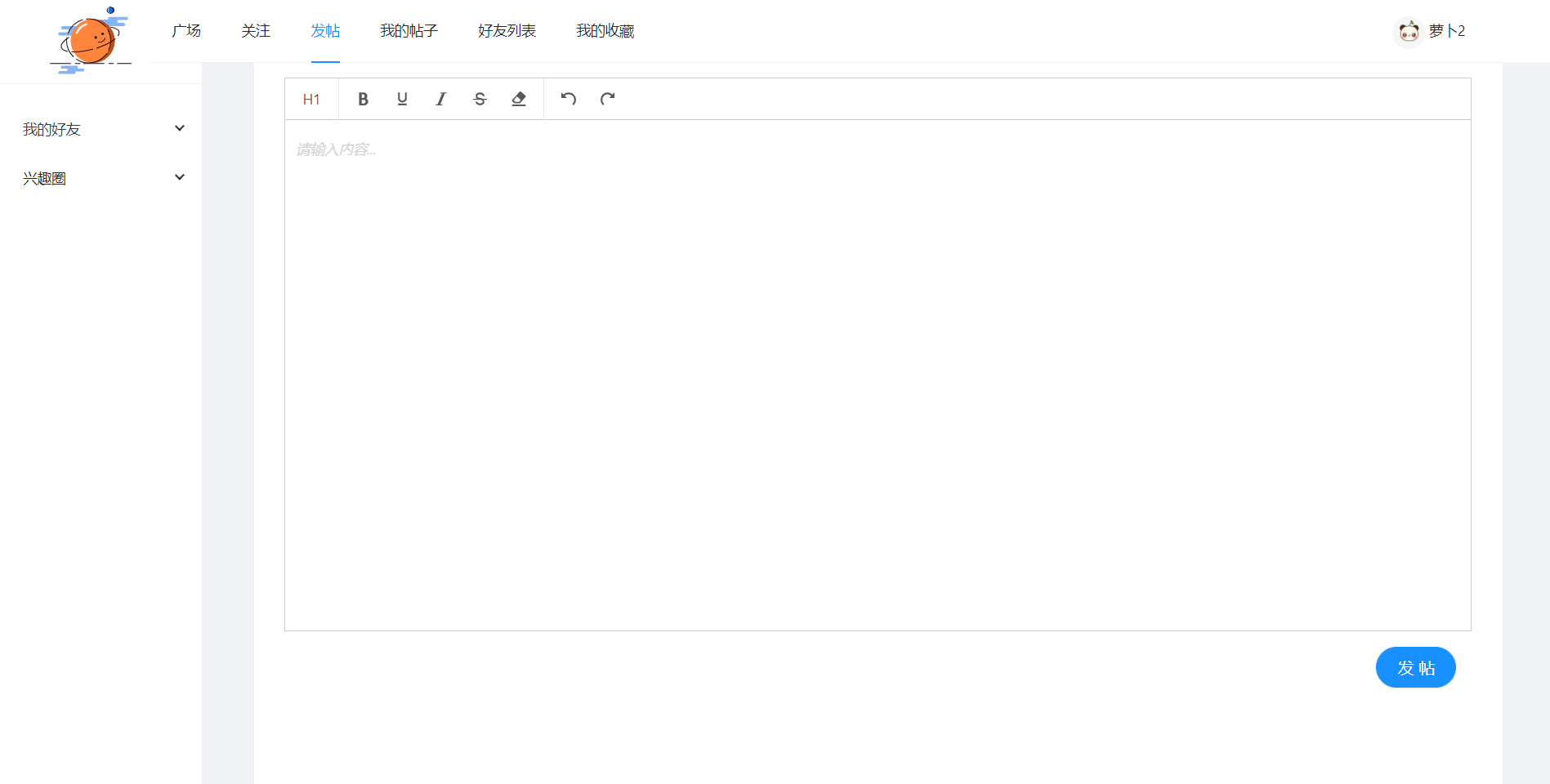
图5.3 用户注册程序流程图

登录页面，需要实现以下功能。通过输入正确的用户名和密码才能进入系统。登陆页面，如图5.4所示。系统登录程序流程图，如图5.5所示。



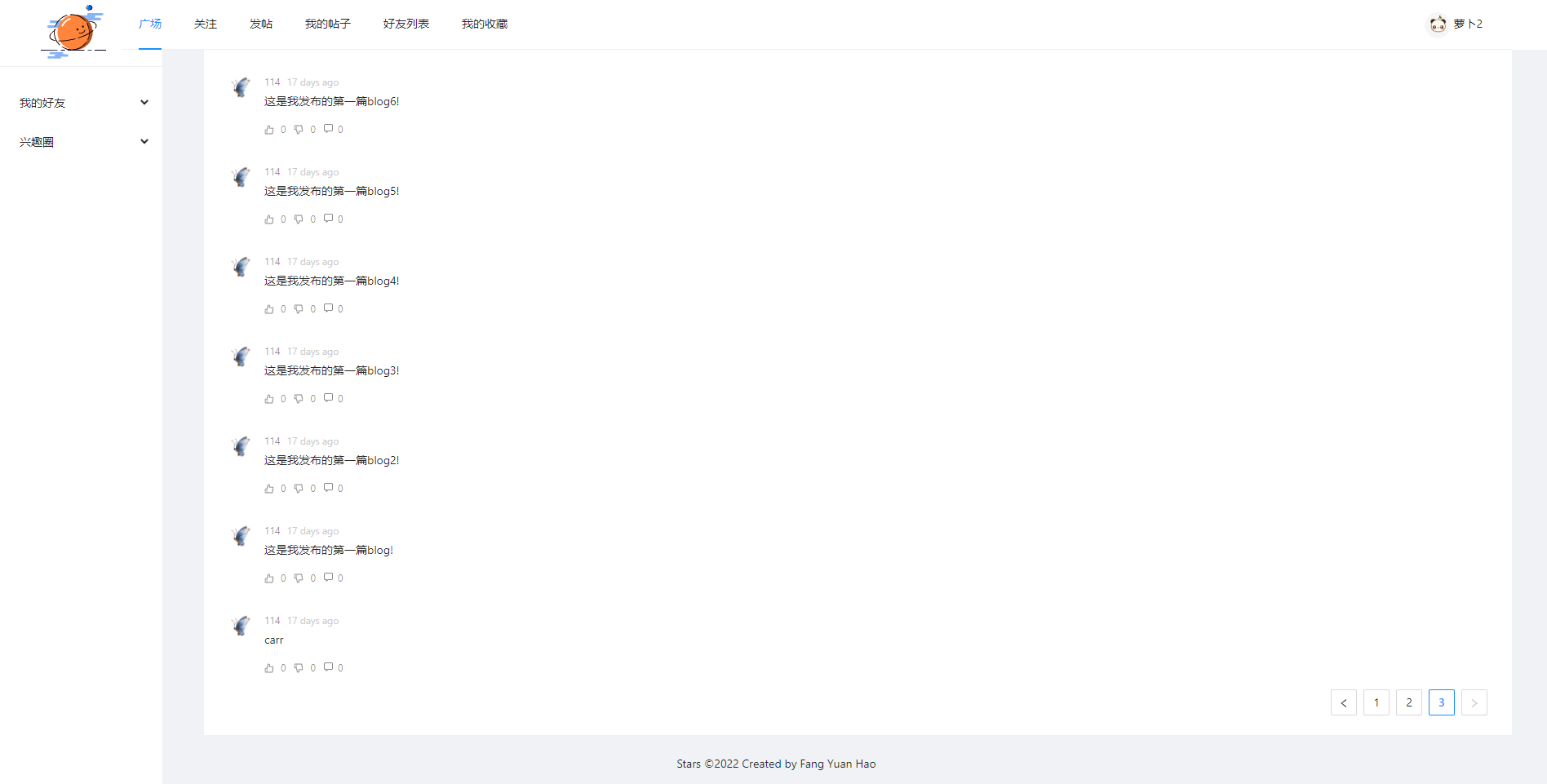
图5.4 “登录”页面

### 5.1.3 发帖页面



### 5.1.4 广场页面

在广场页面可以看到所有人的帖子



### 5.1.5 个人信息页面



## 5.2 基于文本分类的社交系统的测试

### 5.2.1 测试目的

每一个系统，在由系统开发人员设计时，都会仔细考虑以尽量减少其中的漏洞，但是并不能完全杜绝这种错误，因而才有了系统测试这一工作，系统测试并不仅仅是为了证明程序正确的执行了预期的功能，而是为了发现在设计过程中没有考虑到的错误而测试的过程。系统测试的主要工作是，通过与系统的 需求定义作比较，发现系统设计与系统需求定义矛盾或者不符的地方，验证软件系统的功能和性能等满足其在系统设计初所制定的要求。系统与系统设计初的需求定义作比较，主要从几个方面去考虑。（1）要确定预期输出结果，用事先写出的程序预期输出结果去与程序运行输出结果作详细的比对，发现其中存在的错误。（2）检查系统中是否实现在设计初定义的所有需求，是否与设计初定义的需求有不符之处。（3）如果系统实现了，在需求定义中没有定义的需求，要仔细考虑是否合理。从测试方法来看：系统测试可以使用黑盒、手工、自动化、静态、动态方法测试。在系统开发过程中，总会由于各种原因，会在开发的各个阶段产生错误，所以测试活动应该贯穿整个开发过程中，尽早发现错误，消除隐患。测试工作，尽量不要由开发人员完成，因为开发人员会主观的认为自己的开发是完美的，会无意识的漏掉某些方面，也会限定测试思路和测试方向，有很大的局限性。

### 5.2.2 测试环境

## 5.3 核心代码

## 5.4 本章小结

# 结 论

# 致 谢

# 参考文献

# 附 录