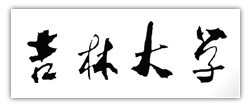


****

**本科生毕业论文（设计）**

**中文题目** 基于微信餐厅服务系统设计与实现

**英文题目** Design and Implementation of WeChat Restaurant Service System Server

**学生姓名** 余海涛 **班级**551702 **学号** 55170237

**学 院** 软件学院

**专 业** 软件工程

**指导教师** 王勇 **职称** 讲师

**吉林大学学士学位论文（设计）承诺书**

本人郑重承诺：所呈交的学士学位毕业论文（设计），是本人在指导教师的指导下，独立进行实验、设计、调研等工作基础上取得的成果。除文中已经注明引用的内容外，本论文（设计）不包含任何其他个人或集体已经发表或撰写的作品成果。对本人实验或设计中做出重要贡献的个人或集体，均已在文中以明确的方式注明。本人完全意识到本承诺书的法律结果由本人承担。

学士学位论文（设计）作者签名：

2020年5月20日

摘要

随着移动互联网的飞速发展，传统媒体为了更加适应新时代的发展需求，迅速向网络媒体转型，于是市场上开始出现各类新闻客户端，以争抢用户流量为核心，展开一场拉锯战。这十几年来，社会生活飞速变迁，很多人因为忙碌而不再去耐心的寻找自己需要的新闻。另一方面，在这种趋势下，人们已经从信息匮乏走向了信息过载，这个时候，如何从海量的信息中筛选出自己需要的那部分往往变成一件很困难甚至是不可能的事情。因此，信息生产者们开始将目光投向推荐系统，于是，利用用户行为进行数据挖掘的各种算法被投向市场并不断迭代，其中，以基于用户和物品的协同过滤算法使用最为广泛。

本次设计的app主要是基于Android平台，服务端使用spring mvc框架，首先客户端通过调用爬虫将百度新闻中的信息爬取下来显示到app上，并将其进行一系列处理，之后把这些新闻整理成JSON类型的数据传送到服务端并插入数据库当中便于之后的工作，接着利用这些用户行为信息通过协同过滤算法进行处理并将处理结果返回客户端的推荐栏并与之前的内容合并显示，系统根据用户的评论数量生成一个兴趣排行榜，用户可以在兴趣排行中看到自己最感兴趣的若干条新闻，同时系统也会根据排行榜以及用户的兴趣动态改变推荐一栏的新闻内容。

App名称为趣新闻，普通用户可以使用的功能包括新闻的分类浏览、用户浏览兴趣排行榜、新闻评论、新闻收藏、新闻搜索、历史记录查询；而管理员除了上述功能以外还可以进行新闻的发布，再次从百度新闻网站上爬取若干条新闻这两个功能。

**关键词：微信小程序、前后端分离、餐厅系统、spring boot、vue、layui**

Design and Implementation of WeChat Restaurant Service System Server

Author: Yu Haitao Tutor: Wang Yong

**Abstract**

With the rapid development of the mobile Internet, in order to better adapt to the development needs of the new era, traditional media quickly transformed to online media, so various news clients began to appear in the market. Over the past decade or so, social life has changed rapidly, and many people are no longer patiently searching for the news they need because they are busy. On the other hand, under this trend, people have gone from lack of information to information overload. At this time, how to filter out the part that they need from the massive information often becomes a difficult or even impossible thing. Therefore, information producers began to turn their attention to the recommendation system, so various algorithms using user behaviors for data mining were put on the market and iterated continuously. Among them, collaborative filtering algorithms based on users and items were the most widely used.

  The app designed this time is mainly based on the Android platform, and the server uses the spring mvcMVC framework. First, the client crawls the information in Baidu News by calling a crawler and displays it on the app, and organizes the news with user behavior into JSON type. The data is sent to the server, and then the user behavior information is used to process through the collaborative filtering algorithm and the processing result is returned to the client ’s recommendation bar and merged with the previous content for display. The system generates an interest leaderboard based on the user ’s comments and the number of favorites. Users can see some of their most interesting news in the interest ranking, and the system will also dynamically change the recommended news content according to the ranking list.

  The name of the app is Fun News. Ordinary users can conduct classified browsing of news, user browsing interest rankings, news comments, news collections, news search, and history query; while administrators can publish news in addition to the above functions The two functions of crawling several news on the Baidu News website. **Key words :** **Crawler, Android, Recommendation System,Collaborative Filtering, News，springmvc**

**目 录**

[第一章 绪论 1](#_Toc11175467)

[1.1课题背景 1](#_Toc11175468)

[1.2研究目的 2](#_Toc11175469)

[1.3研究意义 2](#_Toc11175470)

[1.4研究现状 2](#_Toc11175471)

[1.5研究内容 4](#_Toc11175472)

[1.6论文结构 5](#_Toc11175473)

[第二章 相关知识简介 5](#_Toc11175474)

[2.1 Andorid平台介绍 5](#_Toc11175475)

[2.2开发工具介绍 6](#_Toc11175476)

[2.2.1 IDEA介绍 6](#_Toc11175477)

[2.2.2 Android studio 介绍 7](#_Toc11175478)

[2.2.3开发环境的搭建 8](#_Toc11175479)

[第三章 概要描述 9](#_Toc11175486)

[3.1功能模块描述 1](#_Toc11175487)0

[3.1.1功能名称及对应视图展示 1](#_Toc11175488)0

[3.1.2功能介绍 1](#_Toc11175489)2

[3.2开发及运行环境 1](#_Toc11175490)3

[3.3准备内容 1](#_Toc11175491)3

[第四章 功能实现 1](#_Toc11175492)4

[4.1登录注册功能 1](#_Toc11175493)4

[4.1.1功能说明 14](#_Toc11175494)

[4.1.2数据库表结构说明 14](#_Toc11175495)

[4.1.3关键代码说明 14](#_Toc11175496)

[4.1.4实现逻辑 16](#_Toc11175497)

[4.1.5异常数据提醒 16](#_Toc11175498)

[4.1.6实现结果 17](#_Toc11175499)

[4.2 新闻的分类浏览功能 17](#_Toc11175500)

[4.2.1需求分析 17](#_Toc11175501)

[4.2.2可行性分析 1](#_Toc11175502)8

[4.2.3关键代码说明 18](#_Toc11175503)

[4.2.4实现逻辑 19](#_Toc11175504)

[4.2.5实现结果 2](#_Toc11175505)0

[4.3 新闻浏览排行榜功能 2](#_Toc11175506)0

[4.3.1需求分析 2](#_Toc11175507)0

[4.3.2可行性分析 2](#_Toc11175508)1

[4.3.3数据库表结构说明 2](#_Toc11175509)1

[4.3.4关键代码说明 21](#_Toc11175510)

[4.3.5实现逻辑 2](#_Toc11175511)2

[4.3.6实现结果 2](#_Toc11175512)2

[4.4新闻评论功能 2](#_Toc11175513)2

[4.4.1需求分析 2](#_Toc11175514)2

[4.4.2可行性分析 2](#_Toc11175515)3

[4.4.3数据库表结构说明 2](#_Toc11175516)3

[4.4.4关键代码说明 2](#_Toc11175517)3

[4.4.5实现逻辑 2](#_Toc11175518)3

[4.4.6实现结果 2](#_Toc11175519)4

[4.5新闻推荐功能 24](#_Toc11175520)

[4.5.1需求分析 24](#_Toc11175521)

[4.5.2可行性分析 25](#_Toc11175522)

[4.5.3数据库表结构说明 25](#_Toc11175523)

[4.5.4协同过滤算法说明](#_Toc11175523)......................................................................................................26

[4.5.5关键代码说明 2](#_Toc11175525)9

[4.5.6实现逻辑 3](#_Toc11175526)0

[4.5.7实现结果 3](#_Toc11175527)0

[4.6新闻收藏功能 3](#_Toc11175528)1

[4.6.1需求分析 31](#_Toc11175529)

[4.6.2可行性分析 3](#_Toc11175530)1

[4.6.3数据库表结构说明 3](#_Toc11175531)2

[4.6.4关键代码说明 3](#_Toc11175533)2

[4.6.5实现逻辑 3](#_Toc11175535)2

[4.6.6实现结果 3](#_Toc11175536)3

[4.7新闻搜索功能 3](#_Toc11175537)3

[4.7.1需求分析 3](#_Toc11175538)3

[4.7.2可行性分析 3](#_Toc11175539)4

[4.7.3关键代码说明 3](#_Toc11175542)4

[4.7.4实现逻辑 3](#_Toc11175543)4

[4.7.5实现结果 3](#_Toc11175544)5

[4.8爬虫部分 3](#_Toc11175545)6

[4.8.1需求分析 3](#_Toc11175544)6

[4.8.2可行性分析 3](#_Toc11175544)6

[4.8.3具体操作步骤 3](#_Toc11175544)6

[4.8.4关键代码说明 4](#_Toc11175544)0

[4.8.5实现逻辑 4](#_Toc11175544)1

[4.9新闻发布功能 4](#_Toc11175571)1

[4.9.1需求分析 4](#_Toc11175572)1

[4.9.2可行性分析 4](#_Toc11175573)1

[4.9.3数据库表结构说明 4](#_Toc11175574)1

[4.9.4关键代码说明 4](#_Toc11175575)2

[4.9.5异常数据提醒 4](#_Toc11175576)2

[4.9.6实现逻辑 4](#_Toc11175544)2

[4.9.7实现结果 4](#_Toc11175544)2

4.10历史记录查询功能..................................................................................................................43

[4.10.1需求分析 4](#_Toc11175544)3

[4.10.2可行性分析 4](#_Toc11175544)3

[4.10.3数据库表结构说明 4](#_Toc11175544)4

[4.10.4关键代码说明 4](#_Toc11175544)4

[4.10.5实现逻辑 4](#_Toc11175544)4

[4.10.6实现结果 4](#_Toc11175544)4

4.11个人信息管理功能..................................................................................................................45

[4.11.1功能说明 4](#_Toc11175544)5

[4.11.2关键代码说明 4](#_Toc11175544)5

[4.11.3实现逻辑 4](#_Toc11175544)6

[4.11.4实现结果 4](#_Toc11175544)6

[第五章 总结与展望 4](#_Toc11175577)7

[5.1总结 4](#_Toc11175578)7

[5.2展望 4](#_Toc11175579)9

[参考文献 5](#_Toc11175580)0

[致谢 5](#_Toc11175581)1

1. **绪论**

## 1.1概述

自2014年我国开始大规模地部署4G网络以来，互联网便开始从pc互联网朝着移动互联网方向高速发展，与此同时，O2O的模式也随之萌芽且逐渐发展成熟。随着网络普及率的提高，以及移动设备互联网应用的深入，“在线点餐”课题已经出现，这类服务开始在北京、上海等经济发达的一线城市开始试运行，并受到了餐饮行业商家和消费者的一致欢迎。至今，移动互联网、O2O模式、智能移动端管理系统的概念已经发展完善，餐饮市场上各式各样的智能下单服务如雨后春笋般地在我国各大城市涌现而出，手机、平板电脑等移动智能设备和二维码出现在了大部分餐厅的点餐桌上，“扫码下单”也成为了餐桌上的标语，这标志着智能服务系统正式普及性地走进了餐厅。

本文将简要介绍我国餐饮行业近年与互联网结合发展情况以及研究课题的现状，阐述智能服务系统解决方案为餐饮行业发展提供强大动力的事实。然后会介绍如何搭建一个前后端分离的B/S结构餐厅服务系统（系统包括两个B端一个S端。B端分为餐厅食客使用的微信小程序，和餐厅工作人员使用的后台管理系统网页。S端为运行在linux服务器上，以spring boot为主体框架搭建的负责数据存储和业务处理的服务器端程序）。本文将重点论述该基于微信餐厅服务系统的功能设计思想和实现过程、程序中数据的处理和存储的方式以及多端信息交互的方法。其中会用到目前较新的技术框架和方案，提供最为便捷的开发思路和体系，并会总结我在开发过程中得到的经验，为其他开发者提供有效的参考。

## 1.2背景介绍

餐饮业是重要的服务行业，受到国家的高度重视，因为饮食直接关系到我们人民的生命健康和生活水平。餐饮业是最贴近人民日常生活，最能影响人民生活质量的行业之一。2020年公布数据表明，我国餐饮业零售额度为12352亿元，体现我国餐饮行业规模之大。相比2019年的10345亿元，同比增长19.4%，体现我国人民生活质量提高，更多人选择餐厅就餐，在饮食方面消费的投入增加。餐饮行业发展如此迅速，离不开“在线点餐”对餐饮行业服务和管理高效化的技术支持。若餐厅还是延用传统的营业模式，必然不会有当今餐饮行业的井喷式发展。

目前餐饮服务行业与互联网结合的服务系统产物大体可分为两种。

一种是以美团，饿了么为首的外卖服务软件，这类软件的出现和普及给餐饮行业带来了跨时代意义的改变。它们出现之前，市场上存在的外卖服务十分稀少，就算有也基本都是由商家独立提供，即食客打电话给餐厅订外卖，餐厅派自家员工为食客派送。这种外卖形式受限极大，能有商家订餐电话的食客必然是曾经到店用餐过的食客，这使得食客的订餐成本提高，选择权受限。而外卖软件和商家入驻普及后，食客可以通过外卖软件，以图文并茂的形式，看到多家餐厅的菜品，并且还能看到其他食客对餐厅和菜品的评价，因此大幅扩大了食客的选择范围。近年外卖行业在我国飞速发展，大量商家和外卖骑手加入外卖行业，为我国提供众多的就业创业机会，为我国经济发展提供了强大动力。从2011年开始，外卖行业在中国兴起，经过几年的发展，于2019年中国外卖行业的市场规模就达到了惊人的5779.3亿元。2020年受到疫情影响，传统餐饮销售和客流明显下降的情况下，外卖行业逆势上扬，新增外卖相关企业670000多家。虽然外卖行业发展迅猛，但由于我国的餐饮行业监管制度没有跟上变化，外卖行业的负面问题也层出不穷，卫生条件差，料理包加热出餐等问题成为消费者谨慎点外卖的理由，还是有很多消费者选择门店就餐。

企业的创新需要跟上时代的变化，即使是小小的餐饮店也应如此，餐饮行业与互联网结合的第二种产物便是餐厅的在线自助点餐系服务统，对于线下餐饮门店，传统餐饮门店的管理和服务多为纯人工服务管理，对于市井小店，由店员用纸币记录点菜单进行服务。若是较为前沿的餐厅则可能采用柜台机进行餐厅管理，这种系统多为封闭的，不与客户交互，全由店员操作的嵌入式系统。消费者与柜台的信息交互还需要服务员充当媒介，即服务员记录客人点餐，再将订单递到柜台，由收银员添加订单，通知后厨制作，这种模式的效率显然是较低的。在线自助点餐服务系统将成为主流。在提供自助点餐服务的餐厅，消费者进入餐厅后，只需找个喜欢的位置坐下，拿出手机扫描餐桌上的点餐二维码，即可看到餐厅菜单，相比传统的纸质菜单本，手机点餐可更灵活地编辑图片文字信息，还可让消费者添加评价信息，使菜单信息内容更加丰富，且更具有参考性。

## 1.3微信点餐的优势

相比传统的餐厅服务模式，“扫码点餐”模式有着多方面的优势。首先对食客而言，传统就餐流程需要呼叫服务员，用纸质菜单点餐，服务员在旁记录点餐内容，食客再与服务员核对，等待服务员将点餐单送至收银台处，由收银员手动下单，这才完成点餐步骤，食客最后还要到收银台结账支付，才完成整个用餐流程。而“扫码点餐”模式中，食客只需一部手机即可独自完成从点餐、下单最后支付的所有步骤，这即优化了点餐步骤也同时提升了食客的就餐体验，让食客在这个生活节奏快速的时代节约了宝贵的时间。而在餐厅店家方面，我调查采访了市里多家使用“扫码点餐”服务方式的餐厅，结果表示：“扫码点餐”模式减少了服务员的服务流程，使得同样规模的餐厅需要的服务员人数减少，节省了约30%的人力成本。点餐流程的优化，让食客的用餐时间平均减少约10分钟，进而提高了餐厅的翻台率，使餐厅平均增加约30% 的营业收入，这个数字在节假日餐厅生意火热的时候则会更高。此外，将餐厅服务系统引进餐厅，还为餐厅管理带来了许多便利。例如，使用服务系统可以更高效快速地自定义编辑餐厅的菜单，餐厅管理员在后台管理系统上对菜单内容修改后可以直接反映到食客手机上看到的菜单界面上，而不像传统方式一样，若更改菜单内容则需要在多本菜单本上涂改，甚至重新打印菜单，这也为餐厅节约了成本。

既然是“扫码点餐”模式，那么可选的实现方式有这以下几种：Web应用、移动客户端以及小程序。Web应用的好处是开发难度低，但实际中基于网页的方式并不能为用户带来良好的体验。移动客户端虽然可以给用户最优的使用体验，但也提高了了用户的使用门槛，必须下载该移动APP才能使用，这在一些临时就餐场景中是十分麻烦的，且为了适配主流的两种移动设备系统Android和IOS，开发者需要开发两个版本的客户端，这大大增加了开发成本，所以也并不适合使用。综合考虑后决定，选择使用小程序的形式实现客户端的点餐系统，它能带来不错的用户体验，无需下载，点击即用，降低了用户的使用门槛。此外还具备强大的跨平台能力，大大降低了开发难度和成本，缩短开发时间，提升开发效率。而小程序中亦有多种选择：微信小程序、支付宝小程序、快应用都可以做到点击即用。但是快应用不支持IOS首先被淘汰；支付宝虽是应用最广泛的支付平台，有7.11亿活跃用户（2020年数据），但它的业务更倾向于单纯的支付，用户黏度不如微信。微信作为最火热的社交应用，拥有更大的用户基数，更强的社交性质，更大的用户黏度，选择微信平台开发小程序最为合适的做法。

## 1.4研究意义

社会变化至今，不论是工业生产活动还是日常生活活动，都朝着更高效、更高品质的方向发展，人们的生活节奏越来越快，而饮食是每一个人在每一天都需要面对的问题，餐饮行业也必然最求高效化、高品质化的解决方案，传统的餐饮作业模式已经不适合现在这个科技改变生活的年代，让餐厅数字化、网络化是时代发展的趋势。这次课题研究将研发一套为餐厅管理员和消费者使用的餐厅服务系统解决方案，让实体门店餐饮行业也感受到移动互联网发展带来的便利。

## 1.5研究现状

前面已经分析了消费者客户端技术路线的选择策略，选择使用最符合开发和运作需求的微信小程序客户端形式。微信小程序开发更多是使用微信公众平台提供的官方小程序原生框架mina，它由微信官方提供，系统功能强大，结构轻便简洁。它提供了视图层元素的描述方法和风格定义方法，也给出了逻辑层处理的结构定义，可简单直接地处理用户的操作，同时微信官方还提供可满足大部分开发和应用场景的系统API供调用。

关于餐厅员工使用的餐厅后台管理系统，选择采用web页面形式进行前端独立开发。如今前后端分离开发已经成为主流，Vue作为简单易学的前端轻量级框架，存在强大的数据双向绑定功能，拥有组件化开发，数据与结构分离的先进思想，同时有性能优异，运行速度快的优点，成为前端开发的热门框架。而Layui拥有丰富的简洁美观的前端组件库，模块化编程，兼容性强，是为开发人员提供的优秀ui框架。于是餐厅后台管理系统前端页面决定采用Vue+Layui的技术路线进行开发。

服务器端的Java web应用开发从最早的Servlet技术开始已经经历了许多变化，先是java Servlet 作为一种改进的CGI（公共网关接口）出现，解决了每次请求都要启动一个系统进程（一个JVM）的问题，大大降低了系统开销。后来Web编程脚本语言（PHP/ASP/JSP）兴起，JSP（java server page）也随之出现，它的主要作用还是动态展示网页内容。随着业务的复杂化，设计模式对开发的效率作用更为突出，各路框架横空出世，MVC、ORM成为了主流，其中spring+springMVC+Mybatis便是MVC、ORM模式的一个优秀经典实例。spring boot作为spring框架的扩展，消除了大量繁琐的XML配置，内嵌入了Tomcat服务器且不需要手动部署，同时利用注解完成了大部分自动配置功能，大大提高了web应用的开发效率，于是本文选择使用spring boot作为服务器后端的主体框架。

## 1.6论文结构

本文通过调研使用场景，分析了各类相关的实现技术，选择了合适的开发工具，通过查阅相关书籍、论文以及网络课程，最终实现基于Android的今日头条app的设计与开发——趣新闻，并已适配Android 9，因此Android 9及以上版本的手机均可以安装运行。

本文是《基于Android的今日头条app的设计与开发》，论文的组织结构如下：

第一章绪论，介绍研究背景、研究目的、研究意义、Android应用开发、爬虫以及推荐算法相关内容，期间查阅了许多资料。

第二章相关知识简介，介绍本次开发过程中所能够涉及到的相关技术、语言、框架等知识，主要包括Android开发相关内容、爬虫的应用、spring mvc框架的配置和使用以及基于内容的协同过滤算法的原理简介。

第三章概要描述，介绍了大致的设计思路，还有系统的各个模块和整体框架。

第四章功能实现，本章将对各个功能模块和算法实现原理进行详细的说明，按照模块来进行分析，包括需求分析，可行性分析，功能描述，设计实现等。

第五章总结与展望，对本次毕业设计《基于Android的今日头条app的设计与开发》一题中存在的问题进行复盘，分析其中难点以及解决的策略，总结有用的知识。

**第二章 相关技术介绍**

## 2.1 Apache Maven

### 2.1.1 Maven简介

Maven，若翻译成中文，是“专家”、“内行”的意思，正如它的名字一样，他在项目中充当着“管家”的地位。它是著名的Apache软件基金会用纯Java语言开发的一个开源项目(<https://maven.apache.org>)，是一个广泛使用的软件项目管理工具。Maven尤其适合Java项目，利用Maven可以清晰方便地构建约定熟成的Java项目目录结构，管理和导入项目依赖，以及配置项目编译打包方式等。

### 2.1.2 Maven安装和配置方法

Win 10操作系统的开发环境下：

1. JDK安装并配置环境变量，过程省略。
2. 前往Apache Maven官方页面选择合适版本的maven程序进行下载。
3. 解压下载的压缩包，安装到合适目录下，例如“D:\Java\apache-maven”目录，保证路径中没有中文字符。
4. 进入我的电脑-高级系统设置-环境变量设置页面，新增环境变量MAVEN\_HOME，赋值为maven的安装路径，即“D:\Java\apache-maven”。再编辑Path环境变量，在后追加“%MAVEN\_HOME%\bin;”内容。添加完成后在CMD中输入“mvn --version”命令，能正确识别并显示Maven版本信息时，表示环境变量设置成功。
5. 配置Maven本地仓库：新建文件夹“D:\Java\apache-maven\repository”作为本地仓库位置，然后打开“D:\Java\apache-maven\conf\settings.xml”文件，在settings节点内添加localRepository节点，值为本地仓库文件夹路径，至此就配置好了本地仓库。当我们的项目需要某个jar包，会向maven获取，maven首先向本地仓库查询，找到则返回；若未找到，maven会从远程仓库下载jar包，并保存在本地仓库中。
6. 国内网络下，为了提高远程仓库访问速度，可以将远程仓库设置为国内镜像­，我国提供maven镜像服务的机构有阿里云、腾讯云、网易等。

## 2.2 Spring Boot

### 2.2.1 Spring 系列简介

说到Spring Boot，就得先谈谈Spring家族。在Spring框架出现以前，J2EE领域从最初的Servlet开始，经历了许多的变化。传统J2EE项目中，使用Servlet、JSP、JavaBean、JDBC、EJB、JSF、JNDI等技术，逐渐形成和定义了经典的三层组件结构：客户端上的客户层组件、J2EE服务器上的Web层组件、J2EE服务器上的业务层组件。但这时候，web应用的开发还是基于Web Form的结构的，其视图和逻辑紧密耦合，页面是基于文件URL渲染的，这使得代码难以维护和扩展，非常不利于模块化业务的编写。正因如此，MVC的概念被人提出，即“模型-视图-控制器”的设计模式。

图2-1 MVC结构

MVC结构将页面视图与业务层分离，只需统一接口，彼此分开编码，互不影响，具有耦合性低，重用性高，易于维护，部署的优点。Spring MVC Framework则是MVC模式的一项优秀实践。

讲Spring MVC之前，先要介绍一下Spring的由来：Spring之前的时代，J2EE以SUN公司研发了EJB，Java后端都依照着Enterprise Java Bean的标准进行开发。但EJB运行在EJB容器服务器上，Servlet以及JSP却运行在Web容器上，为了让Web容器能够调用EJB内的业务功能，每开发一个EJB，都需要创建多个配置文件和接口，这样导致配置的工作量不亚于开发的工作量，这给测试人员也带来了更多繁文缛节的流程。同时Web容器通过网络调用EJB容器，必然十分依赖网络质量，这导致服务器的性能难以稳定。种种原因导致当时EJB的规范饱受开发者诟病。2002年，美籍作者Rod Johnson 在其著作《Expert One-on-One J2EE Design and Development》中指出EJB的缺陷，并提出了Spring的概念。

如今Spring已经是主流的Java Web框架，虽是一个轻量级的应用框架，但它同时具备卓越的性能和新颖的思想，因此受到众多后端开发人员的喜爱。Spring之所以卓越，是由于它的两大核心——IOC（Inverse of Control，控制反转）和AOP（Aspect Oriented Programming，面向切面编程）。IOC其实就是基于控制反转思想的一个Java Bean的容器，它取代了EJB臃肿和低效的开发模式，使Bean实例的创建不再由调用者管理，而是由Spring 的IOC容器接管，IOC容器会负责创建实例，接管实例的生命周期，并协调实例之间的依赖关系，完成依赖的自动注入。ApplicationContext就是Spring中的一种IOC容器，它比基础的BeanFactory多了一个功能，在初始化时对所有实例的属性进行是否已注入的自检，避免在使用实例时抛出空指针异常。但Spring不论是向IOC容器添加Bean还是为Bean注入依赖，都需要采用XML文件进行配置，使得步骤繁琐。AOP则基于动态代理设计模式的思想，目的是减少代码的冗余量，消除重复代码。Spring 的AOP中定义了“前置、后置、环绕、异常、引介”五种通知类型，将重复的代码置于通知中，使程序员更加专注于核心业务逻辑的编写，真正做到了模块间的松耦合。开发人员利用Spring的AOP更高效地实现事务处理、异常处理、日志管理以及权限控制等功能。但AOP也需要编写XML文件来配置切面、切入点等，相对繁琐。

介绍完Spring，回归到Spring MVC Framework。先前说到，MVC的目标就是将用户界面和业务逻辑分离，用控制层来协调匹配用户界面和业务逻辑之间的映射关系，大大降低两者之间的耦合性，提升代码的可扩展性、可重用性。Spring MVC与原有的Spring框架集成使用，给出如下方案：

Java web

前端 视图层

控制层

业务层

数据访问层

NoSQL

数据库

夹带AOP事务管理的模型层

图2-2 Spring MVC结构

Spring MVC利用Spring的IOC容器，将控制层、业务层、数据访问层的Java Bean存储在IOC容器中，规范管理了各个组件，并用控制反转的方式调用。Spring MVC也利用了Spring的AOP特性，为模型层扩展了非常方便实现的事务管理功能。相比之前更为火热的Struts2框架，Spring MVC的执行效率更高，与Spring框架的配合更加简便，于是后来逐渐取代了Structs2。

Spring和Spring MVC的出现，集成了Java web开发技术发展演化出的先进思想，对后端开发人员来说是更称手的工具，但Spring框架中存在大量的XML配置内容，让开发人员不光要用JAVA代码编写业务逻辑，还要实时修改配置文件内容，在适配第三方框架时，更是要新建各种xml、properties文件，这让项目的结构变得非常复杂，代码变得难以维护，当时人们称Spring为“配置地狱”。于是，在Java 5推出“注解（Annotations）”的新特性，Java 6.0允许自定义注解，Java 8支持多重注解等一系列Java版本的更替中，Spring开始将注解大范围应用到框架中，将复杂的配置过程改为Java代码内的声明式注解来实现，极大幅度地降低了项目结构的复杂程度，提高了编码开发的效率。虽然Spring有简化其配置过程，但简化得不够彻底，不够智能，真正彻底简化的是Spring Boot。

为简化Spring的搭建、配置和开发过程，Spring Boot横空出世，它由Prvotal团队研发，是基于Spring框架的扩展，它完整地提供了消除XML配置的解决方案，但又同时支持使用XML配置以兼容旧的开发模式。现在一个Spring Boot项目，可以只需要一个application.yml文件，即可完成Spring Boot服务应用的全部配置，例如程序的端口号，数据库源，数据库连接池，静态资源地址，服务响应最大时间……等等配置，都可以十分简洁清晰的写在application.yml文件中。此外，Spring Boot提供了“starters”风格的POM依赖，将各种开发环境场景封装，提供“开箱即用”的一站式服务解决方案。例如Spring Boot只需用Maven引入一个spring-boot-starter-web依赖，即可启动和运行web应用程序，而不像之前Spring需要同时引入spring-web和spring-webmvc依赖。如今，Spring已经拥有了非常丰富的工具链，许多优秀的第三方框架都会有一个“spring-boot-starter-xxx”依赖版本。Spring Boot的另一个优点是强大的Spring应用自动化配置功能，所有配置都可以采用注解的形式完成，添加一个配置Bean，只需在类定义前使用@Configuration注解；注入一个业务Bean，使用@Service注解；控制器内配置方法接口的Http路由，使用@PostMapping、@GetMapping……等注解。之前所有需要XML文件配置的模块，都支持在代码中直接插入注解的形式进行的方便设置。Spring Boot除了简化了开发过程中的步骤，还提供了灵活的部署能力。Spring Boot内嵌入了Tomcat、Jetty以及undertow三种服务器。在Spring Boot之前，部署web应用要首先配置Tomcat 服务器，将web应用打包成war包，再部署在服务器中运行。而在Spring Boot中，只需将项目打包成\*.jar文件，用java -jar \*.jar 命令即可直接部署，项目默认会运行在Spring Boot内嵌的Tomcat服务器上，也可以通过排除tomcat依赖，导入jetty或undertow依赖来更改默认的内嵌服务器。当然，如果你想要以传统的war包方式运行在Nginx或其他服务器上，只需在pom文件的packaging节点中设置值为war，即可轻松切换到打war包的方式。

一言以蔽之，Spring Boot凭借其自身轻量，使用简易，starter组件丰富而功能强大且易扩展的特点，是Java社区中最具影响力的技术框架。

### 2.2.1 Spring Boot 使用方法

介绍中也有提到，Spring Boot作为一个轻量级框架，只需用Maven导入jar包后即可使用。得益于Spring Boot的“starters”模式，starter组件的宗旨是“约定大于配置”，我们只需要按照约定的格式创建项目结构，组件会自动配置，无需操心。

1. 创建一个空白的maven项目
2. pom.xml文件中添加parent节点，引入spring-boot-starter-parent，如图：

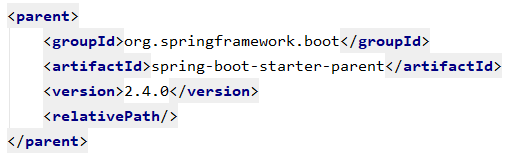


图2-3 spring-boot-parent

1. 对于一个java web项目场景，在dependencies节点内导入spring-boot-starter-web依赖即可，该依赖内部包含了web应用开发所需的必要依赖环境：spring-web、spring-webmvc、spring-boot-starter-tomcat等。

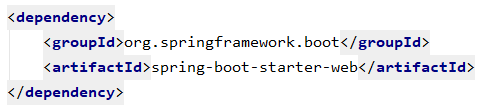


图2-4 spring-boot-web依赖

1. 创建spring boot启动类：



图2-5 spring-boot-web依赖

1. 至此，一个基本的Spring Boot项目已经搭建完成，运行启动类，即可在默认端口80上开启一个spring boot服务。在此基础上，可根据项目需求导入数据库访问、权限控制等组件，完善服务功能。

## 2.3 Mybatis

### 2.3.1 Mybatis简介

后端必然要与数据库交互，Mybatis便是一款简化与数据库交互过程的半自动化的Java持久层框架。它的特点是支持动态SQL、高级映射。

与Java最基本的JDBC相比：

1. Mybatis将SQL语句和Java代码进行了分离，Java代码定义了接口，SQL语句放在xxxMapper.xml的配置文件中，利用了参数映射方式，将Java函数的参数映射到SQL语句的变量中。同时通过输出映射，将查询语句的结果自动转化为Java对象。
2. Mybatis支持动态SQL语句，让SQL语句的构建更加灵活，减少了大量的代码冗余。

### 2.3.1 Mybatis使用方法

1. 用Maven导入mybatis-spring-boot-starter依赖，指定适当的版本号
2. 在application.yml文件中配置mybatis映射文件地址：

mybatis:

mapper-locations: classpath:mapper/\*.xml

1. 在项目源码目录和资源目录下均分别新建mapper文件夹
2. 源码目录mapper文件夹内新建Java接口，接口使用@Mapper注解标注，接口内声明SQL方法。
3. 实现SQL方法：
   1. 注解方式实现，在Mapper接口类中的SQL方法前使用@Select、@Insert、@Update、@Delete等注解，在注解内书写SQL语句。
   2. XML文件配置实现，在资源目录的mapper文件夹内新建与Mapper接口类同名的.xml后缀文件，在XML文件内用xml节点的方式配置实现SQL方法。该方式实现还需指定namespace映射到源码目录的Mapper接口下，并且每个<insert>、<select>、<update>、<delete>节点需要指定id值，且其id值必须与Mapper接口的方法名一致，

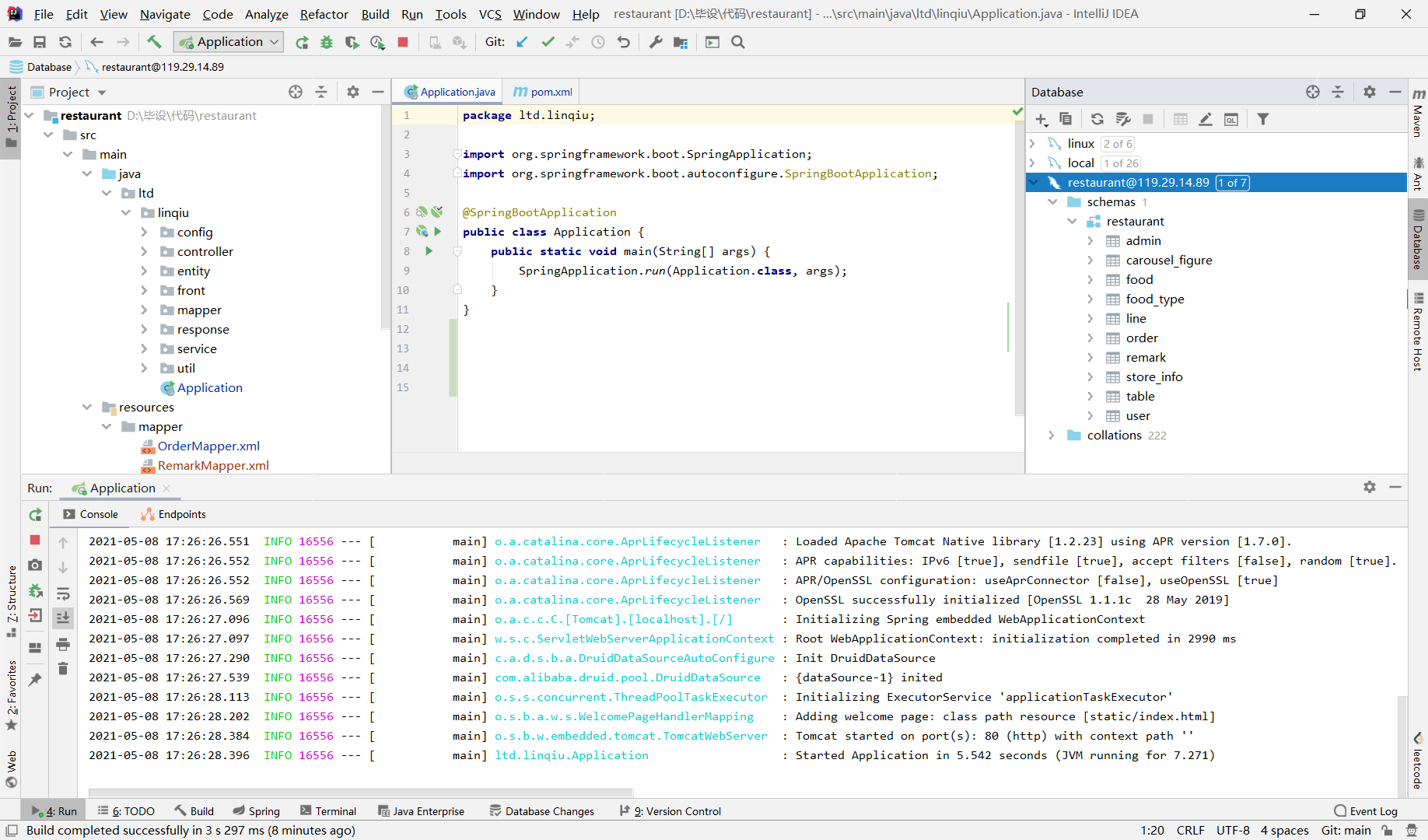
这两种方式均可以实现SQL方法，但是两者的适用场景不同，如果用注解的方式实现，更适合的是比较简单的SQL语句，这种方式更加简便，不需要创建多余的XML配置文件，只需在原本接口上加一行注解。但是注解方式也有弊端，当面临比较复杂的SQL业务时，注解方式书写SQL语句反而变得更加困难，也难以维护。例如当书写动态查询方法时，需要额外使用<script>节点将其他节点包括在内，出现引号和双引号时，还需要使用转义字符，而在XML文件内书写，格式更加清晰，也更容易维护。

## 2.4 开发工具介绍

### 2.4.1 IDEA

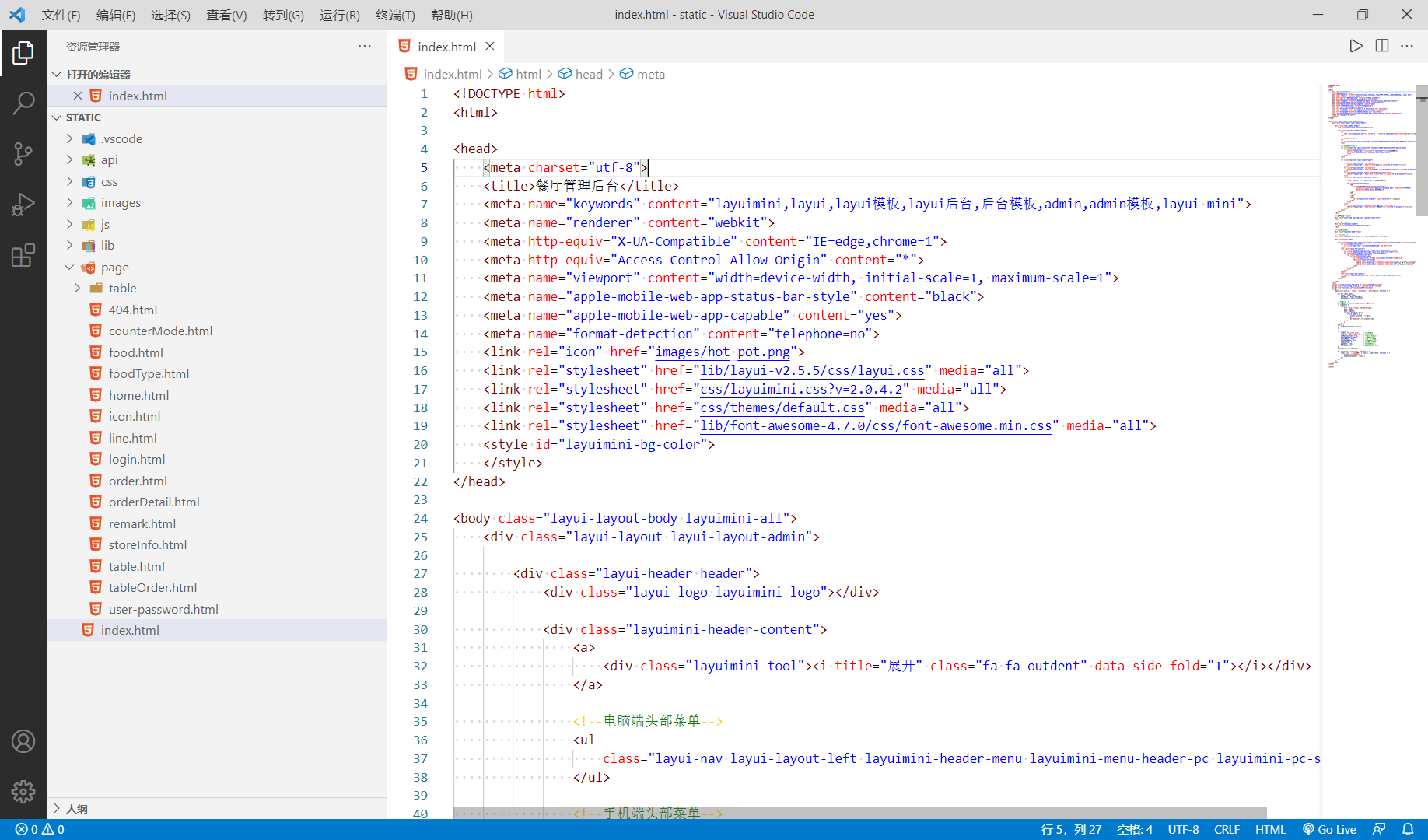
Idea是JetBrains公司开发的用于Java代码编写的集成开发环境，有最好的代码补全、内容格式化、代码高亮显示体验，在编码过程中，还能智能地提供代码改进策略。不仅如此，开发者为Idea开发了许多好用的插件，lombook就是具有代表性的一个插件，它让Java Entity的编写变得非常简单。

不仅是代码编写的体验上，Idea同时拥有开发过程中需要的大部分工具：支持Git智能管理，点击按钮即可自动提交git commit、git push、git pull等操作；支持SSH终端，不需要打开额外窗口即可通过SSH登陆云端的Linux服务器；支持可视化数据库连接和管理，在开发环境下即可直观的查看数据库表单内容，创建、编辑表单；支持SFTP协议连接，可将云端服务器的文件目录映射到本地IDE上，也可将本地主机的文件拖拽上传到云端服务器，大大简化了部署难度。不夸张地说，众多光环于一身IDEA是Java程序员最好用的集成开发环境。



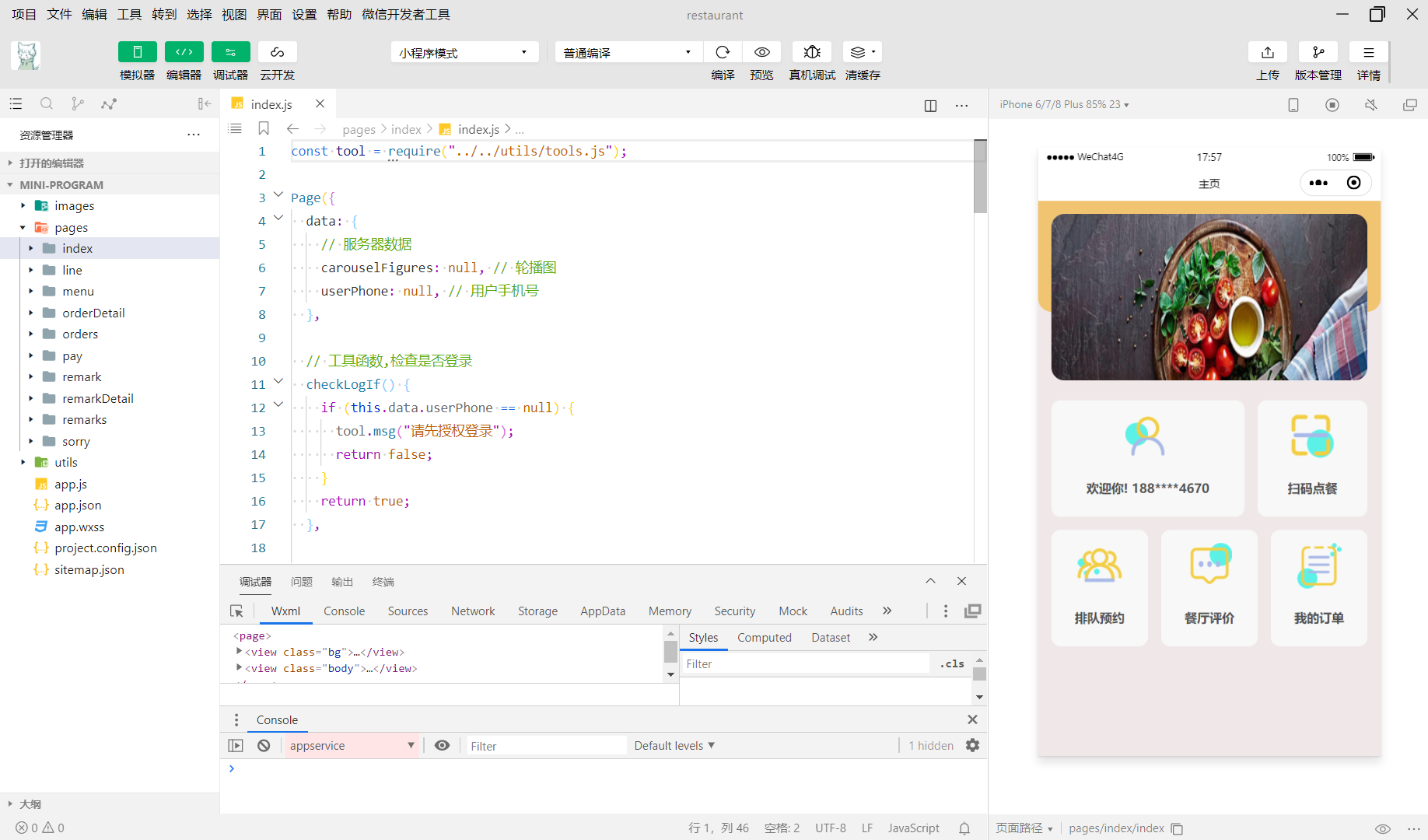
### 2.4.2 VSCODE

vscode作为一款非常轻量和简单的文本编辑器，十分适合用来编写前端代码。本次项目的前端由vscode配合其live server插件进行开发。



### 2.4.3 微信web开发者工具

微信web开发者工具由微信公众平台官方推出，可用于微信小程序和微信公众号网页的前端开发。提供了许多微信平台的API，如获取用户信息、发送Http请求、获取用户会话文件、调用用户摄像头进行二维码扫描等。最好用的是提供了移动端模拟器，可以直观看到项目运行在移动端时的状态。



**第三章 需求分析**

## 3.1食客用户的需求

对于开发人员来说，开发微信餐厅服务系统，开发人员的用户是餐厅的老板。而餐厅使用该微信餐厅服务系统的目标，是服务餐厅的用户，也就是前来就餐的食客。利用微信餐厅服务系统，餐厅能够为食客提供更便捷高效的服务，让食客更快更方便地点餐，更方便地结账，进而提升食客就餐体验，也同时提高餐厅的翻台效率，提升餐厅的口碑，最终为餐厅带来更高的收益。所以说，该微信餐厅服务系统最重要的环节在于对餐厅食客用户的需求满足，这也是判断该系统成功与否的关键。因此在对食客用户需求分析时，应当想得更加全面，更加具体。我在分析食客用户需求时，从一个开发人员的身份转变到一个食客的角度，考虑到以下食客用户需求：

* 进入系统：

作为一名食客，进入餐厅的目的就是吃饭，如果为了吃饭还要手机下载一个APP，这对食客来说是非常不便且不情愿的。因此进入系统的方式选择用微信扫描小程序二维码，直接无需下载即可进入系统。

* 用户登录：

食客使用系统需要有个身份标识，用来查看指定的就餐记录，那么用户登录就变得有必要了。最方便快捷的登录方式就是使用微信小程序API，用手机号码直接登录。

* 排队预约：

餐厅平常不忙时可以直接入座就餐，但若是餐厅生意火爆的时候，就需要有一个排队功能，让食客有序地就餐。并且排队功能应该区分不同的餐桌类型，让不同用餐人数的食客团体选择不同座位数量的餐桌进行排队，这样能最高效的安排餐厅资源。

* 浏览菜单：

系统应当提供一个清晰规整且易操作的界面，以分类的形式展示餐厅所提供的食物，食物应该提供完整的信息，包括食物图片、名称、价格、份量等。

* 购物车：

为了方便食客点餐，系统应当提供购物车功能，可以将菜品加入购物车中。

* 生成订单：

将全部要点的菜品加入购物车后，生成订单，显示收费内容，如餐品费用，可选服务费、餐位费等其余费用，并提供支付按钮。

* 历史订单：

食客可在小程序内查看自己在餐厅的消费记录，查看历史账单，账单还应该可分类筛选：进行中的订单、待评价的订单、已评价的订单。

* 餐厅评价：

订单完成后，食客可以针对这次就餐体验，进行评价，评价形式为评分+文字+图片。并且食客在小程序中，应当要能看到其他食客的评价。

## 3.2管理员用户的需求

对于餐厅管理员来说，使用该系统的方式是在

## 3.2开发及运行环境

硬件平台：常规搭载Windows的PC机

软件要求：

使用语言：JAVA

数据库：MYSQL

操作系统：win10专业版

IDE：IntelliJ IDEA、Android Studio

开发框架：spring mvc

## 3.3准备内容

在开发开始之前，我对于推荐系统、spring mvc框架以及Android开发都不是特别熟练，尤其是推荐系统之前没有接触过。因此我首先阅读了项亮所著的《推荐系统实践》[5]，了解了基本的基于用户和内容的协同过滤算法详细原理，书中所用算法都是用python编写，而且有官方的数据集可作为训练集来验证算法，我由于是个人开发，而且也找不到那么详实的官方数据或者调查问卷数据，因此就把主要原理抽出来用java代码重写了一下，在具体实现上做了一些简化，以此来更加方便的应用到我的项目中来。对于Android开发这一块，一直都不是特别熟练，这次开发是把大二曾经做的两个网课项目结合起来进行了简单的整理，因为当前app的界面设计架构大同小异，我就在之前的项目基础上进行了改进，形成了这个毕业设计项目的初始框架。对于spring mvc框架，通过其他小型项目的练习，熟练了使用方法。至此，开发本次项目所需的准备工作都已完成。

1. **功能实现**

## 4.1登录注册功能

### 4.1.1功能说明

本次设计的登录分为管理员和普通用户，管理员账号固定为10000，密码为123456，管理员账号是在一开始就写进数据库的，不需要额外注册。普通用户需要登录注册界面进行注册。管理员和普通用户登录后的界面是略有不同的，管理员可以发布新闻，爬取新闻，而普通用户则不具备以上权限，只能进行新闻的浏览、搜索和个人信息的管理。

### 4.1.2数据库表结构说明

User的集合（对应于mysql数据库的tb\_user表），表权限没有作限制，因为读写接口在应用中仅仅开放某个用户自己那一条。管理员账号不允许注销。如下表4-1：

表4-1：tb\_user表结构

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段说明** | **对应mysql的字段** |
| userNo | String | 用户编号（主键） | user\_no(varchar) |
| userName | String | 姓名 | user\_name(varchar) |
| password | String | 密码 | password(varchar) |
| userAvatar | String | 头像 | user\_avatar(text) |
| label | String | 标签 | label(text) |
| userSex | String | 性别 | user\_sex(text) |
| userBirth | String | 出生年月 | user\_birth(text) |

### 4.1.3关键代码说明

前端LoginActivity中的关键代码如下表4-2：

表4-2 LoginActivity关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onResume | 判断当前用户是否登录 |
| onClickView | 根据传入参数来判断跳转到注册页面还是登录页面 |
| onBtnLoginClick | 将接受的数据进行简单的判空操作，若不为空则将用户名和密码整理成JSON数据包，将其缓存后把数据提交到服务器 |
| onTvRegisterClick | 跳转到注册页面 |

前端RegisterActivity中的关键代码如下表4-3：

表4-3RegisterActivity关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onViewClicked | 根据点击的按钮来判定执行哪个函数 |
| onIvAvatarClick | 处理点击头像后弹出手机的选择相片或者拍照的对话框 |
| onRegisterSexClick | 处理点击性别选择后的动作 |
| onBtnRegisterClick | 处理点击注册按钮后进行的动作，将数据进行异常检查，无异常后将数据提交到服务器 |
| onTakePhotoClickListener | 根据传入的参数来判定执行调用相机的函数还是执行调用相册的函数 |
| onActivityResult | 回调函数 |
| execCamera | 调用手机的相机 |
| execGalery | 调用手机的相册 |
| showDatePicker  onMultiListDialogResult | 分别为调用年月日选择器、标签多选窗口 |

后端UserService中的关键代码如下表4-4

表4-4 UserService关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| addUser | 将从前端接受到的用户数据插入数据库 |
| updateUser | 更新用户数据 |
| deleteUser | 删除用户数据 |

### 4.1.4实现逻辑

注册和登录各一个界面，第一次启动app会自动跳转到登录页面，如果已经注册了账号就直接输入用户名和密码，点击登录后数据会通过spring mvc框架提交到后台服务器，服务器进行核对后将信息反馈到前端进行登录；如果没有注册过账号，就点击注册按钮，此时会跳转到注册界面，在每个项目都填写完毕后点击注册，数据提交到服务器后，后台将数据插入到数据库中，并反馈给前端显示注册成功，这时候用户可以使用刚才注册的账号来进行登录。

**4.1.5异常数据提醒**

异常登录情况如下表4-5：

表4-5 异常登录内容

|  |  |
| --- | --- |
| **异常原因** | **异常提醒** |
| 账号存在但是密码不匹配 | 登录失败，请检查账号和密码是否正确 |
| 账号不存在 | 登录失败，请检查账号和密码是否正确 |
| 账号或者密码输入为空 | 登录失败，请检查账号和密码是否正确 |

注册异常情况如下表4-6：

表4-6 注册异常内容

|  |  |
| --- | --- |
| **异常原因** | **异常提醒** |
| 账号经过检查已存在 | 账号已存在 |
| 头像为空 | 内容不能为空 |
| 账号为空 | 内容不能为空 |
| 密码为空 | 内容不能为空 |

### 表4-6 注册异常内容（续表）

|  |  |
| --- | --- |
| **异常原因** | **异常提醒** |
| 性别为空 | 内容不能为空 |
| 出生日期为空 | 内容不能为空 |
| 标签为空 | 内容不能为空 |

### 4.1.6实现结果

实现结果如下图4-1：

****

图4-1 登录注册实现结果

## 4.2新闻的分类浏览功能

### 4.2.1需求分析

我们在阅读新闻的时候，有时是漫无目的的，有时是有明确的需求的。例如最近新冠肺炎的新闻比较热门，用户对这种常规的新闻比较感兴趣，有的用户则只想看科技类的新闻。对于这类用户，我们就需要为他们把新闻分好类别，以便于他们可以根据自己的需求来阅读新闻。而对于另外一些用户，他们打开app只是为了看一下当前有什么新闻，并没有明确自己要看什么，这种时候就需要将各类新闻随机抽取一些整理到一个栏目中，供用户来观看。因此，在经过上边的分析思考后，决定对被抓取到app中的新闻进行分类。

### 4.2.2可行性分析

Android中有许多可用的滑屏切换实现方式，这里采用最为常见的viewPager +n\*Fragment的方式来组织大面板，而大面板中标签处理需要使用FragmentPagerAdapter来实现滑屏的切换，因为是官方封装好的库，所以只需要在创建标签页对应的适配器类，把需要覆盖的方法写好再添加自己的业务逻辑就可以很方便的实现标签页的效果以及滑屏切换。

### 4.2.3关键代码说明

前端部分如下：

MainActivity页面关键代码如下表4-7

表4-7 MainActivity页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| swicthFragment | 切换fragment页面 |
| onViewClicked | 根据点击的按钮来进行首页、排行榜、新闻发布、我的信息这四个tab页面的切换 |
| onClickMenuListener | 监听按钮的点击，将点击的按钮颜色加深，未点击的不作处理。同时监听发布按钮是否被点击，若被点击则去跳转到对应的页面进行处理 |

HomeFragment页面关键代码如下表4-8

表4-8 HomeFragment页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| initTabs | 初始化标签 |
| setSelectColor | 设置被选中标签的颜色 |
| onClickView | 如果点击了搜索按钮，就跳转到搜索页面 |

LifeFragment页面关键代码如下表4-9（NBAFragment、NewsFragment、PhoneFragment、RecommendFragment、TechnologyFragment这些标签页由于只是爬取的内容不同，其他地方完全相同，因此这里不再列出详细解释）

表4-9 LifeFragment页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onLoad | 从服务器获取文章列表 |
| onClickArticleClick | 点击某条新闻跳转进入新闻详情页面 |
| onLongClickArticle | 长按确定是否收藏该条新闻 |
| onFabRefreshClick | 通过Jsoup进行爬虫，将爬取的数据显示并提交 |
| onWarningConfirmDialogResult | 将点击了收藏的新闻通过服务器提交 |

ArticleDetailActivity页面关键代码如下表4-10

表4-10 ArticleDetailActivity页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| AddHistory | 将查看的这条新闻的历史记录写入数据库 |
| Onload | 根据文章编号把对应的新闻从数据库中取出并显示 |
| OnClickView | 根据传入参数判定是返回还是跳到评论 |

后端部分如下表4-11：

表4-11 后端部分关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| queryArticleList | 查询文章列表 |
| addCollect | 将前端要收藏的新闻添加到数据库 |
| addOrderList | 添加要发布的新闻到数据库中 |

### 4.2.4实现逻辑

首先switchFragment这个函数，通过传入的参数来判定当前要显示的是哪个tab，然后把当前的正在显示的tab隐藏,转去show需要显示的tab,返回的是一个FragmentTransaction类型的结果，在onViewClicked中传入这个要显示的页面参数，根据传入的id来判定填入的参数是代表哪个tab页面，至此实现底部栏根据tab进行切换；通过Fragment的FragmentPagerAdapter来实现滑动切换上部菜单，以此实现新闻的分类浏览；新闻内容通过爬虫取到数据之后整理成JSON对象后存入数据库，之后的前台显示是根据文章编号通过spring mvc来访问到后台数据库后提交给前台进行显示。关于数据爬虫的详细实现原理将在后边其他模块进行解释。

### 4.2.5实现结果

实现结果如下图4-2

****

图4-2 新闻分类浏览实现结果

## 4.3 新闻浏览排行榜功能

### 4.3.1需求分析

在用户刚刚使用本app的时候，由于系统中并没有很多用户的行为信息，因此此时的推荐就是随机的，没有根据的，这时候用户可以直接去浏览排行榜的新闻来获取资讯，因为排行榜的新闻是其他用户留下过行为信息的，排行是根据这些信息的数量来排序，显然这代表了这些新闻的热门程度，因为用户不会对一条自己完全不感兴趣的新闻作评论。当某些新闻的评论数很多的时候，我们有理由认为大多数人对这些新闻是可能感兴趣的，因此这时候整理出这样一个排行榜，有助于刚使用app的用户快速找到自己想要浏览的新闻，以此获得更棒的使用体验。

**4.3.2可行性分析**

因为许多用户可能会在阅读完一条新闻之后点击评论，那么通过统计一条新闻的评论数量就可以粗略判断这条新闻的热门程度，以此来生成一个排行榜。

### 4.3.3数据库表结构说明

Comment集合（对应于mysql数据库的tb\_comment表），详细定义如下表4-12：

表4-12 tb\_comment表结构

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段说明** | **对应mysql的字段** |
| commentNo | String | 评论ID | comment\_no(varchar) |
| userNo | String | 用户ID | user\_no(varchar) |
| articleNo | String | 文章ID | article\_no(varchar) |
| content | String | 内容 | content(text) |
| createTime | String | 创建时间 | create\_time(varchar) |

### 4.3.4关键代码说明

RankFragment页面关键代码如下表4-13

表4-13 RankFragment页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onLoad | 从数据库中取出评论信息并根据数量降序排列显示 |
| onClickRank | 监听排行榜的点击事件，用户点击排行榜新闻可以进入新闻详情页面 |
| queryRankArticleList（后端） | 查询数据库并将结果返回前端 |

### 4.3.5实现逻辑

首先用户点击排行榜按钮，此时前台的onLoad函数中，通过spring mvc将请求提交到服务器，服务器收到请求后执行queryRankArticleList函数，内部处理时候执行查表语句，将结果返回到集合中，再讲该集合整理整JSON对象，最后把结果返回到前端进行显示。值得注意的是每次点击排行榜按钮，系统都会去重新查询一遍数据库以获得最新的热门新闻排行榜。

### 4.3.6实现结果

新闻浏览排行榜实现效果如下图4-3所示：

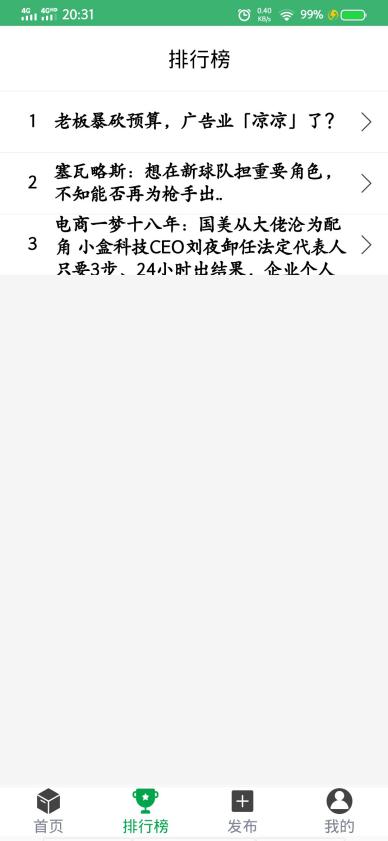
****

图4-3 新闻浏览排行榜实现结果

## 4.4新闻评论功能

**4.4.1需求分析**

许多用户在浏览新闻后，并不是退出就了事，而是期望看到其他人关于这个新闻中所透露出信息的看法，同时也希望就一些观点与他人进行沟通。因此，对于一款新闻资讯类app，提供评论功能就显得十分必要。同时，后台的开发人员也希望用户去评论，因为这样就可以收集到更准确的行为特征来分析用户对某些新闻的感兴趣程度，以便更好的服务用户。

**4.4.2可行性分析**

该功能为常规的界面设计布局和数据库调用。

**4.4.3数据库表结构说明**

Comment集合（对应于mysql数据库的tb\_comment表），详细定义如下表4-12：

表4-13 tb\_comment表结构

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段说明** | **对应mysql的字段** |
| commentNo | String | 评论ID | comment\_no(varchar) |
| userNo | String | 用户ID | user\_no(varchar) |
| articleNo | String | 文章ID | article\_no(varchar) |
| content | String | 内容 | content(text) |
| createTime | String | 创建时间 | create\_time(varchar) |

### 4.4.4关键代码说明

CommentActivity页面的关键代码如下表4-14

表4-14 CommnetActivity页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onLoad | 从数据库中取出当前文章评论信息并显示 |
| onClickView | 监听按钮，根据输入判定是返回还是评论 |
| onBtnSentClick | 处理发送评论的功能 |
| queryCommentList（后端） | 查询评论集合 |
| addComment | 增加评论 |

### 4.4.5实现逻辑

评论页面的实现逻辑比较直接，首先在点击评论按钮后拉起CommentActivity页面，此时系统会根据当前文章的ID去查找对应的评论列表并显示，当用户在输入框中输入了评论后点击评论，这时候会作一些判定，我们规定一个用户只能对一条新闻评论一次，因此这时候会去数据库中查询该文章对应的评论列表中是否有当前用户的ID，如果有的话会提示“已评论”，并不再更新评论列表；如果用户未对该新闻作过评论那么就将该请求通过spring mvc框架被服务端接受到，服务端将该评论相关信息插入数据库并将信息反馈给前端，前端再去更新评论列表。

### 4.4.6实现结果

新闻评论实现效果如下图4-4所示：

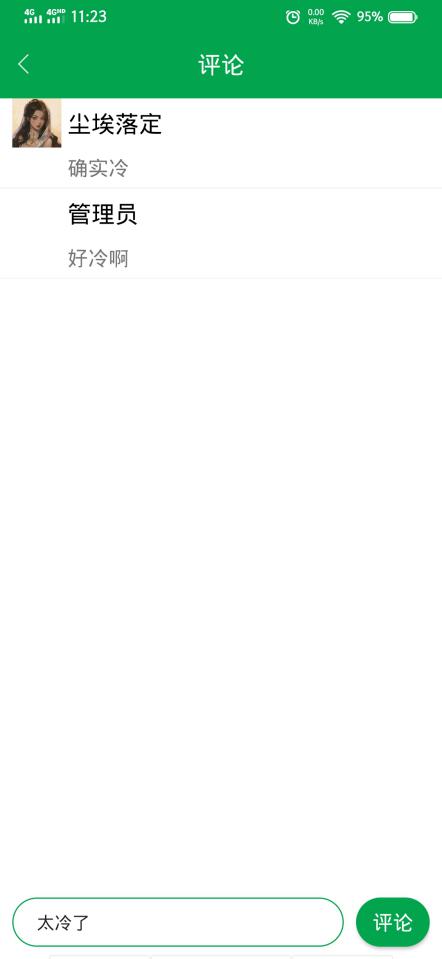


图4-4 新闻评论实现效果

**4.5新闻推荐功能**

### 4.5.1需求分析

在如今这个信息爆炸的时代，每个人都无时无刻不面临着信息过载的问题，各种条目和类别的资讯往往在我们接入互联网的那一刻就扑面而来，如果不对这些信息加以筛选和处理，我们往往会陷入混乱而不知所措。当我们作为信息的生产者和搬运者的时候，当然希望呈现给用户的东西都是他们感兴趣的，愿意看的。因此我们就需要在推送信息之前就对用户首先有一个大致的了解，然后随着用户与产品的交互的增加而不断地深入这种了解。根据用户的喜好进行新闻资讯的推荐，以此来大大节约用户的精力和时间，提供更便捷的服务。

### 4.5.2可行性分析

主要算法部分我参考了项亮的《推荐系统实践》[4]，对其中的基于用户的协同过滤和基于内容的协同过滤部分作了仔细阅读。基于用户的协同过滤适用于用户信息足够充分的情况，这样便于通过各项指标来衡量算法的性能，但是本次毕业设计由于是个人开发，因为一些众所周知的原因，我也没有条件去作大量的用户调查，因此本次的设计充分参考了《推荐系统实战》中基于内容（物品）的协同过滤算法，同时在用户注册时候就必须得到用户的标签信息用来作为过滤和推荐的指标，以此来构建整个新闻推荐系统的核心部分。

### 4.5.3数据库表结构说明

Article集合（对应于mysql的tb\_article表）详细定义如下表4-15

表4-15 tb\_article表结构

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段说明** | **对应mysql的字段** |
| articleNo | String | 文章ID | article\_no(varchar) |
| articleType | String | 文章类别 | article\_Type(varchar) |
| userNo | String | 用户ID | user\_no(varchar) |
| title | String | 标题 | title(text) |
| content | String | 内容 | content(text) |
| pic | String | 图片 | pic(text) |
| createTime | String | 创建时间 | create\_time(varchar) |

### 4.5.4 协同过滤算法说明

1. 准备知识（欧氏距离相似度）

最初是用在计算欧几里得空间中的两个点的距离，假设x、y是N维空间中的两个点，他们之间的欧几里得距离可以依据公式4-1计算[4]：



公式4-1

从这个公式中我们可以看出，当n=2的时候，欧几里得距离表示的是平面上两个点之间的距离。而当欧几里得的这个距离用来表示相似度的时候，可以采用公式4-2来进行换算，距离越小，相似度越大。



公式4-2

2. 算法实现原理及过程

（1）从数据库中取出所有和用户、文章有关的数据，构造用户和相关的文章数组，如下所示：

String[] users = new String[]{"张三", "李四", "王五", "赵六", "王七"};

String[] articles = new String[]{"文章1", "文章2", "文章3", "文章4", "文章5", "文章6", "文章7"};

（2）通过查找所有文章类型和用户喜欢的标签类型，构造用户对文章的兴趣矩阵（用户喜欢的标签为注册时候选择），矩阵如下所示：

int[][] allUserArticleStartList = {

{1, 0, 1, 0, 0, 1, 1},

{0, 0, 0, 1, 1, 1, 0},

{1, 0, 0, 1, 0, 0, 0},

{0, 1, 1, 1, 1, 0, 0},

{0, 1, 0, 0, 1, 1, 1}

};

代码生成过程如下：

Integer[][] allUserArticleList = new Integer[row][col];  
 // 所有用户  
 List<String> users = new ArrayList<>();  
 List<String> articles = new ArrayList<>();  
 for (int i = 0; i < row; i++) {  
 UserModel user = userList.get(i);  
 users.add(user.getUserName());

//取得用户标签  
 String label = user.getLabel();  
 // 所有文章  
 for (int j = 0; j < col; j++) {  
 ArticleModel article = articleList.get(j);  
 articles.add(article.getTitle());  
 // 用户喜欢列表  
 String type = article.getArticleType();  
 int value = 0;  
 if (label.contains(type)) {

//将该用户对应的文章位置置1  
 value = 1;  
 }  
 allUserArticleList[i][j] = value;  
 }  
 }

（3）转换用户文章列表（取得每个用户对各个文章的兴趣向量，此向量为行向量）

targetUserCommentedArticles = Arrays.asList(allUserArticleCommentList[targetUserIndex]);

（4）计算文章的相似度

// 4.1 计算全部文章的相似度

convertRow2Col();  
 articleABSimilaritys = new HashMap();  
 for (int i = 0; i < allArticleCommentList.length - 1; i++) {  
 for (int j = i + 1; j < allArticleCommentList.length; j++) {  
 articleABSimilaritys.put(i + "" + j,  
 calcTwoArticleSimilarity(allArticleCommentList[i],allArticleCommentList[j]));  
 }  
 }

// 4.2 计算两个文章相似度（欧氏距离）

private static double calcTwoArticleSimilarity(Integer[] article1Stars, Integer[] article2Starts) {  
 float sum = 0;  
 for (int i = 0; i < article1Stars.length; i++) {  
 sum += Math.pow(article1Stars[i] - article2Starts[i], 2);  
 }  
 return Math.sqrt(sum);  
 }

// 4.3 数组行转列

private static void convertRow2Col() {  
 for (int i = 0; i < allUserArticleCommentList[0].length; i++) {  
 for (int j = 0; j < allUserArticleCommentList.length; j++) {  
 allArticleCommentList[i][j] = allUserArticleCommentList[j][i];  
 }  
 }  
 }

（5）获取全部待推荐文章并检测这些文章中是否存在和排行榜前三重合的内容，若有，则将这些新闻过滤掉。最后将结果返回前端。

//5.3 获取全部推荐文章并排序

for (int i = 0; i < targetUserCommentedArticles.size(); i++) {

if (targetUserCommentedArticles.get(i) == 0) {

for (int j = 1; j < targetUserCommentedArticles.size(); j++) {

if (targetUserCommentedArticles.get(j) == 1) {

articleSimilaritys.put(i, articleABMatrix[i][j]);

}

}

}

}

recommendList = new ArrayList<Map.Entry<Integer, Object>>(articleSimilaritys.entrySet());

Collections.sort(recommendList, new Comparator<Map.Entry<Integer, Object>>() {

@Override

public int compare(Map.Entry<Integer, Object> o1, Map.Entry<Integer, Object> o2) {

return o1.getValue().toString().compareTo(o2.getValue().toString());

}

});

}

### 4.5.5关键代码说明

前端RecommendFragment页面关键代码如下表4-16所示：

表4-16 RecommendFragment页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onLoad | 向服务端提交请求拉取推荐栏新闻 |
| onClickArticleClick | 点击新闻进入详情页面 |
| onLongClickArticle | 长按弹出是否收藏对话框 |
| onWarningConfirmDialogResult | 若点击收藏则请求服务端将该条新闻加入数据库 |

后端部分关键代码如下表4-17所示：

表4-17 后端部分关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| queryRecommends | 协同过滤算法 |
| calcRecommendArticle | 获取全部推荐文章 |
| calcAllArticleSimilaritys | 计算所有文章的相似度 |

表4-17 后端部分代码（续表）

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| calcTwoMovieSimilarity | 计算两个文章的相似度 |
| converRow2Col | 数组行转列 |

### 4.5.6实现逻辑

本算法实现逻辑较为直接，在RecommendFragment页面中提出请求后，该请求被传递到服务端，服务端根据用户和文章数据，执行协同过滤算法，将计算结果再返回前端，最终用户得到了推荐栏的新闻列表。

后端的协同过滤算法部分的思路可以分以下四步：

1.得到用户的标签，然后根据该标签结合目前爬取的新闻生成一个用户对文章的兴趣矩阵。

2.利用欧式距离的计算公式将每两个文章之间的相似度计算出来，得到一个相似度矩阵（对称矩阵）。

3.从客户端取得此时登录的用户ID，根据这个用户ID可以得到该用户的标签并从之前的矩阵当中抽取对应的向量，这个向量中包含了用户的的兴趣信息。对应坐标为0的位置说明用户对这类新闻不感兴趣，对应坐标为1的位置说明用户对这类新闻有兴趣。

4.通过结合上边的情况，从相似矩阵中，横坐标抽取用户不感兴趣文章，纵坐标抽取用户感兴趣文章，以此可以最终得到需要给用户推荐的文章列表，最后根据相似度来给这个列表排序，从中剔除列在排行榜前三的文章，将结果返回给用户。

### 4.5.7实现结果

以下分别是普通用户的推荐界面和管理员的推荐界面，因为用户和文章都比较少，所以看到的推荐效果不是很明显，值得注意的是排行榜前三的新闻一定会出现在每个人的推荐列表当中。实现效果如下图4-5



图4-5 新闻推荐实现效果

**4.6新闻收藏功能**

### 4.6.1需求分析

用户有时候看到一条新闻后，觉得这条新闻传递的讯息非常重要，希望每次登陆app的时候都能够方便的查阅这条新闻。为了满足用户的这种需求，这里为app增加了新闻收藏功能，用户在浏览后如果需要收藏只需要长按该新闻在弹出的窗口中确认即可。

### 4.6.2可行性分析

本模块的功能为常规的页面设计和数据库连接，主要就是在用户长按某一条新闻后可以弹出对应界面让用户选择是否收藏该新闻。

### 4.6.3数据库表结构说明

Collect集合（对应于mysql的tb\_collect表），详细定义如下表4-18所示：

表4-18 tb\_collect表结构

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段说明** | **对应mysql的字段** |
| collectNo | String | 收藏ID | collect\_no(varchar) |
| userNo | String | 用户ID | user\_no(varchar) |
| articleNo | String | 文章ID | article\_no(varchar) |
| createTime | String | 创建时间 | create\_time(varchar) |

### 4.6.4关键代码说明

CollectActivity页面关键代码如下表4-19所示：

表4-19 CollectActivity页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| OnLoad | 加载收藏列表 |
| OnClickView | 返回上一级 |
| OnClickCollectClick  onWarningConfirmDialogResult | 取消收藏 |
| queryCollectList | 查询数据库中的收藏信息并返回 |

### 4.6.5实现逻辑

本部分功能模块主要就是在用户长按某一条新闻后，弹出一个对话框，询问用户是否要收藏该新闻，如果点击了确定，那么系统将会访问服务器，服务器执行addCollect函数，将要添加的新闻信息加入到数据库中，并给予前端反馈。用户在点击个人中心的收藏按钮时，系统访问服务器执行queryCollectList函数来查询数据库，并将结果显示到客户端。用户长按收藏列表中的某一项时候会弹出对话框询问用户是否删除该条新闻的收藏。

### 4.6.6实现结果

新闻收藏功能的实现结果如下图4-6，从左到右依次为主界面页面收藏、收藏列表界面、删除收藏反馈：



图4-6 新闻收藏部分实现效果

## 4.7新闻搜索功能

### 4.7.1需求分析

用户有时候会有非常明确的阅读需求，比如某个用户今天就是想知道美国新冠肺炎的确诊情况，那么这时候我们就应该为用户提供一个搜索的功能，以便他能够很方便的阅读到这方面的新闻讯息。如果不提供这个功能，用户就只能根据标签类型来费力寻找，这样的设计师不符合用户需求也是不合理的。同时，用户可能也需要系统记忆他的搜索记录，以便下次搜索时候直接点击搜索而不用重新输入。

### 4.7.2可行性分析

由于我在将数据爬取之后后台直接存入了数据库当中，而mysql数据库本身具有模糊查询的功能，因此只要得到关键字就可以根据关键字进行新闻的查询，在获得结果够返回给用户。

### 4.7.3关键代码说明

SearchActivity页面关键代码如下表4-20所示

表4-20 SearchActivity页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onLoad | 加载搜索记录 |
| onTvTrashClick | 清空搜索记录 |
| onBtnSearchClick | 新闻检索 |

SearchResultActivity页面关键代码如下表4-21所示

表4-21 SearchResultActivity页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onLoad | 加载查询到的文章集合 |
| onTvTrashClick | 清空搜索记录 |
| onWarningConfirmDialogResult | 将搜索记录暂时存放在内存中 |

### 4.7.4实现逻辑

SharedPreferences(简称SP)是Android中很常用的数据存储方式，SP采用key-value(键值对)形式，主要用于轻量级的数据存储。本次工具类中使用了这个方法来暂时存储搜索记录。在前端输入了要搜索的关键词后点击搜索，系统通过springmvc框架访问到服务端的queryArticleList函数，该函数访问数据库的配置详情代码截取如下：

<select id="queryListByMap" resultMap="articleMap">

select \* from tb\_article

<where>

<if test="articleType != null and articleType != ''">

and article\_type = #{articleType}

</if>

<if test="searchContent != null and searchContent != ''">

and (title like concat('%', #{searchContent}, '%') or content like concat('%', #{searchContent}, '%'))

</if>

<if test="userNo != null and userNo != ''">

and user\_no = #{userNo}

</if>

</where>

order by create\_time desc

</select>

由标红色的地方指示了本次查询的条件可以根据内容或者标题来查询，模糊查询为mysql内置的功能。查询成功后将结果返回。

**4.7.5实现结果**

新闻搜索功能实现结果如下图4-8，中为关键词“华为”搜索结果，右为关键词“疫情搜索结果”。



图4-8 新闻搜索功能实现效果

## 4.8爬虫部分

**4.8.1需求分析**

想要进行新闻的推送，首先当然要有新闻，目前市面上主流的新闻资讯类app都是通过爬虫来进行数据的获取的。因为只有那些大型的门户网站才能做到大量生产信息，本次毕业设计是个人开发，因此通过爬取其他网站的最新资讯为我所用显得十分方便。将爬取的网站信息整理成JSON数据后可以方便的存入数据库中以供后边操作。

### 4.8.2可行性分析

Java爬虫使用Jsoup最为常见，因为它提供了丰富的选择器和便捷的操作，只要几行代码就可以爬取到需要的信息，因此本次爬虫部分就采用Jsoup插件来实现。

### 4.8.3具体操作步骤

1.在app的gradle目录中，需要引入jsoup包

implementation 'org.jsoup:jsoup:1.11.3'

2.打开百度新闻，以“体育”为例（http://news.baidu.com/sports），爬取红色框里面的信息

利用下面代码进行连接：

Connection conn = Jsoup.connect("https://news.baidu.com/sports");  
 Document doc = conn.get();



图4-9

3.以Chrome浏览器为例，按下F12网页元素审查窗口。



图4-10

在图4-10里面用鼠标点击 “1”出的查找元素icon按钮，然后移动到 “2” 所在的元素上面，点击出现下面的页面，绿色框里面的内容就是我们要爬取的内容。



图4-11

4. 找到绿色框里面，包裹新闻列表ul的外部容器div。在该div上面的class（如：b-left），可以看做第一个识别子。 然后找到这个div下面ul新闻列表的class（如：ulist）,利用下面代码进行获取相关的元素集合：

Elements elsList = doc.select("div.b-left").select("ul[class=ulist ]");

5. 在第4步的基础上，找到ul下面的子节点，即：li。然后查找li标签中的子元素的值。



图4-12

利用下面代码进行获取取得title和详情页的url：

for (int i = 0; i < elsList.size(); i++) {  
 Elements dataElsList = elsList.get(i).select("li");  
 for (int j = 0; j < dataElsList.size(); j++) {  
 String title = dataElsList.get(j).text();  
 String url = dataElsList.get(j).select("a").attr("href");

}

}

6. 通过在第5步里面取得的详情页面的url连接，来获取图片和内容。



图4-13

利用下面代码进行获取详情页的内容和图片：

// 获取详情页的内容和图片  
 Connection connDetail = Jsoup.connect(url);  
 Document docDetail = connDetail.get();  
 Elements elsDetailList = docDetail.select("div.article-content");  
 String articleContent = "";  
 for (int k = 0; k < elsDetailList.size(); k++) {  
 // 内容  
 Elements elsContentList = elsDetailList.select("p");  
 StringBuilder sbContent = new StringBuilder();  
 for (Element elsContent : elsContentList) {  
 String content = elsContent.select("span[class=bjh-p]").text();  
 sbContent.append(content);  
 sbContent.append("\n");  
 }  
 articleContent += sbContent.toString();  
 }  
 // 图片  
 String pic = "";  
 Elements elsPicList = elsDetailList.select("div[class=img-container]");  
 if (0 < elsPicList.size()) {  
 pic = elsPicList.get(0).select("img").attr("src");  
 }

至此，爬取数据的步骤结束了，在实际的应用中，爬取信息的不同，相对应的查找的节点也会不同。本质上原理都是一样。

### 4.8.4关键代码说明(详细说明已经在上边列出)

该段爬虫代码在每一个标签页中都有使用，以LifeFragment页面为例，见表4-22

表4-22 爬虫函数

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| OnFabRefreshClick | 在管理员点击刷新按钮后进行数据爬虫 |

### 4.8.5实现逻辑

本部分功能通过使用Jsoup爬虫插件，利用html的标签特性和Jsoup带有的强大选择器很方便的将数据从网络上爬取出来为我所用。

## 4.9新闻发布功能

**4.9.1需求分析**

作为app的管理员，应该具有一定的新闻资讯的管理职能，资讯类app除了作为各类新闻资讯的搬运者，也应该适当承担新闻资讯生产者和传播者的职能，如今日头条就是从最初的搬运者变为了现在的自成一家，头条系已经成长为国内知名的资讯门户。

### 4.9.2可行性分析

这部分功能为管理员提供了新闻发布的接口，新闻发布者登录管理员，将要发布的文章信息等填写后点击发布，该文章就会出现在对应的栏目下边，后边的用户在浏览的时候就可以看到。

### 4.9.3数据库表结构说明

Article集合（对应于mysql的tb\_article表）详细定义如下表4-23

表4-23 tb\_article表结构

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段说明** | **对应mysql的字段** |
| articleNo | String | 文章ID | article\_no(varchar) |
| articleType | String | 文章类别 | article\_Type(varchar) |
| userNo | String | 用户ID | user\_no(varchar) |
| title | String | 标题 | title(text) |
| content | String | 内容 | content(text) |
| pic | String | 图片 | pic(text) |
| createTime | String | 创建时间 | create\_time(varchar) |

### 4.9.4关键代码说明

NewsAddActivity页面关键代码如下表4-24所示：

表4-24 NewsAddActivity页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onViewClicked | 功能选择函数 |
| onIvAvatarClick | 点击头像按钮选择图片 |
| onBtnReleaseClick | 监听到发布按钮点击后把新闻同时发布到前端和后端 |
| addOrder（后端） | 访问数据库并将发布信息添加到数据库 |
| onTakePhotoClickLister | 根据传入的参数判定是调用相机还是调用相册 |
| execCamera | 调用相机 |
| execGallery | 调用相册 |

### 4.9.5异常数据提醒

NewsAddActivity页面异常数据提醒如下表4-25

表4-25 NewsAddActivity页面异常数据提醒

|  |  |
| --- | --- |
| **异常原因** | **异常提醒** |
| 新闻图片为空 | 内容不能为空 |
| 新闻类别为空 | 内容不能为空 |
| 发布内容为空 | 内容不能为空 |

### 4.9.6实现逻辑

用户点击新闻发布按钮后跳转到NewAddActivity页面，在用户填写好所有信息后点击发布，这时候系统会访问数据库将发布的新闻插入到数据库当中，同时将刚刚发布的新闻更新的对应栏目的资讯列表里。

### 4.9.7实现结果

新闻发布界面、发布效果、点击详情效果如下图4-12所示

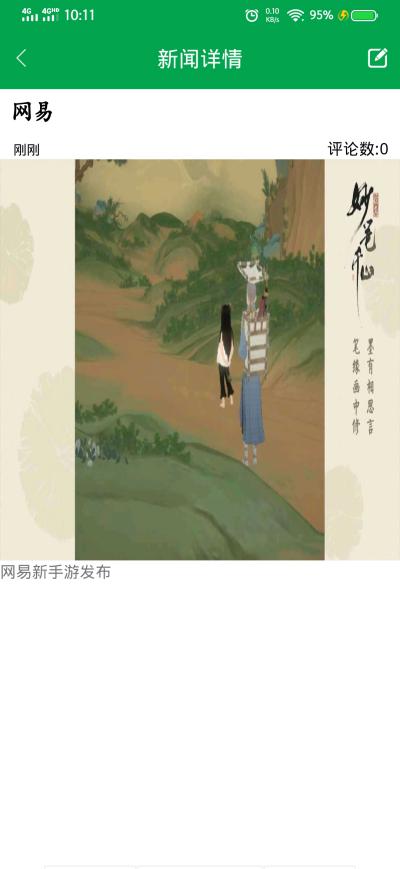


图4-12 新闻发布实现效果

**4.10历史记录查询功能**

**4.10.1需求分析**

很多时候用户在相隔一段时间后会有回看自己曾经浏览过的新闻的这样一种习惯，这个功能并非鸡肋，虽然很多人只是随手翻看新闻就完事，但是也有相当一部分人在浏览过一些新闻后一直存有印象，因此就有一定的概率想要去查找自己浏览过的东西，以便获取自己第一次浏览没有注意，而后来却非常需要的信息。因此，这个功能专门为这类用户而添加。

**4.10.2可行性分析**

这部分功能涉及页面布局为简单的LinearLayout,在用户点击一条新闻后，系统为用户添加相应的浏览记录到数据库。没有涉及特别复杂的设计。

### 4.10.3数据库表结构说明

History集合（对应于mysql的tb\_history表），详细定义如下表4-26所示：

表4-26 tb\_collect表结构

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段说明** | **对应mysql的字段** |
| historyNo | String | 收藏ID | collect\_no(varchar) |
| userNo | String | 用户ID | user\_no(varchar) |
| articleNo | String | 文章ID | article\_no(varchar) |
| createTime | String | 创建时间 | create\_time(varchar) |

### 4.10.4关键代码说明

HistoryActivity页面关键代码如下表4-27所示：

表4-27 HistoryActivity页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onLoad | 访问数据库拉取历史记录列表 |
| onClickHistoryClick  onWarningConfirmDialogResult | 删除历史记录 |
| onClickView | 返回上一级 |
| queryHistoryList(后端) | 后端数据库查询 |

### 4.10.5实现逻辑

这部分实现逻辑比较直接，就是在用户点击某条新闻后将对应的文章信息添加到浏览记录表中，在点击浏览记录按钮后跳转到浏览记录页面的同时将新闻列表从浏览记录表中拉取出来显示。长按某一条浏览记录跳出对话框确认用户是否删除该条浏览记录。

**4.10.6实现结果**

新闻浏览记录的实现结果如下图4-13所示：



图4-13 历史记录查询功能实现效果

## 4.11个人信息管理功能

**4.11.1功能说明**

这部分功能为前边的功能做了一些整理，主页面有设置功能，这里就是用于用户注销以登录其他账号；有收藏功能，点击这个功能可以查看收藏列表；有历史记录功能，点击可以查看用户浏览新闻的历史记录；我的作品功能（仅管理员有此功能），可以查看管理员账号自己发布或者从网络中爬取的新闻列表；点击头像进入个人资料管理，用户可以在这个界面对自己的头像、昵称、密码、性别、出生年月这些信息进行修改。

### 4.11.2关键代码说明

MeFragment页面的关键代码如下表4-28所示：

表4-28 MeFragment页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| onClickView | 根据输入的参数来确定执行跳转的页面 |
| initView | 判定是否是管理员以提供不同的界面 |

PersonActivity页面的关键代码如下表4-29所示：

表4-29 PersonActivity页面关键代码

|  |  |
| --- | --- |
| **函数名** | **函数功能** |
| initView | 将各组件的数据初始化 |
| onClickView | 根据传入参数来确定要执行的功能 |
| onIvAvatarClick  onTakePhotoClickLister | 处理头像点击事件 |
| onInputTextDialogResult | 处理文本框输入 |
| onListDialogResult | 处理列表类对话框输入 |
| uploadNick | 访问数据库，更新昵称 |
| uploadPassword | 访问数据库，更新密码 |
| uploadSex | 访问数据库，更新性别 |
| uploadBirth | 访问数据库，更新出生日期 |

### 4.11.3实现逻辑

这部分就是给用户提供一个简单的功能整理，使得用户能够很方便的去处理相关功能。点击每一个按钮都会去拉起一个新的Activity，然后进入到那个Activity中去处理相应的事情。

**4.11.4实现结果**

个人信息管理功能的实现结果如图4-14所示：



图4-14 个人信息管理功能实现效果

第五章 总结与展望

## 5.1总结

Android开发其实我在大二的时候就有接触了，那个项目是在刚刚学习了java后利用假期的时间做的，整个项目是跟着视频结合资料来一点一点写的，基本上的框架都在那个时候弄明白了，但是有很多细节的东西没有深究，就是达到了“知道这东西这么用就是能出这个效果但并不知道是因为什么”的程度。这次毕业设计因为是在家里完成，期间遇到的最多的问题就是网络情况，可能是开发工具的升级导致“找不到可用的证书”作为一个时隐时现的bug一直存在项目的构建过程中，最后没办法把对应网站的证书都下载下来配置到jdk的目录下才解决。这次开发也同时参考了许多论文、书籍以及网课资料，又添加了许多工具类到项目中，相较于大二的进步是对于一些用法的原理能够更深刻的理解了。

Android Studio这款IDE如果能适应好就是很不错的开发工具，但是有很多时候要面临莫名其妙的报错，有时候甚至重新启动一下这些错误就没有了，不过大多数时候项目构建失败是因为各个组件的版本不一致或者jar包下载超时。最新版本的Android Studio在进行布局文件的书写时候甚至可以完全不用写代码，直接选择组件然后拖动到对应的位置就好，刚开始觉得非常方便非常简单，后来发现这种方式会导致代码结构的混乱，导致自己在前端出现组件数据传递的问题时回来看而一团乱麻的情况，于是后边布局文件就都自己写了。同时在我开始做Android布局文件时候就一直专门留意了市面上流行的app的设计结构，发现几乎所有app的界面布局都是类似的，即主界面上边有搜索栏和可供滑动的标签页，下边有功能选择按钮这一栏，每一个详情页面其实都可以是LinearLayout的布局，因为这种布局是最符合目前手机的尺寸和显示样式的。

对于IDEA这款IDE软件，我对其印象深刻是因为其对于maven项目和gradle项目构建的天然优势，因为在eclipse中很多时候想要创建一个springmvc框架的项目需要手动配置很多东西，而这些操作我并不能记得那么深刻，都是在需要的时候去网络上搜索教程然后跟着配置；而IDEA在项目创建时候项目结构就是完整的，配置框架等也十分简单，只需要在build.gradle文件中将对应框架的包引入进来点击构建，IDEA就会自动的把需要的包全部下载并导入，十分方便。

在动手构建项目初期，我先后阅读了《Android从入门到精通》[6]《基于Jsoup的通用网页采集系统的设计与实现》[7]以及《基于jsoup的Web页面自适应转换系统的实现》[13]，观看了教学视频《Jsoup项目实战》[8]对于Android的基础开发和如何利用java进行爬虫有了一个初步的了解，方便了后续开发的进行。

对于推荐系统这一部分，主要理论是参考了《推荐系统实践》[5]、《个性化推荐算法设计》[9]、《统计学习方法》[10]，实现方式因为我对于python不是很熟悉，就采取java来实现，期间也参考到了许多的网课视频教程，最主要研习的是《Android高级应用开发课程》《清华大学--数据挖掘：推荐算法》[7]这两个课程。

在论文的组织结构上，参考了《软件工程》[11]这本书，每个功能部分都严格遵循需求分析、可行性分析、数据库结构说明、关键代码说明、实现逻辑、实现效果这几个步骤，让我在写论文的时候不至于没有头绪。

最后想说一下推荐系统的部分，比较可惜的是这次推荐算法最后的结果因为用户数量太少，所以最后算出的推荐列表中，前边的推荐指数都挺接近的，这就导致在实际使用时候只有排行榜部分起到了很好的热门新闻推荐效果。我因为之前按照视频教程上的说明，从那个视频指定的网站上下载到了一个训练集(电影推荐系统的数据)，使用python在执行协同过滤算法之后，确实能起到一定的推荐效果，那一份数据是有4000多条，因此我对这个算法最后的效果是肯定的。诚然，市面上那些成熟的推荐类应用使用的算法要比本文使用的复杂许多，评测标准也更加严格，即便这样，我们日常使用时候依然会有生硬的感觉。因为其中许多指标都是在统计数据出来后根据大多数情况来加入的，有经验成分在里边，其实并不是完美的适用于所有人。另一方面，本文所使用的推荐算法是基于用户的兴趣情况来进行计算并推荐的，但其实在实际生活中，有大量的用户并没有体现出明显兴趣意向的项目，用户并非对他们不感兴趣，甚至有可能是非常有兴趣的，如果能有算法能够高效的将这类项目推荐给用户，那么用户就会有惊喜感。但是就目前市面应用推荐类算法的主流软件中，在这一方面的实际效果并不理想。对于这类算法，其前景是广阔的，但是鉴于目前的情况，有理由认为其还有很大的发展空间。

## 5.2展望

通过这次毕业设计，我的个人技能又得到了很大的提升，同时通过查阅资料和观看学习视频，自学能力也更加强了。对于我以后从事的职业，学习将是一个终身的话题，只有不断学习，不断实践，才能不断提升自己的竞争力，在未来的生活和工作中才能有更好的表现。同时也希望能够通过自己的努力，写出一款能够给更多人提供服务的好用的实用的软件。

**参考文献**

[1]Rui Ren,Lingling Zhang,Limeng Cui,Bo Deng,Yong Shi. Personalized Financial News Recommendation Algorithm Based on Ontology[J]. Procedia Computer Science,2015,55.

[2]吴想想. 基于Android平台软件开发方法的研究与应用[D].北京邮电大学,2011.

[3]孙骏雄. 基于网络爬虫的网站信息采集技术研究[D].大连海事大学,2014.

[4]曹一鸣. 基于协同过滤的个性化新闻推荐系统的研究与实现[D].北京邮电大学,2013.

[5]项亮.推荐系统实践.[D].人民邮电出版社.2012.

[6]孙更新.Android从入门到精通.[D].电子工业出版社.2011.10

[7]毛凯. 基于Jsoup的通用网页采集系统的设计与实现[D].电子科技大学,2015.

[8]网课《Android高级应用开发课程》《清华大学--数据挖掘：推荐算法》《Jsoup项目实战》

[9]赵亮,胡乃静,张守志.个性化推荐算法设计[J].计算机研究与发展,2002(08):986-991.

[10]李航.统计学习方法.[M].清华大学出版社

[11]Ian Sommerville.软件工程（原书第9版）[M].机械工业出版社出版,2011

[12][MySQL查询优化的研究和改进](http://kns.cnki.net/kcms/detail/detail.aspx?filename=2009037357.nh&dbcode=CMFD&dbname=CMFD2009&v=)[D]. 孙辉.华中科技大学 2007

[13]顾问,曹阳.基于jsoup的Web页面自适应转换系统的实现[J].电子技术与软件工程,2017(18):200-202.

**致谢**

经过了很长一段时间的学习和努力，我终于完成了这次的毕业设计的编写和毕业论文的写作，首先想向在这次设计过程中给我提供帮助的柴胜老师表示特别的感谢，感谢他能在今年这个特殊的时期在各种方面给予我的指导。同时也很感谢学校能在疫情期间开放了学校的vpn,使得我能够在家里访问校内图书馆和上知网等学术网站查阅资料。

当然，还要感谢在我进行这次毕业设计遇到问题的过程中给予我建议和帮助的同学们，最后感谢尚硅谷提供的免费课程《电影推荐系统实战》以及《硅谷新闻》，通过这两个项目的跟进学习，使得我对于app的开发以及推荐算法的使用有了更加深入的理解。

通过这次的毕业设计，使我能够系统的梳理一个新闻资讯类app的前后端实现，当然由于个人能力有限，我只是把今日头条中的部分功能作了简化实现，实际的应用场景肯定比本文所使用的技术要先进的多、严谨的多。这次设计使我收获良多，因此想要再次感谢在这次的设计和开发中为我提供帮助的所有同学和老师。