南京大学

本科毕业设计

院　　系　软件学院

专　　业　软件工程

题　　目　高考志愿填报推荐系统后端的设计与实现

年　　级　2019　学　　号　191850031

学生姓名　范子君

指导教师　葛季栋　职　　称　副教授

提交日期　2023 年 5 月 24 日

南京大学本科毕业论文（设计）
诚信承诺书

本人郑重承诺：所呈交的毕业论文（设计）（题目：高考志愿填报推荐系统后端的设计与实现）是在指导教师的指导下严格按照学校和院系有关规定由本人独立完成的。本毕业论文（设计）中引用他人观点及参考资源的内容均已标注引用，如出现侵犯他人知识产权的行为，由本人承担相应法律责任。本人承诺不存在抄袭、伪造、篡改、代写、买卖毕业论文（设计）等违纪行为。

作者签名：范子君

学号：191850031

日期：2023年5月27日

南京大学本科生毕业论文（设计、作品）中文摘要

题目：高考志愿填报推荐系统后端的设计与实现

院系：软件学院

专业：软件工程

本科生姓名：范子君

指导教师（姓名、职称）：葛季栋副教授

摘要：

高考作为中国教育体系中的一项重要制度，对社会而言是公平选拔出具备优秀学习能力的人才的关键渠道，对个人而言是不断完善自我认识和实现价值理想的重要途径。高考志愿填报是考生需要面临的至关重要的决策，关系到考生未来的职业发展方向和才能潜力发挥。而填报高考志愿存在着诸多难问题，首先，由于国内院校数量巨大以及专业背景繁杂，学生很难高效获取志愿填报所需要的完备信息，其次，部分学生对自身的职业规划模糊或对自身性格能力是否适合理想专业持困惑怀疑态度，从而在填报志愿的决策过程中迷茫，最后，院校专业的报录比和最终录取分数的不确定性给考生的志愿填报带来隐患。针对上述问题，我们为高考考生开发了高考志愿填报推荐系统。

高考志愿推荐填报系统的前端界面采用 Vue.js 进行框架设计，并基于 Element UI 进行组件式开发；后端采用 Spring Boot 框架构建项目，并连接 MongoDB 数据库；最后在系统设计原则的指导下组装各模块形成完整系统。

在具体实现中，高考志愿填报推荐系统分为用户信息模块、院校信息检索模块、专业信息检索模块和志愿推荐模块。在关键的志愿推荐模块，高考志愿填报推荐系统通过筛选和过滤大量信息向考生提供填报志愿所需要的有效信息来引导考生正确决策，并且通过考生的性格、高考专业和分数排名来分录取可能性推荐院校和专业。

高考志愿填报推荐系统完善了传统高考志愿填报系统的信息检索和推荐功能，对考生用户来说具有较高的可用性，切实解决了考生志愿填报的需求。系统在云平台部署，可以根据业务需求和用户访问量动态调整服务器资源，从而实现应用程序的弹性伸缩和灵活性。除此之外，MongoDB 分布式部署可以将数据和计算负载分散到多个节点上，从而提高系统的横向扩展能力和数据安全性。

关键词：高考；志愿；推荐；Spring Boot；MongoDB

南京大学本科生毕业论文（设计、作品）英文摘要

THESIS: Design and Implementation of the Back-end of College Preference Intelligent Recommendation System

DEPARTMENT: Software Institute

SPECIALIZATION: Software Engineering

UNDERGRADUATE: Zijun Fan

MENTOR: Associate Professor Jidong Ge

ABSTRACT:

As an important system in China, the gaokao is a key channel for the society to fairly select talents with excellent learning ability, and an important way for individuals to continuously improve their self-understanding and realize their value ideals. Volunteering for the college entrance examination is a crucial decision that candidates need to face, which is related to their future career development direction and talent potential. There are many difficult problems in filling in the college entrance examination volunteers. First of all, due to the huge number of domestic colleges and universities and the complex professional background, it is difficult for students to efficiently obtain the information required for volunteer filling. Secondly, some students are confused and skeptical about their own career planning or whether their personality ability is suitable for the ideal major, so as to be confused in the decision-making process of filling in the volunteer, and finally, the uncertainty of the application ratio and final admission score of the college major brings hidden dangers to the candidates' volunteer filling.

In view of the above problems, the college entrance examination volunteer filling recommendation system provides candidates with the effective information required to fill in the volunteer by screening and filtering a large amount of information to guide candidates to make correct decisions, and recommends colleges and majors through the candidates' personality, college entrance examination majors and score rankings.

In terms of specific development mode, the front-end interface adopts Vue.js for framework design and component-based development based on Element UI. The back-end uses the Spring Boot framework to build the project, connects to the MongoDB

database, and finally assembles each module to form a complete system under the guidance of system design principles.

The system introduced in this thesis improves the information retrieval and recommendation functions of the traditional college entrance examination volunteer filling system, which has high usability for candidate users and effectively solves the needs of candidates for voluntary filling out. The system is deployed on the cloud platform, and server resources can be dynamically adjusted according to business needs and user visits, so as to achieve elastic scaling and flexibility of applications. In addition to this, MongoDB distributed deployment can spread the data and computing load across multiple nodes, thereby improving the horizontal scalability of the system and data security.

KEYWORDS: College Entrance Examination; Recommend; Application; Spring Boot; MongoDB

目　录

插图目录

表格目录

第一章　绪论

1.1　研究背景及意义

高考制度经过多年改革，不断调整和完善，效率、科学与公平是高考追求的三大核心价值，它们从不同维度驱动着高考制度的持续现代化。

效率是高考制度人才选拔的功能追求,[1] 它推动高考录取制度历经 “分数为主选”、“兼顾志愿选拔” 到 “双赢性选拔” 三个发展阶段。随着近年高考报名人数的日趋增加，在有限时间内顺利实现全国数百万乃至上千万考生与数千所高校、数百种专业之间的科学匹配，高考的效率优势无法替代，同时提高择校的科学性也因此成为高考制度面临的重大挑战。繁多的院校和专业信息、每年高考政策的改革以及各院校招生计划的变动让高考志愿的填报困难重重。[2]面对海量的院校填报信息，考生往往很难在短期内做出比较符合自身意愿的合理选择，进而导致报考事故的发生。

地域、城乡、阶层等维度上的基础教育资源分配不均，致使高考制度面临的公平挑战异常严峻。而 “高考难，填报更难”,[3]研究表明，高考志愿填报取向对弱势家庭文化和职业背景的学生的录取匹配度有显著影响，但对优势家庭文化和职业背景的学生的影响不显著。因此在择校层面做到决策公平，是直接推进当代高考制度完善的一步。

新高考改革的重要举措之一是探索实行 “专业 + 学校” 的平行志愿填报模式，倡导学生在志愿填报取向上实现从 “总分匹配” 到 “专业导向” 的转变。[4]相较于 2017 年参加高考的学生，2018 年参加高考的学生在填报志愿时倾向于 “专业优先” 的概率显著更高。

高考志愿填报推荐系统能够丰富和平衡考生在择校决策方面的信息和眼界，增加考生对志愿专业和高校的了解与信心。在考生实现兴趣和专业统一的同时，提高个人志愿与社会需要的契合度，对社会发展和考生个人都具有重要的意义。

1.2 国内外研究现状

早期的高考志愿推荐平台更多是基于往年高考录取分数和排名来推荐高校，未考虑到考生学科偏好、生涯规划等个性化因素，难以满足考生的个性化推荐需求,[5]例如 2010 年的高考志愿网报系统性能优化与志愿预测分析，采用统计学习方式，基于历史录取数据构建高考志愿预测分析模型。

[6]较为成熟的推荐系统基于 B/S 结构，利用海量数据进行全面计算，快速准确地为考生提供有价值的参考。数据的提取、数据清洗、数据分析等实现高考填报系统推荐系统的实现。[7]数据来源主要为历年的考生志愿填报相关数据和考生的分数、兴趣爱好等相关数据，对这些数据构建关联，进行分析，使用推荐算法，实现考生输入自己感兴趣的专业、地域等信息，系统推荐供考生参考填报的高考志愿。

更进一步，研究者开始选择更为高效的推荐算法来加强志愿服务平台的个性化和适配性。国内研究者主要研究个性化推荐算法、协同过滤推荐算法、模糊聚类算法等。传统的协同过滤算法没有考虑用户的自身信息的影响，存在数据稀疏性、扩展性差等弊端。[8]李华基于用户情景模糊聚类的协同过滤推荐算法，根据用户情景信息利用模糊聚类算法得到情景相似的用户群分类，在进行协同过滤前预先通过 Slope One 算法填充用户-项目评分矩阵，改善了数据稀疏性和实时性。

高考志愿推荐系统仍有较大的优化空间。[9]谷宝柱对高考志愿选择行为的影响因素研究把 37 个变量通过因子分析和研究需要划分为个人因素、家庭因素、高校因素、国家社会因素、经济因素五个因素，为推荐系统提供了参考方向。[10]黄泽鑫以考生选择的专业为主体，将考生爱好、父母职业、经济条件和专业就业率等四个因素作为义原，提出基于知网概念相似度计算推算出考生最佳专业选择集合。依据考生成绩位次及最佳专业选择集合，通过院校专业录取概率模型，计算出考生被院校专业录取概率，为考生推荐出冲击型志愿、稳妥型志愿、保底型志愿。

国内国外文献总体呈现数量少、相关度低的态势。近年来，人工智能算法不断创新和进步，从简单的数理统计算法，如 Logistic 回归算法和线性回归算法，到复杂的智能算法，如 CNN，相信高考志愿填报推荐系统会迎来新的研究高潮。

1.3 本文的主要工作

本文的主要工作是从概括到具体地介绍高考志愿填报推荐系统的研究和开发方法。首先检索与高考志愿推荐系统相关的资料，总结出国内外研究现状。之后介绍高考志愿推荐系统后端开发和部署用到的重要技术。然后开始对系统进行总分结构的介绍，首先介绍系统的需求分析，用图片辅助展示系统目标、用户特征、功能性需求分析、用例描述和非功能性需求，在阅读上述五个过程中可以了解到系统需求及其设计思路和实现方向。之后是系统设计，首先描述总体设计，从系统总体架构设计和系统模块划分的角度出发，阐述系统的开发方向，接着介绍了本人负责的后端中模块详细设计，重点介绍了根据功能将系统分成的用户信息相关功能模块、院校信息搜索功能模块、专业信息详情展示模块、问卷模块，在这个部分使用了类图来从数据角度阐释信息存储和传递。然后介绍系统后端功能模块的实现，按照上一个部分确定的后端模块介绍，展示代码的实现逻辑以及解释代码设计，结合重点代码和说明文字可以了解后端系统的内部构成。之后的总结与展望部分，总结了系统开发的意义和目标以及本文的整体撰写过程和呈现效果，分析了当前高考志愿推荐系统的优缺点和改进空间。最后是致谢，感谢所有帮助过我的人们。

1.4 论文的组织结构

本文一共七章，具体章节如下：

第一章：绪论。简述研究背景和意义，总结国内外研究现状，总述主要工作和论文组织架构。

第二章：相关技术概论。概述本文所介绍的高考志愿推荐系统构建所用的主要技术。

第三章：系统需求分析。明确高考志愿填报推荐系统的目标和范围，对收集到的需求进行整理和分析，识别需求间的关系和优先级以及可行性和实现难度，最终确立需求。

第四章：系统总体设计。将软件系统划分为若干个模块或组件，并描述这些模块或组件之间的交互关系、数据流向和系统结构。主要借助软件开发过程中的图形化建模工具来展示系统的总体设计。

第五章：功能模块的后端具体设计和实现。分模块具体说明后端的实现，专注于逻辑的实现。

第六章：总结与展望。对本文的工作进行回顾和总结，对不足之处提出改进意见以及就本系统的未来应用前景进行展望。

第二章　相关技术概论

2.1　Spring Boot

Spring Boot 是一套基于 Spring 的开源框架，旨在通过自动配置项目、提供快速开发工具等方式来快速构建 Java 应用程序，适用于构建 RESTful Web 服务、开发云原生应用程序、开发企业级应用程序和实现数据处理任务。

Spring Boot 可以识别注解和匹配类路径，从而实现自动配置项目运行所需第三方库，简化了开发者在配置方面所做的工作。Spring Boot 还提供了嵌入式 web 服务器，应用程序可以打包成可执行的 jar 包，并部署到云环境中，简化了软件设计开发的下游工程。Spring Boot 社区还有大量扩展和插件来支持构建相对复杂的微服务项目，例如 Spring Cloud、Spring Batch 等等。

在实际开发中,[11]开发者可以使用 Spring Boot 提供的大量实用的开箱即用的扩展和插件，其中核心的组件包括 Spring Boot Starter、Actuator、Web、Test、DevTools。引入 Starter 组件就相当于引入一系列开发常用的依赖库和自动配置；Actuator 组件能够监控和管理项目的配置信息、健康状态和性能指标等；Web 组件提供了嵌入式 Web 服务器和应用程序支持，常用的有 Tomcat 和 Jetty；项目的单元测试和集成测试通过 Test 组件支持；DevTools 主要提供自动重启的功能，可以加速开发和测试流程。

2.2　Maven

Maven 是一个流行的构建工具和项目管理工具，可以自动化构建和管理 Java 项目，具有强大的构建、依赖管理、插件系统、多模块支持等功能。Spring Boot 常与 Maven 一起使用，可以更轻松地构建和管理 Spring Boot 应用程序。

Maven 作为强大的构建工具，可以自动化构建 Java 项目，并处理依赖关系和编译顺序。Maven 使用 POM（Project Object Model）文件来描述项目的结构和依赖关系。Maven 提供了强大的依赖管理功能，可以自动下载所需的依赖库，并

确保依赖库的版本一致性。Maven 使用中央仓库（Central Repository）来管理依赖库，中央仓库是一个全球性的公共仓库，包含了大量的 Java 依赖库。如果中央仓库中不存在所需的依赖库，Maven 还支持自定义仓库和本地仓库，方便管理项目的依赖。

Maven 提供了强大的插件系统，可以扩展 Maven 的功能。[12]插件可以用于编译、测试、打包、部署等不同的阶段，也可以用于生成报告、文档等。Maven 自带了大量的插件，还支持自定义插件，方便开发人员扩展和定制 Maven 的功能。

Maven 支持多模块项目的构建，可以将一个大项目拆分成多个子模块，并统一管理。每个子模块都有自己的 POM 文件，可以独立地进行编译、测试、打包、部署等操作。Maven 可以自动处理子模块之间的依赖关系，并确保编译顺序正确。

Maven 拥有庞大的生态系统，包括大量的插件、工具和框架。除了 Maven 自身提供的功能之外，还有很多第三方工具和框架可以与 Maven 集成，例如 Jenkins、SonarQube、Spring 等。这些工具和框架可以帮助开发人员更好地构建和管理 Java 项目。

2.3 MongoDB

MongoDB 是一个流行的功能强大的开源文档数据库，采用了面向文档的数据模型，可以轻松地存储和查询复杂的数据结构，还具有分布式架构、高可用性和容错性的优点，对于需要存储和查询复杂数据结构的应用程序来说是一个非常好的选择。

MongoDB 采用了面向文档的数据模型，数据以文档的形式存储在集合（collection）中。[13]每个文档都是一个键值对的集合，可以包含不同的字段和值。文档中的字段可以是任何类型的数据，包括嵌套文档和数组。与传统的关系型数据库相比，MongoDB 的文档模型更加灵活，可以更好地满足现代应用程序的需求。

MongoDB 的查询语言采用了 JSON 格式的文档，非常直观和易于理解。MongoDB 支持丰富的查询操作，包括范围查询、复杂的嵌套查询、聚合查询等。MongoDB 还支持全文本搜索和地理空间查询等高级功能。

MongoDB 采用分布式架构，可以很容易地进行水平扩展。MongoDB 使用副本集（replica set）来提高可用性和可靠性。[14]副本集是一组 MongoDB 服务器的集合，其中包含一个主服务器（primary）和多个从服务器（secondary）。主服务器处理所有的写操作，并将更新传播到从服务器上，以保证数据的一致性。

MongoDB 提供了丰富的功能来提高可用性和容错性。在副本集中，如果主服务器故障，MongoDB 会自动选择一个从服务器作为新的主服务器，并继续提供服务。[15]此外，MongoDB 还支持自动分片（sharding）来处理大规模数据集。自动分片可以将数据集分割成多个部分，分别存储在不同的服务器上，从而提高可扩展性和性能。

2.4 MD5 算法

MD5 (Message-Digest Algorithm 5) 是一种常用的哈希算法，它能够将任意长度的消息转换为一个固定长度的哈希值。在计算机系统中，通常使用 MD5 算法对密码进行加密处理，然后再将加密后的密码存储到数据库中，以提高密码的安全性。

具体来说，[16]使用 MD5 对密码进行加密可以防止明文存储，如果直接将用户的密码明文存储在数据库中，一旦数据库被攻击或泄漏，那么攻击者可以轻松地获取到所有用户的密码，造成非常严重的后果。而使用 MD5 对密码进行加密后，即使数据库被攻击或泄漏，攻击者也无法直接获取用户的密码明文。MD5 还可以防止彩虹表攻击，彩虹表是一种常见的密码破解技术，它通过预先计算出大量的哈希值和明文密码的对应关系，从而快速破解哈希值。而使用 MD5 对密码进行加密后，即使攻击者获得了哈希值，也需要进行繁琐的计算才能破解出原始密码，防止了彩虹表攻击。

需要注意的是，MD5 算法虽然能够提高密码的安全性，但也存在被暴力破解的风险。因此，在实际使用中，[17]还需要结合其他安全措施，如加盐、多次哈希等，来进一步增强密码的安全性。

2.5 Docker

Docker 是一种流行的容器化技术，它可以将应用程序及其依赖项打包到一个可移植的容器中。[18]容器是一个运行时的实例，是基于镜像创建的。容器拥有自己的文件系统、网络、进程空间和资源限制，可以被启动、停止、删除和重启。Docker 的容器化技术可以大大简化应用程序的部署和运行，同时提高了环境的可移植性和安全性。

镜像是 Docker 中的重要概念，是一个静态的文件，包含了一个完整的应用程序及其依赖项。镜像可以通过构建、下载或导入镜像来获取应用程序及其依赖项。每个镜像都有一个唯一的标识符，称为镜像 ID。[19]Docker 官方提供了 Docker Hub 仓库，也可以自己搭建私有仓库来存储和分发 Docker 镜像。容器是一个运行时的实例，是基于镜像创建的。容器拥有自己的文件系统、网络、进程空间和资源限制，可以被启动、停止、删除和重启。仓库是存储和分发 Docker 镜像的地方。Docker 官方提供了 Docker Hub 仓库，也可以自己搭建私有仓库。

Dockerfile 是一个文本文件，包含了构建 Docker 镜像的指令。Dockerfile 中包含了从哪个基础镜像开始、如何安装依赖、如何配置环境等指令。Dockerfile 还支持构建缓存，可以避免重复构建相同的步骤。

Docker 提供了一系列常用命令来管理容器和镜像，包括 docker pull、docker run、docker ps、docker stop/start、docker rm、docker images、docker build、docker push 等。

总之，Docker 是一种流行的容器化技术，可以帮助开发人员和系统管理员更轻松地构建、部署和运行应用程序。Docker 的核心概念包括容器、镜像、Dockerfile 和常用命令。Docker 的容器化技术可以大大简化应用程序的部署和运行，提高环境的可移植性和安全性。

第三章　系统需求分析

3.1　总体描述

3.1.1　系统目标

高考志愿推荐填报系统是采用前后端分离开发的单体 WEB 应用，前端主要使用 Vue 框架进行开发，后端使用 Spring Boot 框架开发连接 MongoDB 数据库。本高考志愿填报推荐系统致力于为用户提供一个可以检索院校专业信息以及获得志愿填报推荐的一站式平台。具体功能方面，系统主要提供院校信息检索，专业信息检索和志愿推荐三大功能。

3.1.2　用户特征

高考志愿填报推荐系统的目标用户是即将面临志愿填报的高考考生，考生希望了解院校和专业信息，本系统提供了相关平台，考生用户可以参考系统推荐的志愿填报方案来选择志愿。综合考虑，我国高考考生的最大特征在于对高等院校和各个专业没有详细的了解，缺少填报志愿的具体方向、目标和经验，导致考生对志愿填报搜集信息然后进行决策的方法和流程并不熟悉，因此本系统在前端页面方面尽量做到易学性，让用户简单快速地上手操作，功能方面同时为考生提供获取校和专业信息的渠道，以及通过考生分数和性格推荐院校和专业的入口辅助考生决策。

3.2　功能性需求分析

高考志愿填报推荐系统的主要功能如下。

1. 个人信息管理。用户可以填写并保存个人基本信息，如高考分数，排名，生源地，选科。此外，用户还可以通过填写问卷来完成性格和职业测试，测试结果也会被保存在用户个人信息中。

2. 院校信息检索。用户可以通过名称检索院校，同时还可以对检索结果进行排序或筛选来更容易找到要找的院校。在检索到目标院校后，院校的相关数据如院校基本信息，院校图片，专业历年分数线等会以可视化的形式呈现。
3. 专业信息检索。用户可以通过名称或者专业代码检索专业，在检索到目标专业后，可以查看目标专业的介绍，就业前景和知名院校等信息。
4. 志愿推荐。用户在填写了个人基本信息后即可进行基本的志愿推荐，填写系统提供的测试问卷则可以使志愿推荐的结果更加合理，志愿推荐系统会根据用户情况分别列出推荐的院校和专业列表供用户参考。

系统用例图如图3-1所示

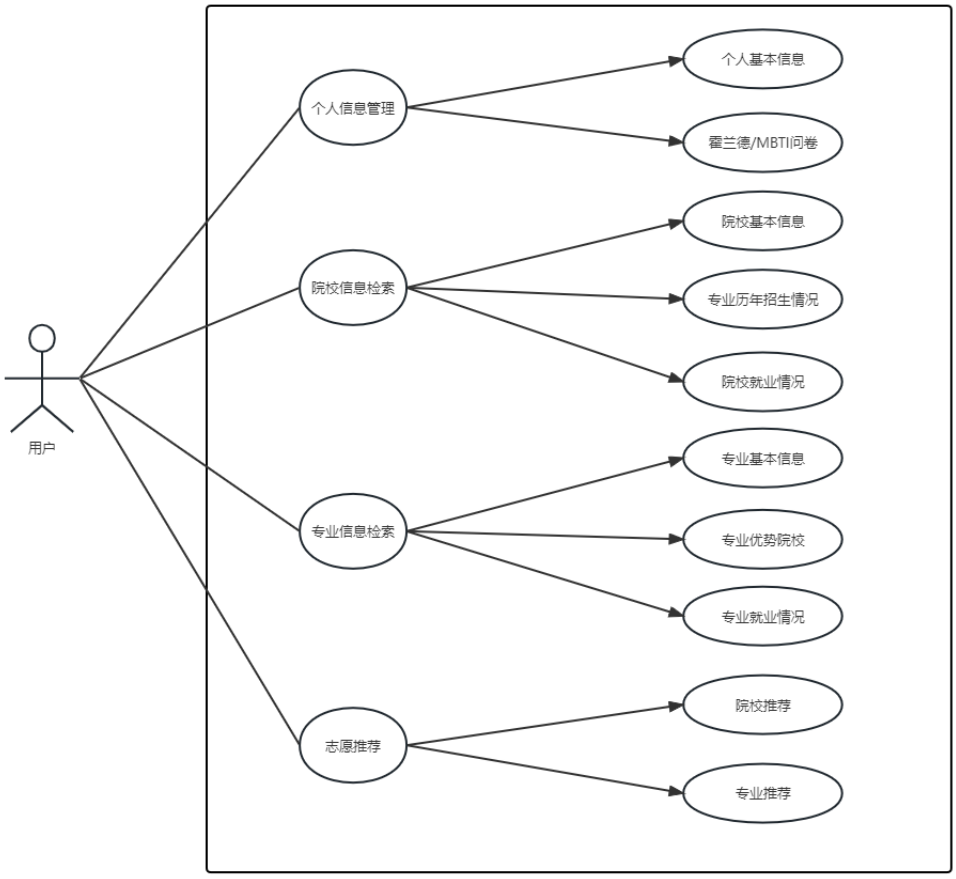

图 3-1　系统用例图

3.3 用例描述

1. 用户进行个人信息管理的用例描述如表3-1所示。此用例描述了用户填写个人信息的流程。

表 3-1 用户进行个人信息管理的用例描述

项目	内容描述
ID	02
用例名称	个人信息管理
参与者	用户，即高考考生
描述	用户填写个人基本信息和问卷，如有需要可以进行修改
触发条件	用户打开个人基本信息或问卷页面
前置条件	用户处于登录状态
后置条件	如果用户修改个人信息或文件内容，系统需要将其保存
正常流程	1. 用户打开个人基本信息或问卷页面，系统显示信息 2. 用户修改信息，系统返回修改后的数据
拓展流程	无
特殊需求	无

2. 用户进行院校信息检索的用例描述如表3-2所示。此用例描述了用户检索查看院校信息的流程。

表 3-2 用户进行院校信息检索的用例描述

项目	内容描述
ID	02
用例名称	院校信息检索
参与者	用户，即高考考生
描述	用户输入院校名称关键词，系统检索名称包含关键词的院校并展示，用户选择系统展示的某个院校，系统展示院校的详细信息
触发条件	用户输入院校名称关键词并检索
前置条件	用户处于登录状态
后置条件	无
正常流程	1. 用户输入院校名称关键词并检索 2. 系统以列表的形式展示名称包含关键词的院校 3. 用户选择系统列出的某个院校 4. 系统展示用户选择的院校的详细信息
拓展流程	无
特殊需求	无

3. 用户进行专业信息检索的用例描述如表3-3所示。此用例描述了用户检索查看专业信息的流程。

4. 用户进行志愿推荐的用例描述如表3-4所示。此用例描述了用户在填写了个人信息后，系统根据推荐算法给出推荐院校专业的过程。

表 3-3　用户进行专业信息检索的用例描述

名称	内容描述
ID	03
用例名称	专业信息检索
参与者	用户，即高考考生
描述	用户输入专业名称关键词或者专业代码，系统检索相关专业并展示，用户选择系统展示的某个专业，系统展示专业的详细信息
触发条件	用户输入专业名称关键词或者专业代码并检索
前置条件	用户处于登录状态
后置条件	无
正常流程	1. 用户输入专业名称关键词或者专业代码并检索 2. 系统以列表的形式展示相关专业 3. 用户选择系统列出的某个专业 4. 系统展示用户选择的专业的详细信息
拓展流程	无
特殊需求	无

表 3-4　用户进行志愿推荐的用例描述

名称	内容描述
ID	04
用例名称	志愿推荐
参与者	用户，即高考考生
描述	用户在填写了个人信息后，选择志愿推荐，系统展示推荐的院校和专业
触发条件	用户点击志愿推荐按钮
前置条件	用户处于登录状态且已经填写个人基本信息
后置条件	无
正常流程	1. 用户点击志愿推荐按钮 2. 系统显示推荐的院校和专业列表
拓展流程	无
特殊需求	无

3.4　非功能性需求分析

易用性：系统应符合日常使用习惯，操作方便，用词易于理解，用户无需参考手册即可直接使用系统，同时，系统应对用户的操作给出显式的反馈，并对可能出现的误操作提供一定的容错处理。

时间性能：用户的操作应在 500ms 内得到反馈，对于耗时更久的大数据量操作等，应出现加载提示以减少用户焦虑

安全性：系统应只允许已经登录的用户访问内容，此外，系统需要保证用户的个人信息不会泄露

可扩展性：系统应采用模块化组件化开发，尽可能降低耦合提高复用，便于日后的功能扩展

支持性：系统应正常在绝大部分浏览器上运行，需要适配不同分辨率的不同种类浏览器

第四章　系统设计

4.1　总体设计

4.1.1　系统总体架构设计

本节使用 “4+1” 视图模型来描述系统的架构设计，即系统的总体架构设计图、逻辑视图、物理视图、处理视图和开发视图。

系统的总体架构分为交互层，执行层和数据层。其中交互层负责和用户的交互，并向执行层传输交互数据，执行层负责处理逻辑，数据层则负责向执行层提供需要的数据。

系统的总体架构设计图如图4-1所示。

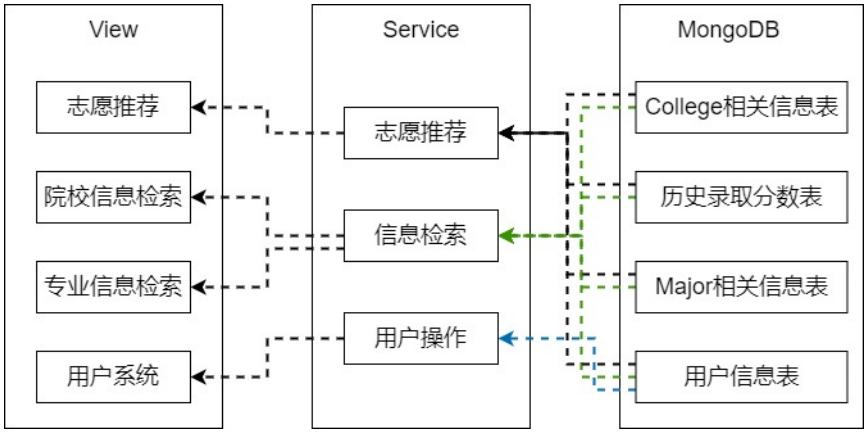

图 4-1　系统总体架构设计图

图4-2描述了系统的逻辑视图。交互层将交互数据传递给业务层，业务层再去从数据层获取数据来处理业务。

以登录行为为例，用户登录时的数据和行为会由 SignIn.vue 收集并向业务层发起请求，业务层的 UserImpl 会处理该请求并对数据层中 User 表的相关数据进行更新处理。

物理视图描述了系统如何部署于硬件上。本系统的前端页面运行于浏览器中，而后端以及数据库则运行于服务器上。

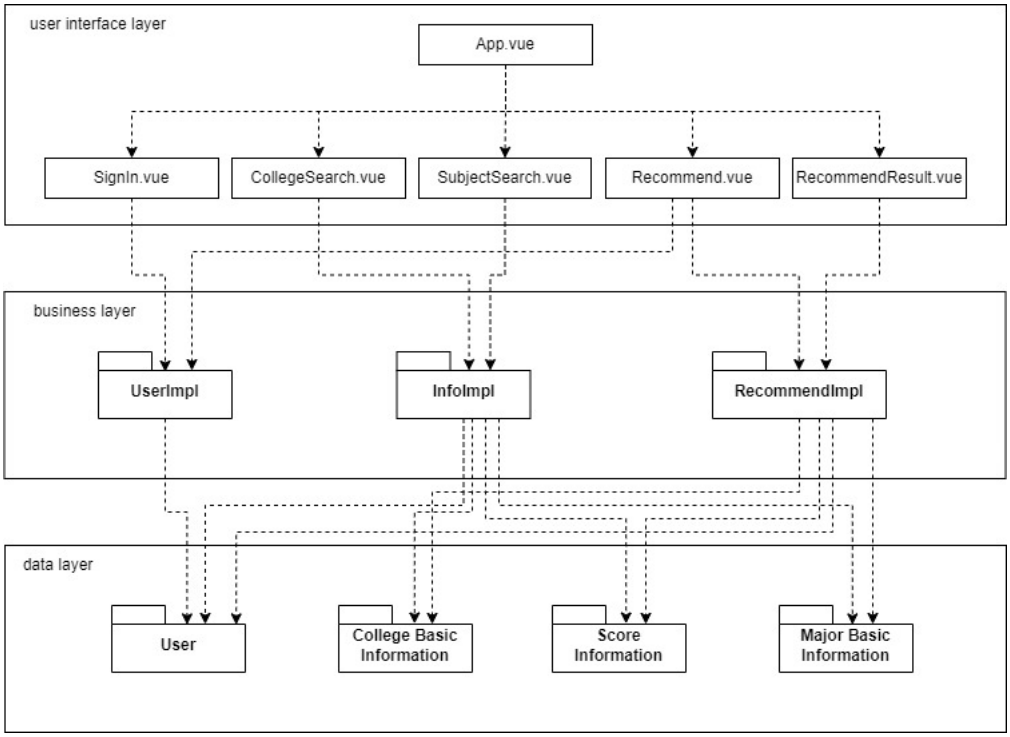

图 4-2　逻辑视图

系统的物理视图如图4-3所示。

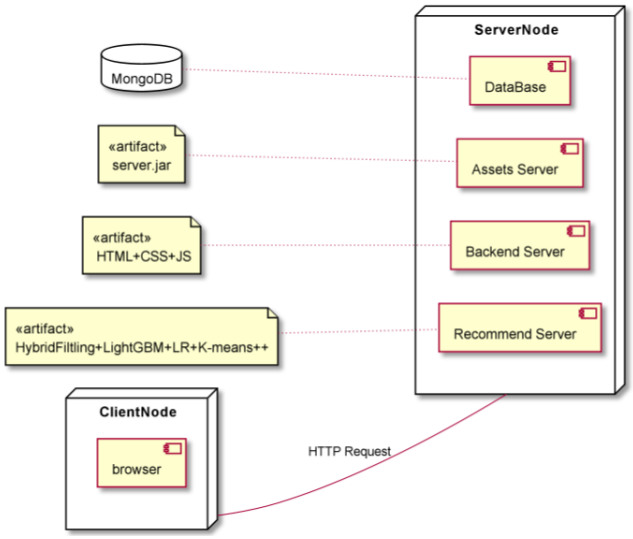

图 4-3　物理视图

图4-4描述了处理视图。处理视图描绘的是系统软件组织之间的通讯时序，数据的输入和输出，在本处以时序图的形式来表示。

图4-5描述了开发视图。开发视图清晰地展现了系统的开发逻辑，包括前后端各个模块的分层架构和它们之间的交互情况。

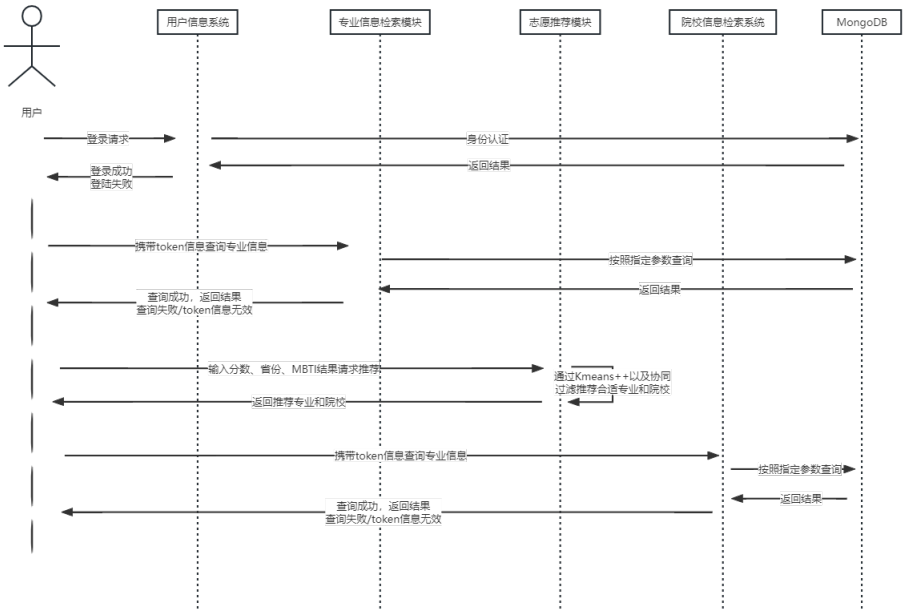

图 4-4　处理视图

4.1.2　系统模块划分

高考志愿推荐系统的模块划分依据功能可大致划分为用户信息模块、院校信息检索模块、专业信息检索模块和志愿推荐模块，见图4-6。

用户信息模块设计网络平台的基本功能，如登录注册、展示和修改用户基本信息，还包含本推荐系统所需要的问卷模块，用户在此模块填写 mbti 和 holland 性格测试问卷，系统推荐时参考用户的性格测试结果。

院校信息检索模块首先给用户提供院校信息搜索的功能，系统根据用户实时输入的查询请求在索引中查找匹配的相关字段并展示给用户，用户可以输入关键词，系统根据用户输入实时更新相关院校选项，用户可从提供的院校选项中选出想要了解的院校。此时院校详情模块展示系统经筛选和汇总的对用户有效的信息，包括院校基本图片和信息、院校开设的专业以及用户所在省份的专业近年分数线和变化趋势、就业去向、国内升学率和出国率等，上述信息经前端渲染，用直观的图像展示给用户，引导用户参考填报志愿时需要的信息。

专业信息检索模块同上述的院校信息检索模块，其中的专业详情模块专注于展示专业基本信息、专业优势院校及其评级和用户所在省份分数线、专业就业率、薪酬及其工作年限相关变化趋势。

志愿推荐模块包含较多的处理过程，可继续划分为数据获取模块、数据处理

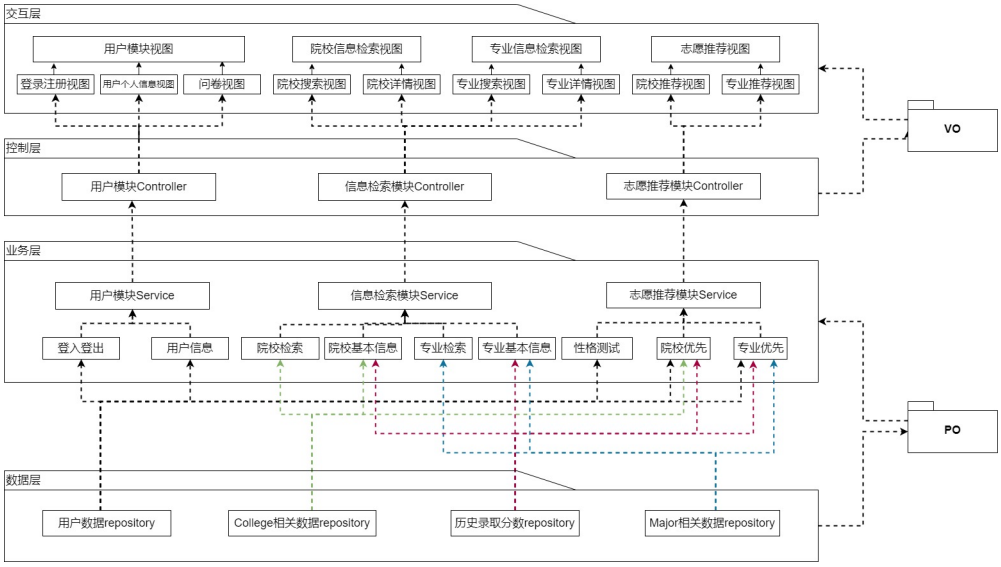

图 4-5　开发视图

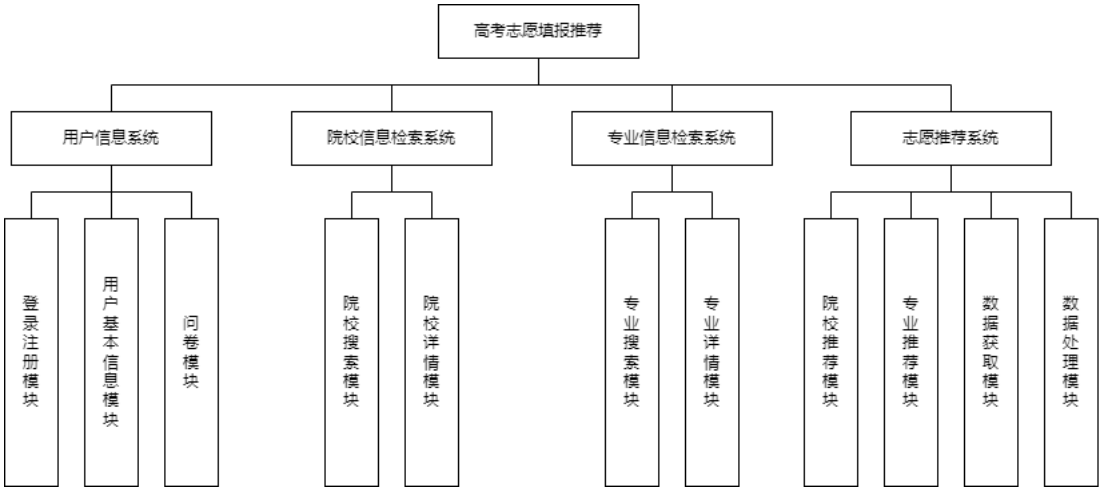

图 4-6　系统模块划分图

模块、院校推荐模块和专业推荐模块。数据获取模块主要由网络爬虫从网页自动抓取大致所需数据，并存入 MongoDB 数据库表中；数据处理模块则主要由爬虫和后端完成，爬虫对获取到的数据进行清晰等二次操作并以合理的方式存储进数据库中，后端连接数据库表完成获取具体所需数据并建立接口返回给前端数据；院校推荐模块和专业推荐模块则主要由算法负责，根据协同过滤等算法提供不同层级的推荐列表给前端展示。

4.2 后端模块详细设计

本部分主要介绍后端项目的整体包结构如图4-7，以及重要功能在后端完成中所构建的实体类及其属性和与其他实体类的关系。

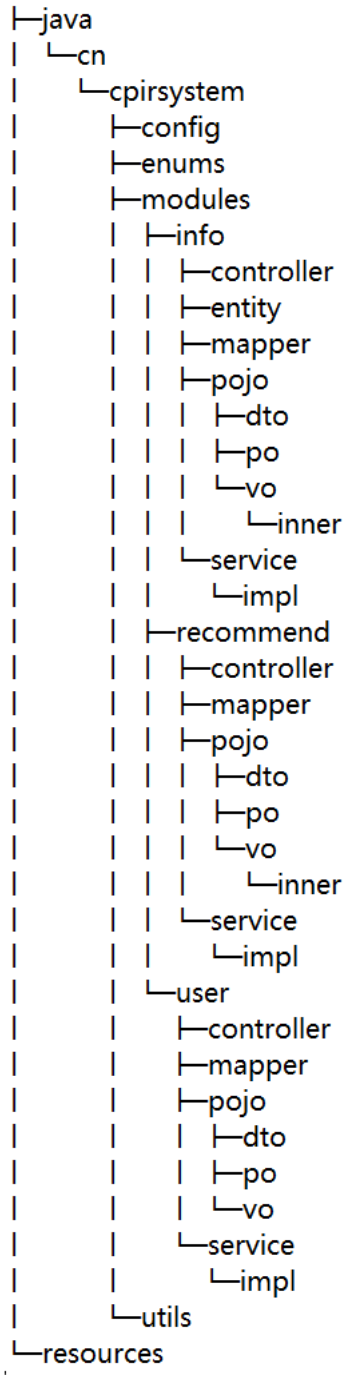

图 4-7　项目包结构示意图

cn.cpirsystem：根代码包，包含项目的核心配置文件和 Spring Boot 启动类。

cn.cpirsystem.config：配置代码包，包含应用程序的配置类。

cn.cpirsystem.enums：存放枚举类型相关的类，定义省份以及选科和数据库中字段值的对应关系。

cn.cpirsystem.utils：工具类代码包，包含一些通用工具类，包含 MD5 加密解密工具类、全局返回值类等。

cn.cpirsystem.modules：业务模块包，根据系统业务分模块，包含 info、recommend、user 模块，其中 info 模块负责院校和专业信息检索和详情展示，recommend 模块负责推荐所需问卷展示和结果计算、user 模块负责登录注册和用户基本信息展示。

controller：控制器代码包，包含处理 HTTP 请求和响应的控制器类，通常将请求转发给适当的 Service 层，以完成具体的业务逻辑，并将结果返回给用户。

entity：实体类包，包含后端具体实现时涉及的类，这些 java 对象与系统交互无关。

mapper：对象关系映射包，包含一组用于查询、添加、更新、删除等操作的接口，将数据访问相关的代码与业务逻辑代码分离，使得代码更加清晰、易于维护和测试。

pojo：包含 dto 数据传输对象、po 持久对象、vo 值对象。

pojo.po：持久化对象包，用于数据库表映射，是和数据库一一对应的实体类，包含数据表的所有或部分字段。

pojo.vo：表现层对象包，代表前端需要显示的数据。用于表示业务模型中的数据结构，包含业务模型中需要展示的字段，以及用于表示业务模型的状态和行为的属性和方法。

pojo.dto：数据传输对象包，负责接收前端传到后端的数据。

service：服务代码接口包，包含业务逻辑的接口。

service.impl：服务代码实现包，包含实现 service 中接口的实体类，实体类调用 Repository 来访问数据库，从而实现数据持久化。

4.2.1 用户信息相关功能模块

用户信息模块的基本功能有登录、注册、修改密码、获取用户信息，更新用户信息。VO 和 PO 的关系如图4-8所示，UserPO 中的 hldAnswer 数组表示用户填写的 holland 性格测试问卷各项答案集合，而 hldRes 表示后端根据 hldAnswer 和

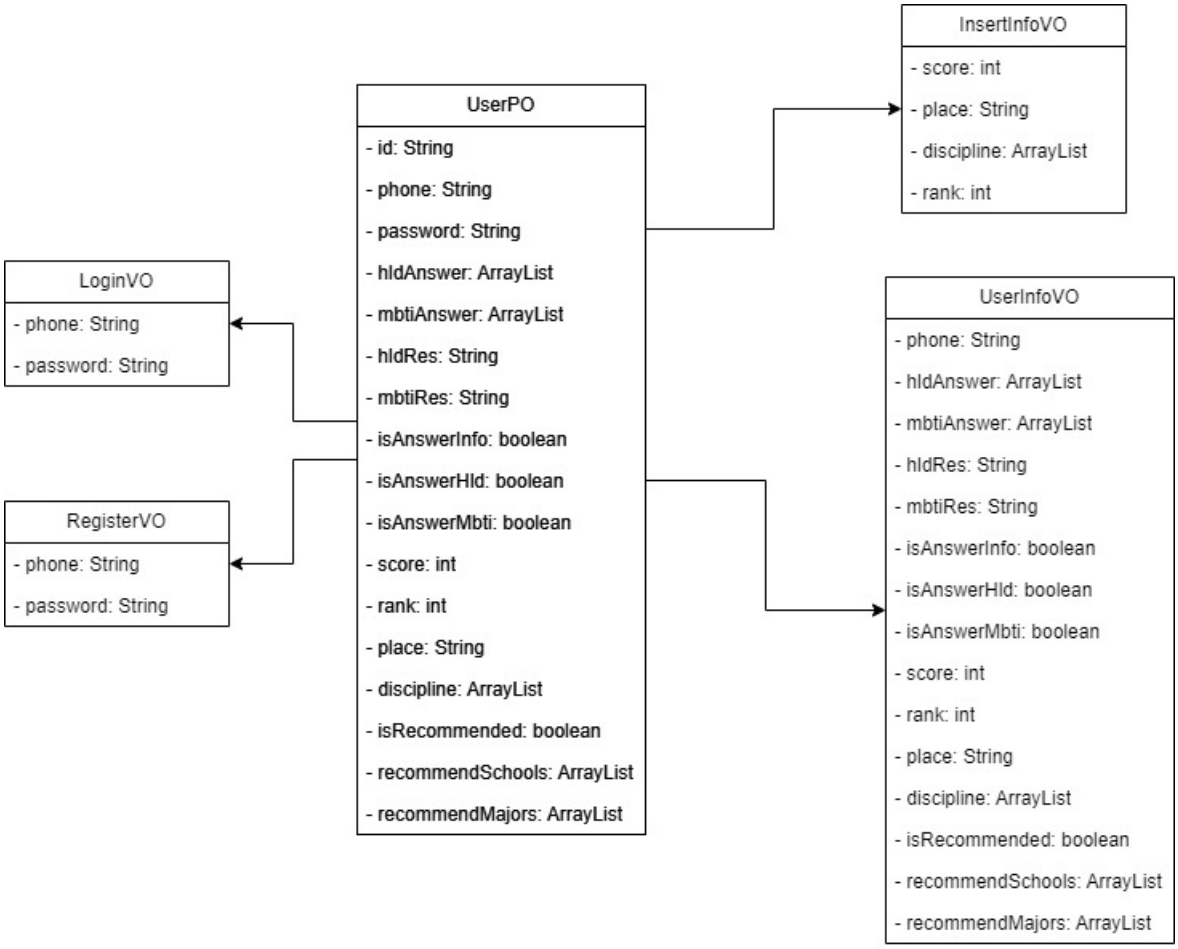

图 4-8　用户信息相关功能模块类图

holland 性格测试问卷的评分标准计算出的用户 holland 性格类型，isAnswerHld 这个布尔值表示用户是否作答过 holland 性格测试问卷，主要用于前端渲染数据等操作时参考，mbtiAnswer、mbtiRes、isAnswerMbti 与上述含义类似，discipline 数组表示考生考试选科。

用户登录和注册均通过键入手机号码和密码，注册时后端通过调用接口在数据库的 User 表中创建该用户字段，生成用户 ID 作为用户标识符，将用户密码经过 md5 加密后存入数据库字段，同时进行一些特定字段的初始化，例如 hldAnswer 和 mbtiAnswer 设置成默认数组，将 isAnswerInfo、isAnswerHld、isAnswerMbti、isRecommended 设置成 false 等等。登录时通过请求头的 Authorization 属性返回 token 给前端，前端进行本地存储，之后需要获取用户相关信息时，前端通过传递 token 给后端，后端通过 token 解析出用户电话和密码，然后检索数据库得出相关信息。

获取用户信息时，前端传递 token，后端查询数据库 user 表得到相关信息，

并返回给前端，注意此时返回的信息通过 UserInfoVO 承载，而 UserInfoVO 实际上比 UserPO 少了 id 和 password 字段，原因在于 id 字段是为了数据库存储和检索存在，与前端展示无关，并且减少 password 的传递从根本上减少密码泄漏的风险，提高了系统的安全性。

更新用户信息时涉及的具体信息有考生用户高考分数、所在省份、所选参考科目以及全省排名，以上四个信息是进行考生学院和专业推荐必备的信息，因此在考生用户希望推荐院校时，系统会检查考生用户是否填写上述四个信息，如果没有则提示用户更新信息。为便于确定考生科目，前端采用提供科目选项的方式引导用户填写后端可直接识别的科目信息，后端通过考生所在省份的高考政策和上述获取的考生参考科目确定考生可报考的院校，并将结果返回给算法端。

4.2.2 院校信息搜索功能模块

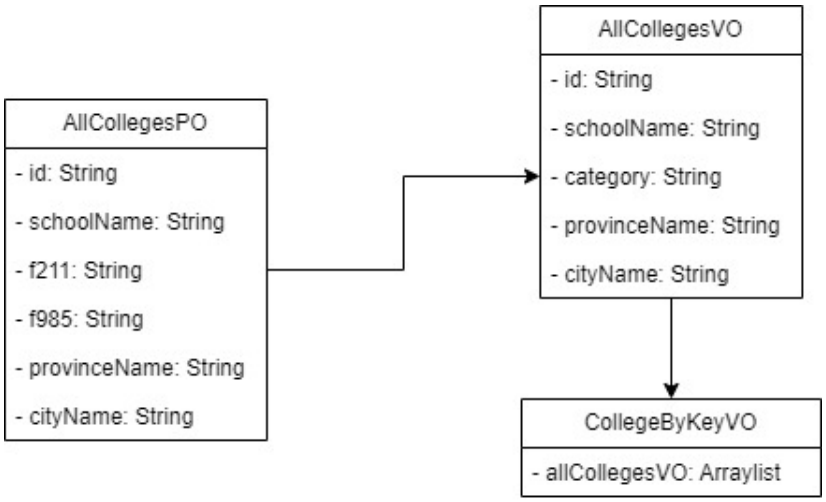

图 4-9 院校信息搜索功能模块类图

院校信息搜索功能模块提供查询所有院校列表和获取子字符串匹配院校列表的功能，VO 和 PO 及其之间的关系类图如图4-9所示，其中 f211 和 f985 在数据库中是字符串的格式，因此映射为实体类 AllCollegesPO 中的对象时也是 String 类型，当 f211 字段值为 “1” 时表示该校属于 211 院校，当值为 “2” 时则反之，f985 的含义同理。

在获取所有院校列表时前端需要获取院校的直接类别，即如果院校在数据库中的 f985 字段为 “1”，则返回给前端的 category 字段值为 “985”，如果院校在数据库中的 f985 字段为 “2”、f211 字段为 “1”，则返回给前端的 category 字段值

为 “211”，如果两个字段值都为 “2”，则返回 “本科院校”。上述过程通过后端进行逻辑判断和结果封装。

获取子字符串匹配院校列表时，数据库基于模式匹配算法进行匹配查询，根据具体情况进行优化，例如使用索引来加速查询，或者使用全文搜索等更高级的查询技术来实现更复杂的子字符串匹配查询。在后端具体实现时调用 mongoRepository 提供的框架，使用正则表达式来实现直接获取 CollegeByKeyVO 的对象并返回给前端，该对象封装了 AllCollegesVO 的数组。

4.2.3 院校信息详情展示模块

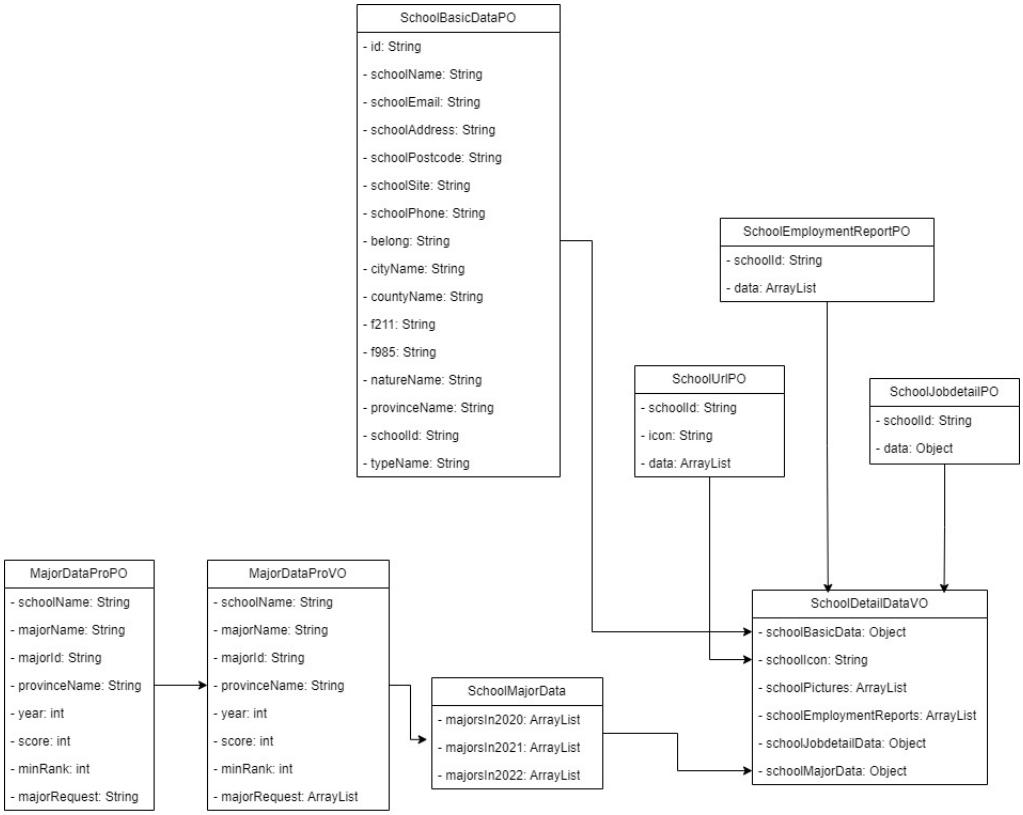

图 4-10 院校信息详情展示模块类图

院校信息详情展示模块涉及到的实体类较多，主要原因在于系统为了给考生用户提供对报考院校有用的信息，从多方数据源获取数据，这就意味着后端需要建立多个对应实体类从而从多个数据库表中获取数据，获取到的数据还需经过处理和加工，有些涉及时间限度的数据还需要以表格的形式进行前端展示，然后后端建立数据传输对象，以对象或列表的形式给前端返回数据。

SchoolBasicDataPO 实体类包含了院校的基本信息，其中 natureName 值为

“公办” 或者 “民办”、typeName 字段的值为 “综合类” 或 “专科类”，上述信息后端可以通过院校 id 从 SchoolBasicData 表中直接检索出。

系统需要获取当前院校考生所在省份的各个专业及其近三年分数线，这个功能点的实现需要查询与专业和院校都相关的数据表 MajorDataPro 表，MajorDataPro 表对应的实体类为 MajorDataProPO，其中的 majorRequest 字段表示该院校专业要求的考生参考科目，是长度为六的 01 字符串，不同位置的 1 表示不同的所需科目，后端需要将其解析为科目组成的字符串返回给前端，因此 MajorDataProVO 中的 majorRequest 是 ArrayList 类型。最后，系统需要近三年的分数线，所以将按照年份检索到数据包装入 ArrayList 中，并将近三年的各专业排名 ArrayList 包含入最终的返回对象 SchoolMajorData 中。

SchoolUrlPO 中的 icon 属性表示院校校徽图片的部分 url，后端需要加上 url 前缀以形成可用 url 并装入最终传递给前端的对象 SchoolDetailDataVO 的 schoolIcon 属性中。data 对象则是一个 url 列表，包含多张代表院校形象的图片，同样最终装入 SchoolDetailDataVO 的 schoolPictures 属性中返回给前端。

SchoolEmploymentReportPO 中的 data 数组中存放的是近三年就业质量报告的对象，每个对象包括年份，就业质量报告名称和可直接访问的 URL，后端将 data 封装入 SchoolDetailDataVO 的 schoolEmploymentReports 属性中传递给前端。

SchoolJobDetailPO 中的 data 对象包含以下属性：jobrate、province、company、abroad 等，分表代表院校毕业生就业率、就业所在省份、进入的公司、出国进修率等。

4.2.4 专业信息详情展示模块

专业信息详情展示模块主要提供专业详情数据，具体包括专业的基本信息、专业强势院校（specialSchool）、专业就业率（jobrate）、专业学习内容、专业类型、专业毕业就业情况。

其中的专业基本信息通过 MajorIdFinishedPO 基本直接映射组成，包含专业代码、专业名称、专业就读年限、专业所属大类、专业开设院校等级（level1_name）、专业开设大类（level2_name）和专业开设小类（level3_name）。

专业强势院校的获取需要通过多表查询并最终将结果拼装入 MajorInfoVOInner 类型的 ArrayList 中。MajorInfoPO 中的 specialSchool 数组包含了该专业的强

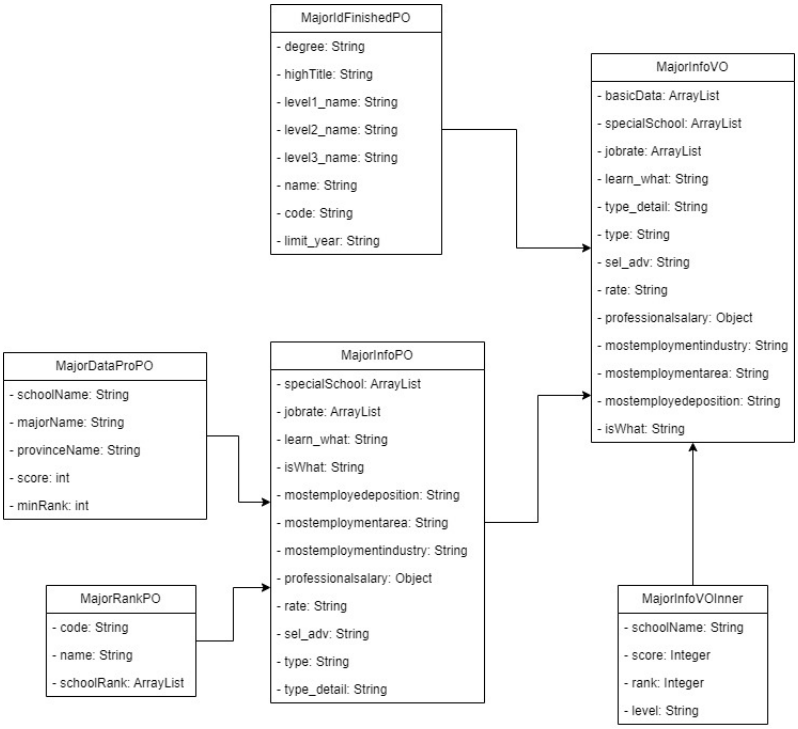

图 4-11　专业信息详情展示模块类图

势院校名称，高考志愿填报推荐系统还需要获取强势院校在考生所在省份的录取分数线、最低排名以及院校评级。录取分数线、最低排名的获取需要通过 MajorDataProPO 获取，院校评级需要通过 MajorRank 获取。

4.2.5　问卷模块

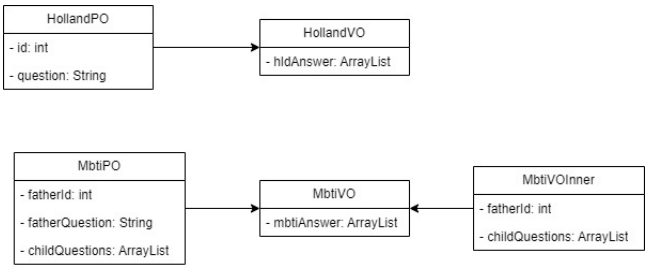

图 4-12　问卷模块

问卷模块的存在原因在于本高考志愿填报系统采用的推荐算法需要根据用户的性格来进行院校和专业推荐，而用户性格就由问卷确定，为增加性格测试结果的可信度和科学性，我们采用了 MBTI 性格测试问卷和霍兰德职业测试问卷。[20]MBTI 全称为 “Myers-Briggs Type Indicator”，是一种个性评估工具，基于心理学家 Carl Jung 的心理类型理论而设计，通过测量人们在以下四个二元对立

维度上的偏好，来描述人们的个性。霍兰德（Holland）职业测试是一种评估人们职业兴趣的工具，基于霍兰德理论，将职业分为六种类型：现实型、研究型、艺术型、社会型、企业型和常规型。

高考志愿填报推荐系统首先需要从后端获取测试问题，之后在用户填写问卷并提交时后端获取问卷结果并根据测试评分标准解析问卷答案，最终得到测试结果并返回给算法部分。这个过程中首先需要重点关注的是数据传递的格式，后端和前端的交互需要用较为简洁且易于渲染的数据格式，后端和算法的交互则需要易于算法解析的特定数据格式，如图4-12所示，由于霍兰德职业测试问卷的每个问题都是判断题的形式，所以对应的 HollandPO 中的 id 字段表示题号、question 字段表示具体具体，当前端请求获取霍兰德职业测试问卷题目时，后端返回 HollandPO 的列表，当考生用户填完问卷并提交时，前端向后端返回一个数组，数组中的每一项如果为 “A”，则表示该判断题答案为正确，如果为 “B”，则反之。由于 MBTI 性格测试问卷分为四大题，每大题有题目要求描述，且第一、三大题为选择题，第二、四大题为判断题，所以采用 MbtiPO 的格式存放数据，fatherId 表示大题题号，fatherQuestion 表示大题题目要求描述，childrenQuestion 是由小题题目和选项组成的数组，而这个数组的内部结构又是由题目和选项组成的。当考生用户填完问卷并提交时，前端像后端返回一个数组，数组是由对象组成的，即 MbtiVOInner 对象，该对象由 fatherId 和 childQuestions 的子问题答案列表组成。

第五章　系统后端功能模块的实现

5.1　用户信息相关功能模块

5.1.1　登录注册

用户注册时提供电话号码和密码，后端通过 RegisterVO 对象获取对应的数据。首先，通过调用 MongorRepository 的接口方法查询数据库中是否已存在该用户信息，即向方法 findByPhone 中传入待注册用户的电话号码进行查询，如果方法返回的 Optional 对象为空时，则表示该用户尚未注册，系统可允许用户进行注册，反之则向前端返回 “该用户已存在” 的提示 message。之后，后端根据 md5 加密算法对用户自定义的密码进行加密，此时通过 MongoRepository 的 insert 接口向 user 表插入用户字段，包括用户电话、加密的用户密码和系统生成的用户 id。最后，为便于系统之后的开发和操作逻辑，还要设置一系列默认值，如问卷答案初始值、一系列供前端渲染判断的布尔值等。更新操作通过 mongoTemplate 提供的 update 接口，传入 Query 参数指定待更新的目标字段。用户登录时后端通过 loginVO 获取用户电话号码和密码，将密码通过 MD5 工具类转化成哈希值并存储到变量中待检测。首先，使用 MongoRepository 的 findByPhone 接口，根据电话号码查询数据库，如果返回的 Optional 对象为空，即系统不存在相关用户，后端直接返回 “登录账号或密码错误” 的 message，这里遵循 “最小化权限” 原则，不具体说明登录失败的原因，只告诉用户登录是否成功，主要是为了提高安全性，防止给攻击者提供利用这些信息来破解密码或攻击系统的机会，如果后期确实需要向用户提供更具体的信息，系统可以通过其他方式，如邮件或短信等方式提供。之后，系统通过 findByPhoneAndPassword 的方法检索用户，如果查询到用户，则开始使用工具类 JwtUtils.getJwtToken 生成 token，最后向前端返回 token，之后的每次前端请求必要时会通过请求头传递 token。具体代码如图5-1。

```
@Override
public Result login(LoginVO loginVO){
    String phone = loginVO.getPhone();
    String pwd = MD5.encrypt(loginVO.getPwd());

    Optional<UserPO> user0 =
        userRepository.findByPhone(phone);

    if(!user0.isPresent()){
        return Result.error().message("登录账号或密码错误");
    }
    Optional<UserPO> user =
        userRepository.findByPhoneAndPassword(phone, pwd);
    if(user.isPresent()){
        UserVO userVO = user.get().UserVO();
        String[] res = getRes(userVO);
        return Result.ok().data("token",
            res[0]).data("phone", res[1]);
    }
    return Result.error().message("登录账号或密码错误");
}
```

图 5-1　登录注册

5.1.2　获取、更新用户信息

如图5-2获取用户信息则直接通过 MongoRepository 提供的接口检索出 UserPO，之后再通过 BeanUtils.copyProperties 进行相似类赋值得到 UserVO，并传给前端。更新用户信息时需要更新的具体字段通过 InsertInfoVO 设置和获取，后端通过 mongoTemplate.updateFirst 方法更新数据库 user 表中的特定字段的特定属性。

5.2　院校信息搜索功能模块

院校信息搜素主要向前端提供两个接口，“获取所有院校列表” 和 “实时获取包含前缀的院校列表”。获取所有院校列表直接使用 MongoRepository 提供的 findAll 接口，并将 AllCollegesPO 转化成 AllCollegesVO 传递给前端。实时获取包含前缀的院校列表是为了给考生用户搜索时提供辅助选项的，后端采用模糊搜索的方式实现，具体实现时使用 MongoRepository 的 findBySchoolNameLike 接口，其中接口命名中的 Like 表示模糊查询。具体代码如图5-3

```
    @Override
    public Result getUserInfo(HttpServletRequest request){
        String token = request.getHeader("Authorization");
        String phone = JwtUtils.getCheckPhoneByJwtToken(token);
        Optional<UserPO> user =
            userRepository.findByPhone(phone);
        UserInfoVO userInfoVO =
            UserSession.User2UserInfoVO(user.get());
        if(userInfoVO == null){
            return Result.error().message("未找到用户信息");
        }
        return Result.ok().data("userInfo", userInfoVO);

    }

@Override
    public Result insertInfo(String token, InsertInfoVO
        insertInfoVO) {
        String phone = JwtUtils.getCheckPhoneByJwtToken(token);
        Query query = new
            Query(Criteria.where("phone").is(phone));
        Update update = new Update();
        update.set("place", insertInfoVO.getPlace());
        update.set("score", insertInfoVO.getScore());
        update.set("discipline",insertInfoVO.getDiscipline());
        update.set("rank", insertInfoVO.getRank());
        update.set("isAnswerInfo",true);
        mongoTemplate.updateFirst(query,update, "user");
        return Result.ok().message("更新用户信息成功");
    }
```

图 5-2　获取、更新用户信息

5.3　院校信息详情展示模块

在院校信息详情展示模块中，后端要向前端提供较为丰富的数据，需要进行多表查询。首先，根据用户电话号码查询 user 表得到用户所在省份 id，做这一步的原因在于为下面用户所在省份院校分数线的查询提供查询参数。之后，通过 majorDataProRepository 的 findBySchoolNameAndProvinceNameAndYear 方法查询近三年分数线并存日返回对象 schoolMajorData 中。然后，通过 schoolBasicDataRepository 的 findBySchoolName 的方法直接查询院校基本信息，此处只需要学校名称。之后，根据学校名称从 SchoolEmploymentReport 表中查询学校近年就业报告，具体实现使用的是 schoolEmploymentReportRepository 的 findBySchoolId 方法，此时获得的是一个包含就业报告的对象，后端需要判断该对象是否为空，如果不为空再取出其中的就业报告并放入返回对象中。之后用类似

```
@Override
   public Result getCollegesByKey(String key){
      List<AllCollegesPO> majorIdPOList =
         allCollegesRepository.findBySchoolNameLike(key);
      CollegesByKeyVO collegesByKeyVO = new
         CollegesByKeyVO();
      ArrayList<AllCollegesVO> arrayList = new ArrayList<>();
      for(int i = 0; i < majorIdPOList.size(); i++){
         arrayList.add(majorIdPOList.get(i).toAllCollegesVO());
      }
      collegesByKeyVO.setAllCollegesVO(arrayList);
      return Result.ok().data("colleges",arrayList);
   }
```

图 5-3　院校信息搜索功能模块

的方式从 schoolJobdetailDataPro 表中获取具体的院校毕业生就业信息。最后从 schoolUrlDataPOOptional 表中获取学校的图片 url，直接获取的数据库中的 url 是省略前缀的不可用 url，后端需要进行拼接，然后返回给前端可用的 url，具体的实现通过 LinkedHashMap 结构获取数据库表数据。

5.4　专业信息详情展示模块

专业信息详情展示模块同上述的院校信息详情展示模块，也需要提供较多的数据。首先通过获取专业基本信息，通过继承了 MongoRepository 接口的 majorIdFinishedRepository 接口中的 findByCode 方法获取 majorIdFinishedPO，之后再通过 BeanUtils.copyProperties 方法将 majorIdFinishedPO 存入返回数据 majorInfoVO 中。之后，需要获取考生用户所在省份，为了便于之后获取专业强势院校的近年分数线。然后，通过 majorInfoRepository 的 findByName 方法获取 majorInfoPO，majorInfoPO 中包含了 specialSchool 列表，表中存储专业强势院校的名称，后端通过名称在 majorDataPro 数据表中检索该院校该专业的近年分数线，这里数据可能存在缺失，需要判断是否存在，如果数据没有完全获取，则向前端返回 “部分院校数据未爬取”message。最后，还需要修改 majorInfoPO 中的 jobrate，原 jobrate 保存的是字符串形式的多院校百分值，前端需要的数据是整型的平均百分值，后端通过正则表达式提取出数据再进一步计算得出最终结果。具体代码如图5-4

```
LinkedHashMap linkedHashMap = (LinkedHashMap)j;
          String rate = (String) linkedHashMap.get("rate");
          //正则表达式提取
          List<String> strs = new ArrayList<String>();
          String pattern = "(\\d+)";
          Pattern p = Pattern.compile(pattern);
          Matcher m = p.matcher(rate);
          while(m.find()) {
             strs.add(m.group());
          }
          int rate1 = Integer.parseInt(strs.get(0));
          int rate2 = Integer.parseInt(strs.get(1));
          linkedHashMap.replace("rate",(rate2+rate1)/2 + "%");
          newJobrate.add(linkedHashMap);
```

图 5-4　专业信息详情展示模块

5.5　问卷模块

问卷模块设计两个问卷，MBTI 性格测试问卷和 Holland 职业性格测试问卷，后端需要向前端提供的接口是对这两个问卷的问卷内容获取和问卷结果计算。问卷内容以较为容易解析的方式存储在数据库中，可以直接通过 hollandRepository 和 mbtiRepository 提供的 findAll 接口获取并返回给前端。在计算 MBTI 性格测试问卷时，需要根据较为复杂的评分标准进行计算。评分标准形如题 1 用户选 A 则 I 性格因子值 +1，选 B 则与 I 性格因子相反的 E 性格因子值加 1，所有题统计结束后，如果 I 性格因子的值大于 E 的，则用户最终是 I 型人格，即用户最终的 MBTI 性格测试结果包含 I，反之包含 E，类似于 I、E 这样相对的性格因子还有三对。具体实现方式是，后端首先用 ArrayList 直接接收前端返回的用户作答结果列表，之后用 HashMap<Character,Integer> 存储八种性格因子和其对应的用户选择个数，初始值都为 0，再用一个 trueList 字符列表存储选 A 时值要加一的性格因子。做好上述准备后，遍历前端传过来的用户填写答案数组，如果该题值为 A 则将 HashMap 中以对应的 trueList 该题的值的键值加一，这样以空间换时间的方法避免了大量的 if-else，提高了程序解析的效率。最后再依据 MBTI 性格测试评分标准比较性格因子，得到最终的由 4 个性格因子字符组成的 MBTI 结果字符串，具体过程如图5-5。

5.6 后端效果的实现

院校信息搜索模块如图5-6所示，左边的搜索框会展示给用户相关院校提示信息列表，并根据用户输入后端实时返回更精确的提示列表。

院校信息详情展示模块的后端效果如图5-7，页面的数据从接口 getDetailed-Description 中获取，页面从上到下依次有后端通过 schoolUrlVO 传递的院校图标和院校图片，之后是院校基本信息，前端通过 schoolBasicDataVO 获取，再下面是院校专业近三年分数线及其变化趋势如图5-8，最后是前端加工渲染后的高可视度的 schoolJobdetailVO 和 schoolEmploymentVO。

专业信息搜索的效果同院校信息搜素，此处不再赘述。专业信息详情展示模块的效果如图5-9，从上到下首先展示的是 majorInfoVO 中的专业基础信息，之后展示的列表是从 majorDataProPO 中获取的专业优势院校相关信息，由于部分院校信息的保密性，分数线和排名存在暂无数据的情况，再然后就是前端经渲染用图表展示的 majorInfoVO 中关于就业率和薪酬的数据。

MBTI 性格测试模块如图5-10所示，Holland 职业性格测试类似，此处不再赘述。

```
private String calculateMbtiUtil(ArrayList mbtiAnswer){
      HashMap<Character,Integer> resHashMap = new
         HashMap<>();
      resHashMap.put('E',0);
      resHashMap.put('I',0);
      resHashMap.put('S',0);
      resHashMap.put('N',0);
      resHashMap.put('T',0);
      resHashMap.put('F',0);
      resHashMap.put('J',0);
      resHashMap.put('P',0);
      Character trueList[] =
            {'J','P','S','E','N','F','P','E','J','J',
            'P','I','S','E','N','F','P','I','E','J',
            'P','I','E','N','P','I','I','J','N','F',
            'T','S','P','E','I','J','N','F','T','S',
            'P','I','J','N','F','T','S','I','J','N',
            'F','T','S','I','N','F','T','S','J','I',
            'S','E','N','F','P','I','E','J','T','J',
            'T','S','N','F','T','S','N','F','T','S',
            'F','T','S'};
      ArrayList<String> answers = new ArrayList<>();
      for(int i = 0; i < mbtiAnswer.size(); i++){
         MbtiVOInner mbtiVOInner = (MbtiVOInner)
            mbtiAnswer.get(i);
         ArrayList childQuestions =
            mbtiVOInner.getChildQuestions();
         for(int j = 0; j < childQuestions.size(); j++){
            String s = (String)childQuestions.get(i);
            answers.add(s);
         }
      }
      for(int i = 0; i < answers.size(); i++){
         String s = answers.get(i);
         if(s.equals("A")){
            Character c = trueList[i];
            int t = resHashMap.get(c);
            resHashMap.put(c,t+1);
         }
      }
      String res = "";
      res += resHashMap.get('E') > resHashMap.get('I') ? "E"
         : "I";
      res += resHashMap.get('S') > resHashMap.get('N') ? "S"
         : "N";
      res += resHashMap.get('T') > resHashMap.get('F') ? "T"
         : "F";
      res += resHashMap.get('J') > resHashMap.get('P') ? "J"
         : "P";
      return res;
   }
```

图 5-5　问卷模块

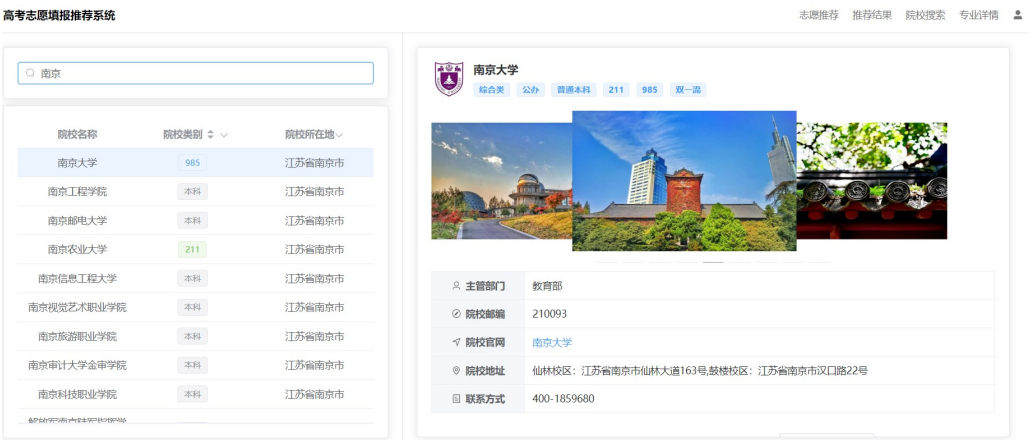

图 5-6　院校信息搜索功能模块效果

图 5-7　院校信息详情展示模块效果

图 5-8　近五年分数线变化趋势

图 5-9　专业信息详情展示模块效果


志愿推荐
推荐结果
院校搜索
专业详情
系统说明
①填写基本信息后即可进行志愿推荐
②MBTI性格测试和霍兰德职业兴趣测量均为可选项，完成的测试越多，推荐结果就越准确
MBTI性格测试
一. 哪一个答案最能贴切的描绘你一般的感受或行为?
1. 当你要外出一整天，你会
计划你要做什么和在什么时候做
说去就去
2. 你认为自己是一个
较为随兴所至的人
较为有条理的人
3. 假如你是一位老师，你会选教
以事实为主的课程
涉及理论的课程
4. 你通常
与人容易混熟
比较沉静或矜持
5. 一般来说，你和哪些人比较合得来?
富于想象力的人
现实的人
6. 你是否经常让
你的情感支配你的理智
你的理智主宰你的情感

图 5-10　问卷模块效果

第六章　总结与展望

6.1　总结

高考形势长势凝聚着较高的社会关注度，而除了关注考生在学习方面的问题，还重视关键的志愿填报内容，如何进行正确的志愿信息搜集和志愿填报建议，一直是考生需要面临的问题，而填报决策这方面的信息差和经验往往较难打破。对于这一问题，我们开发了高考志愿填报推荐系统。本文所提出的系统首先解决了志愿和专业推荐的功能，同时提供了对考生填报志愿有重要参考意义的信息，例如院校几年录取分数波动大的专业、专业强势院校的录取分数线和专业评级等，并且系统对上述数据进行加工和处理，以直观的形式如图表来向用户展示，一方面引导较为迷茫的高考考生用户关注重点信息，另一方面也极大降低了考生的搜索难度，帮助考生完成挑选自己心仪院校和专业的任务。

首先，文章在第一章阐述了开发高考志愿填报推荐系统的目的和研究意义，并汇总介绍了国内的研究现状，梳理了其推荐系统的逻辑和方法，指出了推荐中采取的方法，提供了参考意义，最后列出了本文的总体结构。随后，本文简要介绍了 Spring Boot、Maven、MongoDB 等后端使用的重要技术，说明了为何选用这些技术，以及技术的使用场景。之后，本文从功能性描述和非功能性描述说明了系统的需求分析过程，给出了详细的系统用例，并从总体架构的角度对系统进行了细分。其中针对作者负责的部分，对分属后端的用户信息相关功能模块、院校信息搜索功能模块、院校信息详情展示模块、专业信息详情展示模块进行了从数据设计到实现逻辑的描述，并给出了相关类图和关键代码，尽可能清晰地描述了实现思想和实现过程。

6.2　展望

本文介绍的高考志愿填报推荐系统已经能够实现简单但满足用户需求的志愿推荐，基本符合所在省份考生的一般录取情况。能够满足大部分考生的基本

需求，但是系统仍然存在相对不成熟的一面。

首先，在作者负责的后端开发方面，代码架构和逻辑需要进一步优化。除此之外，部分信息的缺失也是一大遗憾，例如最近一年的录取排名、准确的分数和排名对应关系、特殊院校和特殊专业等等，导致这些信息缺失的原因可能在于当地教育厅的保密措施。其次，在院校和专业详细详细信息展示模块涉及到的相关信息仍然需要用户对特定字段二次搜索，例如当考生用户查看院校详细信息时，如果想要查看该校某专业的具体信心，则首先可能需要在本站内搜索，其次如果对该院校的该专业想要进一步了解则还需要前往浏览器。第三点，高考志愿填报系统在实际应用方面，虽然功能较为完善，但面对每年特定短时间段内数据的更新以及突然大量的用户访问，我们的系统从软件到硬件、从爬虫到后端到算法都面临着上线可能发生问题，目前缺乏这方面的大数据量测试。最后，项目在改进前版本的基础上添加了一定的需求创新点，例如展示今年录取分数线变化较大的院校专业、强势专业评级等，但并未开发系统的商业价值，站在用户的角度来说可能忽视系统的重点功能。

参考文献

[1] 李木洲. 效率、科学与公平：高考制度现代化的内部动因[J]. 中国教育学刊, 2021: 44-49.

[2] 王泽卿, 季圣鹏, 李鑫, 等. 基于分数线预测的多特征融合高考志愿推荐算法[J]. 计算机科学, 2022: 254-260.

[3] 朱沛沛, 刘海峰. 70 年回眸：高考制度的创立与新时代变革[J]. 中国高教研究, 2022: 48-53.

[4] 徐伟琴, 钟秉林. “学校优先”还是“专业优先”?——新高考背景下学生志愿填报取向对录取匹配度的影响[J]. 清华大学教育研究, 2022: 81-92.

[5] 高丽丽, 樊彩虹. 基于大数据的高考志愿填报推荐系统的设计[J]. 电子技术与软件工程, 2021: 213-215.

[6] 刘明奇, 程江珂, 陈晓兰. 高考志愿填报辅助决策系统的设计与实现[J]. 现代信息科技, 2022: 38-40.

[7] 黄戴琴, 周强, 虞飞华. 基于大数据分析的“浙江新高考”志愿填报辅助推荐平台研究[J]. 电脑安全与技术, 2019: 75-78.

[8] 李华, 张宇, 孙俊华. 基于用户模糊聚类的协同过滤推荐研究[J]. 计算机科学, 2012: 83-86.

[9] 王红丽. 学生高考志愿选择影响因素的实证研究——以国内一流大学为例[J]. 时代教育, 2015: 45-46.

[10] 黄泽鑫. 基于知网高考志愿辅助系统的设计[J]. 网络安全技术与应用, 2019: 41-43.

[11] SURYOTRISONGKO H, JAYANTO D P, TJAHYANTO A. Design and Development of Backend Application for Public Complaint Systems Using Microservice Spring Boot[J]. Volume, 2017.

[12] KESHANI M, VOS S, PROKSCH S. On the relation of method popularity to breaking changes in the Maven ecosystem[J]. Journal of Systems and Software, 2023.

[13] 刘云汉, 郭继光. 基于 MongoDB 分片集群的海量数据存储设计[J]. 电子技术与软件工程, 2022: 254-258.

[14] 赖杰. 基于 MongoDB 的分布式事务优化与实现[J]. 电子科技大学, 2022: 71.

[15] 冯国军, 贺占庄, 吕瑛. 基于 DBAF 算法的 MongoDB 负载均衡策略[J]. 微电子学与计算机, 2021: 52-55.

[16] RIVEST R. The MD5 Message-Digest Algorithm[J]. Internet Request for Comments, 2020.

[17] 靳燕. 基于 MD5 算法的文件完整性检测系统分析及设计[J]. 网络安全技术与应用, 2019: 36-38.

[18] 廖若飞. 基于 Docker 容器技术的 SpringWeb 部署研究[J]. 现代信息科技, 2022: 100-103.

[19] 龙平, 周超, 徐旭东, 等. 基于 Docker 技术的信息系统自动化运维分析[J]. 自动化应用, 2022: 65-67.

[20] FATIMA N, GUL S, AHMED J, et al. A rule-based machine learning model for career selection through MBTI personality[J]. Mehran University Research Journal of Engineering and Technology, 2022.

致　　谢

首先，感谢我的导师葛季栋教授。葛老师在整个研究过程中给予我们耐心的指导和悉心的关注，使我们能够顺利完成本次研究。葛老师在我继续求学的过程中提供了很多帮助，希望凝聚了葛老师的想法和多位学长同学的努力的高考志愿填报推荐系统可以帮助更多人。感谢我的家人，感谢他们一直以来对我的支持和鼓励，让我在生活中保持着坚定的信念和积极向上的心态。感谢一同完成项目的吴子🄵、单金明、任毅三位朋友，在研究过程中，我们互相启发和帮助，不断发现并解决问题。感谢母校，感谢她提供了良好的学习研究和生活环境。还有很多人给予了我支持和帮助，由于篇幅所限，未能一一列举，在此，我对所有给予我帮助和支持的朋友表示最真挚的谢意。