

**普**

**通**

**本**

**科**

**毕**

**业**

**设**

**计**

**题 目：基于NodeJs的即时通讯招聘webAPP的设计与实现**

**学 院 软件与物联网工程学院**

**学生姓名 龚巧琳 学 号 0154246**

**专 业 软件工程 届 别 2019届**

**指导教师 汪彩霞 职 称 讲师**

二〇一九年五月

**普通本科生毕业论文（设计）诚信承诺书**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 毕业论文（设计）题目 | | 基于NodeJs的即时通讯招聘webAPP的设计与实现 | | | |
| 学生姓名 | 龚巧琳 | 专 业 | 软件工程 | 学 号 | 0154246 |
| 指导老师 | 汪彩霞 | | 职 称 | 讲师 | |
| 所在学院 | 软件与物联网工程学院 | | | | |
| **诚信承诺**  本人慎重承诺和声明：  我承诺在毕业论文（设计）活动中遵守学校有关规定，恪守学术规范，在本人的毕业论文中未剽窃、抄袭他人的学术观点、思想和成果，未篡改研究数据，如有违规行为发生，我愿承担一切责任，接受学校的处理。  学生（签名）：  年 月 日 | | | | | |

摘 要

在欣欣向荣的21世纪互联网时代中，即时通讯统越来越流行，无论是PC端还是移动客户端，几乎随处可见，其功能性也不再简单的局限于单纯的社交联系，越来越多运用在简化与方便人们日常沟通上。即时聊天招聘webAPP就是其中的一种，大大的简化了求职与招聘的繁琐步骤，简化沟通成本，大大的提升效率。

该项目主要是由注册、登录、完善信息、浏览列表、发起聊天等模块组成。针对不同使用用户跳转不同的APP内置页，求职者注册登录后完善个人信息被系统推送至牛人列表，之后招聘者注册登录后可以在牛人列表找到并与其进行实时沟通。同时求职者也可以在BOSS列表找寻自己心仪的公司BOSS进行沟通面试。这些模块帮助招聘者与求职者快速完成沟通，简化流程。

该项目采用最新React 16+Redux+React Router 4技术开发复杂的单页面应用，并使用node.js+express+socket.io管理实时应用的后端，同时对系统的模块设计与需求分析采用思维导图的形式进行描述，将整个系统的功能结构完整的列出来。

【关键词】NodeJS 即时通讯 React

**Abstract**

In the thriving 21st century Internet era, Instant Messenger are becoming more and more popular. Both PC-side and mobile clients are almost everywhere. Their functionality is no longer limited to simple social connections. More and more applications are being used. In simplifying and facilitating people's daily communication. Instant chat recruitment webAPP is one of them, which greatly simplifies the cumbersome steps of job search and recruitment, simplifies communication costs, and greatly improves efficiency.

The project is mainly composed of modules such as registration, login, perfect information, browsing list, and starting chat. For different users to jump to different APP built-in pages, the job seeker registers and logs in and the personal information is pushed to the list of cattle. After the recruiter registers and logs in, he can find and communicate with him in the cow list. At the same time, job seekers can also find their favorite company BOSS in the BOSS list for communication interviews. These modules help recruiters and job seekers quickly communicate and streamline processes.

The project uses the latest React 16+Redux+React Router 4 technology to develop complex single-page applications, and uses node.js+express+socket.io to manage the back end of real-time applications, while using the mindset for system module design and requirements analysis. The form of the diagram is described, and the functional structure of the entire system is completely listed.

【Key words】NodeJS; Instant Messenger; React

**目 录**

[1绪论 1](#_Toc7281912)

[1.1项目背景与重要问题 1](#_Toc7281913)

[1.2项目的意义和研究的重要性 2](#_Toc7281916)

[1.3研究的内容和主要工作 2](#_Toc7281919)

[1.4论文章节 3](#_Toc7281922)

[2系统的开发方法及相关技术 5](#_Toc7281923)

[2.1后端框架选择 5](#_Toc7281924)

[2.2前端框架选择 6](#_Toc7281927)

[2.3数据库选择 7](#_Toc7281931)

[3需求分析 9](#_Toc7281933)

[3.1可行性分析 9](#_Toc7281934)

[3.2功能需求 9](#_Toc7281938)

[3.3系统用户需求 13](#_Toc7281945)

[3.4数据需求 13](#_Toc7281947)

[3.5非功能需求 14](#_Toc7281948)

[4系统设计 16](#_Toc7281952)

[4.1总体设计 16](#_Toc7281953)

[4.2重要功能模块设计 17](#_Toc7281954)

[4.3数据库设计 23](#_Toc7281959)

[5系统实现 28](#_Toc7281963)

[5.1主要模块实现 28](#_Toc7281964)

[5.2后台管理实现 32](#_Toc7281968)

[5.3主要关键架构与优化 34](#_Toc7281969)

[6总结与展望 39](#_Toc7281974)

[6.1总结 39](#_Toc7281975)

[6.2展望 39](#_Toc7281976)

[参考文献 41](#_Toc7281977)

[致 谢 42](#_Toc7281978)

# 1绪论

1.1项目背景与重要问题

1.1.1项目背景

在我国，互联网的持续发展，截至2018年6月30日，我国网民规模达8.02亿，普及率为57.7%[[1]](#footnote-1)。面对这种形势，在未来的某一天，互联网将真正意义上普及全国。

在当下——信息数字化的时代中，互联网和人们的生活密不可分，紧密相连着。互联网逐步改变着人们获取信息的途径和方式，人们追求便捷效率的方式获取最有价值的信息。与此同时，即时通讯也更加大众化和功能化，运用到生活的各个方面。对我们每一个人来说求职是每一个人必走的人生阶段，对于企业来说，在人才战略上，招聘是非常重要的一环，高效筛选出符合自己公司需求的人才显得尤为的关键。纵观这些因素，一个招聘类的即时沟通APP就有存在使用刚需，将招聘与求职分类整合在一起，这样做的最大好处不仅是信息的筛选获取，更加便捷面试双方即时沟通交流，将所有招聘的信息归置在一个模块，简化寻找的过程与沟通的等待时间。

纵观目前市场已有的招聘类APP大部分都是原生APP，虽然具有更好的访问和兼容能力，但是每一种移动操作系统都需要独立的开发项目，不同的手机系统平台需要使用各自不同的独立语言，开发速度慢、成本高、并且不利于维护。而webAPP是运行在移动设备的浏览器上，所以只需要开发一个项目，可以通过JavaScript来进行开发。利用飞速发展的前端技术，来实现一个webapp版本的招聘软件。

即时通讯招聘webAPP使得收集数据变得简单方便，沟通更加有效，目的明确，使用简单便捷高效。体现现代互联网时代所带来的便捷高效。

1.1.2重要问题

当今，随着前端领域技术的快速发展，许多开发技术方案可以用于开发即时聊天招聘webapp。在技术选型与开发设计的时候，需要站在使用者的角度去思考如何提高用户体验并且优化用户使用感，再去选择正确合适的技术来解决业务问题。使得系统的效率与用户体验完美的契合，并且去有效的规避掉webAPP的缺陷，比如webAPP打开一个页面需要重新加载页面的所元素，访问速度容易受手机终端的性能限制从而导致加载速度慢等问题。

在开发webAPP的时候如果再去选择传统的前端技术，将无法满足我们对产品性能和体验的追求，同时也不再符合当今前端的开发思想了，无法做到前后端分离开发，不利于项目代码的复用与维护。所以，我们要选择合适的前后端框架来进行开发，解决传统模式的弊端，实现现代前端开发的模块化、组件化开发。

除此之外，信息传递的即时性也是我们需要考虑的重要问题之一，该系统的主要目的就是实现快速有效率的沟通，也就是快速响应，并且注重的性能体验，与用户体验，需要的是专业的UI设计。也就意味着它的实现要更加的灵活，迭代程度也会相对的较高。

1.2项目的意义和研究的重要性

1.2.1项目的意义

在过去的前端技术发展中，一直处于一个不太明了的状态，只是简单的页面结构和样式的书写，在近十年中前端技术日新月异开始发展迅猛，前端越来越多的强大，能做的事情也越来越多。相比起三四年前，前端不断刷新的定义，比如大前端/全栈并且在互联网移动化的大趋势下，在移动端，受成本和效率的趋势，跨平台的一站式开发会慢慢的变成趋势。本文以“即时通讯招聘webAPP”为主题，研究实现单页应用、前后端分离、项目工程化开发与前端主流框架的服务端SSR，重点放在研究并且实践移动端高效率的前端可开发生态体系，感受技术革新带给我们的事半功倍。

1.2.2研究的重要性

据有关统计，随着毕业人数的逐年上升，求职人数也跟着逐年上涨。在这个效率为王的时代，人们越来越追求高效率。高效率求职与招聘显得尤为重要，大大的简化了求职与招聘的繁琐步骤，简化沟通成本，大大的提升效率。一个即时沟通招聘webAPP是现在日常不可或缺的实用软件之一，所以非常有必要去研究其产品结构和实现原理的。

1.3研究的内容和主要工作

1.3.1研究的内容

对使用React全家桶技术与NodeJS技术开发的即时通讯招聘平台的设计与实现是本篇论文的主要研究内容。本文在对市面已有的类似系统进行充分的需求分析，提炼出该项目的需求功能。

本文分为五个步骤对系统进行分析设计、实现和完善。第一步，调研了市面上已有相关产品，充分了解这些产品的优缺点。第二步，通过对市面现有的前后端技术进行对比，做出最适合的技术选型。第三步，通过分析列出架构设计，同时进行对应的数据库构建。第四步，对功能的实现进行详细讲解并进一步对项目做出优化，第五步是对该次项目进行对应的总结反思。

在前端技术与标准化开发做深入的研究，去解决数据状态存储、页面结构管理、双向通讯以及模块化开发等问题。

1.3.2本人主要工作

在这次的设计开发的过程中，我的主要工作是先主动分析对现有相关软件的实现和功能却确定本系统的需求与功能。同时对其进行详细的分析设计。确定了系统的具体业务场景，然后再在开发之前进行合适的技术框架选型和数据库设计。随后进行系统的具体开发与实现，开发基本完成时进行系统功能复查和代码规范与项目代码优化。最终撰写论文并思考总结项目里的不足之处与收获。

1.4论文章节

该论文由六章节的内容组成，其中：

第一章：项目调研，讲解了即时通讯招聘webAPP提出的背景和选题意义，并且调研与分析了当前市场相关软件的情况，同时进行优缺点对比，由此整理确定了项目的整体工作目标与方向。

第二章：技术选型，主要介绍了本文所采用的主要技术，包括：前后端成熟高效的框架和方案的选择、系统的代码优化处理方案、代码规范化方案等。并进行详细对比，讲述技术选择的理由和原因。

第三章：需求分析，主要包含了本次毕业涉及的各个模块的具体需求分析与探究，并且画出对应的所需要的图示，并且进一步描述系统的需求。

第四章：系统设计，主要介绍系统的整体设计，并且详细描述拆分重点模块的逻辑实现。通过画出了对应的类图与E-R图来阐述对应关系，实现数据库设计且给出具体的数据集合数据详情。

第五章：系统实现，主要是对系统的具体实现进行详细说明。并且给出一部分重要模块的实现图，同时对系统优化与细节实现部分贴出代码进行说明。

第六章：回顾总结，提出系统未来的研究优化方向同时也反思总结论文研究过程中的得与失。

# 2系统的开发方法及相关技术

即时聊天招聘webAPP系统采用的是NodeJS + mongodb + express + React + Redux + webpack架构，前后端分离。前端方面：使用React实现组件开发，Redux进行管理数据/状态管理，React Router进行路由的配置。后端方面：使用Express 开发web后端，Socket.io管理实时应用，MongoDB/Mongoose实操实现NoSQL数据库。

2.1后端框架选择

2.1.1 NodeJS

NodeJs是一种通过JavaScript语言来开发web服务器的东西，也就是说是一个平台，帮助我们可以快速构建网络服务与应用。

选择NodeJS的三个原因：其一，NodeJs是开源的，有着非常丰富的社区资源，因此在技术上的成本相对较低且不存在使用的付费问题，而且。其二，JavaScript的高并发性能超出了其他语言[[2]](#footnote-2)。NodeJS中的异步机制是基于事件的，并且所有的I/O操作以非阻塞的方式进行，操作的结果由事件循环处理[[3]](#footnote-3)。再加上V8引擎也是同类的佼佼者，所以Node.js的性能也受益其中[[4]](#footnote-4)。其三，NodeJS其强大的功能之一使得我们可以让JavaScript代码运行于服务器端，使得前后端使用同一种语言，同时提供了我们项目所需要的高性能，同时又能降低开发复杂度的架构。虽然还有其他的一些支持JavaScript在服务端运行的平台，但是也正是因为上面三点特征，让我选择NodeJS作为这次的服务端运行平台。

2.1.2 Express 框架

Express框架是NodeJS最流行的轻量级web开发框架，帮助我们快速开发web以及移动端应用。Express将代码背后的具体繁杂操作封装起来，允许开发人员专注于写业务代码[[5]](#footnote-5)。拥有优秀开源生态环境的Express框架具有开箱即用的特点。Express完全是由路由与路由中间件构成的框架，本质上来说就是一个express应用就是为了调用各种middleware。

2.2前端框架选择

2.2.1前端发展大背景

到目前为止，除了浏览器的应用的复杂程度不断提升，前端的开发模式也在不断变化。从最开始的静态黄页时期到服务器组装动态网页数据时代再到后端语言为核心的MVC开发模式，到现在实现跨端、跨界面的前后端分离开发的时代。

前后端分离开发解决方案的开始周期始于纯前端MV\*模式，这是一种设计开发模式，其中数据模型与控制模式和视图分开，一直发展到现在的前端虚拟DOM（用来描述页面DOM树节点之间关系的特殊JavaScript对象）的形式与前后端同构开发的时代。

在前端技术开发模式的演变的同时，互联网的信息呈现方式也在不断的演变，越来越偏向于终端设备屏幕，而且对应的交互也越来越多种多样，也代表着越来越复杂。现代前端web应用内容变得更加冗杂，而且通常也是以多平台的形式展示出来，面对现实需求的多种多样的要求，前端也开始借助特定场景的前端框架来提高开发效率，同时在代码的管理和维护上，也开始逐渐的模块化和组件化，使用拆分管理的概念将复杂化的结构拆分为几个独立的简单结构从而达到解耦的目的。

Web前端技术在不断迅速发展的同时，逐渐能做的事情不再局限在简单的页面层面，也慢慢深入到中间层，逻辑层。但是前端的一路发展到现在不变的宗旨就是，一直持续在以效率和质量为导向发展。

2.2.2 React与其他前端框架对比

本次前端框架选择使用React框架。React是一个用来构建用户界面的JavaScript UI库。React不限定服务器层面的交互和具体代码的组织方式，视图部分才是React的主要关注点[[6]](#footnote-6)。它与之前已有的Angular 和[jQuery](http://caibaojian.com/jquery/" \o "jQuery手册),Backbone.js等框架不同，它的出现改变了JavaScript原有的世界，大力推广了虚拟DOM，并且带来了新的语法——JSX，允许开发者们在JS里面直接书写HTML。

对比Vue，React与vue有着许多的相似之处，就比如都使用了Virtual DOM，都推荐组件化开发等。开发者只需要将路由和数据的全局状态管理交给其他相关的第三方库，将开发专注于核心库[[7]](#footnote-7)。不同点在于两个框架解决问题的关注点不同，Vue关注点在于component的数据视图同步，依赖指令和dom结构，在灵活性上会稍逊于React。React强调的是使用组件的方式聚焦于视图层，借助JSX来写高聚合UI组件，单向数据流模式使得UI组件状态的维护管理更加清晰。

就本次产品来说，它所需要的即时沟通，也就是快速响应，并且注重的是性能体验与用户体验。诸如求职者信息推送和下拉刷新是主要的数据更新来源，除了个人信息推送外，大部分界面都是增量显示而不是更新显示，所以单从数据的复杂和更新度来说，场景并没有很复杂的程度。也就是意味着要更加灵活，迭代程度也会更高。选择借助React灵活先进的技术理念和成熟的生态可以满足我们对该产品的需求。

2.2.3前端开发打包工具

由于React本身并不能实现模块化开发，所以借助webpack前端模块化开发构建工具[[8]](#footnote-8)。该项目选择webpack作为项目的构建和打包工具从而来实现前端工程化。Webpack本质上是module bundler，其提供的强大loader机制让打包机制更加灵活多变[[9]](#footnote-9)。将代码以不同模块的方式进行打包压缩等操作，在优化项目打包的同时实现热启动与热更新等功能。Webpack的强大之处不仅仅是进行打包可以轻松的管理复杂的JavaScript代码和各种繁杂的依赖包关系，更重要的是它的万能模块加载理念，即是所有的资源都可以且模块化，资源处理集成化，使得代码更加可读，项目结构更加清晰，提高开发的效率。

2.3数据库选择

2.3.1 MongoDB

随着互联网技术的飞速发展，数据量呈现爆炸性增长趋势，对互联网应用的数据库提出来高并发读写、高效率存储以及高可拓展性和高可复用性的需求，MongoDB以其独特的优势为web应用提供了可扩展的高性能数据存储解决方案[[10]](#footnote-10)。MongoDB是一个由C++语言编写的非关系型数据库的开源产品。它的存储方式为虚拟内存+持久化存储，采用二进制的高效数据存储[[11]](#footnote-11)。所以特点包括不限于它可以存储丰富的数据格式、操作比关系型数据库快很多、免费开源所以对开发者友好，成本低、部署简单便捷、易于扩展。针对该系统的不同的业务需求，选择MongoDB能够避免浪费很多不必要的资源，同时，Mongo非常适合实时的查询，插入与更新，并且具备实时数据存储所需要的复制与高度的伸缩性。

# 3需求分析

3.1可行性分析

在一个项目开发的最初期应该先做其可行性分析，确定具体的需求、技术实现的难度与方案的可行性，避免出现不必要和无法实现的需求，其目标是合理的设计开发需求，合理利用资源，实现预定的目标。主要讨论三个方面的可行性，确定该系统的开发是完全可以实现的。

3.1.1技术可行性

在该系统开发中，后端使用NodeJS框架，Express作为框架的骨架，代码编写上采用ES6与部分ES7语法标准并使用函数式编程。前端方面采用React全家桶技术，数据库选择使用MongoDB/Mongoose实操实现NoSQL数据库。整体技术成熟，框架环境稳定，在技术层面方案实现可行。

3.1.2经济可行性

系统开发实现所采用的技术均处于开源环境下，不存在版权使用纠纷。技术发展成熟，有稳定强大的社区环境，在市场上使用广泛，其开发与维护成本并不高。并且所使用的开发软件在网上可以免费使用，所以该系统在经济层面也是可行的。

3.1.3操作可行性

在具体操作上来看，系统的使用对象为求职者与招聘者，因此简单直接明了的操作与精简舒适的UI是至关重要的。在界面上精化，在技术上优化交互操作体验，真正切实贴切用户的真实需求，所以该系统在操作上是完全可行的。

3.2功能需求

软件业务需求分析，旨在完成“系统做什么”的问题[[12]](#footnote-12)，是整个系统正确完整开发所需要的。只有明白了正确的受用人群去明确把握业务需求才能设计出符合规范和标准的系统[[13]](#footnote-13)。

3.2.1总体功能需求分析

该系统主要包括注册登录、完善信息、牛人列表、BOSS列表、消息列表、聊天详情六大功能。求职者注册登录之后完善个人信息之后便被系统推送到牛人列表，招聘者注册登录之后便可以在牛人列表找到并与其进行实时沟通，同时招聘者完善公司信息，求职者同样能够自主选择心仪的公司BOSS进行沟通面试。满足了不同使用者的各种需求。整体业务需求关系分析如图3-1所示。

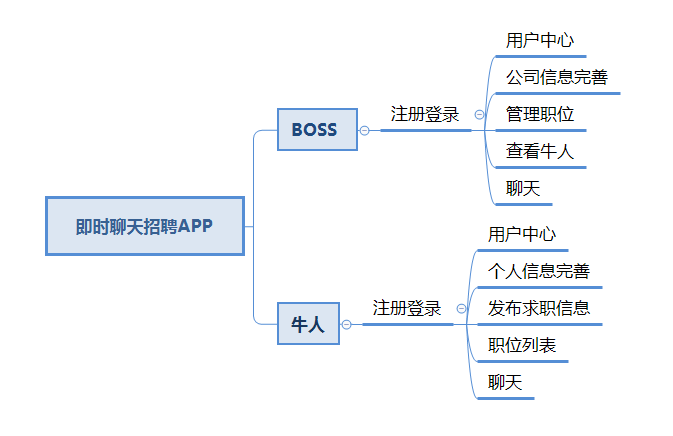


图3-1 整体业务需求分析

3.2.2登录注册功能需求分析

BOSS与牛人两种不同身份都拥有注册功能，对应不同的身份与是否注册登录跳转不同的页面。在注册页面选择用户对应身份，输入自定义用户名与密码，再次确认密码通过完成注册功能。在用户信息确认注册之后，进行登录操作，然后根据用户身份进行对应页面跳转。如图3-2所示：

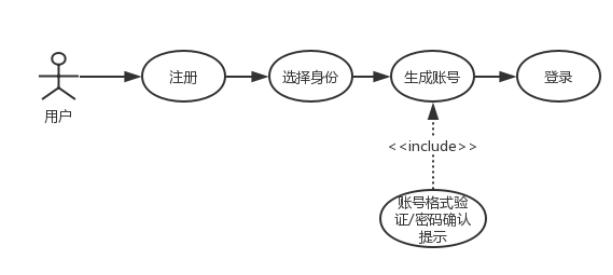


图3-2 注册登录用例图

3.2.3完善信息功能需求分析

当用户成功登录后跳转对应身份的完善信息页面，需要先判断用户事先是否已经完成信息完善，如果未完善，提示其进行完善。BOSS信息包括个人头像、招聘职位、公司名称、岗位薪资、职位要求等表单。牛人信息包括个认头像、求职岗位、个人简介。

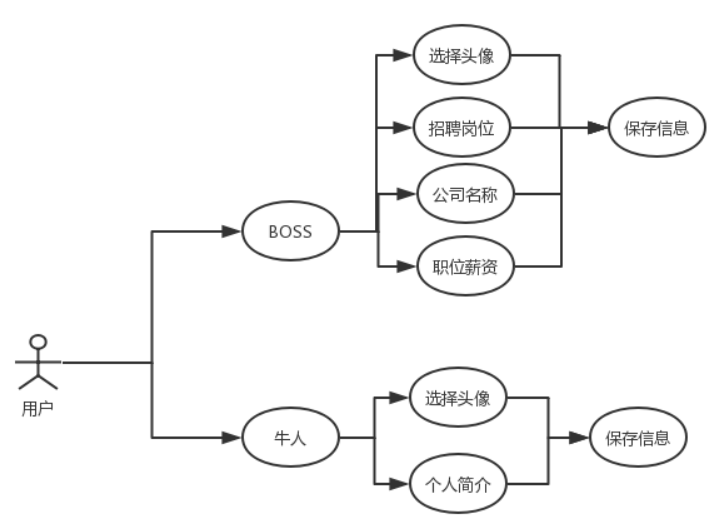


图3-3 完善信息用例图

3.2.4信息列表功能需求分析

用户登录完成并且信息完善的情况下，系统根据不同身份跳转不同的信息列表，BOSS身份跳转到牛人列表，可实时观看牛人发布的个人简介信息，牛人身份跳转到BOSS列表，同样也可以实时看到BOSS发布的公司招聘信息。同时点击其他用户发布的信息，可以发起实时聊天。

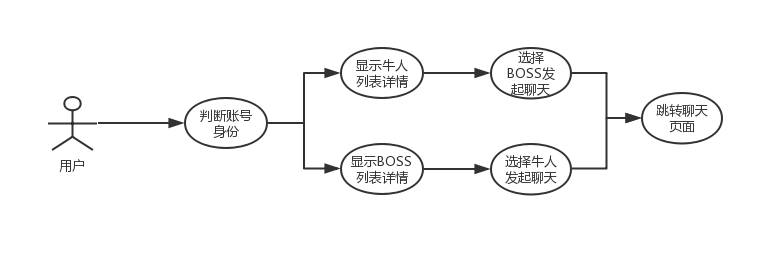


图3-4 信息列表用例图

3.2.5个人中心功能需求分析

个人中心主要展示用户的个人信息，比如头像，用户名称，个人信息详情（根据不同身份显示不同具体信息）。包括退出登录功能。

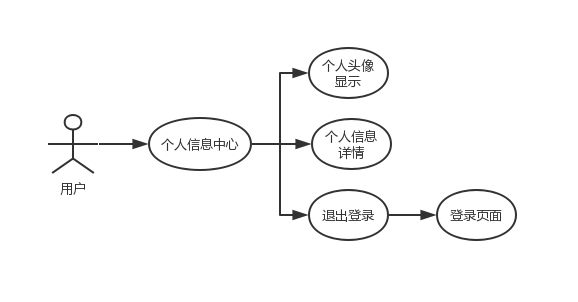


图3-5 个人中心用例图

3.2.6聊天功能需求分析

聊天功能在客户端包括聊天详情与聊天列表两大模块，聊天详情主要为牛人与BOSS实时显示聊天对话，识别显示双方身份头像，聊天未读消息数实时显示，并且支持发送emoji表情。聊天列表主要由聊天信息根据用户分组，列表展示，显示未读消息数量，以及最新消息排序几个功能需求。在服务端主要包括了建立连接、断开连接、监听客户请求三个方面。主要功能模块如图3-6所示：

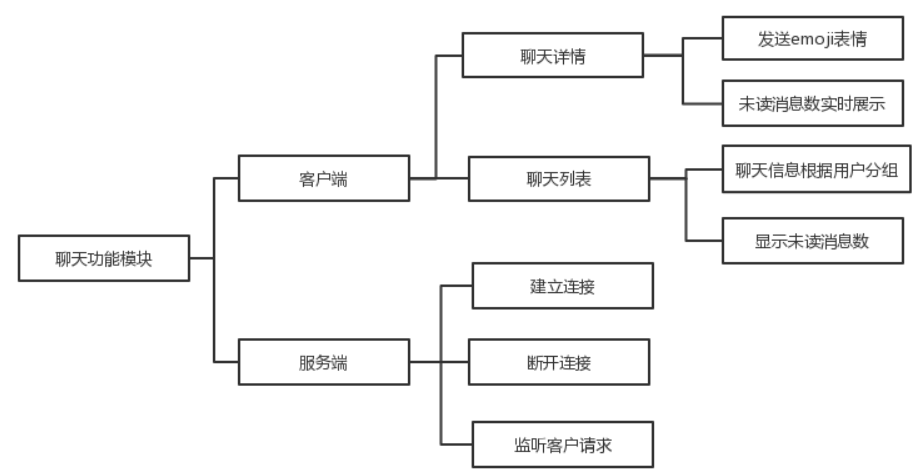


图3-6 聊天功能的模块图

3.3系统用户需求

3.3.1角色分析

该系统涉及的角色如表3.1所示：

表3.1 系统角色信息表

|  |  |
| --- | --- |
| 角色 | 职责或功能 |
| 牛人（求职者） | 注册、登录、发布求职信息、个人信息完善、查看boss列表、聊天 |
| Boss（招聘者） | 注册、登录、管理职位、查看牛人、招聘信息完善、聊天 |

3.4数据需求

该系统中所需要实体类以及各子类之间的关系如图3-7所示，图中

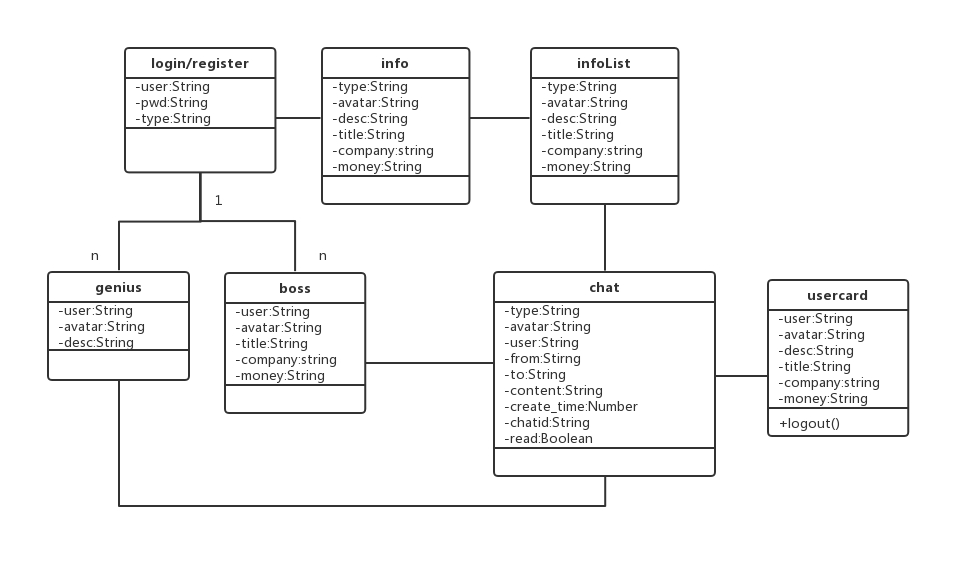


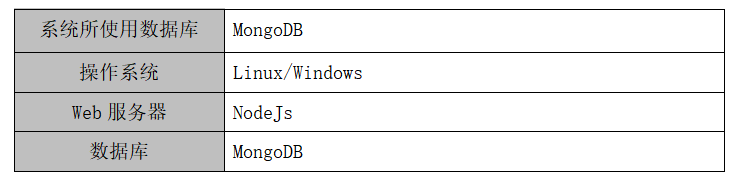
图3-7 系统概念类图

3.5非功能需求

3.5.1环境需求

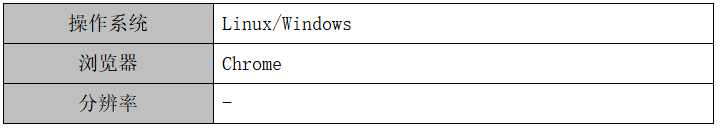
服务端需求如表3.2所示：

表 3.2 服务端环境



客户端环境需求如表3.3所示：

表 3.3 客户端环境



3.5.2性能需求

1. 可维护性

该项目采用组件化开发搭配使用框架，来管理项目的开发代码。同时在代码优化层面做了许多工作，比如提取了公共业务组件、将频繁调用方法提取封装等。使得项目结构清晰明了利于项目的管理与后期的维护。

1. 可扩展性

开发使用框架，使得项目结构清晰，方便定制和扩展特性，在现有功能上拓展，只需要考虑与原来的技术框架相兼容和解耦，相对而言迭代代价较小。可以进扩展功能，具有较强的兼容与扩展性。

1. 性能优化

系统的性能优化主要集中在客户端，因为会有静态文件操作与大量的数据库IO操作，我们选用单线程的node作为服务器端。并且在用户体验上需要优化和改善，确保程序在任何设备上运行流畅，所以前端方面选用了React作为前端技术栈，利用它批量更新插入的特点，避免强制同步布局影响响应和动画。再利用webpack的特点打包不同资源，利用浏览器的缓存去减少页面加载时间，多方面去改善增强用户体验。

3.5.3安全需求

系统的登录注册执行严格的加密原则，采用MD5单向非对称加密。MD5单向加密，只能对数据进行加密，也就是说，没办法对加密后的数据进行解密[[14]](#footnote-14)。在注册登录的时候加密生成对应密文，下次登录时获取密文进行对应校验，防止暴露密码安全。同时对请求作出权限判断，进行安全的跳转，保证用户信息安全和系统数据安全。

# 4系统设计

4.1总体设计

该系统采用的是B/S结构模式，前端只需要将重心放在交互体验与视图层面，将系统底层的功能实现给到服务器端去处理，合理设计和分配客户端和服务端任务。让系统功能的实现核心部分关注在服务器端，简化开发维护的成本。其工作原理如图4-1所示：



图4-1 B/S架构工作原理

同时该项目采用前后端分离开发，让前端专注于业务表示层面，实现页面的业务功能，负担着用户与应用之间的对话功能，不包括业务本身的处理逻辑，使得后期的迭代扩展更加方便，后端专注于数据响应与处理，也就是功能层，在用户成功注册登录后讲信息一次性传递给表示层，避免在表示层和功能层频繁的进行多次数据交换，达到优化交互体验与系统性能的目的。整个系统的技术架构选型设计如图4-2所示：



图4-2系统技术架构设计

4.2重要功能模块设计

4.2.1登录注册模块设计

用户进入APP首页面，点击注册按钮进入到注册页面，输入用户账户与密码用MD5加密的方式传输给数据库，同时前端校验输入是否符合规范，且向后端验证用户名是否存在，成功之后完成注册，进行登录操作，带着cookie向后端去获取用户信息，从而加载页面，同时判断用户是否登录，登录成功进行页面跳转，未登录状态返回登录页面提示登录，同时登录接口中用户信息是存入http header中进行传输，保证用户信息的安全性。

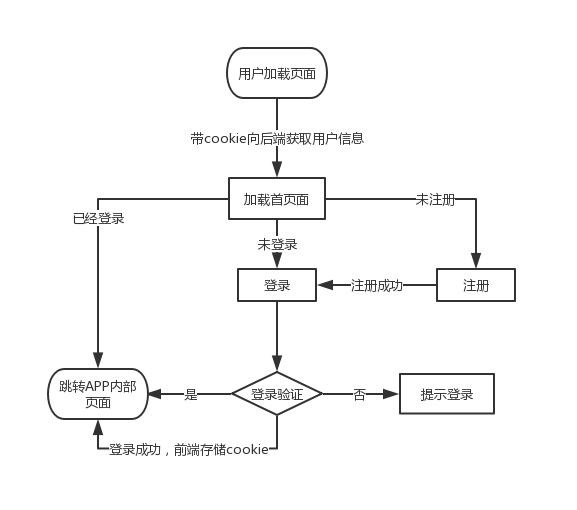


图4-3用户登录模块流程图

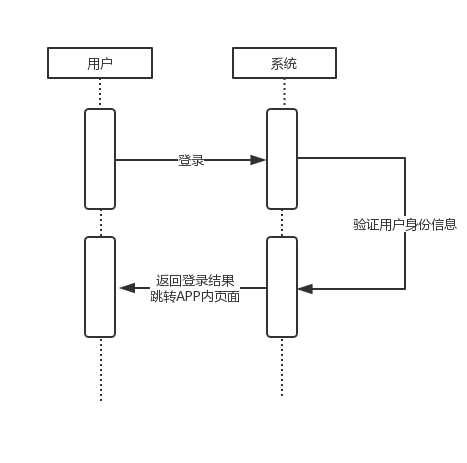


图4-4用户登录模块时序图

基本路径：

1. 用户登录系统进行对应消息验证
2. 显示验证结果反馈给用户进行页面跳转

4.2.2信息模块与列表设计

完善信息模块主要为登录成功之后跳转的页面，通过cookie中的用户信息判断用户身份，是牛人还是boss，若之前就已经完善成功则直接跳转APP内部页面，否则跳转不同的完善信息组件，在完善信息成功之后，再次根据身份的不同去显示不同的列表详情。主要过程如图4-5、4-6所示：

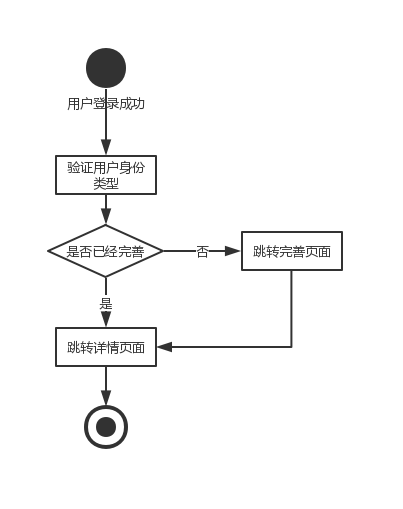


图4-5用户信息完善与列表显示模块活动图

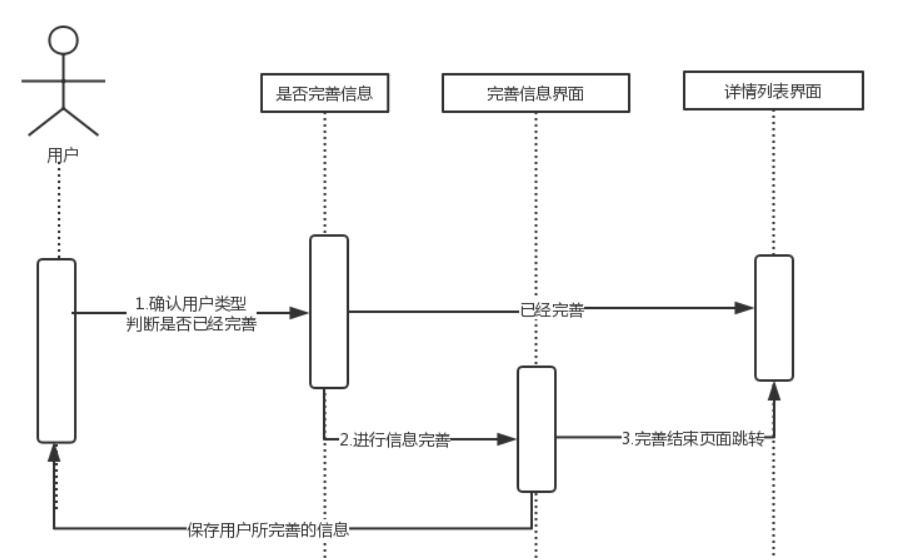


图4-6用户信息完善与列表显示模块时序图

基本路径：

1. 系统确定用户类型并且判断用户信息是否已经完善
2. 未完善的情况跳转完善界面并且保存对应信息
3. 跳转详情列表界面

4.2.3个人中心模块设计

个人中心模块主要由用户信息的显示和退出登录模块组成，主要采用列表的形式来显示具体信息，点击退出登录按钮触发清除用户信息，注销cookie，这个时候会自动刷新页面，要优化体验感，不执行自动更新，则我们在注销时，手动清空redux的数据。

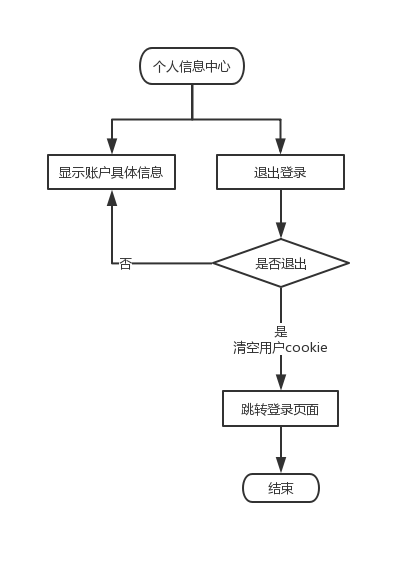


图4-7用户个人信息中心流程图

4.2.4聊天模块设计

聊天模块为本系统最复杂和主要的模块，主要为用户聊天信息的显示和具体聊天页面模块。聊天模块和其他模块不同的地方是，需要在有新消息更新时就实时显示，传统处理聊天相关信息是采用ajax轮询来实现的，但是这样每次请求都要发送对应的HTTP header，传输的数据量比较大，所以选用WebSocket通讯协议并且引入socket.io这个基于事件的实时双向通讯库，来实现前后端实时显示消息，从而实现实时聊天。先对用户进行监听，确定用户已经成功连接，通过用户操作触发相对应触发事件，前后端通过事件进行双向通讯，同时后端可以主动发起数据推送。

用户访问聊天模块界面时还是需要发送HTTP请求来获取对应的HTML与CSS还有JS的，此时如图4-8所示：

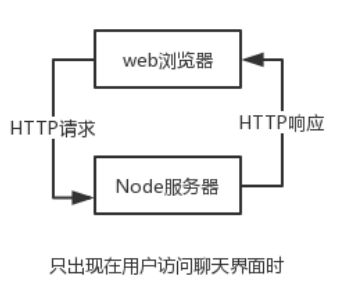


图4-8用户访问聊天界面的请求图

在用户聊天时，Node只需要用一个端口就可以轻松的管理访问页面的HTTP请求与聊天时的WebSocket服务。同时搭配使用express框架，使得对应的web服务更加清晰简单。此时如图4-9所示：

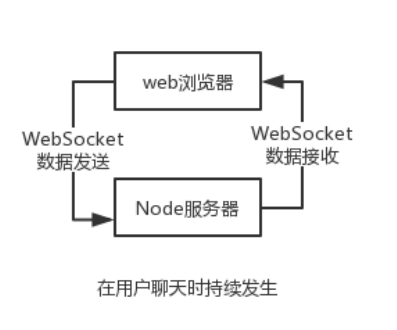


图4-9用户聊天时的请求图

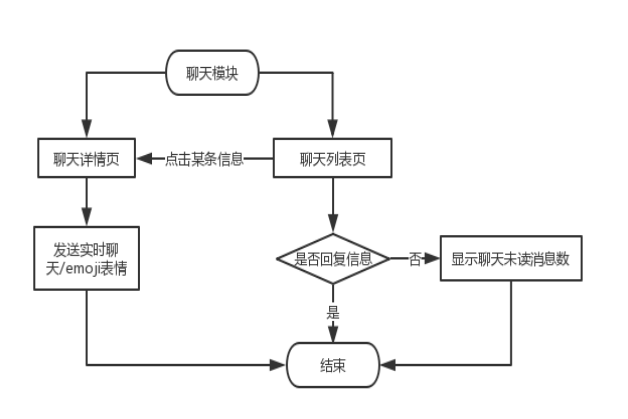


图4-10聊天模块流程图

4.3数据库设计

4.3.1主要实体属性图

1. 求职者/招聘者实体属性如图4-11、4-12所示：

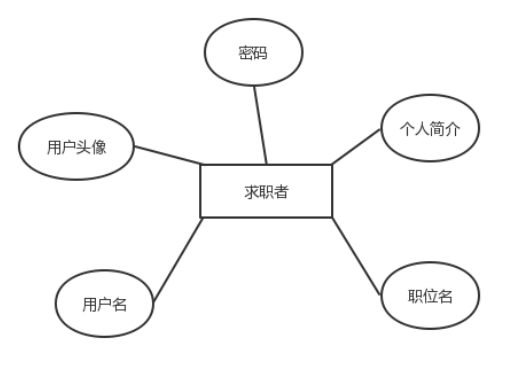


图4-11 求职者实体属性图

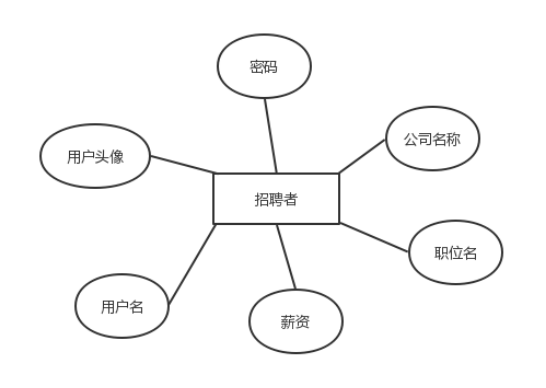


图4-12 招聘者实体属性图

1. 聊天信息实体属性如图4-13所示：

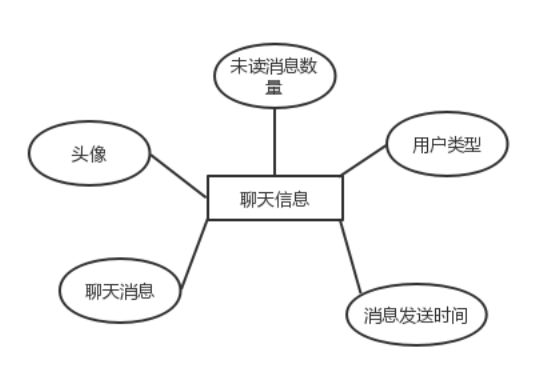


图4-13 聊天实体属性图

1. 牛人列表/BOSS列表实体属性如图4-14所示：

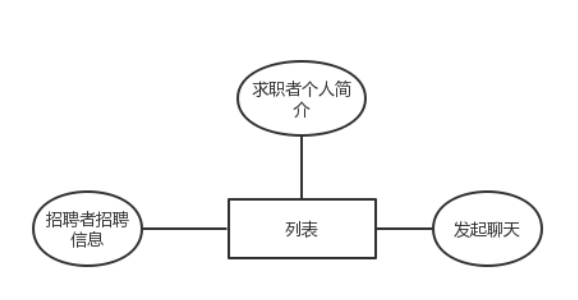


图4-14列表实体属性图

1. 个人信息中心实体属性如图4-15所示：

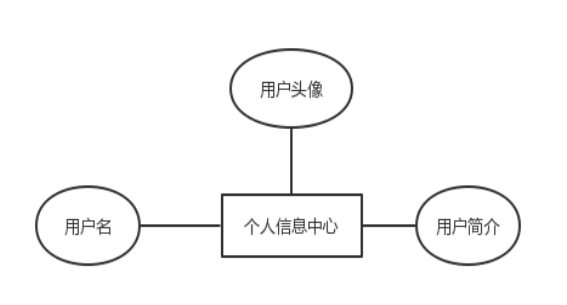


图4-15个人中心实体属性图

4.3.2主要实体关系图

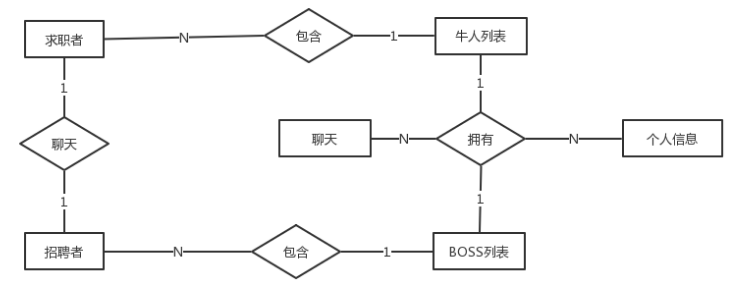


图4-16 系统E-R图

4.3.3主要数据库表设计

由于MongoDB没有其他数据库里的表结构，是非关系型数据库，所以都是以文档集合的形式来存储数据的，也就是Collections——在MongoDB里面叫做集合，也就是文档的集合。由上文的E-R图可以知道该系统至少需要用户（User）、聊天（Chat）这两张collection。具体数据如表4.1所示。

表4.1数据库Collection及说明

|  |  |  |
| --- | --- | --- |
| 编号 | 表名 | 说明 |
| 1 | User | 用户 |
| 2 | Chat | 聊天 |

下面为各个Collection的具体实现：

1. 用户Collection（用户表）

表4.2 用户

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 是否为空 | 备注 | 描述 |
| type | String(20) | Not null | BOSS或者牛人 | 用户类别 |
| user | String(20) | Not null | 用户名 | 用户名 |
| pwd | String(20) | Not null | 密码 | 密码 |
| avatar | String(20) | Null | 用户头像 | 用户头像 |
| desc | String(20) | Null | 详细资料 | 个人简介或者职位简介 |
| title | String(20) | Null | type为任意都有该字段 | 职位名 |
| company | String(20) | Null | 仅BOSS身份有 | 公司名称 |
| money | String(20) | Null | 仅BOSS身份有 | 薪资 |

1. 聊天Collection（聊天表）

表4.3 聊天

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 是否为空 | 备注 | 描述 |
| from | String(20) | Not null | 发送出消息对象 | 发送用户名 |
| to | String(20) | Not null | 接受该信息对象 | 接受用户名 |
| content | String(20) | Not null | 发送的具体信息 | 具体信息 |
| create\_time | Number(20) | Not Null | 默认当前建立时间戳 | 消息发送时间 |
| chatid | String(20) | Not Null | 默认为两个用户的from和to排序后的拼接 | 每一次聊天唯一的标识 |
| read | Boolean | Null | 默认为false，这个字段只对信息接收方有效 | 标识信息是否被读 |

以上两个Collection是本次论文实践设计的系统所需要的信息存储集合。

# 5系统实现

5.1主要模块实现

5.1.1登录注册

APP首次进入时，进入APP首页面去进行常规操作，用户输入注册登录信息。APP的首页面，有登录注册两个功能，如图5-1所示：



图5-1 APP进入首页面

进入首页面之后先进行账号注册，进入到注册界面如图5-2所示：



图5-2 用户注册页面

在注册完成后登录完成将直接进入到信息完善页面，如图5-3所示：

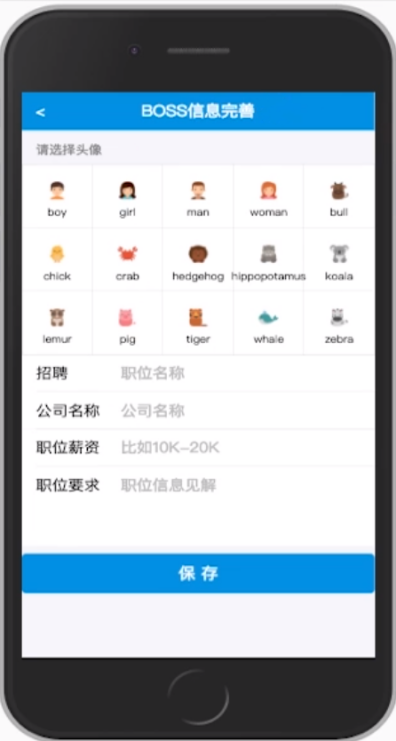


图5-3 用户信息完善页面（以BOSS身份为例）

若在已有账号的情况下，可以选择直接登录，通过账号以及密码的匹配，在数据库中搜索，进行匹配，若查询到对应账号，则登录成功，系统判断个人信息是否完善，未完善跳转完善页面，若已经完善的情况下则显示列表页面，如图5-4所示：



图5-4 列表展示页面（以BOSS身份为例）

登录注册页面的样式均采用antd-mobile组件进行快速开发，将主要核心放在具体功能实现上。登录注册账号安全的实现依赖MD5加密算法，本系统采用utility这个第三方工具库来实现MD5加密算法，同时为防止暴力破解，在密码保存时，手动进行“加盐”，手动添加长串字符串后再次进行MD5加密，这样就算比较完善的加密算法，保证用户信息安全。

5.1.2个人中心

个人中心主要展示用户个人信息，之前所有的用户数据在刚加入APP应用时已经获取得到，并且将数据放入redux中进行管理，用户中心模块，只需要去redux中获取的需要的字段数据，配合antd-mobile的result组件进行展示。本系统的登录状态依赖cookie来进行管理的，当点击退出登录时，去清空cookie里的userid的cookie，同时回到登录页面。具体显示页面如图5-5所示：



图5-5 个人中心页面（以BOSS身份为例）

5.1.3聊天模块

聊天模块分为两个主要部分，聊天列表页与聊天详情页，如图5-6,5-7所示。该模块与其他模块不同，需要实时显示更新消息，所以我们采用Socket.io这个基于事件的实时双向通信库，前后端通过事件进行双向通讯，同时配合使用express框架，快速开发实时聊天模块。

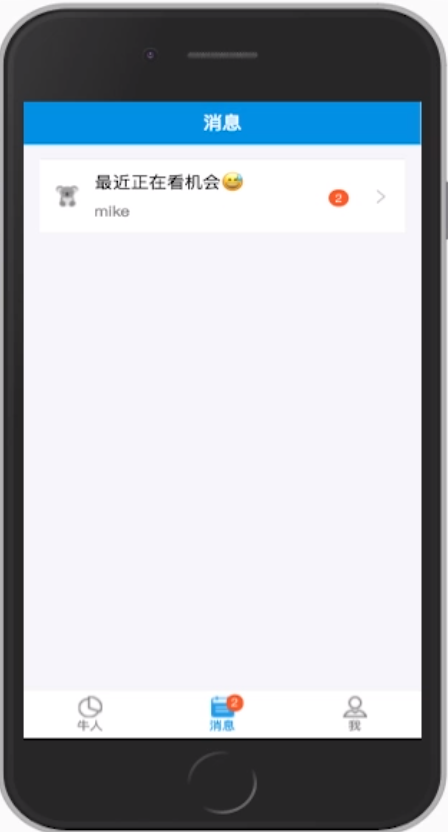


图5-6 聊天列表页面（以BOSS身份为例）



图5-7 聊天内容详情页面（以BOSS身份为例）

求职者发起聊天并且发送emoji表情，招聘者同时收到未读消息，实时的显示未读消息数量，并且在消息列表标红，显示最新提条消息内容。如图5-8、图5-9所示。

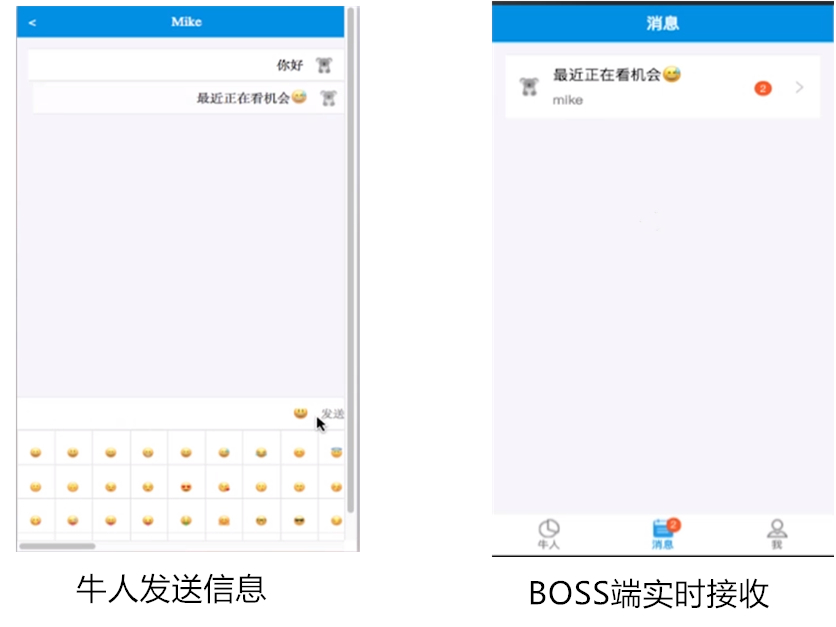


图5-8 实时聊天页面（一）



图5-9 实时聊天页面（二）

5.2后台管理实现

因为项目并没上传到云服务器上，所以后台管理的启动需要在控制台搭配安装的nodemon工具，按照步骤去开启后端服务，如图5-10所示：

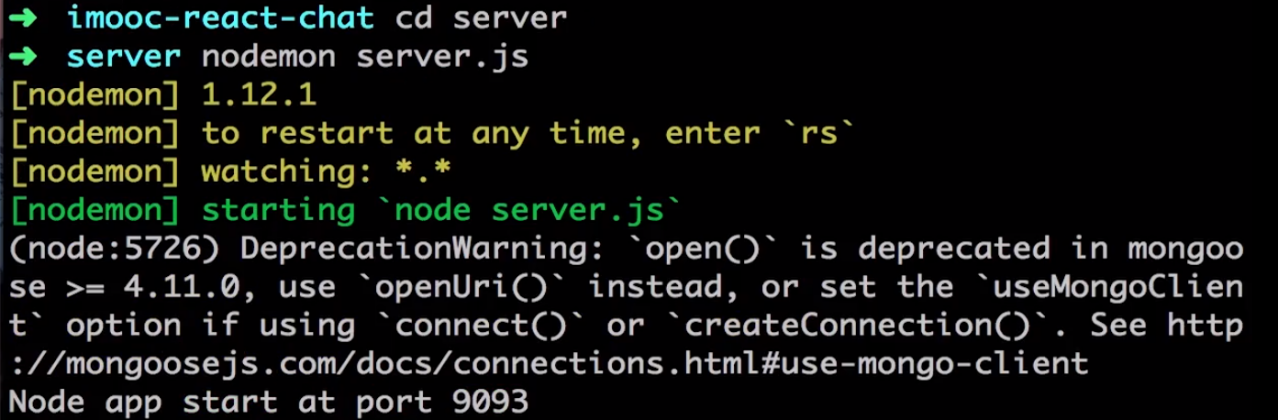


图5-10 server端开启

5.3主要关键架构与优化

5.3.1 项目结构框架与环境搭建

该项目使用webpack脚手架进行项目的构建与打包，首先电脑安装node.js进行对应环境的配置，使用node命令行安装webpack以及其他依赖项。整体项目结构如图5-11所示：

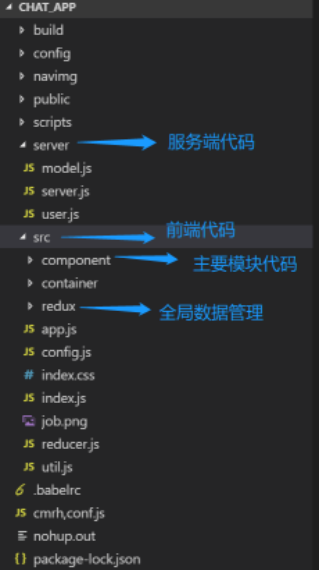


图5-11 整体项目结构

该系统是采用前后端分离搭建单页应用，所以前后端环境是分开搭建的，使用React 服务端SSR，采用后端node环境对页面渲染，同时提供静态资源和接口，前端只需要提供模板，然后页面渲染，页面跳转不需要刷新。但是这样客户端本地有属于自己的server，后端开发过程也有自己的server，两个服务器之间沟通就会产生同源策略的问题，也就是说浏览器会默认拦截不同域名下的沟通请求。

该系统为解决这个问题，采用服务器代理请求的方法，将所有node server下面的请求代理到前端开发下的server，只需要配置Http proxy即可。在项目打包发布时，webpack打包以后node端就可以获取到对应的资源，就不再有同源策略的问题。

5.3.2 服务器接口设计

该系统采用[RESTful](http://www.ruanyifeng.com/blog/2011/09/restful.html" \t "http://www.ruanyifeng.com/blog/2018/10/_blank)风格的api设计，[RESTful](http://www.ruanyifeng.com/blog/2011/09/restful.html" \t "http://www.ruanyifeng.com/blog/2018/10/_blank)是目前最流行的 API 设计规范，用于 Web 数据接口的设计。使得接口设计轻量，直接基于http，并且面向资源，一目了然。同样也使得我们在调用接口时，可以不需要考虑当前的状态，大大的降低了复杂度。

数据服务通过Restful Web Server 为其他服务实体提供统一的数据访问的接口，目的是将数据访问逻辑和业务层逻辑分类，屏蔽底层数据库的细节，提升系统的可扩展性[[15]](#footnote-15)。

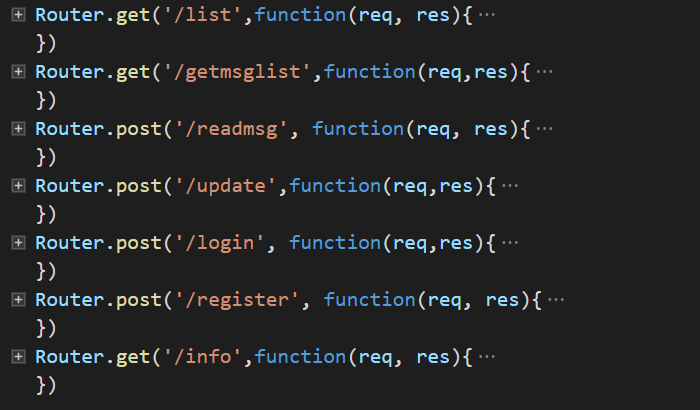


图5-12 项目路由接口设计

5.3.2代码优化实现

1. 使用ESlint+async+await优化异步代码

使用ESlint来规范项目代码，具体配置如图5-13所示：



图5-13 ESlint配置

将代码使用async+await 改造，使得代码没有异步的感觉，同步的形式写异步代码，await确保是同步完成之后才进行下面代码，使用ES7 的写法，使得代码更加优雅。如图5-14所示：

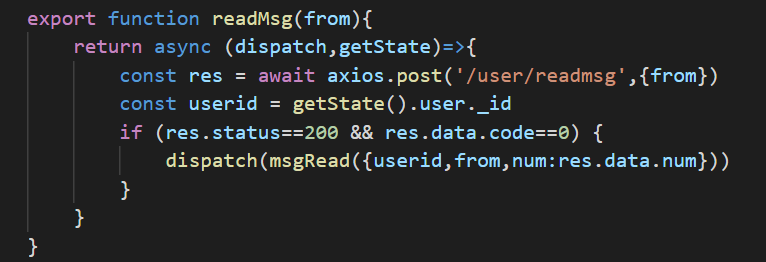


图5-14 async+await写法改造图

1. React服务端渲染SSR

首屏采用SSR技术，前后端均使用ES6/ES7语法，后端使用React本身提供ReactDOMServer来实现服务端渲染，调用里面renderToString方法，在服务端渲染首屏模板，将div React组件变成前端可以编译后的div。这样首屏JS不用执行渲染逻辑，只需要关注一些交互事件。

实际操作使用babal-node配置node里的React环境，修改客户端代码，抽离APP组件，使得部分组件前后端共享，服务端生成DOM结构，使得HTML不再是React生成而是由node生成。

如图5-15所示：

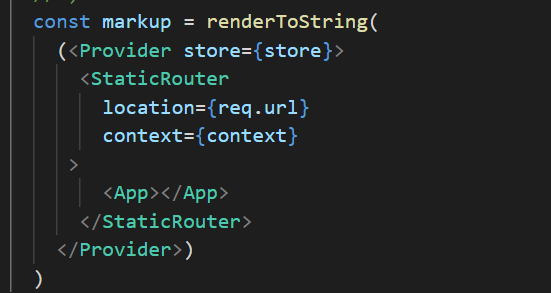


图5-15 服务端渲染写法改造

再使用服务端渲染之后，对项目进行SEO优化配置，利于搜索引擎可以爬取完全渲染的页面，如图5-16所示：

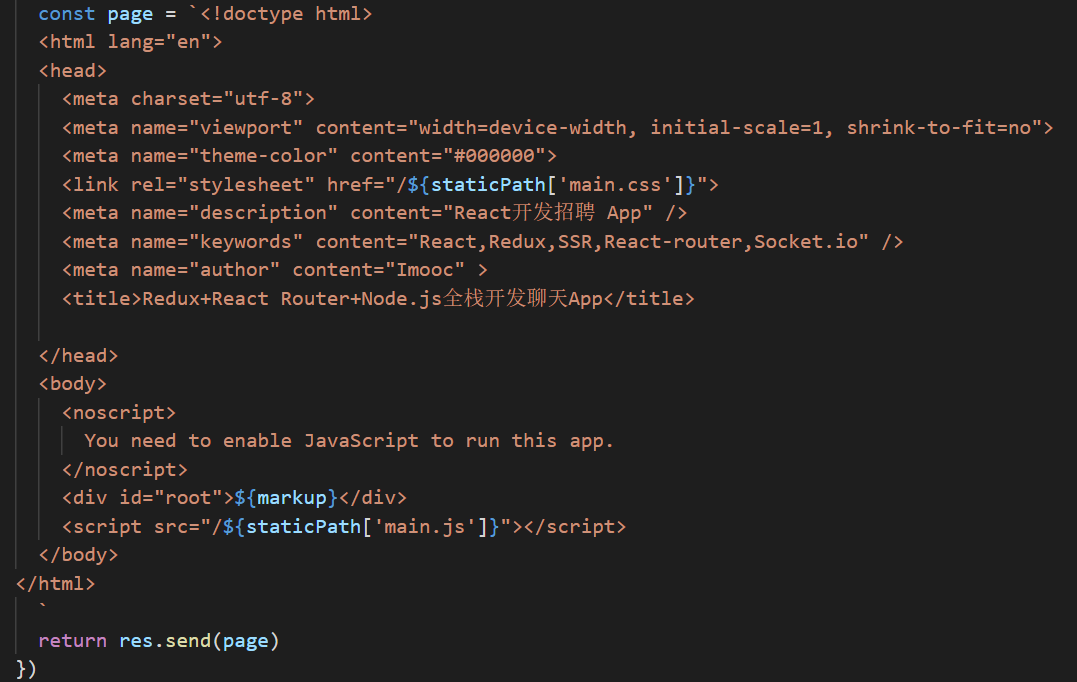


图5-16 页面SEO配置

1. React动画解决方案

使用Ant Motion做React动画，快速在React框架中使用动画，优化用户使用感，使得进入聊天页面，聊天信息出现有动态效果。

# 6总结与展望

6.1总结

本文题目为基于NodeJS的web聊天室的设计与实践。整体回顾，虽然项目不是特别复杂，但是麻雀虽小，五脏俱全。在技术上采用了React全家桶与antd-mobile实现前端技术，后端上使用Socket.io实时通讯、MongoDB存储数据、express做web接口，同时自己实现了redux,并且优化代码使用服务端渲染。在功能上，本系统完成了登录注册、完善详情、列表查看、聊天、个人信息等一系列功能。

在项目的开始阶段，先后阅读了诸多类似系统实现文献资料并且进行了项目所需要的技术储备与学习，同时也亲身体验和感受了现在市场已有的类似APP，先对此类型APP有一定的了解，随后思考构建出系统的主要模块功能，再确定技术选型到开始开发到完整的设计开发前端展示、后端服务、数据库支持，再到系统服务端渲染优化，让我对现在的全栈体系也有了更深层次的认识，也有了对React的进阶学习与node的入门学习，提高了自己的项目实践经验，对前端技术有了更高的追求。在项目后期，完成基本功能结构之后，尝试使用React服务端渲染对项目客户端代码进行重构改造，将项目之前单页应用的弊端消灭掉，更加利于SEO、优化项目结构与提高首屏渲染速度。

对前端前沿新技术与框架的实践，让自己收获颇多，学习实践怎么去完成一个项目到如何优化项目代码与结构再到如何去站在使用者的角度将一个项目的体验提升到极致。对自己未来的开发之路提出来新的思考。

6.2展望

毕业设计的过程是一个学习研究与进步的过程，通过这一次的毕业实践，我对前端开发与实践有了更加深刻的认知，也体会到自己在真正成为一个前端开发者各个方面的不足与差距。虽然项目已经基本完成与进行优化，但是我对于熟练掌握NodeJS的核心技术与高级编程还有很长的路需要去走。要将自己学习到的理论知识与实践相结合，去理解一切技术的最终目的都是为产品实现服务的，在提升知识面与技术能力同时，去自发的驱动自己用技术真正解决问题为产品输出服务，不断学习，热爱技术，在未来能够写出更有质量的代码。

这个基于web版APP系统初步完成，但是系统依旧有值得优化和拓展的地方：

（1）系统功能上比较简单，求职者的个人简介模块相对简单，缺乏多样性，可以后期完善增加视频简历作为信息维度的辅助，方便招聘者更加快速了解求职者信息，也让求职者更好的展现自我。同时可以在聊天模块增加一键发送附件简历的功能，发起预约面试，同时添加公司面试评价，给予不同类型的用户的不同需求上的满足。

（2）在系统技术层面，后期优化可以将项目部署到云服务器上，实现在线云端使用该系统，并且学习实践云服务器部署。同时可以考虑添加爬虫功能，去爬取获得现有公司信息，提供给系统大量真实的公司数据，丰富数据库的设计。

（3）可以未来在优化UI界面的同时，提升APP的实用性，更加的贴合大众的使用习惯，以提升使用者效率为基础去优化开发。

# 参考文献

[1] 中国互联网信息中心（CNNIC）《中国互联网发展状况统计报告》 第42次发布.

[2] 曾探,《JavaScript设计模式与开发实践》[M]中国工信出版社,2015

[3] Michael Abernethy.Just what is Node.js?A ready-to-code server[EB/OL],2011

[4] 朴灵,《深入浅出nodejs》[M]人民邮电出版社,2013

[5] 程桂花,沈炜,何松林，et al.Node.js中Express框架路由机制的研究[J].工业控制计算机, 2016

[6] Developers L O E . React:引领未来的用户界面开发框架[M]. 电子工业出版社, 2015.

[7] VueJs, vuejs.org[D],2018

[8] 冯菲菲,邹连英.基于Webpack及React技术的Scratch互动在线学习平台设计[J].电脑知识与技术, 2018

[9] 占东明,洪家伟,陈希杨, et al.Web新兴前端框架与模式研究[J].电子商务, 2016.

[10] 王光磊. MongoDB数据库的应用研究和方案优化[J].中国科技信息, 2011(20):93-94.

[11] Bain T,2013,《SQL server 2000 and Analysis Services》[M] China Electric Power Press.

[12] Faithe Wempen ,2010:《HTML5 Step by Step》

[13] Gerald Kotonya and Ian Sommerville.Requirements Engineering:Process and Techniques.John Wiley&Sons,2008.

[14] 刘洪涛. MD5加密算法在Web程序中的应用[J]. 科技创新导报, 2006(1):191-192.

[15] 李兴华. 基于WebSocket的移动即时通信系统[D]. 重庆大学，2013.

# 致 谢

本篇论文是在我的导师汪彩霞老师的耐心专业的指导下完成的。她在整个论文书写阶段与项目设计阶段给了我莫大的鼓励与督促，在整体思路和质量把控上给了我很多建议与帮助，在她的帮助下我如期完成了大学中的最后一项任务。我能决定并且走向前端开发，也正是受汪老师的影响，汪老师大一的课，让我领略到了前端世界的精彩与有趣。也正是受老师的带领，让我一直在前端的世界持续性的学习积累去提升自己的知识面和技术能力。这里，我向汪老师表示由衷的感谢。

同时，我要感谢本文所参考引用的学者论文，是你们启发帮助了我完成这次项目的开发和论文的撰写。大学四年让我成长了很多，收获了很多，这里也非常感谢学院及学校的其他导师和我的室友们的帮助与鼓励，很幸运遇见你们。

此篇论文若有不足之处恳请各位老师同学多多指正，感谢！

1. 中国互联网信息中心（CNNIC）《中国互联网发展状况统计报告》 第42次发布. [↑](#footnote-ref-1)
2. 曾探,《JavaScript设计模式与开发实践》中国工信出版社,2015 [↑](#footnote-ref-2)
3. Michael Abernethy.Just what is Node.js?A ready-to-code server[EB/OL],2011 [↑](#footnote-ref-3)
4. 朴灵 .《深入浅出nodejs》人民邮电出版社,2013 [↑](#footnote-ref-4)
5. 程桂花,沈炜,何松林，et al. Node.js中Express框架路由机制的研究[J].工业控制计算机, 2016 [↑](#footnote-ref-5)
6. Developers L O E . React:引领未来的用户界面开发框架[M]. 电子工业出版社, 2015. [↑](#footnote-ref-6)
7. VueJs, vuejs.org[OL],2018 [↑](#footnote-ref-7)
8. 冯菲菲,邹连英.基于Webpack及React技术的Scratch互动在线学习平台设计[J].电脑知识与技术, 2018 [↑](#footnote-ref-8)
9. 占东明,洪家伟,陈希杨, et al.Web新兴前端框架与模式研究[J]. 电子商务, 2016(10):65-66. [↑](#footnote-ref-9)
10. 王光磊. MongoDB数据库的应用研究和方案优化[J]. 中国科技信息, 2011(20):93-94. [↑](#footnote-ref-10)
11. Bain T,2013,《SQL server 2000 and Analysis Services》[M] China Electric Power Press. [↑](#footnote-ref-11)
12. Faithe Wempen ,2010:《HTML5 Step by Step》 [↑](#footnote-ref-12)
13. Gerald Kotonya and Ian Sommerville.Requirements Engineering:Process and Techniques.John Wiley&Sons,2008. [↑](#footnote-ref-13)
14. 刘洪涛. MD5加密算法在Web程序中的应用[J]. 科技创新导报, 2006(1):191-192. [↑](#footnote-ref-14)
15. 李兴华. 基于WebSocket的移动即时通信系统[D]. 重庆大学，2013. [↑](#footnote-ref-15)