

归 档 号：

武汉工商学院毕业论文(设计)

学院：人工智能学院

专业：通信工程 年级：2017 级

学生： 学号：

指导教师： 职称：

题目：基于 VUE 开发的招聘网站设计

2021 年 5 月 5 日

武汉工商学院

本科毕业论文（设计）原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品。本人完全意识到本声明的法律后果由本人承担。

作者签名：

年 月 日

目 录

摘 要.....	I
关键词.....	I
Abstract.....	II
Key words.....	II
1 绪论.....	1
1.1 研究的目的.....	1
1.2 研究的意义.....	1
2 开发语言和工具介绍.....	2
2.1 开发工具 VSCODE.....	2
2.2 开发语言 JavaScript.....	2
2.3 前端框架 VUE.....	2
2.4 后端技术 Node. js.....	2
2.5 数据管理 Mysql.....	2
3 系统总体设计.....	3
3.1 后台管理设计.....	3
3.2 用户端设计.....	3
3.3 公司端设计.....	4
3.4 后端服务器设计.....	4
3.5 数据库的设计.....	5
4 系统详细设计.....	9
4.1 前端详细设计.....	9
4.1.1 用户端设计.....	9
4.1.2 公司端设计.....	22
4.2 后端详细设计.....	30
4.2.1 服务器的搭建.....	30
4.2.2 数据库的连接.....	31
5 系统测试.....	33
5.1 用户端功能测试.....	33

5.2 公司端功能测试.....	38
5.3 后台管理功能测试.....	40
6 总结.....	43
参考文献.....	41
致谢.....	45

基于 VUE 开发的招聘网站设计

摘 要

随着社会的急剧发展，企业工作岗位需求的增加，致使很多大学生和一些社会人士有求职的机会，但是现在的求职招聘网站都比较眼花缭乱，而且广告很多，这个招聘网站就是专门为大学生以及一些社会人士做的招聘网站，简约且功能全面，不会有所谓的广告，同时也会有很多优质工作推荐，都会是相关的专业，让用户使用的时候能够一目了然、迅速定位，同时也给需要招聘的公司提供了一个招纳人才的平台，实现了求职者和招聘者的双方需求。

本系统设计采用的技术点使用的是前端近几年比较流行的 VUE 框架技术，以及对于 XMLHttpRequest 封装请求的 AJAX 技术和 AXIOS 技术，配合 WEBSOCKET 实现了一些进行相应的聊天功能，运用了支付宝开放平台的 API 实现了支付宝沙盒支付功能，后端采用的是基于 NODE.JS 和 EXPRESS 技术搭建后台，完成对于前端任务的请求回应以及相应的数据处理返回等，数据管理使用的是 MYSQL 数据库进行数据的存储，开发工具用的是 VisualStudioCode。最后系统的上线采用了阿里云服务器，以及 NGINX 代理进行上线到公网。

关键词：VUE；招聘；数据库

Design of Recruitment Website Based on Vue

Abstract

With the rapid development of society and the increase of the demand for jobs, many college students and some social people have the opportunity to apply for jobs. However, the job hunting website is dazzled and there are many advertisements. This recruitment website is a recruitment website specially for college students and some social people. It is simple and comprehensive, and there will be no so-called advertisement. At the same time, there will be many high-quality job recommendations, which will be related majors, which enables users to locate at a glance and quickly when using them. At the same time, it also provides a platform for companies that need to recruit talents, which realizes the needs of both candidates and recruiters.

The technology points used in this system are the popular VUE framework technology in the front end, the AJAX technology and AXIOS technology for XmlHttpRequest packaging request, and some corresponding chat functions are implemented with WEBSOCKET. The Alipay sandbox payment function is realized by using API of Alipay open platform. The back end uses the technology of node.js and express to build the background, complete the request response to the front-end tasks and the corresponding data processing return. The data management uses MySQL database for data storage, and the development tool uses visualstudiocode. Finally, alicloud server and nginx agent are used to go online to public network.

Key words: VUE; Recruit; Database

1 绪论

1.1 研究的目的

招聘网站的出现很大程度上减轻了企业招聘人才的压力，也方便了求职者，所以招聘网站的设计需求真实地反映了社会现状。招聘网站拥有用户端和公司端以及后台管理三端。在用户端方面，招聘网站面向的主要用户是刚刚步入社会的大学生和社会求职者，因为用户端涉及到的功能点非常的多，同时开发的技术点和难度也非常的大，所以整个设计任务的重心将会在用户端。致使在用户端中需要完成用户给公司投递简历的功能、相应职位的收藏功能、查看简历投递状态功能，同时实现聊天功能、用户编辑个人简历信息，以及基本的个人信息和修改密码功能、用户实现便签功能，同时实现付费推荐简历功能、同时可以查看用户付费订单功能，功能十分完善，极大程度上增加了用户体验。公司端作为招聘网站的第二核心，承担了职位发布等等，用户端看到的职位都是通过审核认证的公司发布的职位，所以公司端的技术点也有一些难度。公司端刚刚注册的公司需要提交公司材料进行认证审核申请，通过方可发布职位，公司通过后台审核后，会对应的有公司职位发布功能、公司职位管理功能、公司对于相应职位用户投递过来的简历状态的操作等等，功能也是十分的全面。后台管理作为本设计的权限管理中心，它管理了用户端和公司端的所有权限，同时也展示了用户和公司的信息，也完成对于公司的认证审核需求。后台系统需要完成对于用户信息的编辑和删除、实现对于公司认证审核以及注销公司功能、对于公司大数据进行的分析等等，权限限制是为了增加招聘网站的安全性。

1.2 研究的意义

在求职过程中，网络招聘等字眼被越来越多的人熟知，相比起传统的线下招聘，网络招聘的优势就体现的淋漓尽致，网络招聘极大程序降低了用人单位与求职者的成本也节约了时间，从而让跟更多人开始认同这种招聘的方式。同时随着互联网不断的发展，上网人数急剧增加，越来越多的求职者把网络招聘当成自己找工作的首选途径，而企业方面为了节省时间和人力成本，也逐渐依赖网络来招揽人才需求。一个好的工作对于刚步入社会的大学生来说是非常重要的，对于再次求职者来说也是一个好的去处，同时网络招聘也给公司提供了一个能够招聘到优秀人才的平台。在这样的环境和社会的趋势下打造一个网上招聘平台必然会很受欢迎，于是本系统的设计开发想法由此而来。

2 开发语言和工具介绍

2.1 开发工具 VSCODE

VSCODE 全称 Visual Studio Code，它是由微软开发并编写应用于现代 Web 的一个编辑器^[1]，它是比较轻量级的，但是它拥有插件库，可以通过自己装入对应的插件来提高写代码的速率或者规范等等，而且它几乎支持所有的开发语言，同时也支持跨平台开发，例如 Win、Mac 和 Linux 等等都可以流畅的运行。

2.2 开发语言 JavaScript

JavaScript 被简称为 JS，它是一个具有函数优先同时也比较轻量级，具有解释型同时拥有编译型的编程语言^[2]。大家可能对于 JavaScript 不了解不熟悉，但是大家肯定知道脚本，JavaScript 就是脚本语言而人尽皆知的^[3]。它的应用非常的广泛，JavaScript 是一种基于原型以及多范式的脚本动态语言^[4]，它的风格也是特别的独特，具有面向对象和命令式以及声明式的风格。

2.3 前端框架 VUE

Vue.js 它是 MVVM 体系结构的最佳实践、同时也是 JavaScript MVVM 库和用于构建用户界面的渐进式框架^[5]。它以 MVVM 中的 ViewModel 为核心，不仅实现了双向数据绑定，而且是一个相对轻量级的 JS 库，具有简单的 API。由于 Vue 的双向数据绑定特性和技术的形成，实现了项目的热加载^[6]。修改页面代码后，可以立即在浏览器中显示效果，提高开发效率。

2.4 后端技术 Node.js

Node.js 是基于 chromeV8 引擎的 JavaScript（脚本）运行时环境^[7]。它使用事件驱动、无阻塞的 I/O 模型，让使 JavaScript 在服务器开发平台上运行。Node.js 对一些特殊的用例进行了优化，并提供了可选的 api，使 V8 在非浏览器环境下运行得更好。V8 引擎执行 JavaScript 的速度很快，性能也很好。该平台基于 chromejavascript 运行时^[8]，用于构建响应速度快、易于扩展的网络应用程序。

2.5 数据管理 Mysql

Mysql 它是一个关系型数据库管理系统，目前接触的数据库管理系统除了 Mysql 还有非关系型数据库 MongoDB^[9]。但是在本系统中使用的是 Mysql。Mysql 会将数据保存到不同表中，提高了数据的灵活行，同时在查询数据的时候也会大大提高速度和效率。

3 系统总体设计

本系统是软件结构构思加代码实现，所以依靠的是前端框架配合后端技术以及数据存储数据库完成，以下是系统的功能设计图，主要是由后台管理也就是后端权限管理用户端和公司端，在用户端和公司端中都对应有着不同的功能模块，并且都已经实现。系统设计图如图 3-1 所示。

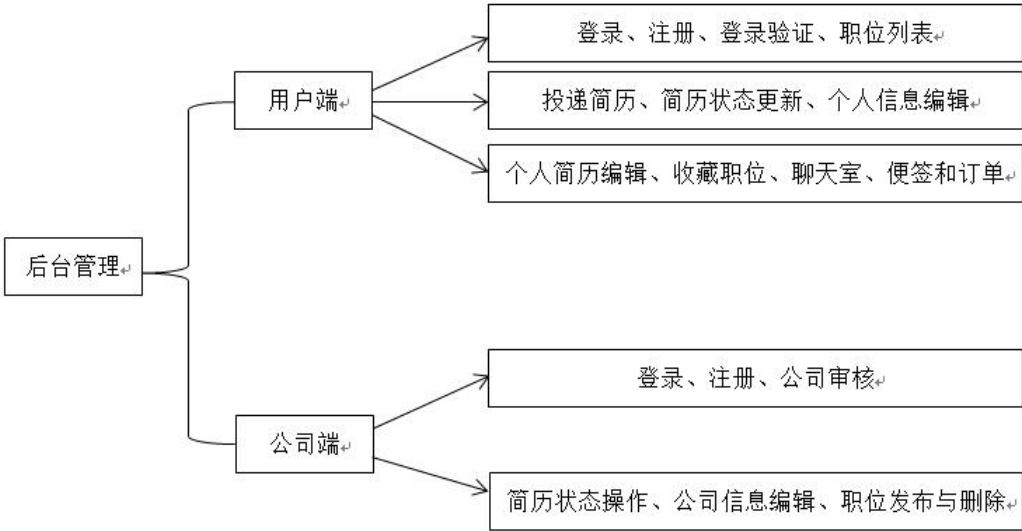


图 3-1 系统功能设计图

3.1 后台管理设计

在系统总体设计中后台管理是管理于用户端和公司端的，在后台管理中，功能的设计初衷也是为了更加方便的去管理用户端和公司端，达到一个完善的效果，所以后台管理的功能中更多的是针对一些权限功能的实现。后台管理功能设计图如图 3-2 所示。



图 3-2 后台管理功能设计图

3.2 用户端设计

在系统总体设计中用户端的设计是功能最多同时也是技术的最复杂的，因为在用户端中展示所匹配的是移动端也就是在手机上去访问，所以对于外观样式的实现等等都会比电脑端更加的复杂点，同时用户端功能点非常的多，但是在 VUE

框架中，是单个页面，也就是说不会打开的新的窗口，所以对于每一块功能其实都是单独的一个文件。同时在用户端中会使用到很多官方文档的 api，例如系统中的订单支付，它是一个大数据推荐简历的一种付费模式，而订单支付系统应用的是支付宝开放平台的 api，产生的回显数据都是由支付宝开放平台反馈给系统在进行数据的处理。用户端功能设计图如图 3-3 所示。

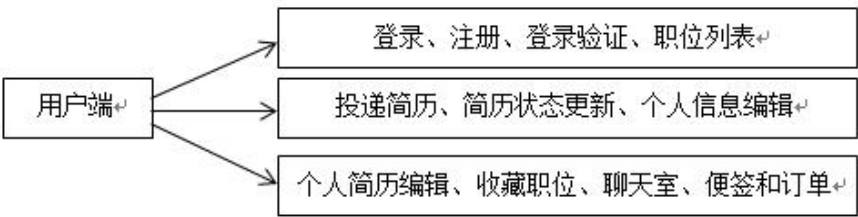


图 3-3 用户端功能设计图

3.3 公司端设计

在系统总体设计中公司端主要是对于职位的发布操作和删除。公司发布工作的前提是需要公司在注册后发送需要的审核材料给后台管理，当后台管理通过递交材料过来的公司认证的时候，职位才能正常的发布，并且即便公司端一时间通过了审核，在后续后台管理端中也能对于公司的审核认证进行更改权限，很大程度上监察了公司是否正常操作。公司端只有通过审核认证后才能正常的发布工作，用户端投递的简历，公司端可以查看并且更改简历的状态，例如更改为待沟通或者是不合适等四种状态，其次就是对于招满的职位进行删除职位等等。公司端功能设计图如图 3-4 所示。

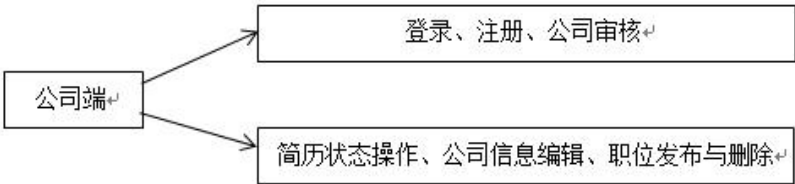


图 3-4 公司端功能设计图

3.4 后端服务器设计

服务器是一个项目的数据处理核心，所有能看到的数据都是通过后端读取数据库而返回的数据，所以服务器的设计尤为重要，在本系统设计中采用的是

Node.js 配合 Express 搭建的后端服务器，因为 Node.js 在轻量级项目和新手来说非常的友好，这也是系统采用的原因。后端服务器设计图如图 3-5 所示。

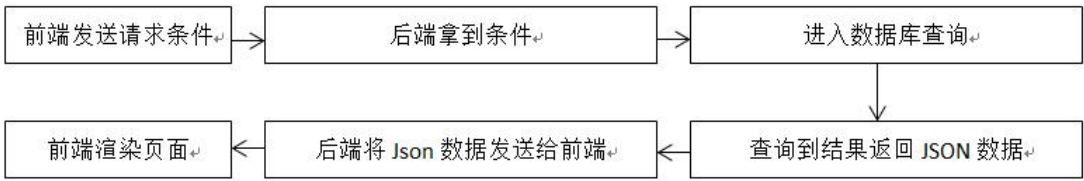


图 3-5 后端服务器设计图

3.5 数据库的设计

数据库的设计对于项目来说也是非常的总要的，因为数据库是存放一个项目数据的库，对于数据库表的设计也是很讲究的，每个表之前都会有对应的联系，在本系统中采用的就是以 ORM 为核心 sequelize 为模型建立的数据库。

(1) 用户信息表

表 3-1 用户信息表

字段	类型	说明
Account	varchar	用户姓名
Password	varchar	用户密码
Name	varchar	用户名字
Province	varchar	用户电话
City	varchar	用户居住城市
Telephone	varchar	用户手机号
Email	varchar	用户电子邮箱
Introduction	varchar	用户个人介绍
Sex	varchar	用户性别
HeadImage	varchar	用户头像
Delete	Int	用户数据是否假删除：1 为存在，0 为假删除

(2) 公司信息表

表 3-2 公司信息表

字段	类型	说明
Account	varchar	公司账号
Password	varchar	公司密码
Name	varchar	公司名称
Logo	varchar	公司图标
Stage	int	公司发展阶段：1:不需要融资,2:天使轮,3:A 轮,4:B 轮,5:C 轮,6:D 轮及以上,7:上市公司
Scale	int	公司发展规模：1:15 人一

		下,2:15-50 人,3:50-150 人,4:150-500 人,5:500-2000 人,6:2000 人以上
Verify	int	公司是否通过审核审核 {0:未通过,1:已通过,2:已认证}
Delete	int	公司数据是否假删除: 1 为存在, 0 为假删除

(3) 管理页信息表

表 3-3 管理页信息表

字段	类型	说明
Account	varchar	管理员账号
Password	varchar	管理员密码
Type	int	管理员账号类型: 1 高级管理人员, 2 管理员
Delete	int	管理页数据是否假删除: 1 为存在, 0 为假删除
adminImg	varchar	管理员头像

(4) 职位信息表

表 3-4 职位信息表

字段	类型	说明
Province	varchar	工作省份
City	varchar	工作城市
Brief	varchar	职位简介
CompanyName	varchar	公司名称
CompanyScale	Int	公司规模: {1:15 人以下, 2:15-50 人, 3:50-150 人, 4:150-500 人, 5:500-2000 人, 6:2000 人以上}
CompanyStage	Int	公司发展阶段: {1:不需要融资, 2:天使轮, 3:A 轮, 4:B 轮, 5:C 轮, 6:D 轮及以上, 7:上市公司}
CompanyLogo	varchar	公司 logo 图片
Type	Int	职位类型 {1:全职, 2:兼职, 3:实习}
maxSalary	Int	最高薪水 format (10, 99) :10k-99k
minSalary	Int	最低薪

		水 format (10, 99) : 10k-99k
Description	varchar	工作详细描述
Degree	Int	学历 {0: 大专, 1: 本科, 2: 硕士, 3: 博士, 4: 不限}
Experience	Int	工作经验 {1: 不限, 2: 应届毕业生, 3: 1-3 年, 4: 3-5 年, 5: 5-10 年}
Delete	Int	公司数据是否假删除: 1 为存在, 0 为假删除

(5) 个人简历信息表

表 3-5 个人简历信息表

字段	类型	说明
HeadImage	varchar	头像
Name	varchar	姓名
Sex	varchar	性别
Birthday	datetime	出生日期
Degree	int	学历 {0: 大专, 1: 本科, 2: 硕士, 3: 博士, 4: 其他}
Experience	int	工作年限 {0: 应届毕业生, 1-10(n): n 年, 11: 十年以上}
Telephone	varchar	电话号码
Email	varchar	邮箱
LiveCity	varchar	所在城市
LiveProvince	varchar	所在省
Brief	varchar	以前从事工作
WorkExperience	varchar	工作经历 JSONString[{company: "", position: "", joinTime: "", leaveTime: "", content: ""}, ...]
EducationExperience	varchar	教育经历: JSONString[{school: "", major: "", endTime: "", degree: ""}]
ExpectWork	varchar	期待工作 JSONString: {name: "前端", type: "全职", city: "南京", salary: 1, addition: "补充说明 (若有)"}
ProjectExperience	varchar	项目经验 JSONString {name: "商城", position: "", startTime: "", endTime: "", description: "", link: ""}
PersonalDescription	varchar	个人描述
Delete	varchar	数据是否假删除: 1 为存在, 0 为假删除

(6) 简历投递信息表

表 3-6 简历投递信息表

字段	类型	说明
Process	int	简历状态:{1:已投递,2:待沟通,3:面试,4:不合适}
InterviewDate	datetime	面试时间(简历面试之后一个月过期)
InterviewPosition	varchar	面试地点
companyID	Int	公司 ID
Delete	Int	数据是否假删除: 1 为存在, 0 为假删除
userID	Int	用户 ID
jobID	Int	工作 ID

(7) 职位收藏表

表 3-7 职位收藏信息表

字段	类型	说明
ID	Int	主键
JobID	Int	工作 ID

(8) 聊天信息表

表 3-8 聊天信息表

字段	类型	说明
ID	int	主键
chatHistory	varchar	聊天记录

(9) 便签信息表

表 3-9 便签信息表

字段	类型	说明
ID	int	主键
Tip	varchar	便签记录

4 系统详细设计

4.1 前端详细设计

4.1.1 用户端设计

在用户端的程序设计中，模块细分了很多，下面将会一一介绍各个功能的实现方法。

① 登录与注册

在前端的用户端登录中，本系统加入了滑块验证码进行登录验证，滑块验证码也是网上公开的一个 api，所以系统中运用的时候只需要进行一些简单的配置就行了。登录和注册都是通过对于请求的方法将输入的账号密码发送给后端，后端拿到数据后，将会根据查询条件去数据库中查询，在滑块验证码正确使用后且账号密码正确的情况下，才会登录到主页中，一旦账号、密码、滑块验证码其中一个错误都登录不进去，同时也会显示错误。注册同样的道理，注册首先会将账号在数据库中查询一次，防止重复账号引起的不必要的问题，注册成功后会跳转到登录页面进行登录。如下是登录和注册的部分代码：

登录：

```
this.$http
    .post('signIn', {
      Account: this.userName,
      Password: this.userPassword
    })
    .then(
      (response) => {
        this.res = response.body
        if (this.res.success) {
          setTimeout(() => {
            this.$vux.loading.hide()
            this.$vux.toast.show('登陆成功！')
            Tools.setStorage('token', this.res.data.token)
            Tools.setStorage('userInfo', this.res.data.user)
            this.$router.push('/Index')
```

```

    }, 1000)
  } else {
    document.getElementById('captcha').innerHTML = "
    this.getGeetCode()
    this.$vux.loading.hide()
    this.$vux.alert.show({
      title: '登录失败!',
      content: this.res.error.message
    })
  }
},
() => {
  this.$vux.loading.hide()
  this.$vux.alert.show({
    title: '登录失败!',
    content: '无法连接服务器...'
  })
}
)

```

注册:

```

var result = this.captchaObj.getValidate()
if (!this.userName) {
  this.$vux.toast.show({ text: '请填写用户名!', type: 'warn' })
  this.userNameState = 'error'
} else if (!this.userPassword) {
  this.$vux.toast.show({ text: '请设置密码!', type: 'warn' })
  this.passwordState = 'error'
} else if (!this.confirmPassword) {
  this.$vux.toast.show({ text: '请确认密码!', type: 'warn' })
  this.confirmPasswordState = 'error'
}

```

```

} else if (this.userPassword !== this.confirmPassword) {
  this.$vux.toast.show({ text: '两次密码输入不一致!', type: 'warn' })
} else if (!result) {
  this.$vux.toast.show({ text: '请完成验证', type: 'warn', time: 1000 })
  this.showNotice = true
  setTimeout(() => {
    this.showNotice = false
  }, 2000)
} else {
  this.$http
    .post('signUp', {
      Account: this.userName,
      Password: this.userPassword
    })
    .then((response) => {
      let self = this
      if (response.body.success) {
        this.$vux.alert.show({
          title: '提示',
          content: '注册成功! ',
          onHide() {
            self.$router.push('/')
          }
        })
      } else {
        this.$vux.alert.show({
          content: response.body.error.message,
          title: '注册失败'
        })
      }
    })
  }
}

```

```

    })
  }
}

```

②工作列表显示和工作详情

工作列表的显示在系统中运用到了一种下拉刷新上拉加载的功能，在手机上这是个非常常见的，所以本系统中也加入了这个功能模块，这样用户在使用的时候就会非常的流畅和方便。同时工作列表的显示也是根据时间来排序的，最近发布的工作会在上面，这个排序显示是后端进行处理的。而工作详情是根据用户点击的工作序号 ID 在进行查询的，最后反馈具体工作的全部数据。如下是工作列表显示和工作详情代码：

工作列表显示代码：

```

getCatesList($loadinState) {
  this.$http
    .post('getAlljobs', {
      token: tools.getStorage('token'),
      currentPage: this.findKey.currentPage,
      countPerPage: this.findKey.countPerPage
    })
    .then((_res) => {
      if (_res.body.success) {
        this.jobs = _res.body.data.rows
        this.jobList = this.jobList.concat(this.jobs)
        this.$loadinState()
        if (_res.body.data.count === this.jobList.length) {
          this.$loadinState(true)
        }
      } else {
        this.$vux.alert.show({
          title: '提示',
          content: '登录失效!'
        })
      }
    })
  }
}

```

```

        this.$router.push('/')
      }
    })
  }

```

工作详情代码：

```

created() {
  this.$http
    .post('getAlljobs', {
      ID: this.$route.params.job_id,
      token: tools.getStorage('token')
    })
    .then((_res) => {
      if (_res.body.success) {
        this.job = _res.body.data.rows[0]
        /* console.log(this.job) */
      } else {
        this.$vux.alert.show({
          title: '提示',
          content: '获取职位失败!'
        })
      }
    })
  }
}

```

③工作收藏

工作收藏功能指的是这份工作用户特别喜欢，其实很多应用都有这个功能，类似于淘宝添加到收藏夹或者购物车功能，并不需要立即投递简历等等，所有收藏的职位都能在‘我的收藏’中看到，同时也可以删除掉。运用的是每次收藏的时候将工作 ID 单独存放到数据库表中，查看的时候通过工作的 ID 进行查询收藏的工作。如下是工作收藏的部分代码：

```

this.$http

```

```

.post('getFavorites', {
  token: tools.getStorage('token')
})
.then((_res) => {
  if (_res.body.success) {
    this.Ids = _res.body.data.favorites
    this.Ids.forEach((element) => {
      this.$http
        .post('getAlljobs', {
          ID: element.JobID,
          token: tools.getStorage('token')
        })
        .then((_res) => {
          if (_res.body.success) {
            this.favjob.push(_res.body.data.rows[0])
          } else {
            this.$router.go('/home')
            this.$vux.alert.show({
              title: '获取收藏失败!',
              content: this.res.error.message
            })
          }
        })
      })
    })
  } else {
    this.$router.push('/home')
    this.$vux.alert.show({
      title: '获取收藏失败!',
      content: this.res.error.message
    })
  }
})

```

```
}  
})
```

④便签功能

便签功能其实是一个比较简单的功能，跟手机便签一样的功能，只是将输入的信息进行了保存到数据库中，后面查看的时候是通过用户登录的 ID 进行查询的，同时也可以不断的修改。如下是便签功能的部分代码：

```
this.$http  
  .post('userTip', {  
    token: tools.getStorage('token')  
  })  
  .then((_res) => {  
    if (_res.body.success) {  
      this.PersonalDescription = _res.body.data.user.Tip  
    } else {  
      this.$router.push('/home')  
      this.$vux.alert.show({  
        title: '提示',  
        content: '获取便签失败!'  
      })  
    }  
  })
```

⑤个人信息编辑

个人信息是用户账号本身登录的信息，基本信息都会有，初始登录的时候都默认为空，需要自己进行编辑修改，下次再次登录或者刷新的时候就是非常完整的个人信息了。如下是个人信息的部分代码：

```
this.$http  
  .post('personalInformation', {  
    HeadImage: this.HeadImage,  
    Name: this.Name,  
    Sex: this.Sex,
```

```

Telephone: this.Telephone,
Email: this.Email,
Introduction: this.Introduction,
Province: this.Province,
City: this.City,
token: tools.getStorage('token')
}))
.then(
  (response) => {
    this.res = response.body
    if (this.res.success) {
      this.$vux.toast.show('修改成功! ')
      /* this.$router.go('/home') */
    } else {
      this.$vux.alert.show({
        title: '修改失败!',
        content: this.res.error.message
      })
    }
  },
  () => {
    this.$vux.alert.show({
      title: '修改失败!',
      content: '请填写完整...'
    })
  }
)

```

⑥个人简历的编辑

个人简历的编辑就需要非常的认真的去对待，因为投递简历后，公司第一时间看到的是简历上的个人信息，同时简历上的信息也会特别的完整，包括技能特长等等。如下是个人简历的部分代码

```
this.$http
  .post('resume', {
    HeadImage: this.HeadImage,
    Name: this.Name,
    Sex: this.Sex,
    Birthday: this.Birthday,
    Degree: this.Degree,
    Experience: this.Experience,
    Telephone: this.Telephone,
    Email: this.Email,
    LiveCity: this.LiveCity,
    LiveProvince: this.LiveProvince,
    Brief: this.Brief,
    WorkExperience: this.WorkExperience,
    EducationExperience: this.EducationExperience,
    ExpectWork: this.ExpectWork,
    ProjectExperience: this.ProjectExperience,
    PersonalDescription: this.PersonalDescription,
    token: tools.getStorage('token')
  })
  .then(
    (response) => {
      this.res = response.body
      if (this.res.success) {
        this.$vux.toast.show('修改成功！')
        /* this.$router.push('/home') */
      } else {
```

```

        this.$vux.alert.show({
          title: '修改失败!',
          content: this.res.error.message
        })
      }
    },
    () => {
      this.$vux.alert.show({
        title: '修改失败!',
        content: '请填写完整...'
      })
    }
  )
)

```

⑦简历支付推荐以及订单功能

很多地方都会运用到大数据进行推送等功能，当支付一定的费用后，系统将会增加工作的曝光率等等，在当时进行系统设计的时候，由于技术不是那么成熟，个人能力有限，所以没想到订阅功能。所以目前的简历支付功能只是增加了自己的成功率，同时支付成功后会跳转到订单页面查看，有支付和未支付的订单。支付功能都是运用的支付宝开放平台的 api，支付工具选择的是支付宝沙盒，需要运用到此功能也需要对接支付宝开放平台的 api 进行配置等等，才能正确的使用。如下是简历支付推荐以及订单功能部分代码：

```

goshop(e) {
  let pro = e
  this.$http
    .post('submitOrder', {
      u_id: tools.getStorage('userInfo').ID,
      o_name: pro.proname,
      o_telephone: pro.desc,
      o_address: pro.serve,
      o_total: pro.price
    })
}

```

```

    })
    .then((_res) => {
        console.log(_res)
        window.location.href =
            tools.apiHost + '/zhiFuBaoWebPage?o_id=' + _res.data.data
    })
}

getUserOrders() {
    this.$http
        .post('getUserOrders', {
            u_id: tools.getStorage('userInfo').ID
        })
        .then((_res) => {
            if (_res.body.code === 0) {
                this.resumes = _res.body.data
                this.resumes.forEach((_resume) => {
                    if (_resume.o_state === this.resumeStateNum) {
                        this.resumeList.push({
                            o_id: _resume.o_id,
                            o_name: _resume.o_name,
                            o_ordertime: _resume.o_ordertime,
                            o_telephone: _resume.o_telephone,
                            o_total: _resume.o_total,
                            o_state: _resume.o_state
                        })
                    }
                })
            }
        })
    }
}

```

⑧搜索功能

搜索功能主要针对的是对于职位的搜索，用户在使用的时候可能会搜索某个相关职位或者某个公司的职位，系统针对于此就增加了搜索功能，用户通过搜索工作名称和公司的关键字就可以实现。如下是搜索功能的部分代码：

```
this.$http
  .post('searchJobs', {
    key: this.$route.query.key,
    token: tools.getStorage('token')
  })
  .then((_res) => {
    if (_res.body.success) {
      this.job = _res.body.data
      if (_res.body.data.length < 1) {
        this.$vux.alert.show({
          title: '提示',
          content: '无结果!'
        })
      }
    } else {
      this.$vux.alert.show({
        title: '提示',
        content: '搜索失败!'
      })
      this.$router.push('/')
    }
  })
```

⑨投递简历功能

投递简历功能就是用户对于心仪的职位投递简历，也就是相当于简历上的信息都能够被对应公司的管理者看到，投递简历其实就是将用户简历的 ID 进行保存对应数据库表，公司在查看的时候，只需要通过简历的 ID 进行查询反馈的

数据就是对应的用户简历信息。同时公司端在操作简历状态的时候，用户端这边也会进行对应的更改。如下是投递简历功能的部分代码：

```
let self = this

this.$http
  .post('getSentResume', { token: tools.getStorage('token') })
  .then((_res) => {
    if (_res.body.success) {
      this.resumes = _res.body.data.resumes
      this.resumes.forEach((_resume) => {
        if (_resume.jobApply.Process === this.resumeStateNum) {
          this.resumeList.push({
            src: tools.imageHost + _resume.CompanyLogo,
            title: _resume.CompanyName,
            desc: _resume.Brief,
            id: _resume.ID,
            minSalary: _resume.minSalary,
            maxSalary: _resume.maxSalary
          })
        }
      })
    } else if (_res.body.error.code === 6) {
      this.$vux.alert.show({
        title: '提示',
        content: '登录已失效!',
        onHide() {
          self.$router.push('/')
        }
      })
    }
  })
```

4.1.2 公司端设计

在公司端的程序设计中，模块功能也细分了，但是没有用户端那么多，下面将会一一介绍各个功能的实现方法。

① 公司账号登录与注册

公司账号的登录和注册都是跟用户端登录注册一个原理，也是需要将输入的账户密码进行验证，通过 Ajax 发送请求到后端进行验证，验证通过才能登录进去，如果账号密码错误会进行对应的错误等等。注册成功后跳转到登录页面再进行登录。如下是公司账号登录和注册的部分代码：

```
var signInFunc = function () {  
    var signInInfo = $('#singInForm').serializeArray()  
    var signData = {  
        Account: signInInfo[0].value,  
        Password: signInInfo[1].value  
    }  
    if (!signInData.Account || !signInData.Password) {  
        myLayer.showErrorMsg('请填写完整信息!')  
    }  
  
    postJson('/api/company/signIn', signData, function (_res) {  
        if (_res.success) {  
            sessionStorage.setItem('token', _res.data.token)  
            sessionStorage.setItem('userInfo', JSON.stringify(_res.data.user))  
  
            myLayer.showSuccessMsg('登录成功', function () {  
                window.location.href = 'modules/back.html'  
            })  
        } else {  
            myLayer.showErrorMsg(_res.error.message)  
        }  
    })  
}
```

```
}
```

② 公司信息编辑

首次登录的公司，公司的信息是不全的，需要公司管理者进行编辑，特别是公司名称、logo、规模等等，都需要进行更新和编辑。如下是公司信息编辑的部分代码：

```
tpl
```

```
.find('#gsxxLogoInput')
.unbind('change')
.bind('change', function (event) {
    tpl.find('#gsxxSubmit').attr('disabled', false)
    tpl.find('.error-span').remove()
    var img = new FormData()
    console.log(event)
    img.append('file', event.currentTarget.files[0])
    $.ajax({
        type: 'POST',
        data: img,
        url:
            tools.apiHost +
            '/picture?explore=company&token=' +
            sessionStorage.getItem('token'),
        cache: false,
        contentType: false,
        processData: false,
        success: function (_res) {
            /* console.log(_res) */
            if (_res.success) {
                tpl.find('#gsxxLogo').attr('src', tools.Host + _res.url)
                subInfo.Logo = _res.url
            } else {
```

```

        myLayer.showErrorMsg(_res.error.message)
    }
},
error: function () {
    myLayer.showErrorMsg('上传失败!')
}
})
})

```

③ 职位发布功能

公司对于缺少的职位可以进行职位发布进行招聘，不过公司在发布职位之前公司必须通过平台的认证审核，这个认证审核是后台管理进行操作，也是对于求职者的一个保障，防止一些诈骗和避免不良的公司对求职者进行诈骗和伤害等等。如下是职位功能的部分代码：

```

tpl.find('#zwfbSubmit').click(function (_event) {
    var needInfo = ['Province', 'City', 'Brief', 'Type', 'minSalary', 'maxSalary', 'Description', 'Experience', 'Degree'];
    var subInfo = {};
    var CompanyInfo = tools.getStorage('userInfo');
    if (!CompanyInfo.Name || !CompanyInfo.Logo) {
        myLayer.showErrorMsg('请先完善公司信息!');
        return false;
    }
    subInfo.CompanyName = CompanyInfo.Name;
    subInfo.CompanyLogo = CompanyInfo.Logo;
    subInfo.token = sessionStorage.getItem('token');
    needInfo.forEach(function (_info) {
        subInfo[_info] = tpl.find('#zwfb' + _info).val();
    });
    /* console.log(subInfo); */
    needInfo.forEach(function (_item) {

```

```

    if (!subInfo[_item]) {
        tpl.find('#zwfb' + _item).after('<span class="error-span">必填! </span>')
    }
});

```

④ 职位管理功能

职位管理功能指的是当公司发布的工作已经招聘到需要的人才，这个职位不在需要进行招聘的时候，公司管理者可以将这个职位进行删除，防止用户继续不必要的投递。以下是职位管理功能部分代码：

```

$jobList.find('.zwgl-detail').click(function (_event) {
    var $tr = $(_event.currentTarget).parents('tr')
    var jobId = $tr.attr('data-id'),
        jobItem
    jobsData.forEach(function (_job) {
        if (_job.ID === parseInt(jobId)) {
            jobItem = _job
            jobItem.Degree = getDegreeById(jobItem.Degree)
            jobItem.Experience = getExperienceById(jobItem.Experience)
            jobItem.Type = getTypeById(jobItem.Type)
            jobItem.Description = jobItem.Description.replace(/\n/g, '<br>')
        }
    })
})

```

4.1.3 后台管理

后台管理中，前面主要也提到了，后台管理的存在只要是对于公司和用户端的一些权限管理和公司审核认证等功能，下面将会介绍后台管理的主要功能。

① 后台管理的登录

后台管理的登录跟之前的一样，都是采用验证的方式去查询，输入的账号密码正确才能登录到后台管理系统中。以下是登录的部分代码：

```

this.$refs.LoginformRef.validate(async (valid) => {
    if (valid) {

```

```

// console.log('开始发送请求')
// console.log(this.$http)
// 当服务器没有开或者失败的情况没有考虑
try {
  const res = await this.$http.post('SignIn', this.Loginform)
  console.log(res)
  if (res.error) {
    this.$message({
      showClose: true,
      message: '账号或密码错误',
      type: 'error',
      duration: 1000
    })
  } else {
    this.$message({
      showClose: true,
      message: '登陆成功',
      type: 'success',
      duration: 1000
    })
    // 存储 token 令牌到 sessionStorage 里面
    window.sessionStorage.setItem('token', res.data.token)
    window.sessionStorage.setItem('Account', res.data.user.Account)
    window.sessionStorage.setItem('ID', res.data.user.ID)
    window.sessionStorage.setItem('adminImg', res.data.user.adminImg)
    // 开始存储 token
    this.$router.push('/home')
  }
} catch (error) {
  this.$message({

```

```

        showClose: true,
        message: error.message,
        type: 'error',
        duration: 1000
    })
}
// 结束 loading
} else {
    console.log('验证失败')
}
})

```

②公司审核认证

公司审核认证就是针对于刚刚注册的公司没有提交审核认证的公司不给予发布工作的权限，公司必须通过审核认证才能发布工作，而给予公司认证通过就是后台管理的权限，可以随时打开和关闭公司发布工作的权限。如下是公司审核认证的部分代码：

```

try {
    const res = await this.$http.post('/verifyCompany', {
        companyID: userRow.ID,
        Verify: userRow.Verify
    })
    console.log(res)
    if (res.error) {
        this.$message.error('公司授权失败')
        console.log(res)
    } else {
        console.log(res)
        this.$message.success(res.success.message)
    }
} catch (error) {

```

```

    this.$message.error(error.message)
    console.log(error)
  }
}

```

③工作查看和管理

工作的所有列表也会在后台进行显示，主要是防止一些工作的重复性和不安全性，防止公司即使通过审核也会发布不良工作等，能够即使掌握最新的消息，同时也包含搜索功能等等。工作查看和管理的部分代码如下：

```

async getUsersList() {
  try {
    this.queryInfo.token = Tools.getStorage('token')
    console.log(this.queryInfo.token)
    const res = await this.$http.post('/getAllJobs', {
      params: this.queryInfo
    })
    console.log(res)
    console.log(res.data)
    if (res.success === false) {
      this.$message.error(res.error.message)
      this.$router.push('/')
    } else {
      this.usersList = res.data.rows
      // 这个数据从服务器返回过来的
      this.totalPage = res.data.count
      console.log(this.totalPage)
      // this.pagenum = res.data.pagenum
    }
  } catch (error) {
    // 使用 message 提示错误信息
    console.log(error)
  }
}

```

```

    }
  },
  // 实现删除职位的功能
  async deleteJobsById(id) {
    try {
      const result = await this.$confirm(
        '您确定需要永久删除该职位吗?',
        '提示'
      )
      console.log(result)
      // 发送请求删除职位的信息
      console.log(id)
      const res = await this.$http.post('/deleteJob', {
        ID: id
      })
      console.log(res)
      if (res.error) {
        this.$message.error(res.error.message)
        console.log(res.error)
      } else {
        this.$message.success(res.success)
        console.log(res.success)
        // 重新更新界面
        this.getUsersList()
      } } catch (error) {
        console.log(error)
        this.$message.error(error)
        console.log(error)
        this.getUsersList()
      }
    }
  }
}

```

4.2 后端详细设计

4.2.1 服务器的搭建

在后端中需要搭建项目需要的服务器，在本系统中运用的是 Node.js 配合 Express 作为后端服务器，好处是方便更加容易入门，同时也减轻了完成项目时候的压力，因为 Node.js 作为后端对于刚刚入门的前端程序员来说非常的友好，而且目前 Node.js 的兼容性更加的完善，功能也逐渐慢慢强大，逐渐在后端中占据了一席之地。如下是搭建后端服务器的主要代码：

```
require('./check-versions')()
var config = require('../config')
if (!process.env.NODE_ENV) process.env.NODE_ENV = JSON.parse(config.dev.env.NODE_ENV)
var path = require('path')
var express = require('express')
var webpack = require('webpack')
var opn = require('opn')
var proxyMiddleware = require('http-proxy-middleware')
var webpackConfig = require('./webpack.dev.conf')
var cookieParser = require('cookie-parser');

//后台管理程序
var bodyParser = require('body-parser');
var logger = require('../lib/logger');
var busboy = require('connect-busboy');
var session = require('../lib/middleware/session');
//路由
var user = require('../routes/user');
var company = require('../routes/company');
var admin = require('../routes/admin');
var picture = require('../routes/picture');
var code = require('../routes/code/register');
```

```

var port = process.env.PORT || config.dev.port
var proxyTable = config.dev.proxyTable
var app = express()
app.use('*', function (req, res, next) {
  res.header("Access-Control-Allow-Origin", "*");
  res.header("Access-Control-Allow-Headers", "Content-Type,Content-Length, Authorization, Accept,X-Requested-With");
  res.header("Access-Control-Allow-Methods", "PUT,POST,GET,DELETE,OPTIONS")
;
  res.header("X-Powered-By", ' 3.2.1');
  if (req.method == "OPTIONS") res.send(200);/*让 options 请求快速返回*/
  else next();
});

```

4.2.2 数据库的连接

在一个项目完成基本的搭建后，系统需要将项目与对应的数据库进行连接，这样项目中的数据库才能保存到数据库中，同时在使用查询的时候也才能拿到数据库中的数据库，所以这样说，连接数据库可以说是一个项目的必要条件，如下是连接数据库的主要代码：

```

let params = {
  user: {
    token: {
      expires: 2000 * 60 * 1000 //单位为毫秒
    }
  },
  salt: 'jurenyoupin',
  db: {
    host: 'localhost',
    dialect: 'mysql',
    port: '3306',
    user: 'root',

```

```
password: '123456',  
database: 'jurenyoupin',  
charset: 'UTF8_GENERAL_CI',  
timezone: '+08:00',  
},  
adminKey: '10086'  
};
```

5 系统测试

5.1 用户端功能测试

系统登录页面如图 5-1 所示。



图 5-1 登录页面

系统登录验证页面如图 5-2 所示。



图 5-2 登录验证页面

系统首页如图 5-3 所示。

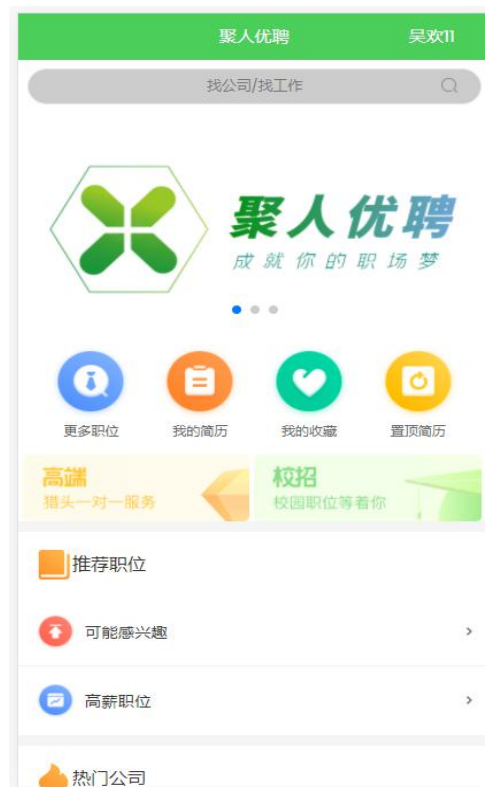


图 5-3 系统首页页面

系统工作列表如图 5-4 所示。

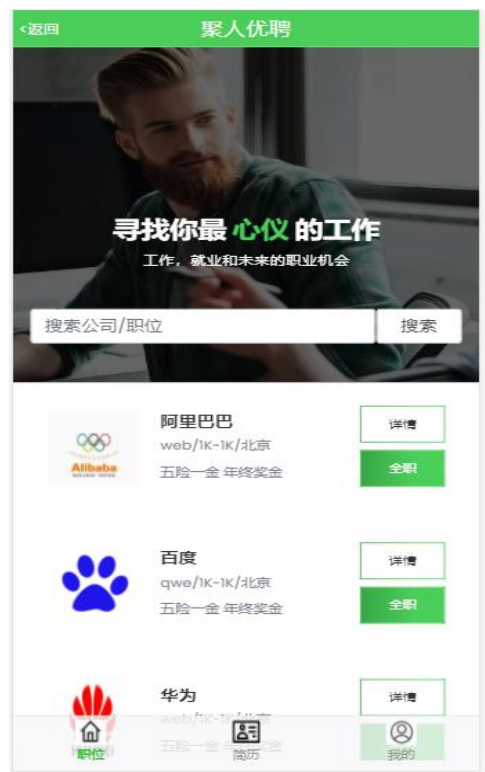


图 5-4 系统工作列表页面

已投递简历页面如图 5-5 所示。



图 5-5 已投递简历页面

工作详情页面如图 5-6 所示。



图 5-6 工作详情页面

个人中心页面如图 5-7 所示。



图 5-7 个人中心页面

我的收藏页面如图 5-8 所示。



图 5-8 我的收藏页面

我的便签页面如图 5-9 所示。

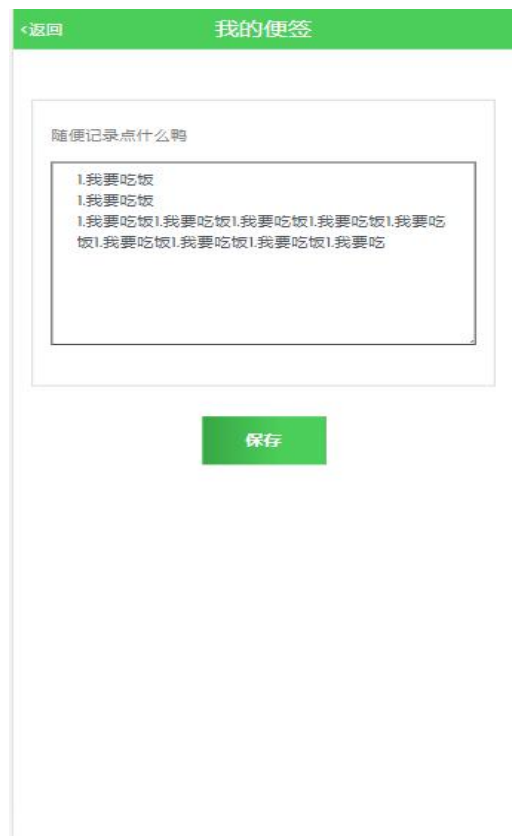


图 5-9 我的便签页面

我的简历页面如图 5-10 所示。

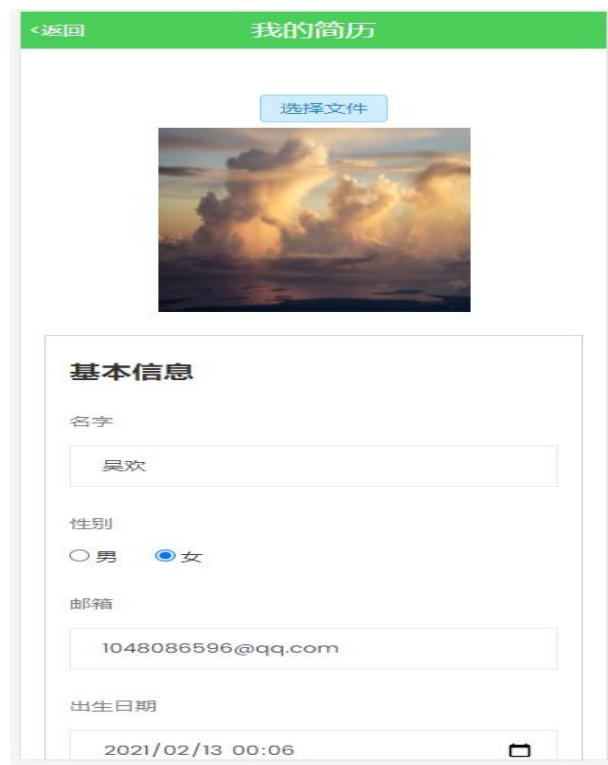


图 5-10 我的简历页面

我的订单页面如图 5-11 所示。



图 5-11 我的订单页面

5.2 公司端功能测试

公司登录页面如图 5-12 所示。



图 5-12 公司登录页面

公司信息页面如图 5-13 所示。

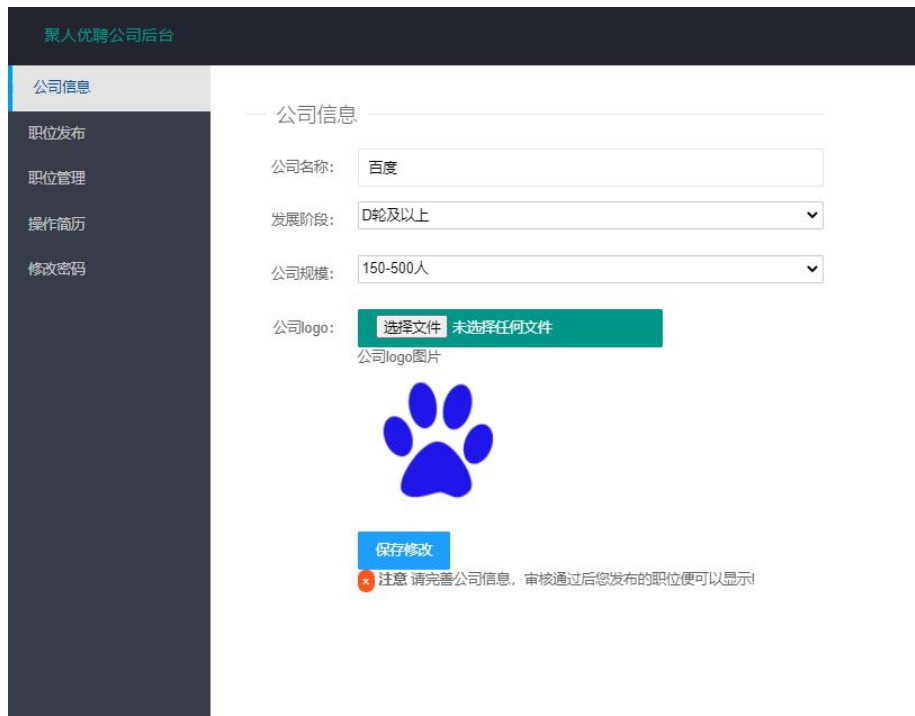


图 5-13 公司信息页面

公司职位发布页面如图 5-14 所示。

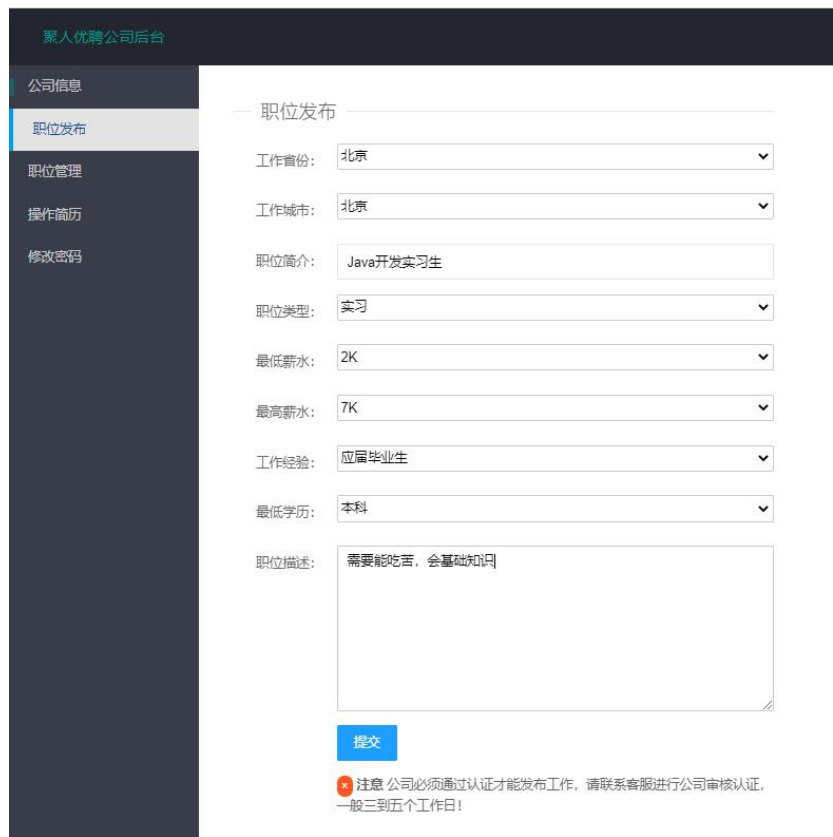


图 5-14 公司职位发布页面

公司职位管理页面如图 5-15 所示。

聚人优聘公司后台

操作

公司信息

职位发布

职位管理

操作简历

修改密码

简介	地点	学历	经验	类型	薪水	描述	操作
js	上海	硕士	5-10年	全职	11K-27K	javascript...	详情 删除
Java	安徽合肥	硕士	1-3年	全职	10K-24K	Java web工程...	详情 删除
web前端	山西大同	博士	3-5年	兼职	12K-21K	这是web前端的职位...	详情 删除
nodejs程序员	山西长治	本科	1-3年	全职	7K-13K	三年工作经验，熟悉...	详情 删除
Java	山西太原	大专	不限	全职	1K-1K	Java程序员，Ja...	详情 删除
android开发	北京	本科	1-3年	全职	10K-18K	熟悉Java语言，熟...	详情 删除
web前端	山西朔州	硕士	1-3年	全职	12K-22K	前端开发工程师，精通...	详情 删除

公司信息

职位发布

职位管理

操作简历

修改密码

未读 待沟通 通知面试 不合适

职位: web前端 申请人数: 1

姓名	性别	邮箱	手机	职业	操作
吴欢11	男	1048086596@qq.com	18271412846	前端	☐ 已投递 ☐ 待沟通 ☐ 待面试 ☐ 不合适

职位: sdasdm1 申请人数: 1

姓名	性别	邮箱	手机	职业	操作
吴欢11	男	1048086596@qq.com	18271412846	前端	☐ 已投递 ☐ 待沟通 ☐ 待面试 ☐ 不合适

图 5-16 公司职位及简历状态页面

5.3 后台管理功能测试

后台管理登录页面如图 5-17 所示。



图 5-17 后台管理登录页面

后台管理员信息页面如图 5-18 所示。

聚人优聘

123 退出

管理员

职位

公司

用户

定位

首页 > 管理员管理 > 个人信息

* 用户名 123

* 密码 123

昵称 一级管理员

图片

重置 确定

图 5-18 后台管理员页面

后台管理员查看工作列表页面如图 5-19 所示。

聚人优聘

123 退出

管理员

职位

职位列表

公司

用户

定位

首页 > 职位管理 > 职位列表

请输入搜索信息

ID	职位	日期	省份	城市	职位名称	公司名称	操作
11		2020-10-22 19:01:57	上海	上海	js	百度	删除
12		2020-10-22 19:02:36	安徽	合肥	Java	百度	删除
13		2020-10-22 19:02:52	山西	大同	web前端	百度	删除
14		2020-10-22 19:03:11	山西	长治	nodejs程序员	百度	删除
15		2020-10-22 19:03:14	山西	太原	Java	百度	删除

共 64 条 < 1 2 3 4 5 6 ... 13 > 前往 1 页

图 5-19 后台工作列表页面

后台对于公司审核认证页面如图 5-20 所示。

ID	Logo	公司名称	公司负责人账号	公司授权	日期	日期	操作
2		百度	123	☒	2017-01-12 17:13:27	2021-01-31 15:49:26	删除
10		华为	111	☒	2020-10-21 14:57:26	2021-01-31 15:12:13	删除
15		阿里巴巴	222	☒	2021-01-31 15:21:19	2021-01-31 15:24:01	删除
16		小米	333	☐	2021-01-31 15:50:32	2021-01-31 15:51:42	删除

图 5-20 公司审核认证页面

后台查看用户列表页面如图 5-21 所示。

ID	用户头像	用户名称	工作意向	电话	邮箱	性别	操作
18		吴欢11	前端	18271412846	1048086596@qq.com	男	删除
28		123	123	123	123	男	删除

图 5-21 用户列表页面

后台查看公司认证统计图页面如图 5-22 所示。

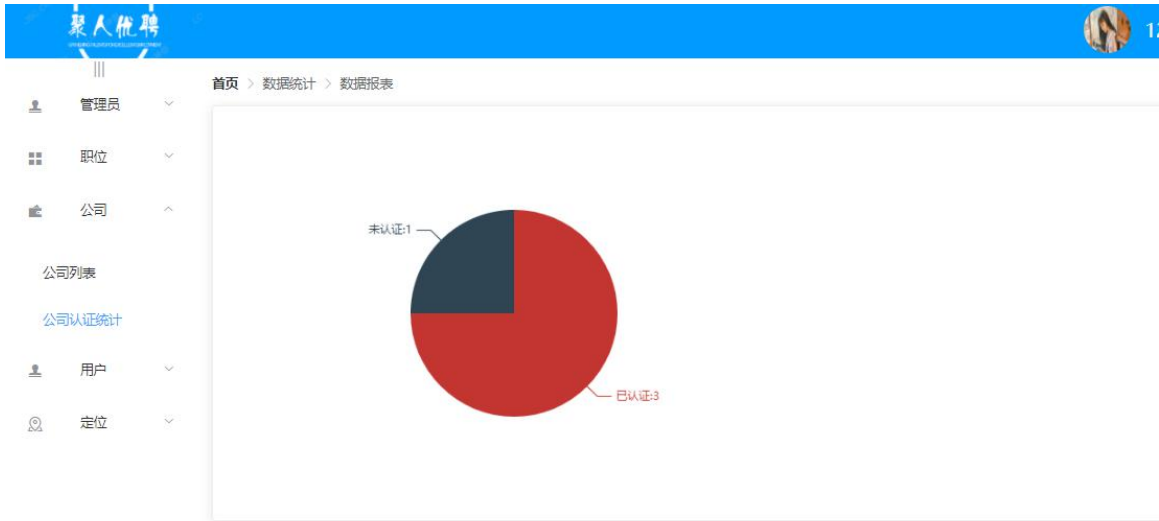


图 5-22 公司认证统计图页面

6 总结

在完成这个毕业项目设计的过程中，我觉得最大的收获应该是自己慢慢的学会了去主动学习，例如在我们软件前端中需要很多的框架或者方法都需要去官网查看对应的 api 才能够配置使用，同时也锻炼了自己的学习能力，在完成配置项目的过程中其实也遇到过很多问题，简单的通常都会自己去解决，难一点就可能会需要去网上进行百度或者问一下身边的人，实在难以解决的可能就会选择去询问老师，同时也收获了很多，掌握到了解决问题的核心。

在选择毕设项目的时候其实自己也犹豫了很久，毕竟这个项目是对于大学的一个总结，最后还是选择了自己刚刚学习过的前端项目作为毕设，第一是因为自己刚刚学习过，作为毕设项目可以当做巩固知识，第二其实就是因为代表自己心里还是慢慢先当一个程序员，不过在我写这篇毕业论文的时候已经是一个前端开发工程实习生了。所以这个项目算是我从通信一个硬件专业转到软件行业的一个开头，意义是非常大的。

在前端这个行业来说，需要有很好的自学能力和逻辑性，因为现在的社会上对于软件行业人才需求还是很大的，同时这几年逐渐流行的前后端分离的技术，现在前端人才也是非常的受欢迎，但是前端框架的流行和更新迭代非常的快，要自己去学习新的框架是必须的，这也是每个前端工程师必备的学习知识，同时在学习新的框架的时候也要不断去提高自己的逻辑性，这样一来写出来的代码质量会高很多，同时也经得住推敲和测试。

最后其实这个项目完成后其实放在了公网上进行过测试，目前没有什么大问题，最后导致阿里云服务器过期了，项目也就不能访问了。总体来说这个毕设项目我个人在目前觉得是功能比较完善的，同时也是自己前端所学的第一个框架做出来的项目，也非常的有意义，所以作为自己的毕业设计项目，感觉非常的合适，同时也希望在以后这个项目有时间我能够用其他框架进行重写一次。

参考文献

- [1] 张意. 计算机软件工程的维护措施和方法[M]. 北京: 清华大学出版社, 2017.
- [2] 易剑波. 基于 MVVM 模式的 WEB 前端框架的研究[J]. 信息与电脑(理论版), 2016, 43(02):115-113.
- [3] 真国诚. 基于计算机软件工程的现代化技术[M]. 北京: 清华大学出版社, 2017.
- [4] 易剑波. Web 新兴前端框架与模式研究[J]. 信息与电脑(理论版), 2016.
- [5] 孔晓. 软件工程中的常用软件生命周期模型[J]. 电子技术与软件工程, 2017, 43(02):115-113.
- [6] 温谦. HTML+CSS 网页设计与布局从入门到精通[M]. 北京: 人民邮电出版社, 2008.
- [7] 刘翔宇. 基于 Vue 的数据可视化系统的设计与实现[D]. 北京: 北京邮电大学 2018.
- [8] 麦冬, 陈涛, 梁宗湾. 轻量级响应式框架 Vue.js 应用分析[J]. 信息与电脑(理论版), 2017, 11(06):15-20.
- [9] 胡雅丽. 基于 BootStrap+JavaScript 语言简易计算器的实现[M]. 太原: 太原学院学报(自然科学版), 2020.
- [10] 王移芝. 大学计算机基础教程[M]. 东营: 中国石油大学出版社, 2020.

致谢

至此，终于完成了毕业设计，心里有点如释重担的感觉，同时也有些许的失落，因为当我写到这儿的时候，代表我们这一届的毕业也即将踏入社会不再是一个学生的身份，但是我们要学的东西还是很多。回顾四年大学旅程，有开心、难过、有努力、有懒惰，转眼间已然临近毕业，在大学教室中上课的时光，那些在学校参加比赛的时光，那些在大学校园，在图书馆复习的时光仿佛就如昨日，再次睁眼，已然过去。昔日同窗室友，皆已各奔东西，这是一场我们自己的人生，这场人生中我们会遇到很多人，他们教会我们很多东西，让我们受益良多，所以我们要珍惜眼前人。

同时感谢老师对于毕业论文的指导和负责，作为我的毕业论文的负责老师，也为我们能够顺利毕业操心了很多，再次感谢。同时也感谢母校这四年的栽培和教学，因为大学学的更多的是为人处世，所以大学的日子往往是美好的向往的，也是最让人怀念的。最后想感谢自己的父母将自己养育成人，从小到大能受到良好的教育；感谢我的朋友，在我实习期间一直鼓励我认真工作实习，陪伴我度过了我最孤独和艰难的时刻。至此，大学四年已经过去，感谢遇见，感谢经历。

最后想用一句话来表达自己的心情吧，‘尚未佩妥剑，转眼便江湖。愿历尽千帆，归来仍少年’。