《信息系统集成与管理》
小组实习报告

学　　院：遥感信息工程学院

班　　级：22F12、22F13

小组成员：杨皓翔 2022302131097

赵慧琳 2022302131061

贾羽佳 2022302131030

实习地点：附三教学楼 203

指导老师：王华敏

2024 年 12 月 5 日

目录

一、实验目的

当前大学生“就业难”已成为广泛关注的社会问题，武汉大学遥感信息工程学院的学生也面临着这样的挑战。而且遥感作为一个相对前沿的领域，其专业人才的就业选择相对有限，而且由于信息不对称，学生很难获取到全面的行业动态和就业机会。为了应对这种困境，我们希望开发一个面向武汉大学遥感在校生及校友的招聘平台。该平台旨在为遥感专业的学生提供一个集中的就业信息汇总，帮助更好地了解行业需求，拓宽就业视野，增加就业机会。通过这个平台，学生可以了解到最新的行业动态、招聘信息，以及与遥感相关的职业发展路径。

二、实验环境

1. IntelliJ IDEA 2024.3
2. mysql 数据库 8.0.36
3. MySQL Workbench 8.0 CE（mysql 数据库可视化操作界面）
4. Vue/cli 5.0.8
5. Jdk17

三、项目介绍

1. 项目概述

该项目是一个前后端分离的 Web 应用，后端基于 Spring Boot 框架，前端采用 Vue.js 技术。后端通过 Spring Boot 提供 RESTful API，包括用户登录验证和企业信息展示功能。前端使用 Vue.js 构建用户界面，并通过 Vuex 进行状态管理。项目集成了 Element UI 库来加速界面开发，并使用 axios 与后端进行通信。后端配置了 MyBatis 与数据库交互，并在 application.properties 中配置了数据库连接和 MyBatis 的 Mapper 文件设置。

2. 项目功能

（1）网页端

网页端是一个使用 Vue.js 框架开发的前端应用程序，它为用户提供了一个直观的用户界面，并与后端服务进行交互。通过 App.vue 根组件，网页端构建了一个包含导航栏和路由视图的基本布局。利用 Vue Router，网页端实现了动态路由

管理，允许根据后端提供的菜单数据动态添加路由，从而展示不同的页面和功能模块。状态管理方面，网页端使用 Vuex 来维护全局状态，例如菜单列表，并通过 mutations 来更新这些状态。前端还集成了 Element UI 库来加速界面的开发，并使用自定义的 global.css 文件来统一管理样式。HTTP 请求通过 axios 库与后端 API 进行通信，处理登录请求和获取企业信息等操作。此外，网页端还提供了退出登录的功能，允许用户在导航栏中一键登出。整个网页端的设计注重用户体验和界面的响应式设计，以确保在不同设备上都能提供良好的访问体验。

（2）服务端

服务端是一个基于 Spring Boot 框架构建的 Java 应用程序，主要负责处理业务逻辑和数据存取。它提供了用户登录验证功能，通过 UserService 中的 validateLogin 方法，接收前端传来的用户名和密码，与数据库中存储的信息进行比对，实现用户身份的验证。此外，服务端还提供了企业信息服务，通过 firmService 中的 getgetfirm 方法从数据库获取企业信息列表，并以 RESTful API 的形式提供给前端。服务端使用 MyBatis 作为 ORM 框架，通过 UserMapper 和 firmMapper 接口定义数据库操作。为了支持前后端分离的开发模式，服务端还配置了跨域资源共享（CORS），允许前端开发服务器发起的跨域请求。所有配置信息，包括数据库连接和 MyBatis 的配置，都在 application.properties 文件中集中管理。整个服务端的设计注重模块化和解耦，以提高代码的可维护性和可扩展性。

3. 项目文件详细概述

（1）后端（Java/Spring Boot）

①Spring Boot 应用入口：HttpserverApplication.java 是 Spring Boot 应用的启动类，标注了@SpringBootApplication 注解，用于启动整个 Spring Boot 应用。

②服务层（Service）：firmService.java 提供企业信息的业务逻辑处理，通过注入的 firmMapper 从数据库获取企业信息；UserService.java 提供用户登录验证的业务逻辑处理，通过注入的 UserMapper 从数据库验证用户名和密码。

③数据访问层（Mapper）：firmMapper.java 是 MyBatis 的 Mapper 接口，定义了与企业信息相关的数据库操作；UserMapper.java 是 MyBatis 的另一个 Mapper

接口，定义了与用户信息相关的数据库操作。

④实体类（Entity）：firm.java 是企业信息的实体类，包含企业名称、经度和纬度；user.java 是用户信息的实体类，包含用户名和密码。

⑤控制器层（Controller）：firmController.java 处理企业信息的 HTTP 请求，提供跨域支持，并返回企业信息列表；UserController.java 处理用户登录的 HTTP 请求，提供跨域支持，并返回登录验证结果。

⑥配置文件：application.properties 是 Spring Boot 的配置文件，包含了应用名称、服务器端口、数据库连接信息、MyBatis 配置等。

⑦MyBatis 配置：在 application.properties 中指定了 MyBatis 的 Mapper 文件位置和类型别名包。

（2）前端（Vue.js）

①主组件：App.vue 是 Vue 应用的根组件，包含导航栏和路由视图。

②入口文件：main.js 是 Vue 应用的入口文件，导入 Vue、Element UI、Vue Router、axios 等依赖，并配置 Vue 实例。

③状态管理（Vuex）：index.js 是 Vuex 的配置文件，定义了全局状态管理，包括菜单列表的状态和 mutations。

④路由配置：在 index.js 中定义了动态路由的添加逻辑，根据从后端获取的菜单列表动态添加路由。

⑤全局样式：在 main.js 中导入了全局样式文件 global.css。

⑥Axios 配置：在 main.js 中配置了 axios 实例，并将其挂载到 Vue 原型上，方便在组件中使用。

这个项目是一个典型的前后端分离的 Web 应用，后端负责业务逻辑处理和数据存取，前端负责展示和用户交互。项目使用了 Spring Boot 框架进行后端开发，Vue.js 框架进行前端开发，Element UI 库来快速构建界面，并且通过 Vuex 进行状态管理，项目具体的文件组织与管理见下图 1。

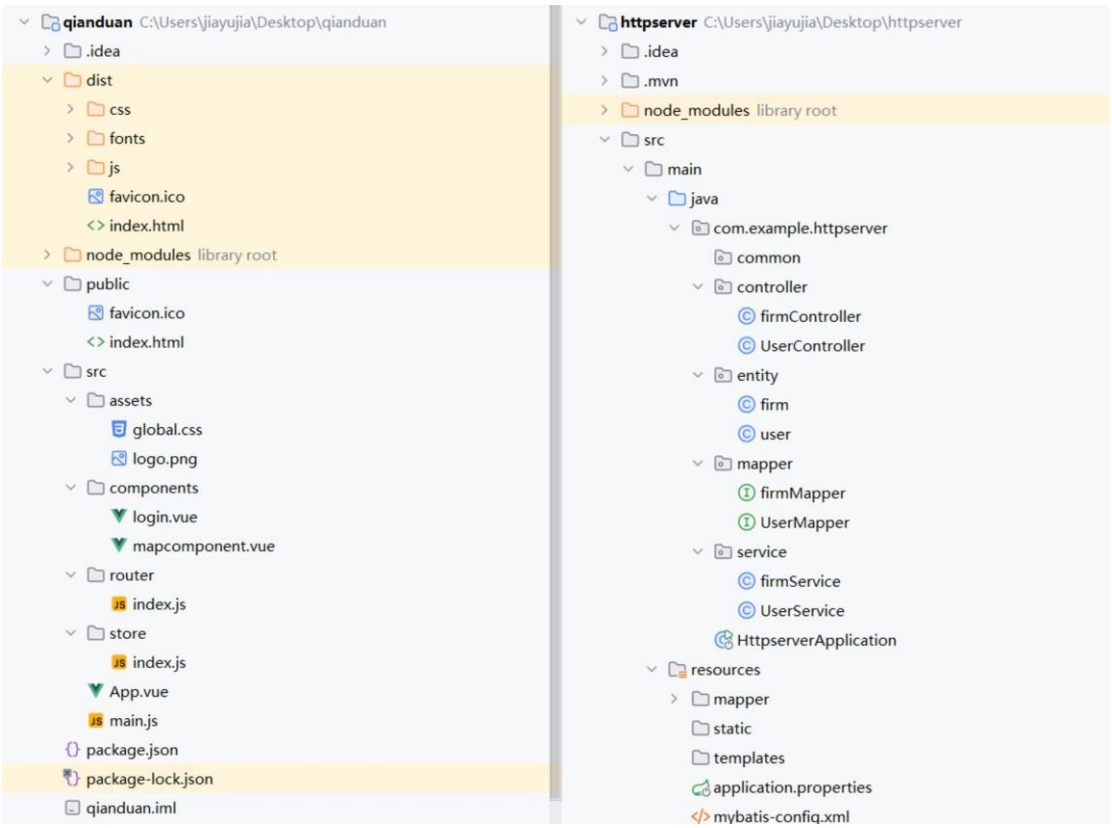

图 1 前后端项目文件概述

四、部署方法

①安装 MySQL 8.0 及以上版本（如果已有可跳过此步骤，如果低于此版本需完全卸载再安装最新版本）。

②启动 MySQL 服务（这个步骤不可或缺，每次使用前都要确保 MySQL 的服务已经开启，否则服务器无法连接 mysql 数据库）。

③打开 MySQL Workbench 8.0 CE，以下面的信息登录 mysql 数据库如下图 2（密码默认为 123456，我设置的为 152935，这个根据实际情况填写）。连接后，新建数据库 db，选项默认不需要改动；之后按下图 3 创建表和字段（注意表的的编码方式选择 utf8，否则不能显示中文，会乱码）。

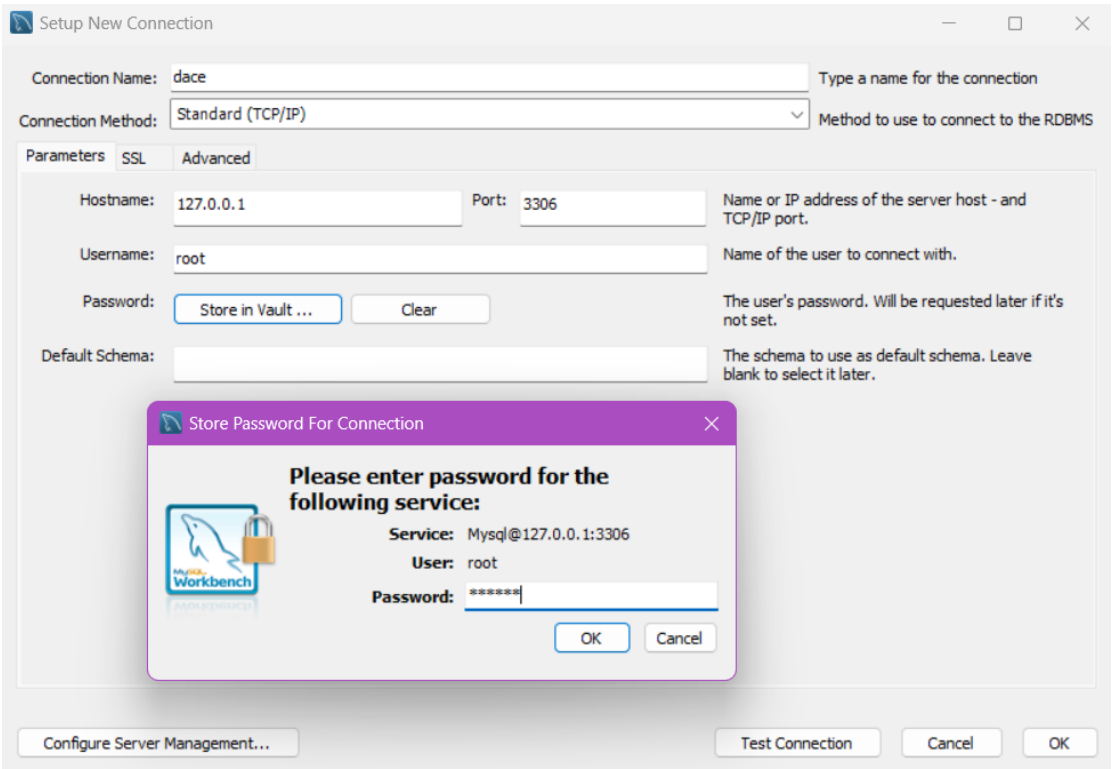

图 2 连接 mysql 数据库

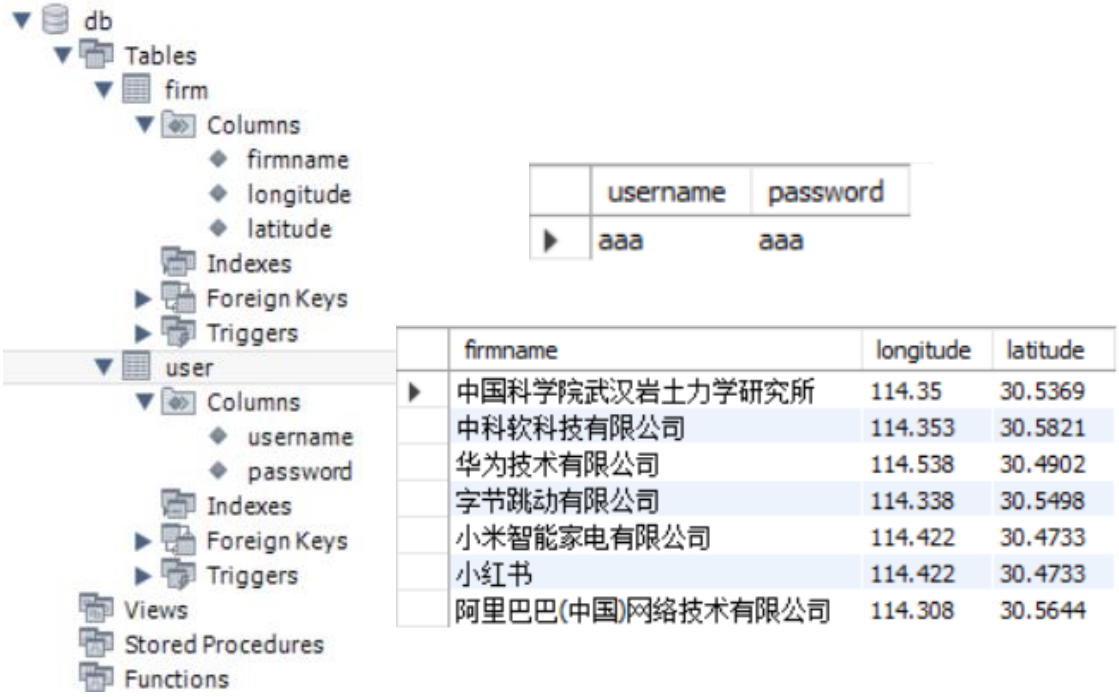

图 3 数据库表和字段

④在 IDEA 中打开 httpserver 项目，配置 jdk17 环境，打开 \httpserver\src\main\resources\application.properties，修改属性为下图 4。

```
spring.application.name=httpserver
server.port=2020
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.username=root
spring.datasource.url=jdbc:mysql://localhost:3306/db?useSSL=false&serverTimezone=UTC&\
  useUnicode=true&characterEncoding=UTF-8&autoReconnect=true
spring.datasource.password=152935
mybatis.mapper-locations=classpath:mapper/userMapper.xml,classpath:mapper/firmMapper.xml
mybatis.type-aliases-package=com.example.httpserver
```

图 4 修改项目数据库属性配置

⑤成功运行后端项目后，打开前端项目，配置 vue 环境，运行后在浏览器访问 http://localhost:8080/，即可跳转到登录界面，系统正常运行。

五、网页前端

1. 技术路线概述

网页端采用 Vue.js 作为前端框架构建用户界面。Vue.js 的组件化特性使得开发大型单页应用变得简单，每个组件负责视图的一部分，易于维护和复用。使用 Vue Router 管理页面路由，实现 SPA（单页应用）的导航功能，使得用户在不同页面间切换无需重新加载整个页面。状态管理通过 Vuex 实现，集中存储和管理所有组件的状态，保证状态的一致性和可预测性。Element UI 库提供了一套完整的 UI 组件，加速了界面的开发进程，并保证了界面的美观和一致性。axios 库用于与后端进行 HTTP 通信，支持 Promise API，使得异步请求更加简洁和易于管理。全局样式通过 CSS 文件统一管理，保持了界面风格的一致性。整体上，网页端技术路线注重用户体验、界面响应性和开发效率，通过现代化的工具和库，实现了一个高性能、易维护的前端应用。

2. 界面 UI

访问 http://localhost:8080/，进入用户登录界面（如下图 5）。

图 5 用户登录界面

用户名输入“aaa”，密码输入“aaa”登录，跳转到企业信息界面，如下图 6。

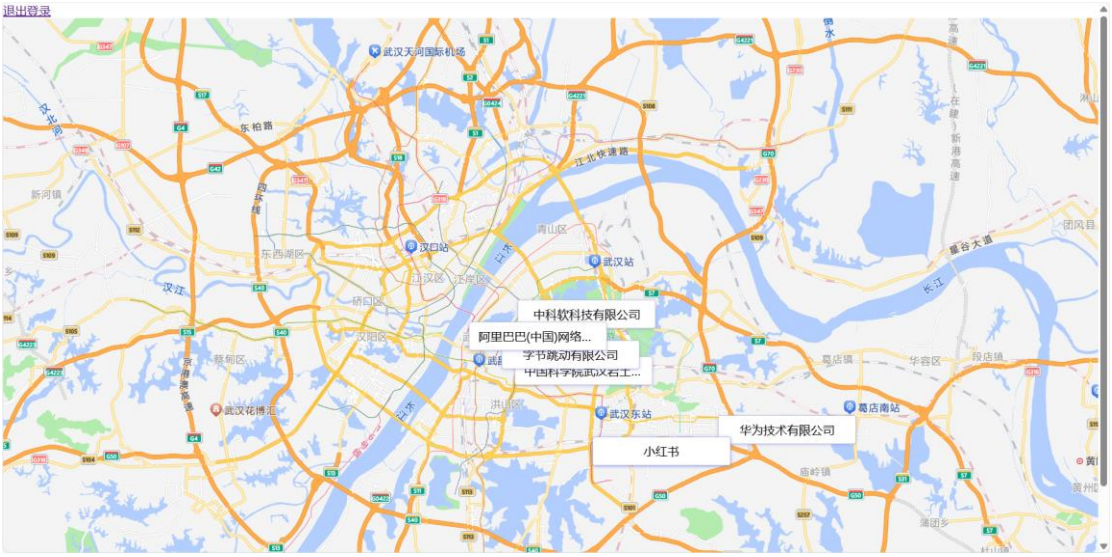

图 6 企业信息显示界面

点击左上角“退出登录”，可退回到用户登录界面。

六、服务端

服务端采用 Spring Boot 框架，用于创建独立、生产级的基于 Spring 框架的应用。通过@SpringBootApplication 注解，项目整合了 Spring Boot 的核心功能。使用@RestController 注解定义了 RESTful 风格的控制器，通过@CrossOrigin 支持跨域请求，使得前端 Vue.js 应用能够从不同的端口安全地访问后端服务。数据访问层使用 MyBatis 框架，通过定义 Mapper 接口与 XML 配置文件实现 SQL 操作的映射，简化数据库交互。实体类使用@Data 注解的 Lombok 库来自动生成

getter 和 setter 方法，减少模板代码。配置文件 application.properties 用于集中管理数据库连接、MyBatis 配置等参数，使得应用配置更加灵活和可维护。整体上，服务端技术路线强调了简洁性、模块化和快速开发，同时保证了良好的性能和稳定性。

七、困难及解决

1. 后端

①数据库连接问题

由于我们是使用的 mybatis 对数据库进行操作，因此需要建立 java 接口文件和 XML 映射文件，其中在 java 接口文件和 xml 映射文件中均运行后报错 select 定义重复。

出问题的原因是当时跟着 csdn 中的教程一步步做，谁知教程误导，教程上说要在 java 接口文件和 xml 映射文件中给同一个 id 同时定义 select *from user where username=#{username}，当时由于过于信任教程，只以为是自己的依赖和配置问题，在这里徘徊许久，后来又去到视频网站重新了解才知道原来 java 接口文件和 XML 映射文件中只能定义一次，要么在接口文件定义，要么在 XML 映射文件定义，反正不能一起定义。所以有时候教程要多看几份，且不能过于轻信教程。

②select *from user where username=#{username}SQL 语句报警

需要在数据库中进行数据库的配置和 sql 语句的调试

③数据库查询返回数据的接受类型问题

在前后端调试时一直显示 500 服务器内部出现问题，经几番排查发现是数据库的数据无法传入 mapper 接口函数中，原因是当时查询了所有数据并将七条数据同时返回，而在 xml 文件中设置的数据接收类型是 firm 实体，而 firm 实体仅对应一条数据的类型。因此需要定义一个 resultMap 作为集合数据的传入载体，并将 firm 实体作为 resultType（如下图 7）。

```
<resultMap id="FirmResultMap" type="com.example.httpserver.entity.firm">
    <result property="firmname" column="firmname"/>
    <result property="latitude" column="latitude"/>
```

```
    <result property="longitude" column="longitude"/>
</resultMap>
<select id="getFirm" resultMap="FirmResultMap"
resultType="com.example.httpserver.entity.firm">
    SELECT * FROM firm
</select>
```

图 7 定义 Firm 的结果映射

④前端访问传入的 URL 读取失败，不识别

在网站调试时一直显示 400，无法解析前端传入 URL。经排查，原因是在后端设置的解析前端 URL 方法为@RequestParam，这种方法默认前端数据传入是在 URL 中，然而前端的数据传输形式是，URL 为仅访问路径，数据全部放在 data 中，因此后端始终无法解析前端 URL。解决方法是将方式改成@RequestBody，并在@Data 中设置 user 类（注意内部类必须设置成静态，否则也会报错），如下图 8。

```
    @PostMapping("/login")
public boolean login(@RequestBody user user) {
    if (userService.validateLogin(user.username, user.password)) {
        return true;
    } else {
        return false;
    }
}
@Data
public static class user{
    String username;
    String password;
}
```

图 8 URL 读取

⑤Bean 依赖注入失败问题

Bean 是 springboot 的一大特性，用于降低耦合度，出现这种报错的原因是没有给对象设置 Bean，需要在定义对象前加入@Autowired 等注解。

⑥mybatis 配置问题

mybatis 配置也出现也许多次报错，不一一概述，下面贴上正确的配置截图，

作为记录：

```
spring.application.name=httpserver
server.port=2020
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.username=root
spring.datasource.url=jdbc:mysql://localhost:3306/geniusjyj?useSSL=false&serverTimezone=UTC&useUnicode=true&charact
spring.datasource.password=sheep031020

# spring.datasource.url=jdbc:mysql://localhost:3306/your_database?useSSL=false&serverTimezone=UTC
# spring.datasource.username=your_username
# spring.datasource.password=your_password
# spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
# MyBatis ??
mybatis.mapper-locations=classpath:mapper/userMapper.xml,classpath:mapper/firmMapper.xml
mybatis.type-aliases-package=com.example.httpserver
```

图 9 mybatis 配置

⑦xml 文件的位置问题

报错总提示找不到 xml 文件，发现如果在配置中设置的是“classpath:”的话；系统默认的路径是非常严格的，首先需要 resource 文件必须放在 src 下面的 main 下面，和 java 文件夹并列；其次 xml 文件必须存放在 resource 下面，否则就会出现找不到的报错，最后重新对文件结构进行排布，问题得以解决。

2. 前端

①无法解析后端传来的响应体

前端一直显示无法解析响应体，而且网站的显示是后端已成功给出响应体，并且响应体就是数组类型，完全与前端代码吻合，但前端的控制台输出始终显示后端传入响应体为 undefined，且无法获取数据。这是前端调试中遇到的最重要、最大、且最有意义的问题——异步问题。出现问题的原因是前端连接后端且从后端获取公司名称和坐标数据需要的时间有些长，因此还没等数据获取完全，就开始执行下一句处理数据的代码，因此处理数据的得到的 response 响应体根本是不完全的响应体，当然处理不成。需要在前端请求数据代码后加入.then(response=>{XXXXX})，其中 XXXXX 为必须在数据获取完全后才能执行的代码，也就是数据处理代码；或者也可以用 async 设置异步函数，并在必须执行完才能接着执行的代码前加入 await。

②npm 依赖下载失败

报错一直显示找不到 npm 的依赖，但在 idea 中直接下载下载失败，在 idea 命令行下载也失败，在 cmd 下载也失败。原因是 npm 的下载要求权限非常高，

必须使用 win+X 调出管理员模式的 cmd，才能下载，而且必须把依赖下载到项目的根目录下，同时要将镜像文件转换成官方版本，否则也会报错证书过期或没有效力等问题。

③高德地图 api 的加载问题

这个地方有许多细小的报错，首先是地图加载不出，发现是地图尺寸设置的问题；其次是第一次加载地图总是显示不出，一定要第二次加载才能显示出来，经调试发现是在配置中重复定义的问题；最后又发现在使用 text 组件是提示不识别 AMAP，经调试发现是只有访问 API 时用 then 传入的 AMAP 才是有效力的，一旦在 then 之外使用 AMAP 且没有传出，那就不会识别，解决方案是将代码加到 then 的内部。

八、思考与总结

这个项目由我们团队三个人共同完成，涉及前端和后端的开发工作。项目采用了当前流行的技术栈，后端基于 Spring Boot 框架，前端则采用 Vue.js 框架，结合 Element UI 库进行界面构建。整体上，项目展现了现代化 Web 应用开发的特点，包括前后端分离、模块化开发、RESTful API 设计等。

杨皓翔同学主要负责后端开发，项目所构建的后端服务是基于 Spring Boot，这是一个广泛使用的 Java 基础框架，用于创建独立、生产级的基于 Spring 框架的应用。通过@SpringBootApplication 注解，项目整合了 Spring Boot 的核心功能，包括自动配置、起步依赖等。后端服务提供了用户登录验证和企业信息展示的 API，使用@RestController 注解定义了 RESTful 风格的控制器，并通过 @CrossOrigin 支持跨域请求，使得前端 Vue.js 应用能够从不同的端口安全地访问后端服务。数据访问层使用 MyBatis 框架，通过定义 Mapper 接口与 XML 配置文件实现 SQL 操作的映射，简化数据库交互。实体类使用@Data 注解的 Lombok 库来自动生成 getter 和 setter 方法，减少模板代码。配置文件 application.properties 用于集中管理数据库连接、MyBatis 配置等参数，使得应用配置更加灵活和可维护。

赵慧琳同学主要负责前端开发，项目前端采用 Vue.js 作为前端框架，这是一个轻量级且高效的 JavaScript 框架，用于构建用户界面。Vue.js 的组件化特性使

得开发大型单页应用变得简单，每个组件负责视图的一部分，易于维护和复用。使用 Vue Router 管理页面路由，实现 SPA（单页应用）的导航功能，使得用户在不同页面间切换无需重新加载整个页面。状态管理通过 Vuex 实现，集中存储和管理所有组件的状态，保证状态的一致性和可预测性。Element UI 库提供了一套完整的 UI 组件，加速了界面的开发进程，并保证了界面的美观和一致性。axios 库用于与后端进行 HTTP 通信，支持 Promise API，使得异步请求更加简洁和易于管理。全局样式通过 CSS 文件统一管理，保持了界面风格的一致性。

贾羽佳同学主要负责项目架构和前后端整合，确保前后端的无缝对接。在项目正式开始之前，对项目的整体架构作出一个清晰的规划：确定使用何种技术栈、设计 RESTful API 的规范、以及如何实现前后端的分离。在整合过程中，确保数据能够正确地在前后端之间传输，同时处理前端状态管理与后端数据同步的问题，以保证用户界面能够实时反映后端数据的变化，除此之外，还负责调试和解决前后端交互中出现的问题，包括 API 响应时间、数据格式转换、错误处理等。

在这个项目中，团队成员之间的合作至关重要。后端开发者需要深入理解业务逻辑和数据库设计，前端开发者则需要关注用户体验和界面设计。在这个项目中，杨皓翔同学通过 Spring Boot 和 MyBatis 提供了稳定的数据服务，赵慧琳同学则利用 Vue.js 和 Element UI 构建了友好的用户界面。这种分工合作，不仅提高了开发效率，也确保了项目的质量和性能。团队成员之间的沟通和协作是项目成功的关键，贾羽佳同学的负责内容确保了团队沟通的有效性和协调性。

尽管项目已经取得了一定的成果，但仍有一些地方可以改进。例如，前后端的接口文档可以更加详细，以便于团队成员之间的沟通和协作。此外，项目可以考虑引入自动化测试，以提高代码质量和减少人为错误。在安全性方面，用户登录验证可以进一步强化，比如引入 JWT（JSON Web Tokens）等安全机制，以保护用户数据和接口安全。在项目架构方面，可以考虑引入微服务架构，将不同的业务模块拆分成独立的服务，提高系统的可维护性和可扩展性。此外，项目可以考虑容器化部署，使用 Docker 和 Kubernetes 等工具，提高部署的效率和可靠性。