# 一、文件结构

## 1. 前端项目结构

xmlyNew 项目名称

├── index.html

├── package-lock.json 项目需要的依赖和对应的版本文件

├── package.json 项目需要的依赖和对应的版本文件

├── public

│   └── favicon.ico 项目图标log

├── src

│   ├── App.vue 主的Vue文件 里面是路由跳转

│   ├── assets

│   │   └── logo.png。项目需要的logo图片

│   ├── components 组建

│   │   └── HelloWorld.vue helloword组建（创建项目自带的）

│   ├── index.css 项目需要的css

│   ├── main.js 引入各种需要的文件

│   ├── router 项目的路由管理

│   │   ├── index.js 填写页面的路由跳转

│   │   └── router.js 填写页面的路由跳转

│   └── view 项目的页面文件

│   ├── Login.vue 登陆页面

│   ├── book

│   │   └── list.vue 图书管理页面

│   ├── document

│   │   └── list.vue 论文管理页面

│   ├── index

│   │   └── index.vue 项目首页页面

│   └── statistics

│   └── list.vue 项目统计管理页面

└── yarn.lock yarn、

10 directories, 17 files

## 2. 后端项目结构

list [error opening dir]

java\_back\_end

├── E:\maven\maven\_repository maven依赖

│   └── org

│   └── springframework

│   └── boot

│   └── spring-boot-starter-parent

│   └── 2.2.1.RELEASE

├── HELP.md

├── hhu.sql 数据库的sql文件

├── java\_back\_end.iml

├── mvnw

├── mvnw.cmd

├── pom.xml

├── src

│   ├── main

│   │   ├── java

│   │   │   └── com

│   │   │   └── sjz

│   │   │   └── cloudcourse

│   │   │   ├── JavaBookManager.java 主启动类

│   │   │   ├── common 封装了需要的各种基本的Java类

│   │   │   │   ├── AjaxResult.java

│   │   │   │   ├── BaseContext.java

│   │   │   │   ├── Constant.java

│   │   │   │   ├── PageUtils.java

│   │   │   │   ├── Query.java

│   │   │   │   ├── R.java

│   │   │   │   ├── RRException.java

│   │   │   │   └── SQLFilter.java

│   │   │   ├── config 项目需要的各种配置类

│   │   │   │   ├── LoginConfig.java

│   │   │   │   ├── MyBatisConfig.java

│   │   │   │   ├── MyMetaObjectHandler.java

│   │   │   │   ├── RedisConfig.java

│   │   │   │   └── WebMvcConfigure.java

│   │   │   ├── controller 请求类

│   │   │   │   ├── BookController.java 图书相关的请求类

│   │   │   │   ├── BuyerController.java 关联表的请求类

│   │   │   │   ├── DocumentController.java 文档相关的请求类

│   │   │   │   ├── UploadController.java 上传相关的请求类

│   │   │   │   └── UserController.java 用户相关的请求类

│   │   │   ├── entity

│   │   │   │   ├── Book.java 图书实体类

│   │   │   │   ├── Buyer.java 图书用户管理实体类

│   │   │   │   ├── Document.java 文档实体类

│   │   │   │   ├── Upload.java 文档用户实体类

│   │   │   │   └── User.java 用户实体类

│   │   │   ├── mapper

│   │   │   │   ├── BookMapper.java 图书Mapper文件

│   │   │   │   ├── BuyerMapper.java 图书用户管理Mapper文件

│   │   │   │   ├── DocumentMapper.java 文档Mapper文件

│   │   │   │   ├── UploadMapper.java 用户文档管理Mapper文件

│   │   │   │   ├── UserMapper.java 用户Mapper文件

│   │   │   │   └── xml

│   │   │   │   ├── BookMapper.xml 用户xml文件

│   │   │   │   ├── BuyerMapper.xml 用户图书管理表xml文件

│   │   │   │   ├── DocumentMapper.xml 文档xml文件

│   │   │   │   ├── UploadMapper.xml 文档用户xml文件

│   │   │   │   └── UserMapper.xml 用户xml文件

│   │   │   ├── service

│   │   │   │   ├── BookService.java 图书业务层代码文件

│   │   │   │   ├── BuyerService.java 用户图书管理表文件

│   │   │   │   ├── DocumentService.java文档务层代码文件

│   │   │   │   ├── Impl

│   │   │   │   │   ├── BookServiceImpl.java图书业务层实现类

│   │   │   │   │   ├── BuyerServiceImpl.java 用户图书关联表实现类

│   │   │   │   │   ├── DocumentServiceImpl.java 文档业务层实现类

│   │   │   │   │   ├── UploadServiceImpl.java 用户文档关联表实现类

│   │   │   │   │   └── UserServiceImpl.java 用户实现类

│   │   │   │   ├── UploadService.java 用户文档管理表接口

│   │   │   │   └── UserService.java 用户业务层接口

│   │   │   ├── utils 放项目需要的工具类

│   │   │   └── vo

│   │   └── resources

│   │   ├── application.properties 项目的配置文件

│   │   └── application.yaml 项目的配置文件

│   └── test

│   └── java

│   └── com

│   └── sjz

│   └── cloudcourse

│   ├── CloudClassApplicationTests.java 测试类

│   └── service

│   └── impl

│   └── DocumentServiceImpl.java 测试类

└── target

├── classes

├── generated-sources

│   └── annotations

├── generated-test-sources

│   └── test-annotations

└── test-classes

37 directories, 54 files

# 二、使用说明

## 1. 前言

如果直接打开后端项目可能会出现大量报错，只要先运行前端项目，再打开后端项目，就不会出现报错了

## 2. 前端项目

0. 需要的环境 nodejs 直接搜索在官网上安装 <https://nodejs.org/en/>，版本最好一致



1. 使用vscode或者idea打开xmlyNew 文件夹
2. 在命令行输入 npm i 安装相关的依赖



1. 输入npm run dev运行项目



1. 结果如图，运行后端项目之后打开图中的网址，就是界面。打开后端项目时，前端项目不要停止运行，用两个窗口分别打开

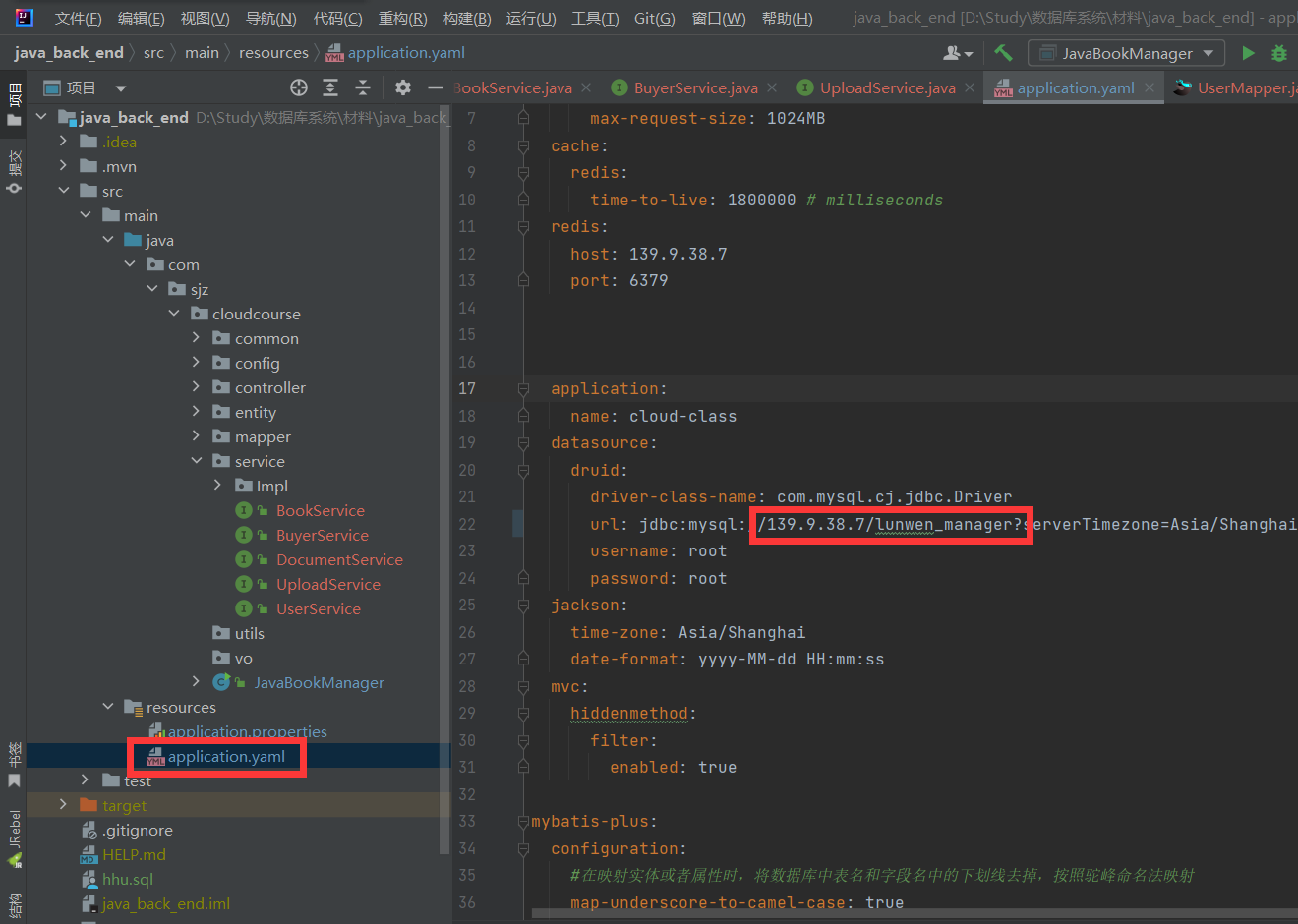


## 3. 后端项目

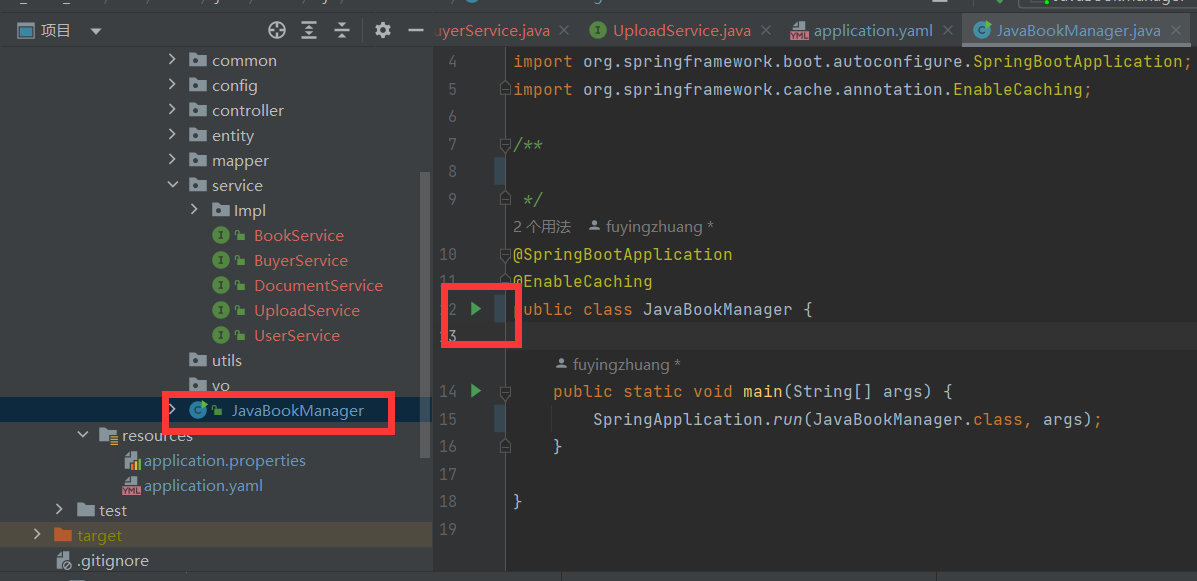
0. 需要的环境 jdk1.8 mysql数据库 maven

1. 使用vscode或者idea打开java\_back\_end 文件夹

2. 在这个位置配置自己的数据库环境，现在文件里的数据库可以用但是顶不住百万级数据



3， 在这里运行项目



4. 运行结果

