# 1 前端

## 1.1 环境条件

安装了node.js v16.20.2、npm 8.19.4、vue，使用vue ui命令

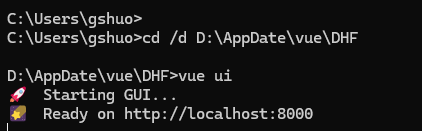
[参考网址](https://blog.csdn.net/qq_39915083/article/details/107931632)

## 1.2 创建vue项目

Cmd命令vue ui 打开图形化界面，以管理打开cmd.

？不能在制定目录创建项目

#使用vue ui命令前，切换到指定目录



？安装elementui插件，是插件不是依赖

#可使用Element组件

## 1.3 使用IDEA前端编码

HTML--<template> <div> </div> </template>，全部在一个div内

Js--<script> </script>

Css--<style> </style>

## 1.4 Element组件

[官网](https://element.eleme.cn/#/zh-CN/component/installation)

创建组件、router动态加载、<router-view>、<router-link>

注意：Element UI（Vue 2 专用），在 Vue 3 项目中，不能使用 element-ui，而应该使用 element-plus。

### 1.4.1 布局容器：

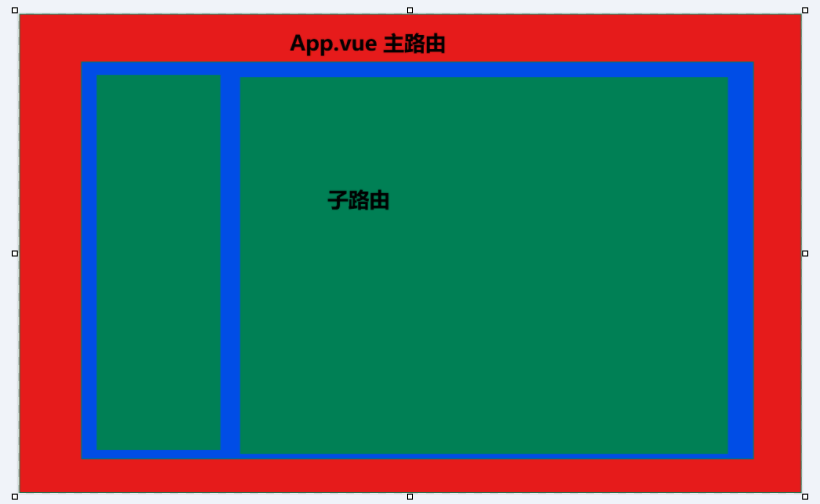
布局中使用了 <el-header>、<el-aside> 和 <el-main>，这是典型的「横向布局 + 顶部栏」。但是这里的问题是：

缺少嵌套容器结构（即：header + container（aside + main）结构）

或者说：el-aside 和 el-main 应该包裹在另一个 el-container 中



### 1.4.2 子路由，动态构建左侧菜单



首先从App.vue进入<router-view>,

子路由通过index,js文件内const routes[ ]的children设置。

动态构建菜单：<v-for>

### 1.4.2 menu与router的绑定

<el-menu>标签添加router属性、

在页面中添加<router-view>标签，它是一个容器、

<el-menu-item>标签的Index值就是要跳转的router。

### 1.4.3 假数据测试

## 1.5 添加axios组件

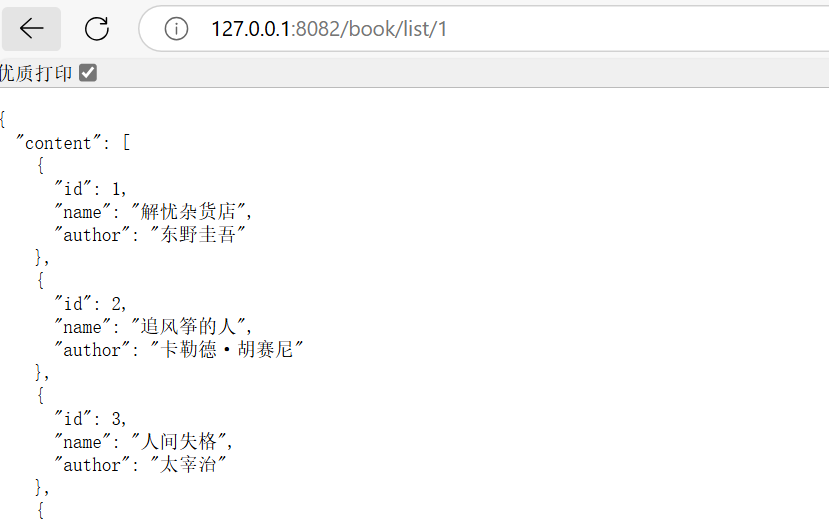
以管理员身份打开IDEA，vue add axios

替换假数据，created()方法。

# 2 后端

[参考网站](https://blog.csdn.net/qq_39915083/article/details/107931130)

后端数据，在浏览器可见，JSON格式



# 3 前后端数据传输

前后端通过网址传递参数，数据。

## 3.1 创建项目

依赖：spring Data JPA，~~LomBok~~表达式，MySQL驱动和springWeb启动器

简单配置：将application.properties改为yml后缀，数据库信息

准备数据库

## 3.2 创建实体类

针对数据库结构创建实体类，

实体类，只包含字段（属性），用于封装数据（相当于数据容器）。

@Entity 与表名进行绑定

@Data 是 **Lombok** 提供的一个注解，包括@ToSpring @Setter @Getter等方法，简化代码。

@Id主键，数据索引依据，唯一值。

？@Data不管用，需要@Override（重写）ToString方法，构造getter和setter等方法。

## 3.3 创建repository

使用了springData Jpa。只需要将实体类和数据库对象绑定起来。

## 3.4 编写controller

首先进行一下单元测试，测试结果是否符合预期。

编写cotroller，结果可以在浏览器显示

## 3.5 解决跨域

Config。。。。。

# 4 数据库操作

## 4.1 查询

## 4.2 添加

通过表单Form添加数据，

通过v-model将表单项的值与数据对象的属性进行绑定，

后端—controller，@PostMapping，操作数据库

前端—axios.post(‘’,this.ruleForm)…

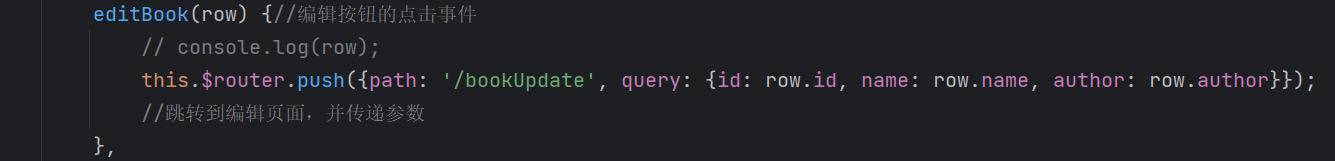
注意添加数据，需要设置实体类主键id**自增**，不然报错。

同时添加数据时，不设置Id具体值。

同时，MySql数据库中的id也要设置自增！！。

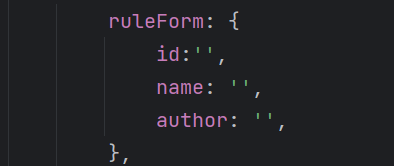
## 4.3 修改

通过表单修改，

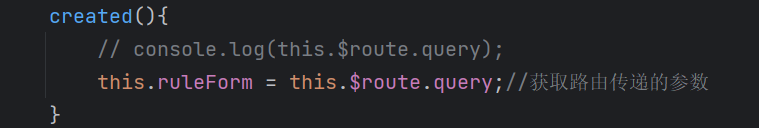


1.跳转路由；2.传递参数；通过查询id修改

修改表单以添加表单为模版，增加了Id属性；



Created(),初始值与原来一样。



其他不变，修改的功能与添加数据功能相似，区别在于id,

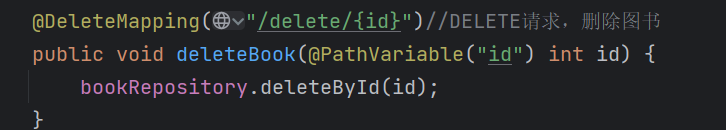
在原有Id上，=修改数据；

不设置Id，=添加数据。

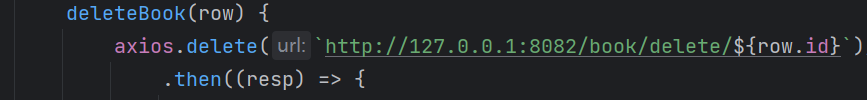
## 4.4 删除

通过ID删除，

后端.deleteById



前端



# 5 Echart绘图

## 5.1 安装ECharts

终端命令：

npm install echarts@5.4.3

main.js中：

import \* as echarts from 'echarts'  
*Vue*.prototype.$echarts = echarts

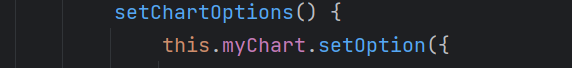
## 5.2 Echart绘制

[资源](https://echarts.apache.org/examples/zh/index.html#chart-type-line)

从Echart官网下载需要的图型代码，

1）option = {}🡺拷到this.myChart.setOption（）里面;

2）其他补充代码🡺拷到setChartOptions()内，且在this.myChart.setOption（）外面。



## 5.3 链接后端数据

针对前端结果创建数据对象以便对接。（可以通过实体类）

注意数据类型一致！！

# 6 前端控制后端的方法运行

前端负责交互，后端执行逻辑运算



## 6.1 vo类（全称：**Value Object**，值对象）

与Entity实体类功能相似，Entity接收数据库的数据，vo类接收前端的数据。

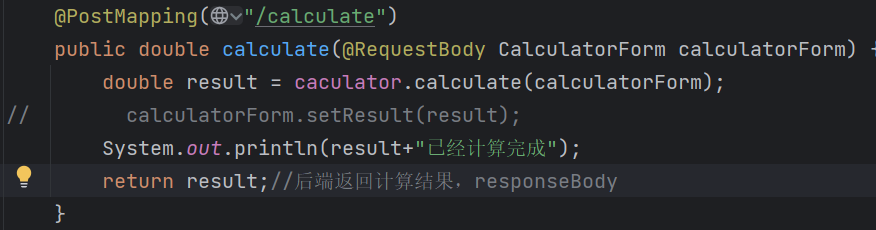
主要作用是用来封装和传输数据。

封装前端传来的数据，当前端发送请求（如表单提交或接口调用）时，可以用VO类来统一封装这些请求参数，便于后续在业务层进行处理。

## 6.2 后端响应



前端发送请求，传递参数，后端程序运行并返回resp



此处后端返回了一个double值；

## 6.3 JSON格式

responseBody—指后端的数据；.then((resp)=>

requestBody--接收前端传入的json数据

# 7 前后端合为一个项目（分离开发完之后）

## 7.1 打包前端项目

使用 npm run build 打包，生成的文件会放到 dist/ 目录。

打包后，你会得到一个包含 index.html、js、css 文件的静态文件夹（dis）

## 7.2 将前端静态资源整合到后端项目

如果你的后端是**基于 Spring Boot** 开发的，可以将打包后的前端静态文件（dist 目录）复制到后端项目的 resources/static/ 目录中。Spring Boot 会自动为你提供静态资源的支持。

## 7.3 修改后端路由配置

* 确保后端 API 路由不会和前端静态文件路径冲突。如果有冲突，通常需要调整 API 路径或者将后端 API 路由加一个前缀（比如 /api）。

例如：

* /api/user → 后端接口
* /user → 前端的页面

### 7.3.1 前后端分离

前端请求 http://localhost:8080/api 时，实际会代理到 http://localhost:8083，后端服务的 API 就可以正常访问了。

### 7.3.2 合为一个项目之后

前后端的会合并，采用的是后端端口。访问前端页面应该换成后端设置的端口。

## 7.4 统一部署

**Spring Boot**：直接打包为一个 .jar 文件，包含了前端静态资源和后端服务，部署到服务器后，前端和后端会在同一个服务中运行。

打包命令：mvn clean package

生成的 jar 文件会包含：

* 所有后端代码；
* 所有前端静态文件

## 7.5 运行.jar文件

所在目录，cmd，

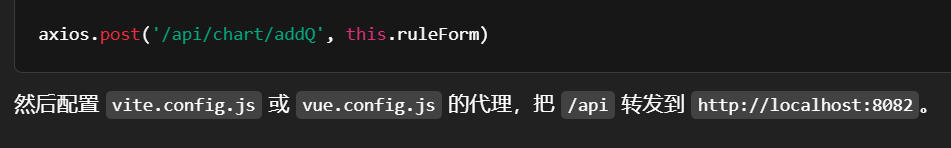
命令：java -jar myapp-1.0.0.jar

**！！！不要使用 localhost 访问，使用 127.0.0.1：8082（后端端口！！）**

因为前端请求用的是：

Localhost是主机名，127.0.0.1是IPv4地址。

可以把接口路径写成相对路径，不要写死？？还需要其他配置：



### 7.5.1双击运行的 **bat 脚本**文件

1）创建run.bat文件

在任意位置创建一个新的文本文件，并将其命名为 run.bat。

2）编辑 run.bat 文件内容

@echo off

echo Starting Spring Boot application...

:: 设置 JAVA\_HOME (根据你本地 JDK 路径修改)

set JAVA\_HOME=C:\Program Files\Java\jdk-20

:: 设置应用的 JAR 文件路径

set JAR\_PATH=D:\AppDate\vue\DHF\dhf\_springboot\target\dhf\_springboot-0.0.1-SNAPSHOT.jar

:: 运行 Spring Boot 应用（使用 start 命令启动新窗口，防止阻塞 bat 脚本执行）

echo Running application...

start "" "%JAVA\_HOME%\bin\java" -jar "%JAR\_PATH%"

:: 等待几秒，确保服务启动（可按需延长等待时间）

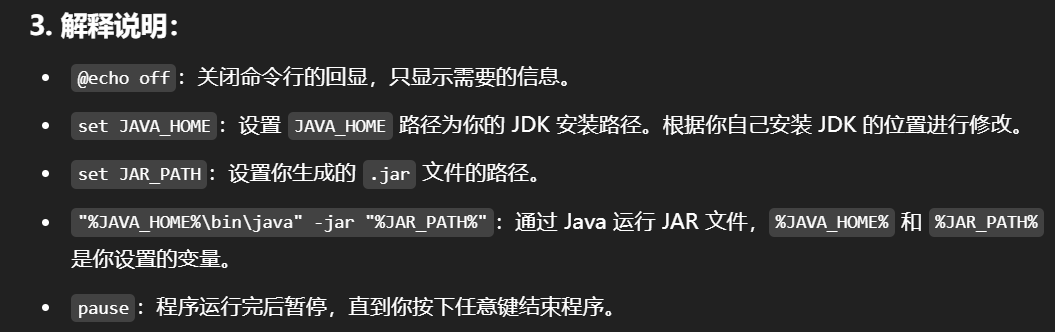
timeout /t 5 >nul

:: 自动打开网页（根据你后端监听的地址改）

start http://127.0.0.1:8082

echo Application started. Browser opened.

pause



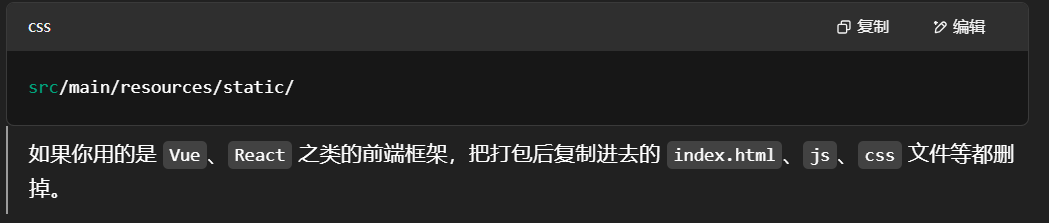
4）运行 run.bat 文件

双击 run.bat 文件，它将自动启动 Spring Boot 应用。

## 7.6 将合并的项目分离

将**已合并的前后端项目再分离**，本质上是把静态资源从后端中分离出来，重新用前后端独立部署的方式来运行。

删除后端中已集成的前端资源



# 8 让别人可以通过浏览器访问

## 8.1 本地局域网联网使用

## 8.2 公网联网使用

# 9 Git版本控制

git下载与安装配置。[Git在IDEA中的使用](https://blog.csdn.net/mucaoyx/article/details/98476174)

首先，在Github中新建一个仓库，建成后会看到以下命令。



## 9.1 提交到Git本地仓库

1. git init：初始化一个新的 Git 仓库。
2. git add README.md：将 README.md 文件添加到暂存区。(git add .：添加所有文件)
3. git commit -m "first commit"：提交更改并附加提交信息 "first commit"。

## 9.2 推送到Github远程

1. git branch -M main：将默认分支名称更改为 main（如果你的默认分支是 master，这一步是必需的）。（main和master仅仅是命名的不同，功能是一样的）
2. git remote add origin https://github.com/gaoshuo888/HiGithub.git：将远程仓库地址添加为 origin。
3. git push -u origin main：将本地的 main 分支推送到远程仓库，并设置为默认的远程分支。

执行完这段命令后，你的本地仓库就会成功关联到 GitHub 上的仓库并且将 README.md 文件推送过去。

## 9.3 解决可能的权限问题--使用SSH推送方式

第一步：生成 SSH 密钥对

ssh-keygen -t rsa -b 4096 -C "gaoshuo888@gmail.com"

第二步：添加 SSH 公钥到 GitHub

打开公钥文件，复制内容到GitHub 账户 SSH 设置页面

notepad "$env:USERPROFILE\.ssh\id\_rsa.pub"

第三步：将 Git 远程地址换成 SSH 模式

git remote set-url origin git@github.com:gaoshuo888/HiGithub.git

第四步：测试连接并推送

ssh -T git@github.com

git push -u origin master

一旦完成了以上配置，以后这个电脑推送 GitHub 就不再需要每次输账号密码，稳定又方便。

# 10 一些注意事项

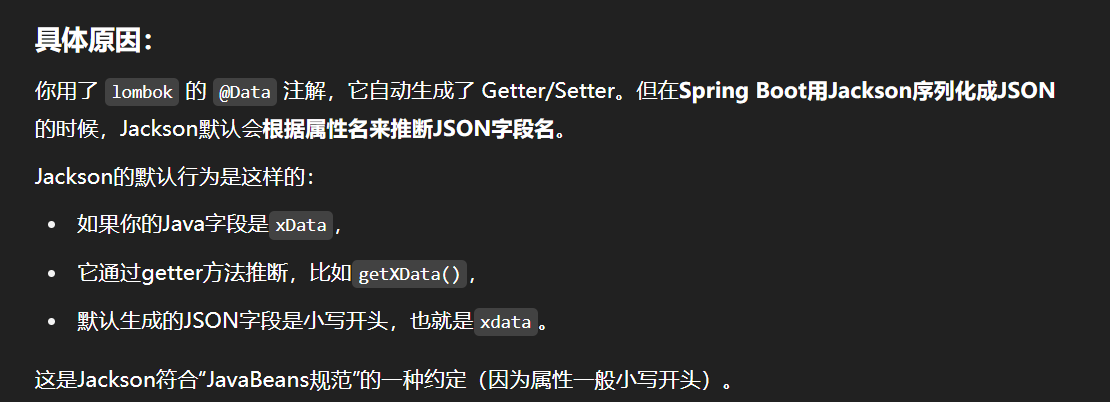
## 10.1 大小写

Java中的类名：Book 🡪 自动对应数据库表名：book（实体类，不需要@Table）；

数据库：parameter\_name🡪自动对应java属性：parameterName （下划线与大写）；

Java中的属性规范为小驼峰，因此若有属性E1，后端输出为e1；

Java:xData 🡪 自动生成getXData 🡪 JSON:xdata



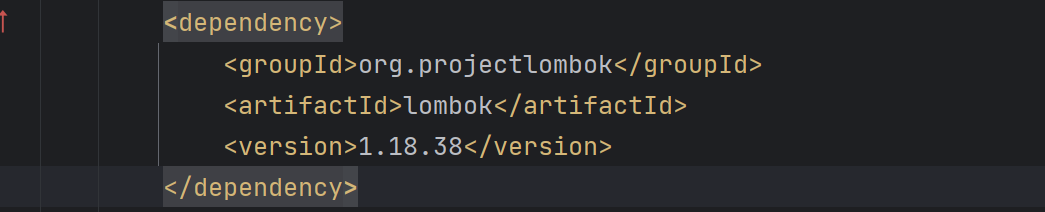
## 10.2 [@Data注解不管用](https://blog.csdn.net/weixin_45875070/article/details/145697952?spm=1001.2101.3001.6650.6&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7EYuanLiJiHua%7ECtr-6-145697952-blog-145074192.235%5Ev43%5Epc_blog_bottom_relevance_base1&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7EYuanLiJiHua%7ECtr-6-145697952-blog-145074192.235%5Ev43%5Epc_blog_bottom_relevance_base1&utm_relevant_index=11)

原因：在创建项目时添加lombok依赖的话，会自动引入一个插件（Spig官方进行了升级，之前没有，新增的)，把这个插件删了就可以了

1.第一步，删除（√）



2.第二步，不能按照网页的方法，需要在mevan官网更换依赖。



## 10.3 实体类对应的数据类型

实体类对应一个对象，而不是一个可以包括多个对象的数组

{

"id": 1,

"name": "E1",

"value": 0.1

}

数组：List

List<Parameter>

## 10.4 其他

（1）子视图—父组件中使用<router-view>占位，在index.js文件中设置好children[ ]，可以使得子组件在父组件的<router-view>位置中展示。