前后端分离

前后端分离就是将一个应用的前端代码和后端代码分开写，为什么要这样做?

如果不使用前后端分离的方式，会有哪些问题?

传统的 Java Web 开发中，前端使用 JSP 开发，JSP 不是由后端开发者来独立完成的。

前端---》HTML 静态页面 ---〉后端 ---》JSP

这种开发方式效率极低，可以使用前后端分离的方式进行开发，就可以完美地解决这一问题。

前端只需要独立编写客户端代码，后端也只需要独立编写服务端代码提供数据接口即可。

前端通过 Ajax 请求来访问后端的数据接口，将 Model 展示到 View 中即可。

前后端开发者只需要提前约定好接口文档（URL、参数、数据类型...），然后分别独立开发即可，后端用postman进行测试,前端可以造假数据进行测试，完全不需要依赖于后端，最后完成前后端集成即可，真正实现了前后端应用的解耦合，极大地提升了开发效率。

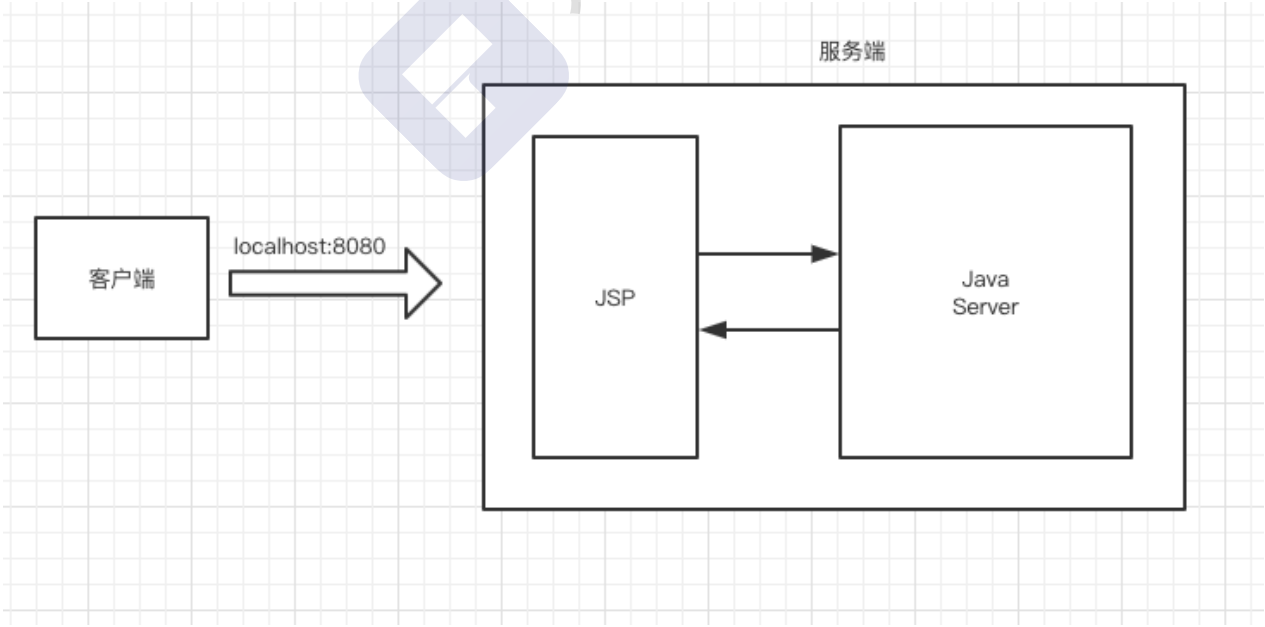
单体---》前端应用 + 后端应用

前端应用：负责数据展示和用户交互。

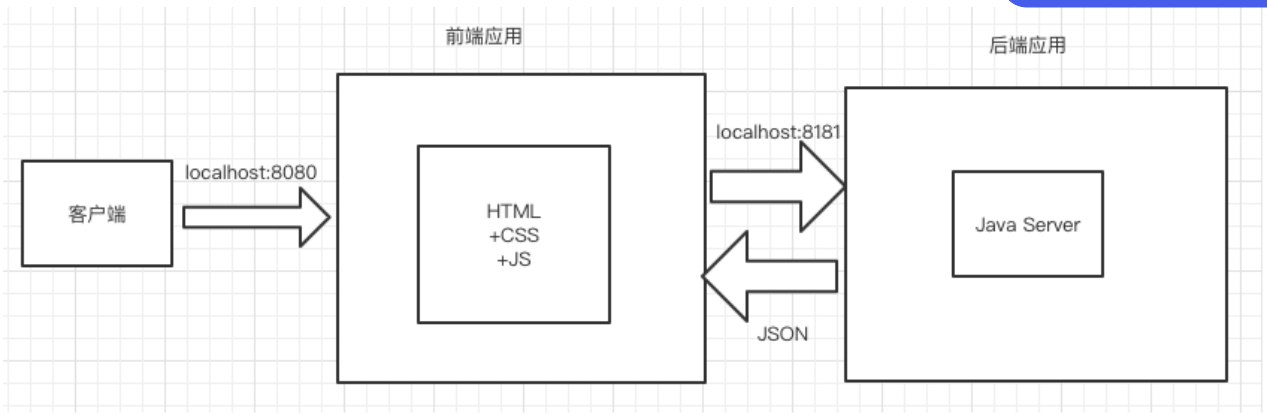
后端应用：负责提供数据处理接口。

前端 HTML ---》Ajax ---〉RESTful 后端数据接口。

传统的单体应用



前后端分离的结构



前后端分离就是将一个单体应用拆分成两个独立的应用，前端应用和后端应用以 JSON 格式进行数据交互。

实现技术

Spring Boot + Vue

使用 Spring Boot 进行后端应用开发，使用 Vue 进行前端应用开发。

Vue + Element UI

Vue 集成 Element UI

Element UI 后台管理系统主要的标签：

- el-container：构建整个页面框架。
- el-aside：构建左侧菜单。
- el-menu：左侧菜单内容，常用属性：

  :default-openeds：默认展开的菜单，通过菜单的 index 值来关联。

  :default-active：默认选中的菜单，通过菜单的 index 值来关联。
- el-submenu：可展开的菜单，常用属性：

  index：菜单的下标，文本类型，不能是数值类型。
- template：对应 el-submenu 的菜单名。
- i：设置菜单图标，通过 class 属性实则。
  - el-icon-messae
  - el-icon-menu
  - el-icon-setting
- el-menu-item：菜单的子节点，不可再展开，常用属性：

  index：菜单的下标，文本类型，不能是数值类型。

Vue router 来动态构建左侧菜单

- 导航1

    - 页面1
    - 页面2
- 导航2
    - 页面3
    - 页面4

menu 与 router 的绑定

1、 标签添加 router 属性。

2、在页面中添加 标签，它是一个容器，动态渲染你选择的 router。

3、 标签的 index 值就是要跳转的 router。

Element UI 表单数据校验

定义 rules 对象，在 rules 对象中设置表单各个选项的校验规则

```
rules: {
  name: [
    { required: true, message: 'error', trigger: 'blur' },
    { min: 3, max: 5, message: '长度在 3 到 5 个字符', trigger: 'blur' }
  ]
}
```

required: true, 是否为必填项

message: 'error', 提示信息

trigger: 'blur'，触发事件