浙江大学实验报告

课程名称：**B/S 体系软件设计** 实验类型：个人上机实验

实验项目名称：物联网应用网站

学生姓名：聂俊哲 专业：软件工程 学号：3180103501

同组学生姓名：无 指导老师：胡晓军

实验地点：曹西 103 实验日期：2021 年 6 月 20 日

物联网应用网站个人开发体会和小结

本次实验中，我前端采用了 React + antd 组件，Mqtt 服务器采用了 mosquitto，后端使用 Spring Boot 框架完成了物联网网站的应用开发。在本次开发中，该应用实现了要求的所有功能：

1. 实现用户注册、登录功能，用户注册时需要填写必要的信息并验证，如用户名、密码要求在 6 字节以上，email 的格式验证，并保证用户名和 email 在系统中唯一，用户登录后可以进行以下操作。
2. 提供设备配置界面，可以创建或修改设备信息，包含必要信息，如设备 ID、设备名称等
3. 提供设备上报数据的查询统计界面
4. 提供地图界面展示设备信息，区分正常和告警信息，并可以展示历史轨迹
5. 首页提供统计信息（设备总量、在线总量、接收的数据量等），以图表方式展示（柱状体、折线图等
6. 样式适配手机端，能够在手机浏览器/微信等应用内置的浏览器中友好显示。

Mosquitto 介绍：

Mosquitto 是一款实现了消息推送协议 Mqtt 的开源消息代理软件，提供轻量级的，支持可发布/可订阅的的消息推送模式，使设备对设备之间的短消息通信变得简单，比如现在应用广泛的低功耗传感器，手机、嵌入式计算机、微型控制器等移动设备。

搭建该服务器，首先需要下载安装该软件，然后进行配置。之后在服务器内部架设线程，订阅指定的 Topic，获取相应的数据并进行处理。

Spring Boot 框架介绍：

SpringBoot 是基于 Spring 的一个新型框架，基于约定优于配置的思想，可以让开发人员不必在配置与逻辑业务之间进行思维的切换，全身心的投入到逻辑业务的代码编写中，从而大大提高了开发的效率，一定程度上缩短了项目周期。其特点有:

1. 为基于 Spring 的开发提供更快的入门体验
2. 开箱即用，没有代码生成，也无需 XML 配置。同时也可以修改默认值来满足特定的需求
3. 提供了一些大型项目中常见的非功能性特性，如嵌入式服务器、安全、指标，健康检测、外部配置等

项目的难点和亮点介绍：

1. 使用 Redis 来存储 Token 并结合拦截器进行身份认证

Redis 是一个开源的、基于内存的数据结构存储器，可以用作数据库、缓存和消息中间件，其具有读写快速的特点，因此当系统具有大量用户的时候，使用 Redis 存储 Token 能够极大地提升效率 。

Token 进行身份认证的流程：

- 客户端在登录时向服务端发起请求，服务端验证通过后，随机生成一个字符串 Token，将 Token 和该用户的序列化信息、有效期等信息一起存入到 Redis 中，并将 Token 返回给客户端。
- 客户端接收到响应后，从响应体中取出 Token，并存到 LocalStorage 中。
- 当客户端发起请求时，Axios 拦截器首先会检查用户的 LocalStorage 中是否具有 Token，如果没有并且该页面需要认证才可访问，则会强制跳转到登录界面;如果有，则将 Token 放到请求头中发送给服务端。
- 服务端接收到响应后，如果用户访问的接口需要登录态，则该请求会被 TokenInterceptor 拦截器拦截，拦截器从请求头中取出 Token 值并在 Redis 中查找对应的信息，如果查找成功则放行，否则返回错误码。

2. 使用 RESTful 风格的接口

REST(Representational State Transfer)REST 指的是一组架构约束条件和原则。如果一个架构符合 REST 的约束条件和原则，我们就称它为 RESTful 架构。

RESTful 架构应该遵循统一接口原则，统一接口包含了一组受限的预定义的操作，不论什么样的资源，都是通过使用相同的接口进行资源的访问。接口应该使用标准的 HTTP 方法如 GET，PUT 和 POST，并遵循这些方法的语义。如果按照 HTTP 方法的语义来暴露资源，那么接口将会拥有安全性和幂等性的特性，例如 GET 和 HEAD 请求都是安全的， 无论请求多少次，都不会改变服务器状态。 而 GET、HEAD、PUT 和 DELETE 请求都是幂等的，无论对资源操作多少次， 结果总是一样的，后面的请求并不会产生比第一次更多的影响。

RESTful 接口具有如下优点:

- 前后端分离，减少流量
- 安全问题集中在接口上，由于接受 json 格式，防止了注入型等安全问题
- 前端无关化，后端只负责数据处理，前端表现方式可以是任何前端语言

- 前端和后端人员更加专注于各自开发，只需接口文档便可完成前后端交互， 无需过多相互了解

3. 采用局部加载的方式提高效率

当需要从数据库查询的表有上万条记录的时候，一次性查询所有结果会变得很慢，特别是随着数据量的增加特别明显，这时需要使用分页查询。

在 Device Message 查询页面，查询设备的消息时，并不是将所有的消息一次性全部查询加载而是按需加载。当用户点击翻页时才会向后端发起请求，后端根据页数在数据库中利用 limit 10 offset 进行查询，将查询到的对应的 10 条数据返回。这样能够减少请求的数据量，提高服务器的性能。

4. 使用 React.memo()来优化函数组件的性能

组件是构成 React 视图的一个基本单元。有些组件会有自己本地的状态(state)，当它们的值由于用户的操作而发生改变时，组件就会重新渲染。在一个 React 应用中，一个组件可能会被频繁地进行渲染。这些渲染虽然有一小部分是必须的，不过大多数都是无用的，它们的存在会大大降低我们应用的性能

为了避免 React 组件的无用渲染，我们可以实现自己 shouldComponent-Update 生命周期函数。当 React 想要渲染一个组件的时候，它将会调用这个组件的 shouldComponentUpdate 函数，这个函数会告诉它是不是真的要渲染这个组件。

而 React.memo(...)是 React v16.6 引进来的新属性。它的作用和 React.PureComponent 类似，是用来控制函数组件的重新渲染的。React.memo(...) 其实就是函数组件的 React.PureComponent。React.memo 会返回一个纯化(purified)的组件 MemoFuncComponent，这个组件将会在 JSX 标记中渲染出来。当组件的参数 props 和状态 state 发生改变时，React 将会检查前一个状态和参数是否和下一个状态和参数是否相同，如果相同，组件将不

会被渲染，如果不同，组件将会被重新渲染。在本项目中，地图组件、图表组件等都使用了 React.memo()进行了优化，提升了性能。

5. 安全性

1.防 SQL 注入攻击

SQL 注入攻击是攻击者在 HTTP 请求中注入恶意的 SQL 命令，服务器用请求参数构造数据库的 SQL 命令时，恶意 SQL 被一起构造，并在数据库中执行。 SQL 注入攻击需要攻击者对数据库结构有所了解才能进行，攻击者可以通过错误回显、盲注等手段获取数据表结构信息。

本项目采取参数绑定的方式来防止 SQL 注入攻击:使用预编译，绑定参数的方法。攻击者的恶意 SQL 会被当作 SQL 的参数，而不是 SQL 命令被执行。

2. 数据校验

用户请求的数据会经过前后端双重校验，即在客户端进行一次校验(如果有问题直接拦截请求)，在服务端也会进行一次校验，防止用户通过模拟请求来绕过客户端数据检验直接提交非法数据。