

浙江大学

本科实验报告

课程名称: B/S 体系软件设计

实验名称: 物联网管理系统

姓 名: 王伟杰

学 院: 计算机学院

系: 软件工程

专 业: 软件工程

学 号: 3210106034

指导教师: 胡晓军

2023 年 11 月 4 日

浙江大学实验报告

课程名称: B/S 体系软件设计 实验类型: _____

实验项目名称: _____ 物联网管理系统

学生姓名: 王伟杰 专业: 软件工程 学号: 3210106034

同组学生姓名: 无 指导老师: 胡晓军

实验地点: _____ 实验日期: 2023 年 11 月 4 日

目录

1 文档介绍.....	5
1.1 编写目的.....	5
1.2 文档范围.....	5
1.3 读者对象.....	5
1.4 术语及缩写解释.....	5
2 项目介绍.....	6
2.1 项目说明.....	6
2.2 需求概述.....	6
2.3 条件与限制.....	7
3 需求分析.....	8
3.1 功能需求.....	8
3.2 性能需求.....	8
3.3 安全性需求.....	9
3.4 可维护性需求.....	10
5 总体设计.....	11
5.1 基本设计概念和流程处理.....	11
5.2 系统架构.....	13
5.2.1 前端架构.....	13
5.2.2 后端架构.....	14
5.3 技术介绍.....	15
5.3.1 前端技术框架.....	15
5.3.2 后端技术框架.....	16
5.3.3 MQTT 服务器.....	17
5.4 接口设计.....	17
5.4.1 用户信息接口.....	18
5.4.2 设备信息接口.....	20
5.4.3 设备消息接口.....	23

5.4.4 MQTT 服务器消息接口	23
6 数据库设计	24
6.1 概念结构设计 (E-R 图)	24
6.2 逻辑结构设计	24
6.3 物理结构设计	25
7 系统出错设计	27
7.1 出错信息	27
7.2 补救措施	27
7.3 系统维护设计	28
8 用户界面	28
8.1 登录页面	28
8.2 注册页面	29
8.3 Home 页面	30
8.3.1 新闻页面	30
8.3.2 个人信息页面	31
8.3.3 关于页面	31
8.4 工作页面	32
8.4.1 统计页面	32
8.4.2 设备页面	33
8.4.3 消息页面	34
8.4.4 设备修改页面	35
8.5 移动端适配	36
9 附录	40
9.1 项目进度安排	40
9.2 备注	40

1 文档介绍

1.1 编写目的

本文档描述软件产品功能设计说明书（SRS）的目的是：

- （1） 定义软件总体要求，作为用户和软件开发人员之间相互了解的基础；
- （2） 提供性能要求、初步设计和用户影响的信息，作为软件人员进行软件结构设计和编码的基础；
- （3） 作为软件总体测试的依据。

1.2 文档范围

物联网管理系统功能设计说明书旨在为物联网管理系统的具体实现提供一个完整、可行、详细的总体设计方案，并综合考虑用户需求、软件设计条件和限制，对该系统的功能目标和功能操作方式进行基本的描述。本文档将根据总体的系统架构图进行全面、细致的设计分析，其内容包括系统实现技术栈、系统功能模块设计、执行概念设计、流程处理设计、数据结构和类设计、数据库关系模型设计、数据库逻辑结构设计、数据库物理结构设计、系统内外接口设计、系统部署方案设计等。此外，本文档还将为系统出错提供一个基本的解决指导和简单说明。

本文档旨在为软件开发人员、软件测试人员、软件维护人员提供清晰完整的开发、测试、维护指导，帮助他们更好地理解系统总体运行生态、系统业务逻辑及其底层设计，并在此基础上进一步完成软件子系统的开发工作。

1.3 读者对象

本说明书的预期读者人员包括软件用户，项目经理、软件开发人员、软件测试人员、系统维护人员等。

1.4 术语及缩写解释

缩写、术语及符号	解释
IPO图	IPO图是输入/处理/输出图的简称，描述输入数据、对数据的处理和输出数据之间的关系。
状态图	状态图描绘一个系统或组件可能假设的状态，并且显示引起或导致一个状态切换到另一个状态的事件或环境。
类图	类图描述了一个软件系统中的类、接口、关系以及它们之间的静态结构。类图用于表示系统中的对象、类以及它们之间的关系，它提供了一种直观、可视化的方式来表示系统的结构。
MVC	MVC(Model-View-Controller)是一种设计模式，常用于开发Web应用程序。它将一个应用程序分为三个部分：模型、视图和控制器。其中，模型是应用程序的核心，表示应用程序的数据和业务逻辑；视图负责呈现数据给用户；控制器负责接收用户输入并调用相应的模型和视图

2 项目介绍

2.1 项目说明

- 项目名称：物联网管理系统
- 任务提出者：浙江大学 B/S 体系软件设计任课老师-胡晓军
- 开发者：浙江大学软件工程系学生-王伟杰
- 用户群：物联网系统管理员

2.2 需求概述

这个项目是一个物联网设备管理平台的网站开发，使用 Web 技术实现以下功能：

1. 搭建一个 mqtt 服务器，接收物联网终端模拟器的数据

2. 实现用户注册、登录功能，验证用户信息的合法性和唯一性
3. 提供设备配置界面，创建或修改设备信息，如设备 ID、设备名称、设备类型等
4. 提供设备上报数据的查询统计界面
5. 提供地图界面展示设备信息，区分正常和告警信息，部分设备类型的历史数据可以展示成历史轨迹
6. 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）
7. *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

2.3 条件与限制

- 服务器运行环境限制(受限于实验条件，使用 PC 作为服务器):
 - 操作系统：Windows 10 或以上版本，或 Linux 系统；
 - 硬件配置：具备 4 核 CPU、8GB 内存以上的配置，以支持系统的基本运行；
 - 软件环境：
Web 服务器：使用 Nginx 1.10 或以上版本作为 Web 服务器；
前端框架：使用 Vue.js 3.0 或以上版本作为前端框架；
后端框架：使用 Node.js 10.0 或以上版本作为后端框架；
数据库：使用 MySQL 5.7 或以上版本作为系统的数据库；
 - 网络环境：使用 HTTP 协议对系统进行传输。
- 开发过程代码规范限制：
 - 遵循一定的代码规范，包括变量命名、函数命名、代码注释等；
 - 避免使用过时、不安全或存在漏洞的代码库和组件；
 - 严格控制代码质量，避免出现内存泄漏、缓冲区溢出、代码注入等问题；
 - 采用版本控制工具，如 Git 等，进行代码管理和协作开发。
- 工具版本限制：
 - Web 服务器：Nginx 1.10 或以上版本；
 - 前端框架：Vue.js 2.0 或以上版本；
 - 后端框架：Node.js 10.0 或以上版本；
 - 数据库管理工具：MySQL；
 - 版本控制工具：Git。

3 需求分析

3.1 功能需求

本项目旨在创建一个物联网设备管理平台的网站，利用 Web 技术来实现以下功能：

1. MQTT 服务器搭建：搭建一个 MQTT 服务器，以接收来自物联网终端模拟器的数据。这将为物联网设备提供一个可靠的数据传输通道。
2. 用户注册与登录：实现用户注册和登录功能，确保用户信息的合法性和唯一性。这将为用户提供访问系统的权限和安全性。
3. 设备配置界面：提供一个用户友好的界面，允许用户创建或修改设备信息，包括设备 ID、设备名称、设备类型等。这有助于管理和跟踪设备。
4. 设备数据查询与统计界面：提供一个界面，用于查询和统计设备上报的数据。这将使用户能够分析设备的性能和状态。
5. 地图界面：展示设备信息并区分正常和告警信息。对于部分设备类型，历史数据可以以历史轨迹的形式展示在地图上，帮助用户更好地理解设备的运动轨迹和行为。
6. 首页统计信息：提供首页统计信息，包括设备总数、在线设备总数、接收的数据量等。这些数据将以图表的方式展示，如柱状图、折线图等，以使用户一目了然地了解系统的整体情况。
7. 手机端适配：确保网站的样式能够适配手机端，并在手机浏览器、微信内置浏览器等应用中友好显示。这将增加用户的便利性和可访问性，使用户可以随时随地访问物联网设备管理平台。

3.2 性能需求

1. 系统配置

系统要具有良好的反应速度，课题要求在良好的网络情况下，本系统应该具有如下时间特性要求：

单个用户在线时：1. Web 响应用户动作时间小于 1 秒。2. 信息搜索操作响应用户动作时间小于 2 秒。

500 个用户同时在线时：1. Web 响应用户动作时间小于 2 秒。2. 信息搜索操作响应用户动作时间小于 5 秒。

2. 访问容量

该系统至少在同一时间内支持 200 个用户并发访问。

3. 服务器配置最低要求

CPU 4 核 2.6G，内存 8.0G，硬盘 7200 转。

4. 可用性

该系统应实现多 Web 浏览器支持：在大多数流行的 Web 浏览器中正确显示和执行，包括 Firefox、Chrome、Edge 等。

5. 数据处理效率

该系统需要具备高效的数据处理能力，包括信息展示、检索、数据导入导出等。管理员需要对职责范围内的所有数据进行管理，因此需要具备高效的数据处理能力。

3.3 安全性需求

1. 保密性

(1) 用于身份验证的用户名和密码应该防止未经授权的用户访问系统，防止没有授权的用户进行管理员的相关操作。

(2) 建立合理的访问控制来防止有权限的管理员跨权限使用系统的资源。敏感数据交换之前要进行加密，密码存储之前应该加密。

(3) 在用户登录期间要防止 SQL 注入和密码强制破解和伪造会话入侵。

2. 完整性

防止非法用户对数据进行无意或恶意的修改、插入、删除，防止数据丢失。

3. 约束性

(1) 为数据库加上一定的约束，对关键性操作如删除、修改进行限制，并对用户进行警示。

(2) 不同身份所拥有的权限不同，只可以进行自己权限内的操作。

4. 用户信息安全性

(1) 着重账户信息安全性设计，做到外界人员无法入侵到系统本身。

(2) 内部人员操作需要留下操作痕迹，使用权管理层可以定期或不定期地维护系统。

5. 访问控制

系统应该实现访问控制机制，以确保只有授权用户可以访问系统资源。这可以通过使用身份验证和授权机制来实现，例如使用用户名和密码进行身份验证，并使用访问令牌或角色来授权用户访问特定资源。

6. 数据加密

系统应该使用加密技术来保护敏感数据的机密性和完整性。例如，在传输过程中，可以使用 SSL / TLS 协议来加密数据，以防止未经授权的用户访问数据。在存储过程中，可以使用加密算法对密码等敏感数据进行加密，以防止数据泄露。

7. 防止攻击

系统应该实现安全措施来防止各种攻击，例如 SQL 注入、密码强制破解和会话劫持等。这可以通过使用安全编码实践、使用防火墙和入侵检测系统等技术来实现。

8. 安全审计

系统应该记录所有用户的操作，以便在发生安全事件时进行调查和审计。这可以通过使用日志记录和审计工具来实现，以便管理员可以检查系统中发生的所有事件，并确定是否存在安全问题。

3.4 可维护性需求

作为一个成熟的系统，在开发初期就应该充分考虑系统的可维护性。对此，我们提出以下几点要求：

(1) 高内聚、低耦合的系统模块划分。开发者需要充分考虑模块内部结构的紧密型及模块间联系的独立性。

(2) 完备、清晰、可读的文档。文档是影响软件可维护性的一个决定因素，一个好的文档应具有简明性和书写风格的一致性，从而提高系统的可读性和可修改性。设计系统时应准备好各类相关文档，方便操作人员的对功能的快速查阅及维护人员的对架构的系统掌握。交付时应文档齐全，说明详尽，且文档描述符合相关标准。

(3) 良好的编程风格：程序内部应有详细的注释和统一的编程格式，结构清晰、注释明

确，使调试、测试人员能快速定位各种错误。对编程风格的具体要求如下：不使用令人捉摸不定或含糊不清的代码；使用有意义的数据名和过程名；适当的、格式正确的注释；使用模块化、结构化的设计方法；具有正确、一致和完整的文档。

（4）严谨的单元测试：对核心模块应编写单元测试，在交互时保证各子模块和系统整体的正常运作。对可测试性的要求如下：具有模块化和良好的结构；具有可理解性、可靠性；能显示任意的中间结果；以清楚的描述方式说明系统的输出，根据要求显示所有的输入。

（5）可扩展性：系统应该具备可扩展性，以便在未来需要添加新功能或模块时，可以轻松地进行扩展。这可以通过使用模块化和面向对象的设计方法来实现，以便可以轻松添加新的类和方法。

（6）易于维护的代码：系统应该编写易于维护的代码，以便在未来需要进行修改或更新时，可以轻松地进行维护。这可以通过使用清晰、简洁、可读性强的代码编写规范来实现，以便开发人员可以轻松理解和修改代码。

（7）自动化测试：系统应该具备自动化测试能力，以便在进行修改或更新时，可以自动化地进行测试，以确保系统的稳定性和正确性。这可以通过使用自动化测试工具和测试框架来实现，以便可以轻松地进行测试和验证。

（8）版本控制：系统应该使用版本控制工具来管理代码和文档，以便可以轻松跟踪和管理系统的版本和变更历史。这可以通过使用Git、SVN等版本控制工具来实现，以便可以轻松地进行版本控制和协作开发。

5 总体设计

5.1 基本设计概念和流程处理

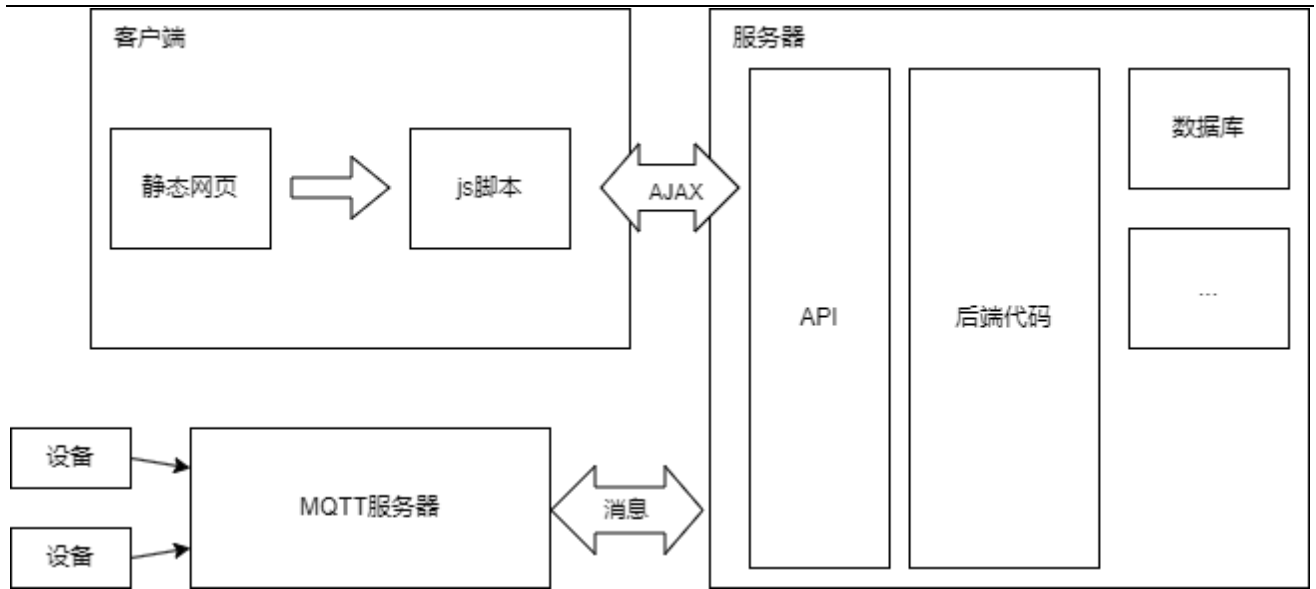
前端：Vue.js 框架

后端：Node.js 框架

数据库：数据库采用 MySQL

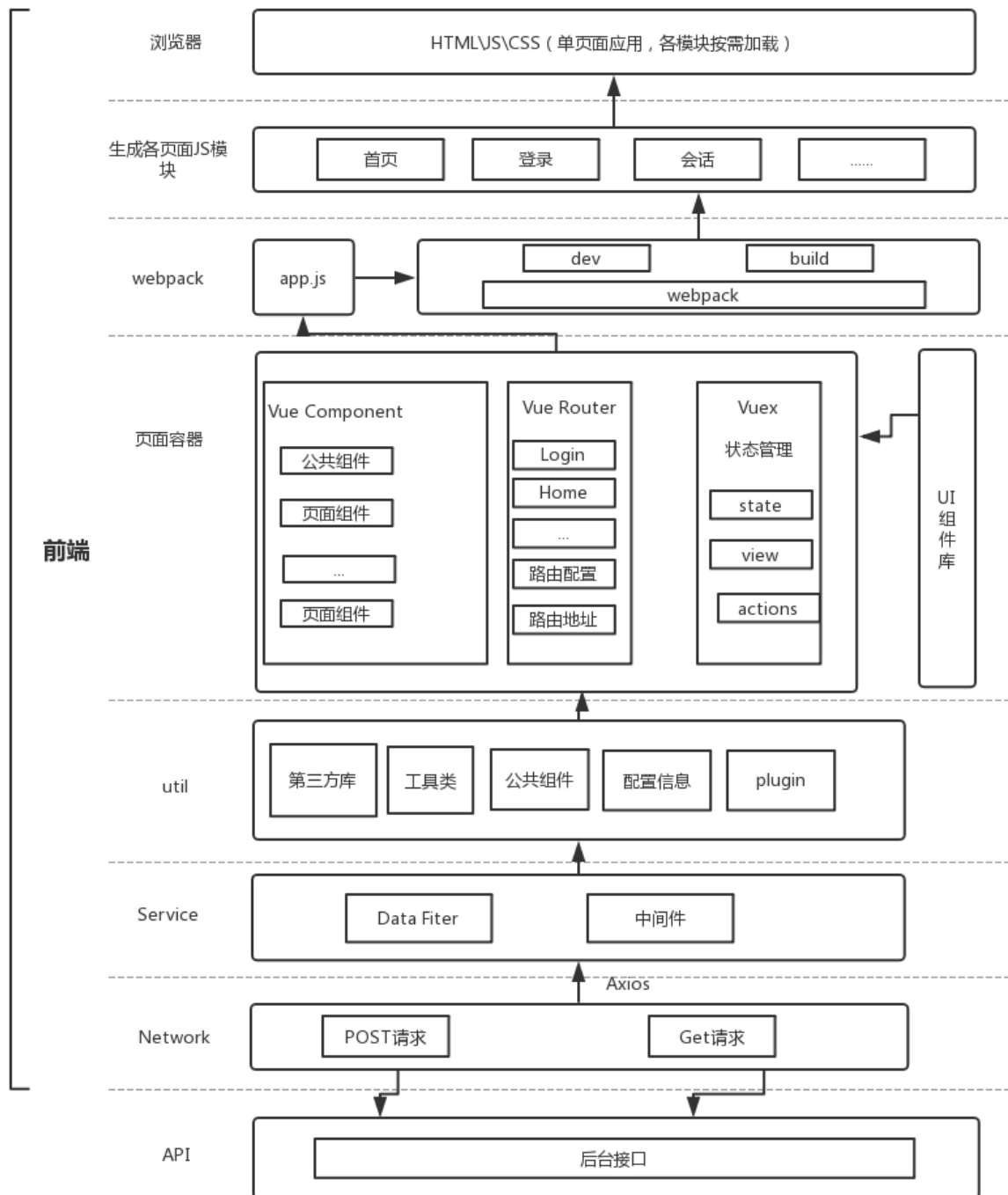
客户端：推荐使用 Chrome、Firefox、Safari、Microsoft Edge 浏览器

处理流程图如下：

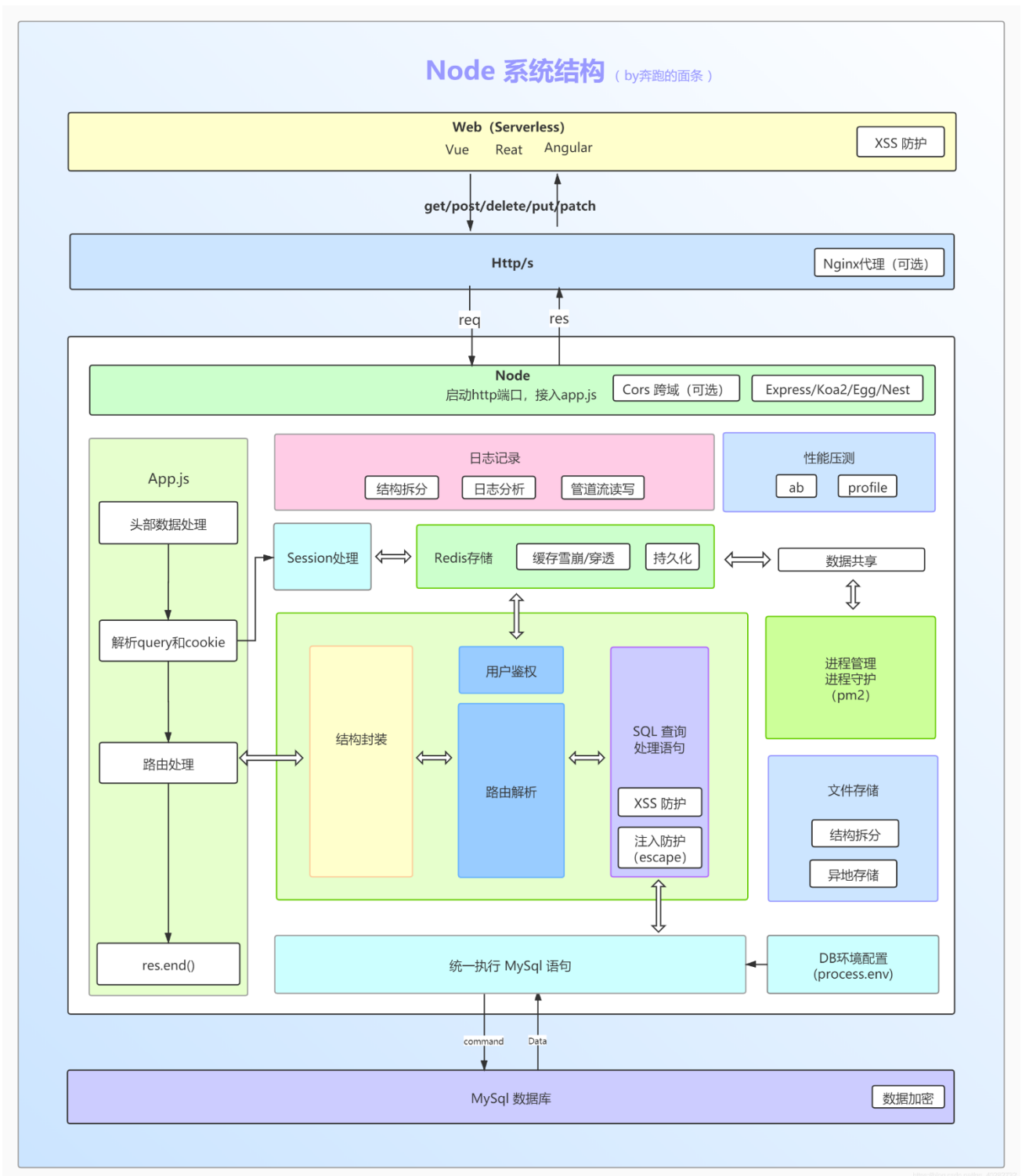


5.2 系统架构

5.2.1 前端架构



5.2.2 后端架构



5.3 技术介绍

5.3.1 前端技术框架

项目前端使用了 **Vue + Element-plus 组件库 + Echarts 图表组件 + 高德地图 Web 服务 API + npm 包管理工具** 作为前端开发的技术栈。

Vue 是一款用于构建用户界面的 JavaScript 框架。它基于标准 HTML、CSS 和 JavaScript 构建，并提供了一套声明式的、组件化的编程模型，帮助开发者高效地开发用户界面。无论是简单还是复杂的界面，Vue 都可以胜任。Vue 的两个核心功能分别是：1. 声明式渲染：Vue 基于标准 HTML 拓展了一套模板语法，使得我们可以声明式地描述最终输出的 HTML 和 JavaScript 状态之间的关系。2. 响应性：Vue 会自动跟踪 JavaScript 状态并在其发生变化时响应式地更新 DOM。

Element Plus 是一个基于 Vue 3 的高质量 UI 组件库。它包含了丰富的组件和扩展功能，例如表格、表单、按钮、导航、通知等，让开发者能够快速构建高质量的 Web 应用。Element Plus 的设计理念是：提供开箱即用的 UI 组件和扩展功能，帮助开发者快速构建应用程序，同时提供详细的文档和教程，让开发者更好地掌握和使用 Element Plus。

ECharts 是一款基于 JavaScript 的数据可视化图表库，提供直观，生动，可交互，可个性化定制的数据可视化图表。ECharts 最初由百度团队开源，并于 2018 年初捐赠给 Apache 基金会，成为 ASF 孵化级项目。ECharts 提供了常规的折线图、柱状图、散点图、饼图、K 线图，用于统计的盒形图，用于地理数据可视化的地图、热力图、线图，用于关系数据可视化的关系图、treemap、旭日图，多维数据可视化的平行坐标，还有用于 BI 的漏斗图，仪表盘，并且支持图与图之间的混搭。

高德地图 JSAPI 2.0 是高德开放平台免费提供的第四代 Web 地图渲染引擎，以 WebGL 为主要绘图手段，本着“更轻、更快、更易用”的服务原则，广泛采用了各种前沿技术，交互体验、视觉体验大幅提升，同时提供了众多新增能力和特性。地图 JSAPI 2.0 与 1.x 版本的 JSAPI 接口基本保持一致，同时提供了适配器方便开发者以零成本将地图应用从老版本 JSAPI 升级至 2.0 版本。延续了 PC 端、移动端完自动适配的特性，兼容 IE9 及以上的所有浏览器环境。

npm(node package manager)，是 nodejs 的包管理器，用于 node 插件管理（包括安装、卸载、管理依赖等），它是随同 NodeJS 一起安装的包管理工具，能解决 NodeJS 代码部署上的很多问题，常见的使用场景有以下几种：允许用户从 NPM 服务器下载别人编写的第三方包到本地使用。允许用户从 NPM 服务器下载并安装别人编写的命令行程序到本地使用。允许用户将自己编写的包或命令行程序上传到 NPM 服务器供别人使用。

5.3.2 后端技术框架

项目后端使用了 Node.js + koa + npm 包管理工具的技术栈，同时使用 MySQL 作为数据库。

Node.js 发布于 2009 年 5 月，由 Ryan Dahl 开发，是一个基于 Chrome V8 引擎的 JavaScript 运行环境，使用了一个事件驱动、非阻塞式 I/O 模型，让 JavaScript 运行在服务端的开发平台，它让 JavaScript 成为与 PHP、Python、Perl、Ruby 等服务端语言平起平坐的脚本语言。Node.js 对一些特殊用例进行优化，提供替代的 API，使得 V8 在非浏览器环境下运行得更好，V8 引擎执行 Javascript 的速度非常快，性能非常好，基于 Chrome JavaScript 运行时建立的平台，用于方便地搭建响应速度快、易于扩展的网络应用。

Koa 是一个新的 web 框架，由 Express 幕后的原班人马打造，致力于成为 web 应用和 API 开发领域中的一个更小、更富有表现力、更健壮的基石。通过利用 async 函数，Koa 帮你丢弃回调函数，并有力地增强错误处理。Koa 并没有捆绑任何中间件，而是提供了一套优雅的方法，帮助开发者快速而愉快地编写服务端应用程序。

npm(node package manager)，是 nodejs 的包管理器，用于 node 插件管理（包括安装、卸载、管理依赖等），它是随同 NodeJS 一起安装的包管理工具，能解决 NodeJS 代码部署上的很多问题，常见的使用场景有以下几种：允许用户从 NPM 服务器下载别人编写的第三方包到本地使用。允许用户从 NPM 服务器下载并安装别人编写的命令行程序到本地使用。允许用户将自己编写的包或命令行程序上传到 NPM 服务器供别人使用。

MySQL 是一个关系型数据库管理系统，由瑞典 MySQL AB 公司开发，属于 Oracle 旗下产品。MySQL 是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL 是最好的 RDBMS (Relational Database Management System, 关系数据库管理系统) 应用软件之一。MySQL 是一种关系型数据库管理系统，关系数据库将数据保存在不同的表中，而不是将所有

数据放在一个大仓库内，这样就增加了速度并提高了灵活性。MySQL 所使用的 SQL 语言是用于访问数据库的最常用标准化语言。MySQL 软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型和大型网站的开发都选择 MySQL 作为网站数据库。

5.3.3 MQTT 服务器

本项目使用的 MQTT 服务器为 EMQX。在后端中集成了一个 MQTT 客户端进行消息接收。

EMQX 是一款开源的大规模分布式 MQTT 消息服务器，功能丰富，专为物联网和实时通信应用而设计。EMQX 5.0 单集群支持 MQTT 并发连接数高达 1 亿条，单服务器的传输与处理吞吐量可达每秒百万级 MQTT 消息，同时保证毫秒级的低时延。EMQX 支持多种协议，包括 MQTT (3.1、3.1.1 和 5.0)、HTTP、QUIC 和 WebSocket 等，保证各种网络环境和硬件设备的可访问性。EMQX 还提供了全面的 SSL/TLS 功能支持，比如双向认证以及多种身份验证机制，为物联网设备和应用程序提供可靠和高效的通信基础设施。内置基于 SQL 的规则引擎，EMQX 可以实时提取、过滤、丰富和转换物联网数据。此外，EMQX 采用了无主分布式架构，以确保高可用性和水平扩展性，并提供操作友好的用户体验和出色的可观测性。

5.4 接口设计

这部分主要介绍前后端接口。本项目采用了 axios 库进行前后端交互。axios 是一个基于 Promise 的 HTTP 客户端，专门设计用于浏览器和 Node.js 的 HTTP 请求。它提供了强大的功能，使我们能够方便地与后端服务器进行通信。前后端的交互在本项目中是通过 HTTP 协议进行的，而 axios 作为前端与后端通信的工具，提供了以下方式来实现交互：

- a. 发送 GET 请求
- b. 发送 POST 请求
- c. 发送 PUT 请求
- d. 发送 DELETE 请求

通过以上方式，axios 库使得前端能够与后端服务器进行各种数据交互，包括获取、提交、更新和删除数据。这些 HTTP 请求可以与后端 API 端点进行交互，实现项目中各种功能，如设备管理、数据展示以及地图信息的获取和显示。axios 的 Promise 基础使得请求处理变得可

靠且容易管理，同时有助于处理错误和异步操作。这种前后端交互方式对于 IoT 管理系统的正常运作非常关键。

5.4.1 用户信息接口

5.4.1.1 用户登录

url	/loginSubmit
参数值	body.username: 用户名 body.password: 密码
返回值	(1)success: true: 登录成功 user: 用户信息 (2)success: false: 登录失败 message: 错误消息，如用户名或密码错误
简介	该接口用于用户登录，接受用户名和密码，验证用户身份。如果验证通过，返回成功标志和用户信息；否则返回失败标志和错误消息。

5.4.1.2 用户注册

url	/registerSubmit
参数值	body.username: 用户名 body.password: 密码 body.email: 电子邮件 body.phone: 电话号码 body.gender: 性别 body.address: 地址
返回值	(1)success: true: 注册成功 user: 用户信息

	<p>(2)success: false: 注册失败</p> <p>message: 错误消息, 如用户名重复、邮箱重复、用户名或密码长度不符合要求等</p>
简介	<p>该接口用于用户注册, 接受一组用户信息, 包括用户名、密码、电子邮件等。在注册前会检查用户名和邮箱的唯一性, 如果通过验证, 创建新用户并返回用户信息; 否则返回错误消息。</p>

5.4.1.3 修改信息

url	/modifyInfor
参数值	<p>body.username: 原用户名</p> <p>body.newUsername: 新用户名</p> <p>body.email: 电子邮件</p> <p>body.phone: 电话号码</p> <p>body.gender: 性别</p> <p>body.address: 地址</p>
返回值	<p>(1)success: true: 修改成功</p> <p>user: 用户信息</p> <p>(2)success: false: 修改失败</p> <p>message: 错误消息, 如用户名不存在、邮箱格式无效、用户名长度不符合要求等</p>
简介	<p>该接口用于修改用户信息, 接受原用户名以及要修改的用户信息。在修改前会检查用户是否存在和邮箱格式是否有效, 如果通过验证, 更新用户信息并返回用户信息; 否则返回错误消息。</p>

5.4.1.4 修改密码

url	/modifyPassword
参数值	body.username: 用户名 body.password: 新密码
返回值	(1)success: true: 修改成功 user: 用户信息 (2)success: false: 修改失败 message: 错误消息, 如用户名不存在、用户名长度不符合要求等
简介	该接口用于修改用户密码, 接受用户名以及新密码。在修改前会检查用户是否存在, 如果通过验证, 更新用户密码并返回用户信息; 否则返回错误消息。

5.4.2 设备信息接口

5.4.2.1 获取设备

url	/getDevice
参数值	body.username: 用户名
返回值	(1)success: true: 获取设备成功 device: 用户拥有的设备列表 (2)success: false: 获取设备失败 message: 错误消息, 如用户名为空或用户没有设备

简介	该接口用于获取特定用户拥有的设备列表。接受用户的用户名作为参数，如果用户名有效，返回用户拥有的设备列表；否则返回错误消息。
----	---

5.4.2.2 修改信息

url	/modifyDevice
参数值	body.oldName: 原设备名 body.newName: 新设备名 body.type: 设备类型 body.status: 设备状态 body.location: 设备位置 body.description: 设备描述
返回值	(1)success: true: 修改设备信息成功 message: 成功消息 (2)success: false: 修改失败 message: 错误消息，如找不到设备
简介	该接口用于修改设备信息，接受原设备名和要修改的设备信息。在修改前会检查设备是否存在，如果设备存在，更新设备信息并返回成功消息；否则返回错误消息。

5.4.2.3 添加设备

url	/addDevice
参数值	body.name: 设备名 body.type: 设备类型 body.status: 设备状态

	body.location: 设备位置 body.description: 设备描述 body.owner: 设备拥有者
返回值	(1)success: true: 添加设备成功 message: 成功消息 (2)success: false: 添加设备失败 message: 错误消息, 如设备名已存在
简介	该接口用于添加新设备, 接受一组设备信息, 包括设备名、类型、状态、位置、描述和拥有者信息。在添加前会检查设备名的唯一性, 如果设备名未被使用, 创建新设备并返回成功消息; 否则返回错误消息。

5.4.2.4 删除设备

url	/deleteDevice
参数值	body.name: 设备名
返回值	(1)success: true: 删除设备成功 message: 成功消息 (2)success: false: 删除设备失败 message: 错误消息, 如设备名不存在
简介	该接口用于删除设备, 接受要删除的设备名。在删除前会检查设备是否存在, 如果设备存在, 将其从数据库中删除并返回成功消息; 否则返回错误消息。这个接口允许用户对其设备进行删除操作, 以管理其设备列表。

5.4.3 设备消息接口

url	/getMessage
参数值	body: 包含一个或多个设备名的数组 body[i].name: 设备名
返回值	(1)success: true: 获取消息成功 message: 消息列表 (2)success: false: 获取消息失败 message: 错误消息, 如设备名为空或设备没有消息
简介	该接口用于获取特定设备的消息列表。接受一个包含一个或多个设备名的数组作为参数, 对每个设备进行消息检索, 然后将消息列表合并为一个完整的消息数组。如果设备名有效且设备具有消息, 返回成功标志和消息列表; 否则返回失败标志和错误消息。

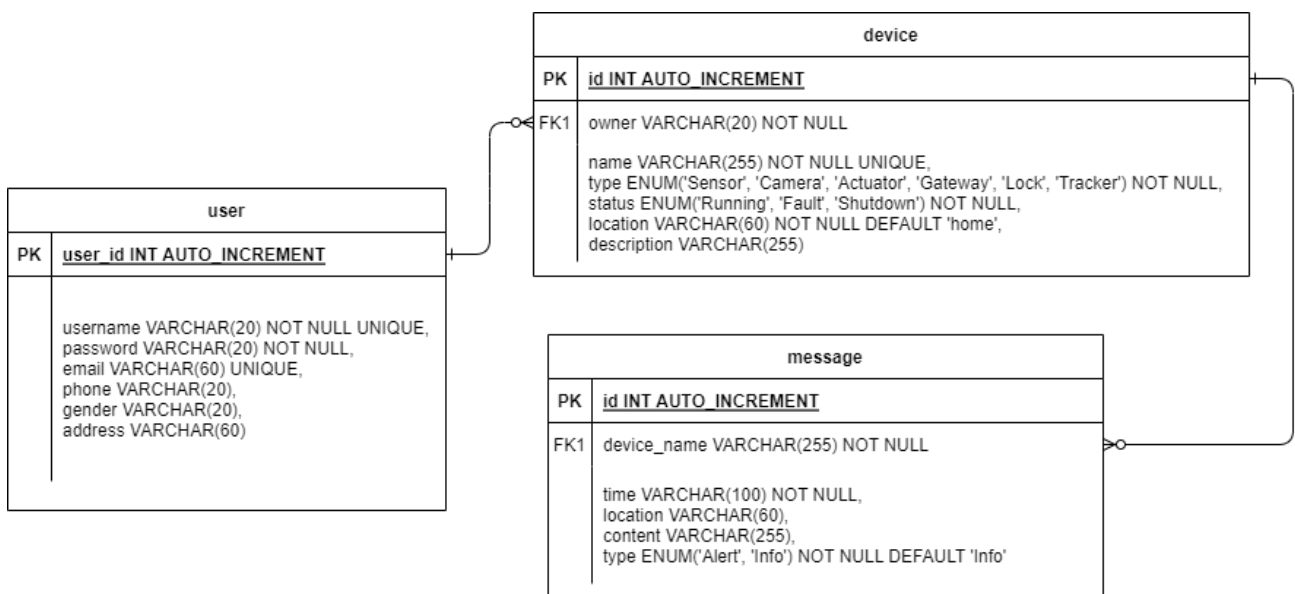
5.4.4 MQTT 服务器消息接口

1. 当客户端（可能是 MQTT 客户端）接收到消息时，会触发“message”事件。
2. 代码首先尝试解析接收到的消息，期望消息的格式是 JSON，包含设备名（device_name）、设备消息（device_message）、设备位置（device_location）和消息类型（type）。
3. 如果消息中不包含 type 字段，代码将默认设置消息类型为 “Info”。
4. 接下来，代码会验证消息的各个字段，确保它们符合期望的格式。这些验证包括确保设备名、消息内容、设备位置以及消息类型的存在和格式正确性。
5. 如果消息格式不正确，代码将输出错误消息并终止处理。

6. 如果消息格式正确，代码会获取当前的时间戳，并将消息信息插入到数据库中（可能是消息表），包括设备名、时间、消息内容、设备位置和消息类型。
7. 代码还会更新与设备相关的位置信息，将设备的位置字段更新为最新接收到的位置信息。

6 数据库设计

6.1 概念结构设计（E-R 图）



6.2 逻辑结构设计

- 用户表

user(user_id, username, password, email, phone, gender, address)

- 设备表

device(id, name, type, status, location, owner, description)

- 消息表

message(id, device_name, time, location, content, type)

6.3 物理结构设计

● 用户表

字段	类型	是否可以为空	是否主键	备注
user_id	INT	F	T	用户唯一 ID 编号, ID 自增
username	VARCHAR(20)	F	F	用户名, 添加唯一约束以确保用户名唯一
password	VARCHAR(20)	F	F	用户密码
email	VARCHAR(60)	F	F	用户电子邮件, 添加唯一约束以确保电子邮件唯一
phone	VARCHAR(20)	T	F	用户电话号码
gender	VARCHAR(20)	T	F	用户性别
address	VARCHAR(60)	T	F	用户地址

● 设备表

字段	类型	是否可以为空	是否主键	备注
id	INT	F	T	设备唯一 ID 编号, ID 自增
name	VARCHAR(255)	F	F	设备名称, 添加唯一约束以确保设备名称唯一
type	ENUM	F	F	设备类型, 可选值为 'Sensor', 'Camera', 'Actuator',

				'Gateway', 'Lock', 'Tracker'
status	ENUM	F	F	设备状态，可选值为 'Running', 'Fault', 'Shutdown'
location	VARCHAR(60)	F	F	设备最新位置，初始值为 "home"
owner	VARCHAR(20)	F	F	设备所有者，与用户表关联，外键
description	VARCHAR(255)	T	F	设备描述

● 消息表

字段	类型	是否可以为空	是否主键	备注
id	INT	F	T	消息唯一 ID 编号，ID 自增
device_name	VARCHAR(255)	F	F	设备名称，与设备表关联，外键
time	VARCHAR(100)	F	F	消息时间，由后端生成
location	VARCHAR(60)	F	F	位置信息，经纬度表示
content	VARCHAR(255)	T	F	消息内容
type	ENUM	F	F	消息类型，可选值为 'Alert', 'Info', 默认值 'Info'

7 系统出错设计

7.1 出错信息

系统输出信息的形式	含义	处理方法
数据库无法连接	数据库配置出错、数据库连接数超过上限	修改数据库配置、限制并发访问量
服务器无法访问	服务器正在维护中、短时间内有大量流量导致服务器瘫痪	联系系统管理员进行紧急处理
无法读取磁盘内容	磁盘受损	对磁盘与数据库进行周期性备份
非法访问	部分用户企图访问管理员界面或后台程序，窃取网站	限制普通用户越权访问，通过各种手段保护后台数据
数据库执行出错	部分用户企图恶意实施 SQL 注入	使用参数绑定的方法进行 SQL 语句构建。对用户的输入进行过滤、检查

7.2 补救措施

1. 系统恢复

系统崩溃后，根据系统运行日志恢复系统和数据，并重新启动系统。

2. 定时备份

(1) 周期性备份数据库中的数据，定期进行校对、更新，以避免数据损失之后难以找回。

(2) 使用 Git 进行完善的版本管理。

3. 人工操作

当出现紧急情况时，数据库管理员人工对数据库中数据进行修改，并做相应的记录。

7.3 系统维护设计

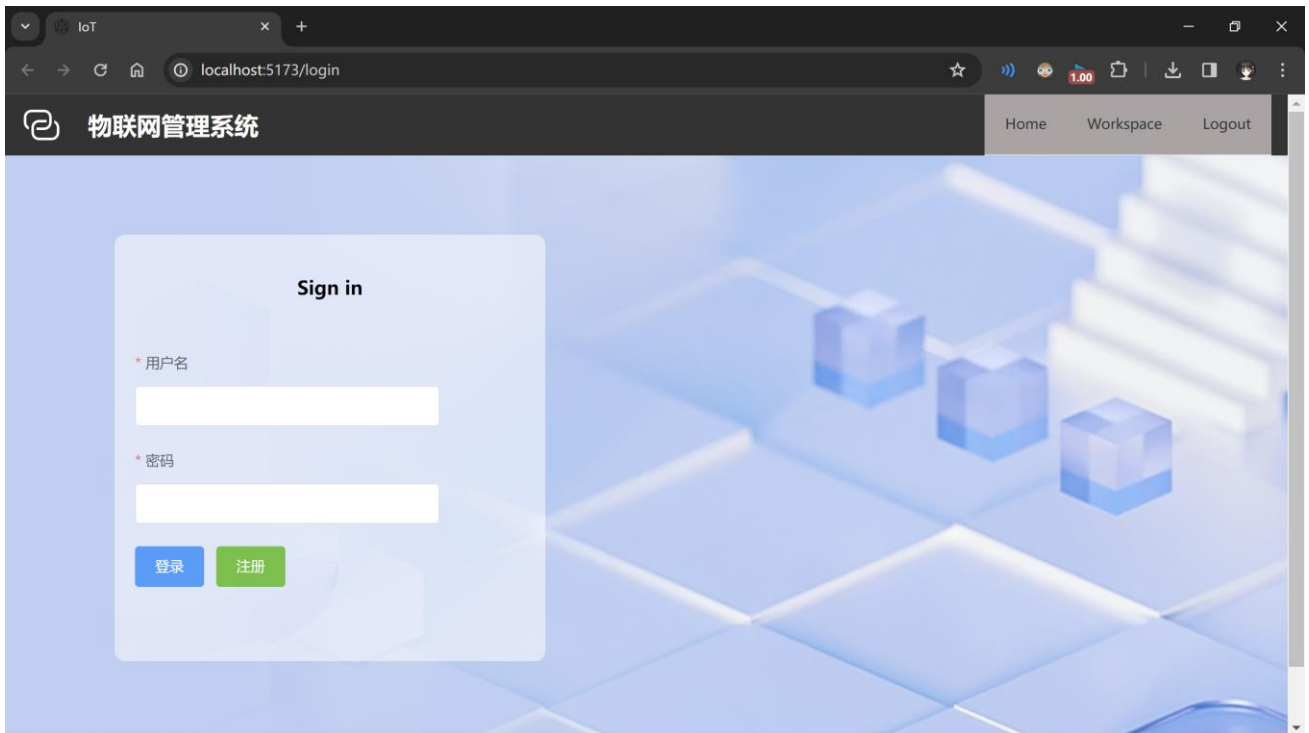
(1) 用户在该系统执行操作时应该留下痕迹，以方便检查系统是否被恶意篡改。同时系统管理员定时查看系统日志，统计非法攻击来源和次数，并针对相应攻击加强安全防范措施。

(2) 系统维护人员及时更新技术漏洞，通过各种手段防止各种对系统的攻击，增强代码的可靠性。

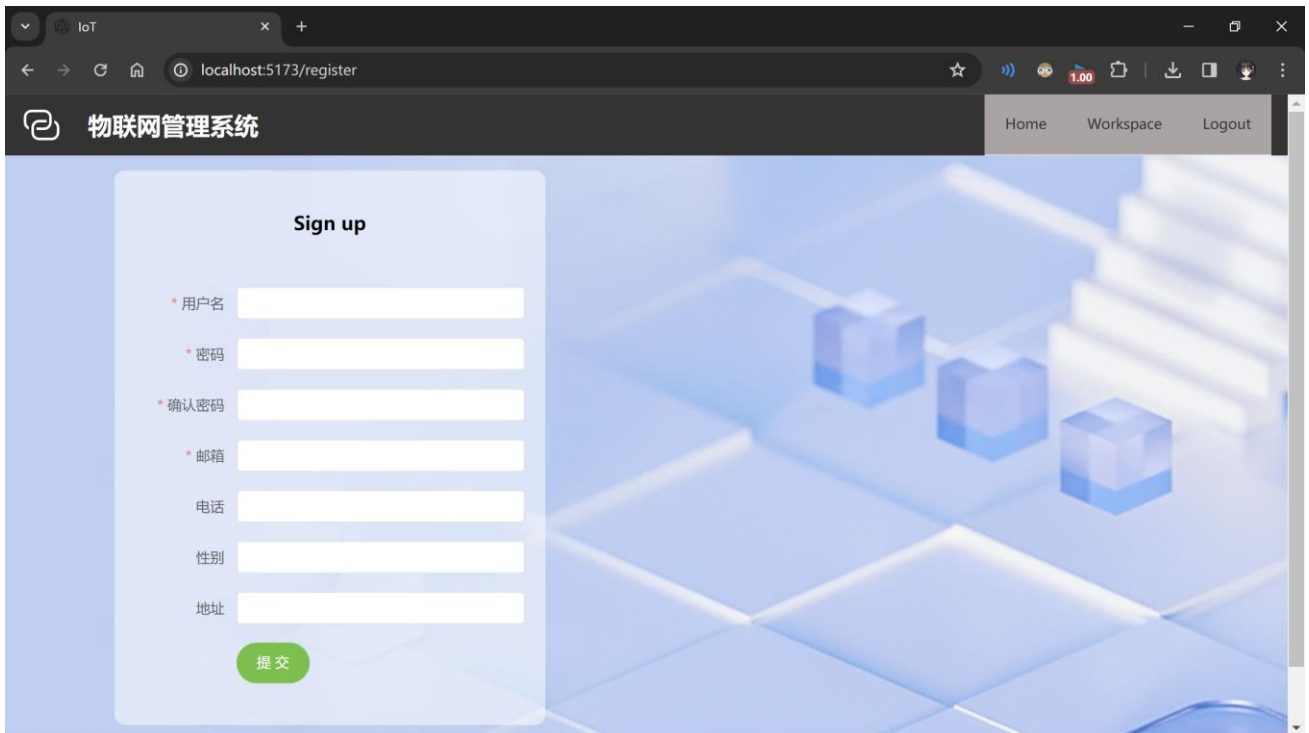
(3) 定期维护数据库，涉及到检查数据库表、检查日志文件等，确保数据库内数据的正确性。

8 用户界面

8.1 登录页面



8.2 注册页面



IoT

localhost:5173/register

物联网管理系统

Home Workspace Logout

Sign up

* 用户名

* 密码

* 确认密码

* 邮箱

电话

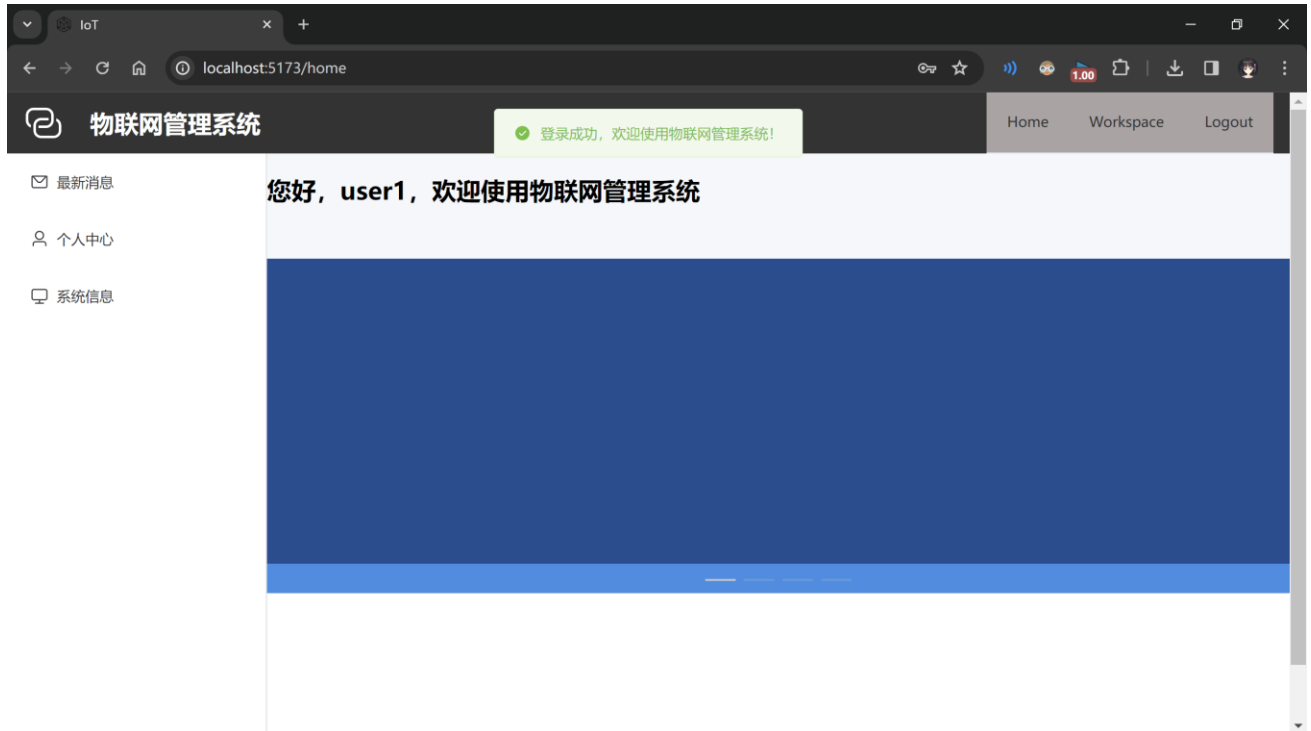
性别

地址

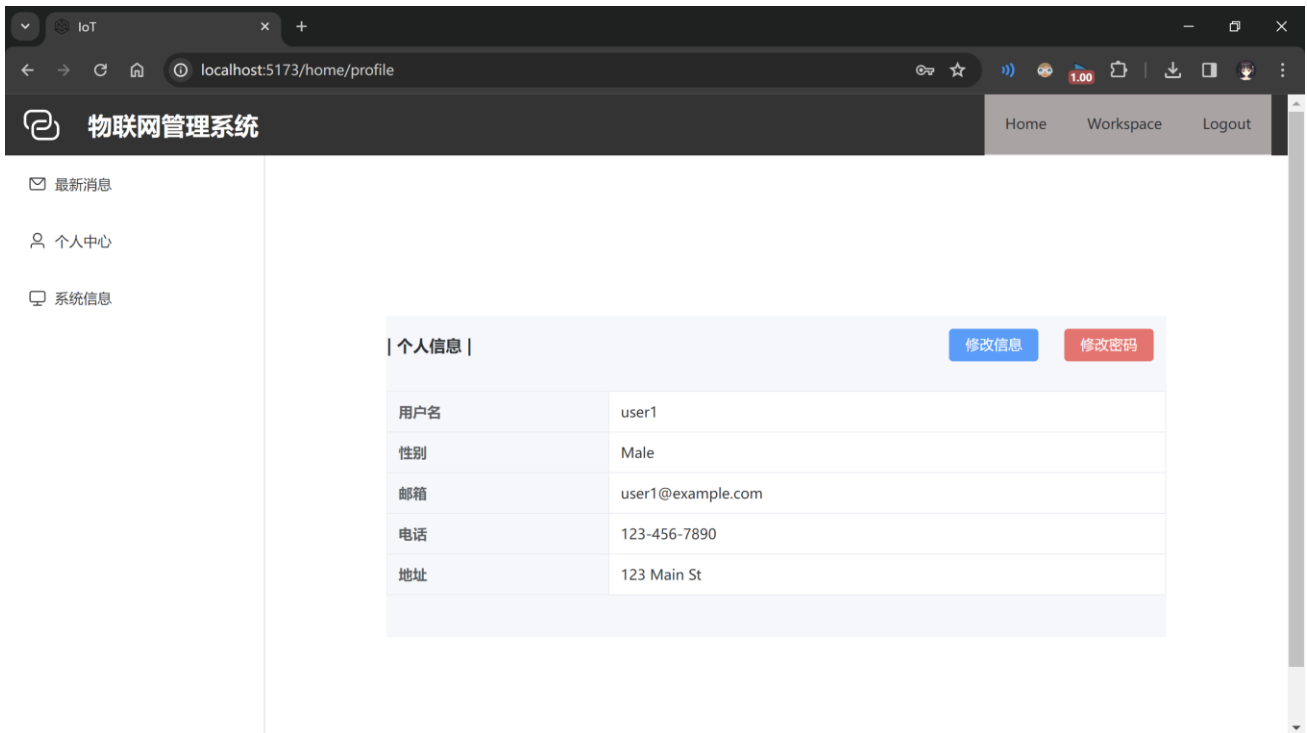
提交

8.3 Home 页面

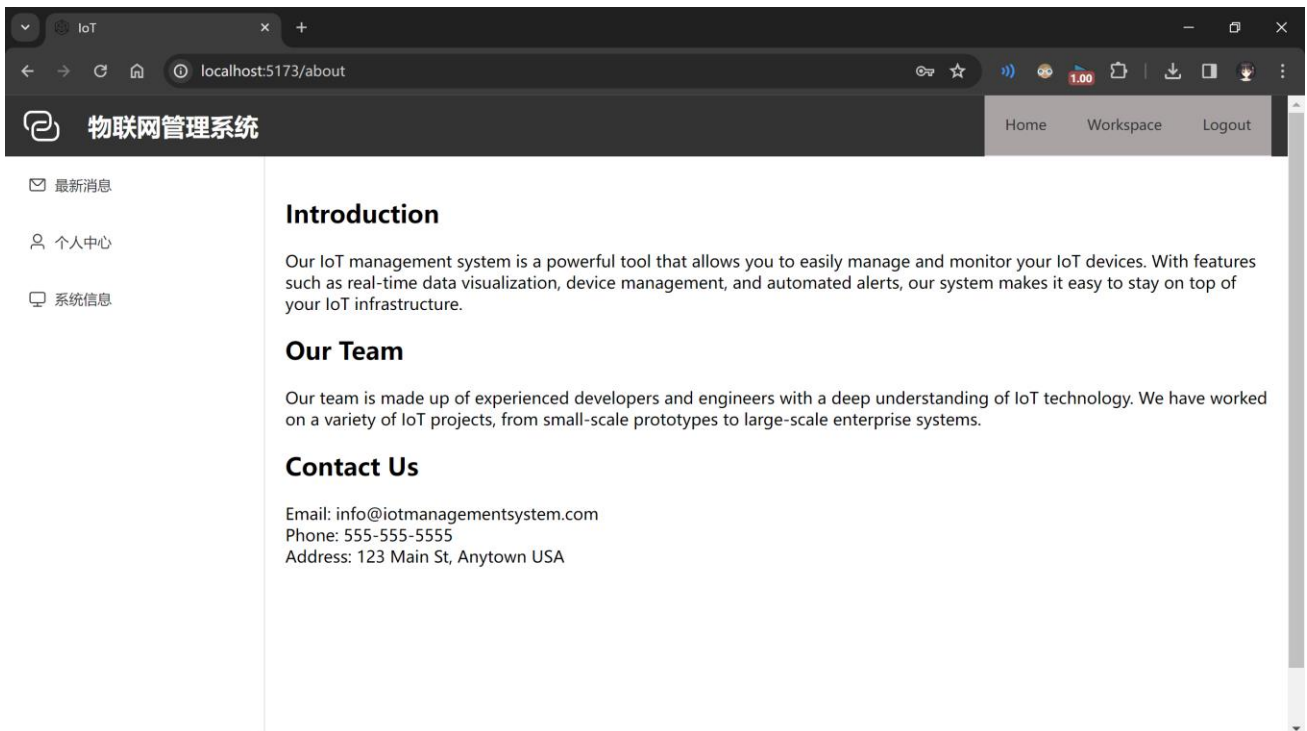
8.3.1 新闻页面



8.3.2 个人信息页面

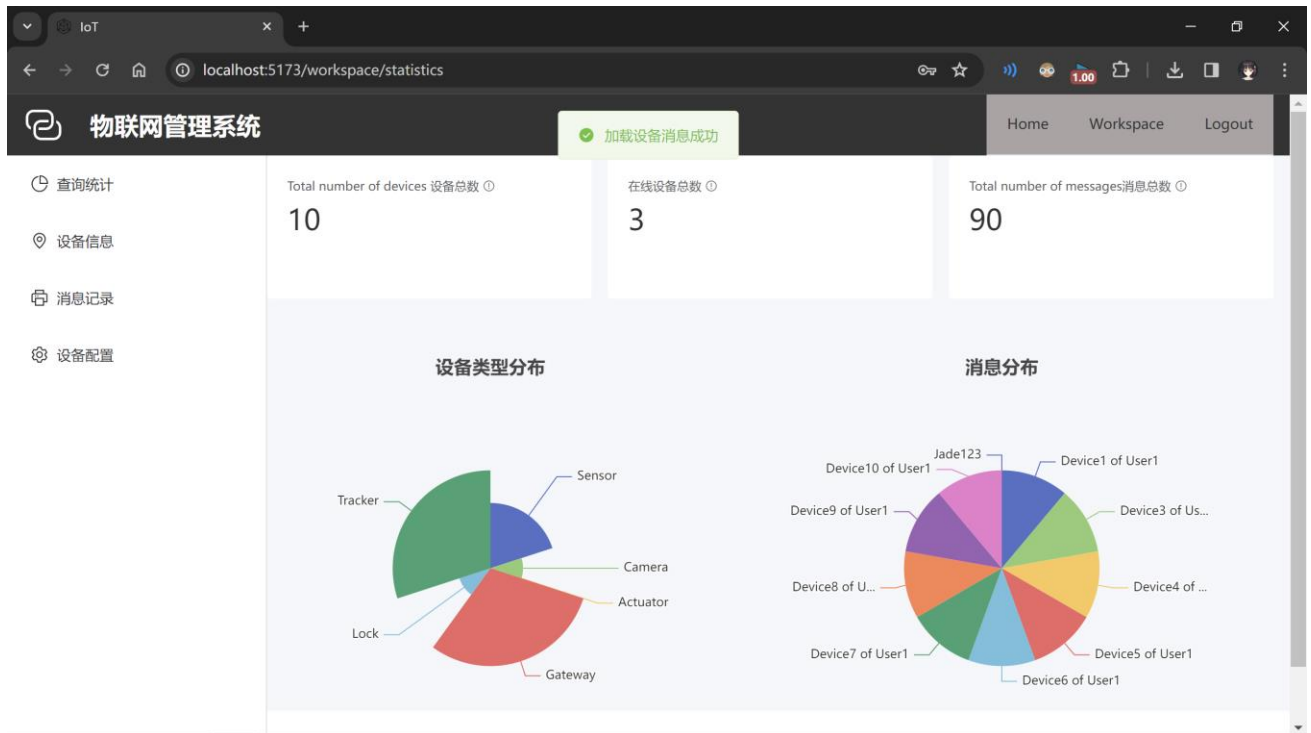


8.3.3 关于页面

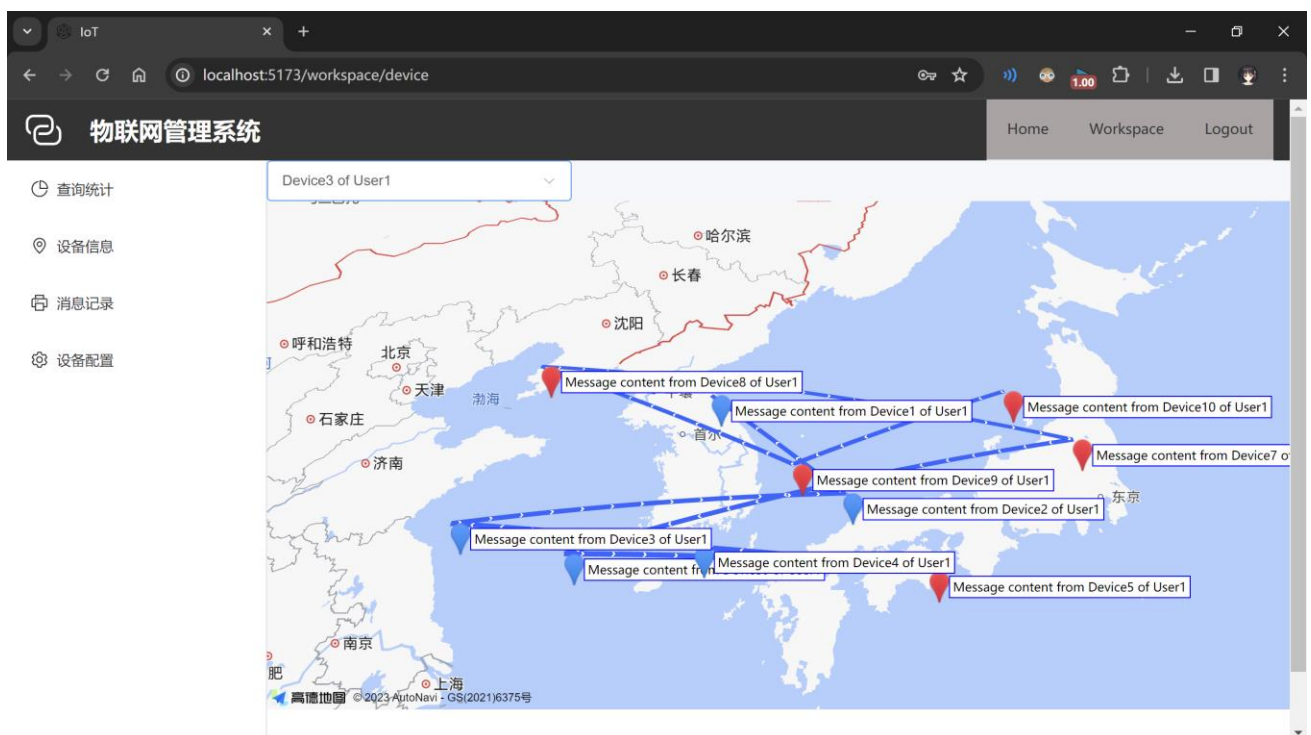
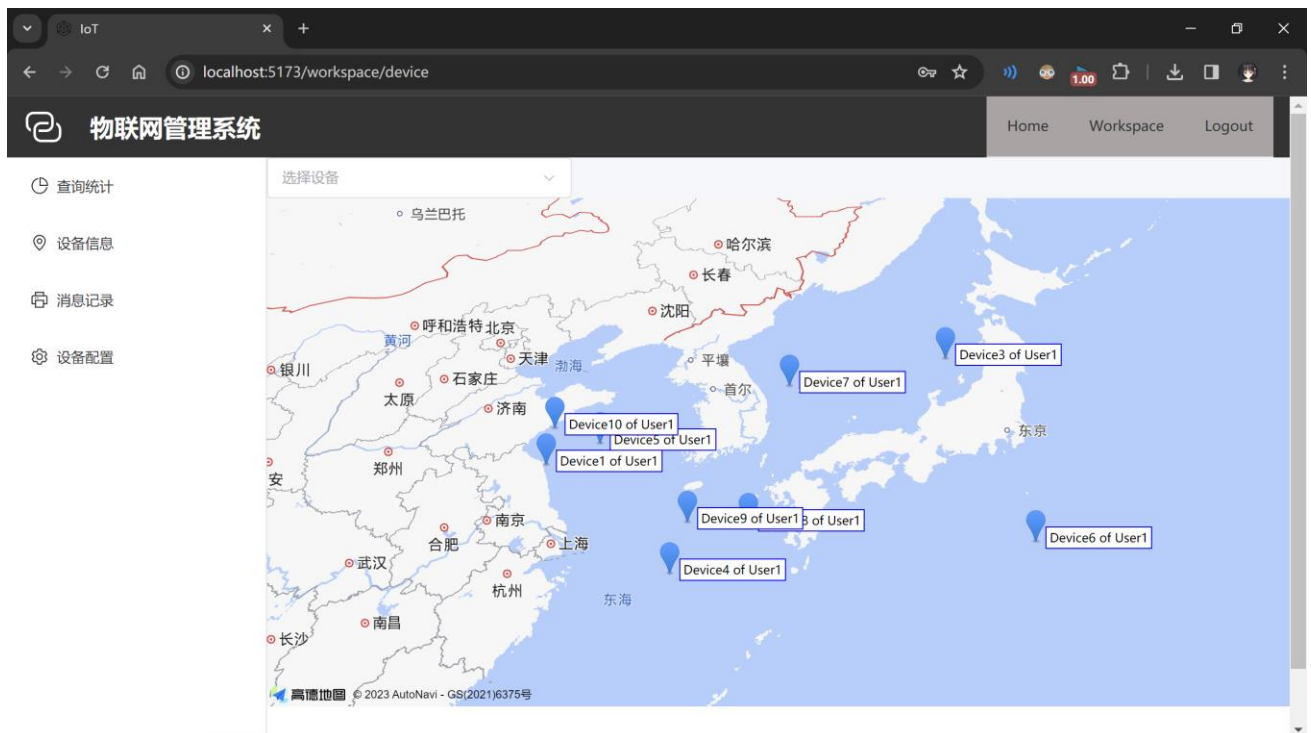


8.4 工作页面

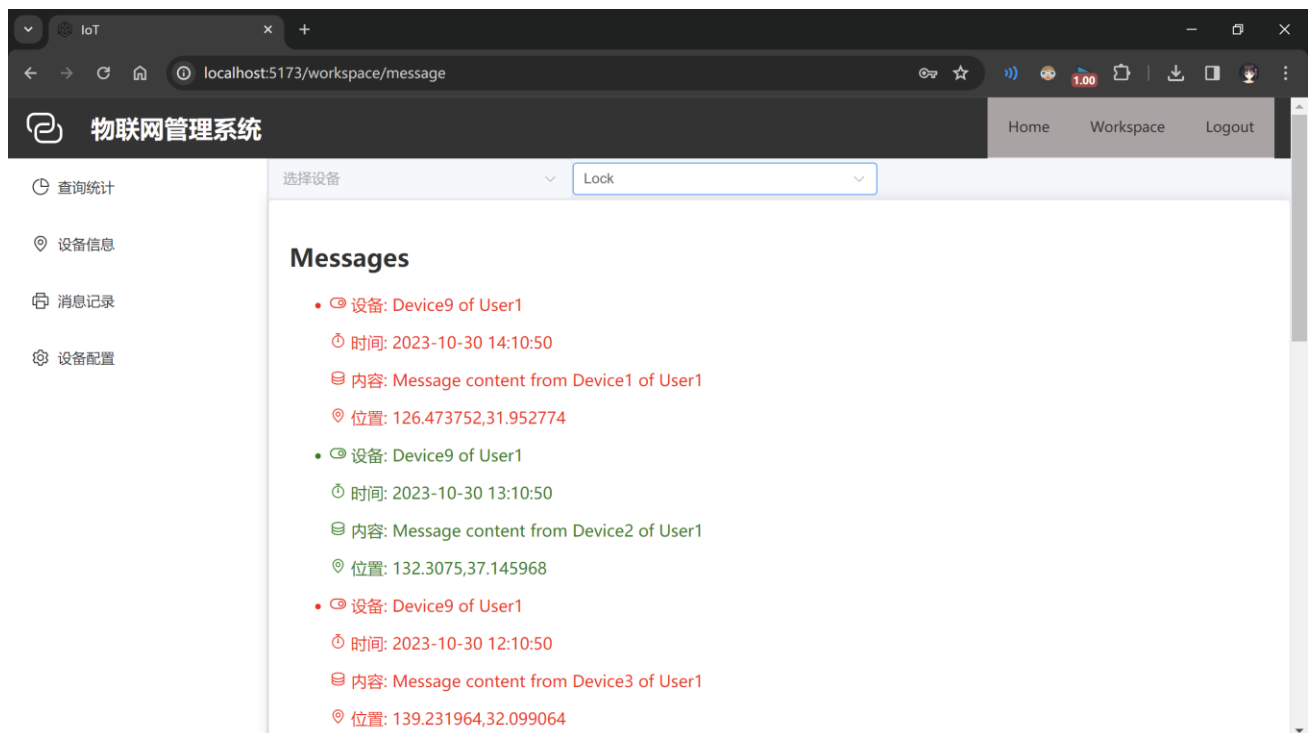
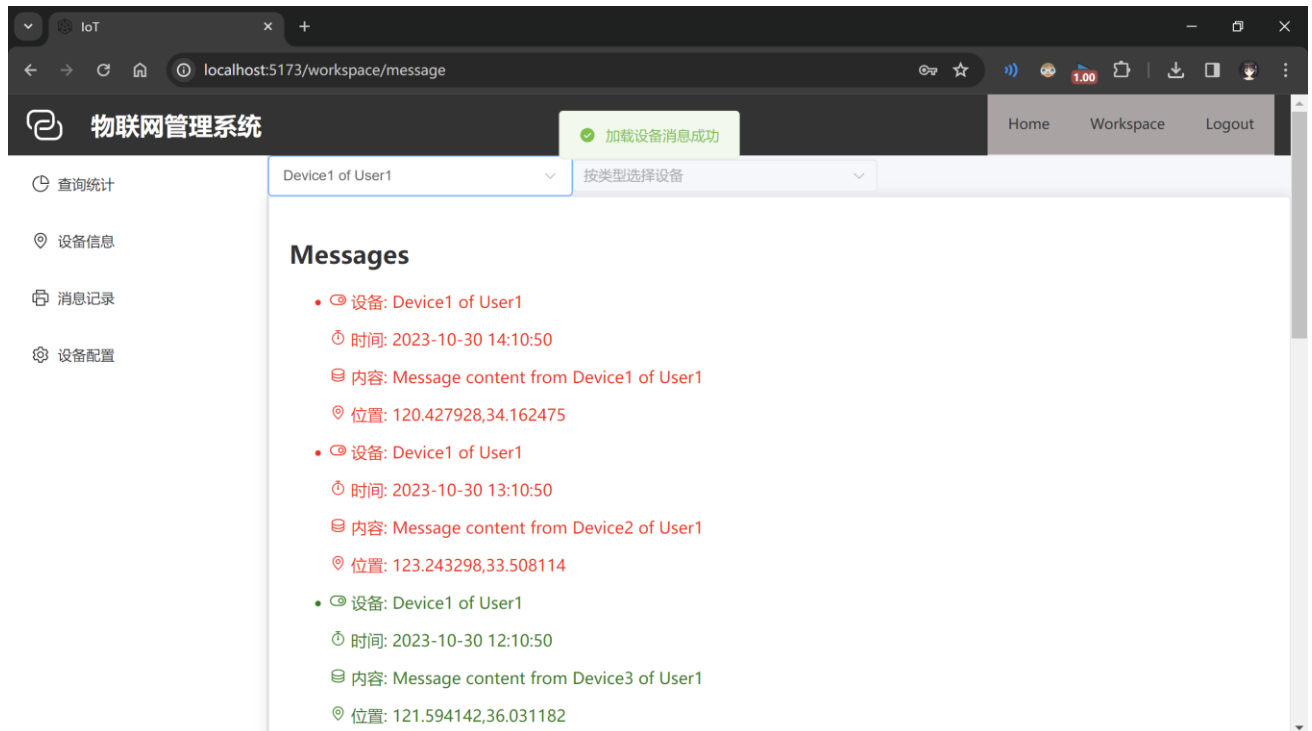
8.4.1 统计页面



8.4.2 设备页面



8.4.3 消息页面



8.4.4 设备修改页面

The screenshot shows the 'Add Device' page of the IoT Management System. The browser address bar is 'localhost:5173/workspace/device/settings'. The page has a dark header with the system name and navigation links. A sidebar on the left contains menu items. The main content area has a 'Select Device' dropdown, a 'Add Device' title, and a form with fields for device name, type, status, location, and description. There are 'Submit' and 'Delete' buttons at the bottom.

IoT

localhost:5173/workspace/device/settings

物联网管理系统

Home Workspace Logout

查询统计

设备信息

消息记录

设备配置

选择设备

添加设备

* 设备名

* 设备类型 Select

* 设备状态 Select

* 设备位置 Format is (longitude, latitude)

设备描述

提交 删除

The screenshot shows the 'Edit Device' page of the IoT Management System. The browser address bar is 'localhost:5173/workspace/device/settings'. The page has a dark header with the system name and navigation links. A sidebar on the left contains menu items. The main content area has a 'Device1 of User1' dropdown, an 'Edit Device' title, and a form with fields for device name, type, status, location, and description. There are 'Submit' and 'Delete' buttons at the bottom. A green success message '修改成功!' is displayed in the header.

IoT

localhost:5173/workspace/device/settings

物联网管理系统

Home Workspace Logout

查询统计

设备信息

消息记录

设备配置

Device1 of User1

修改设备

* 设备名 Device1 of User1

* 设备类型 Camera

* 设备状态 Running

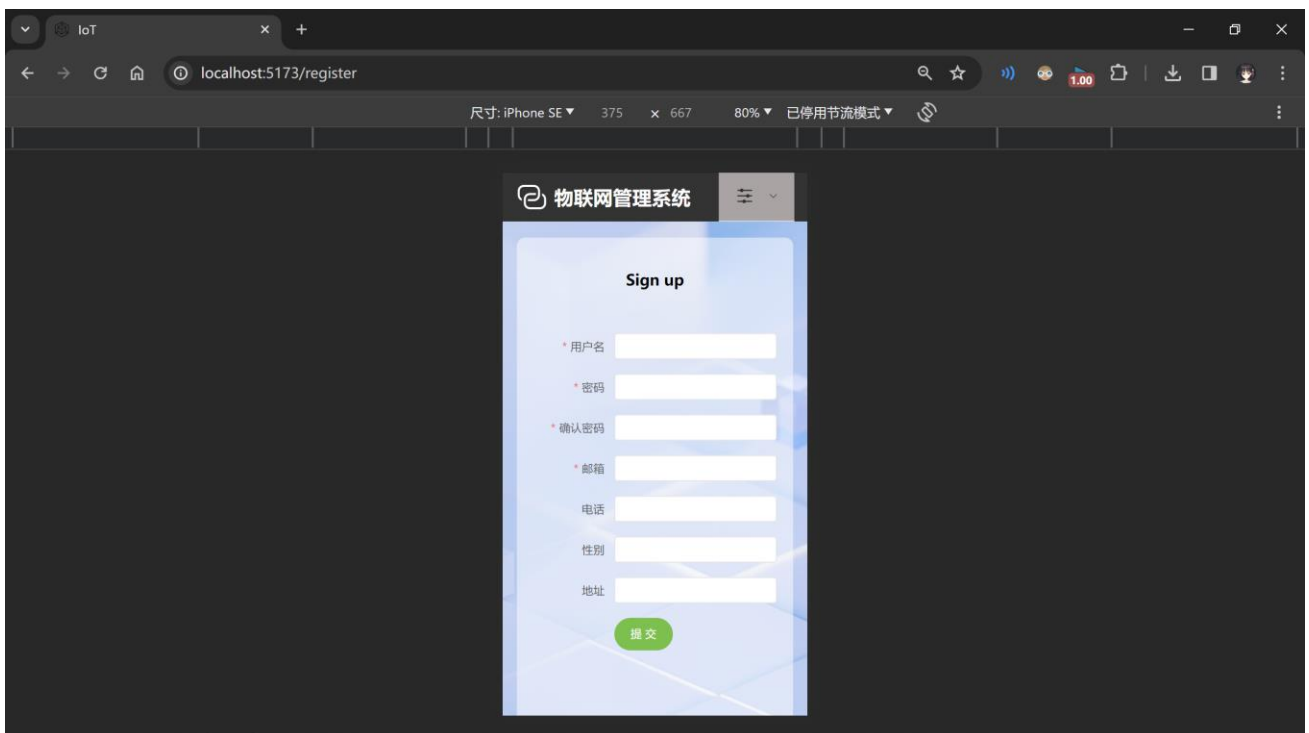
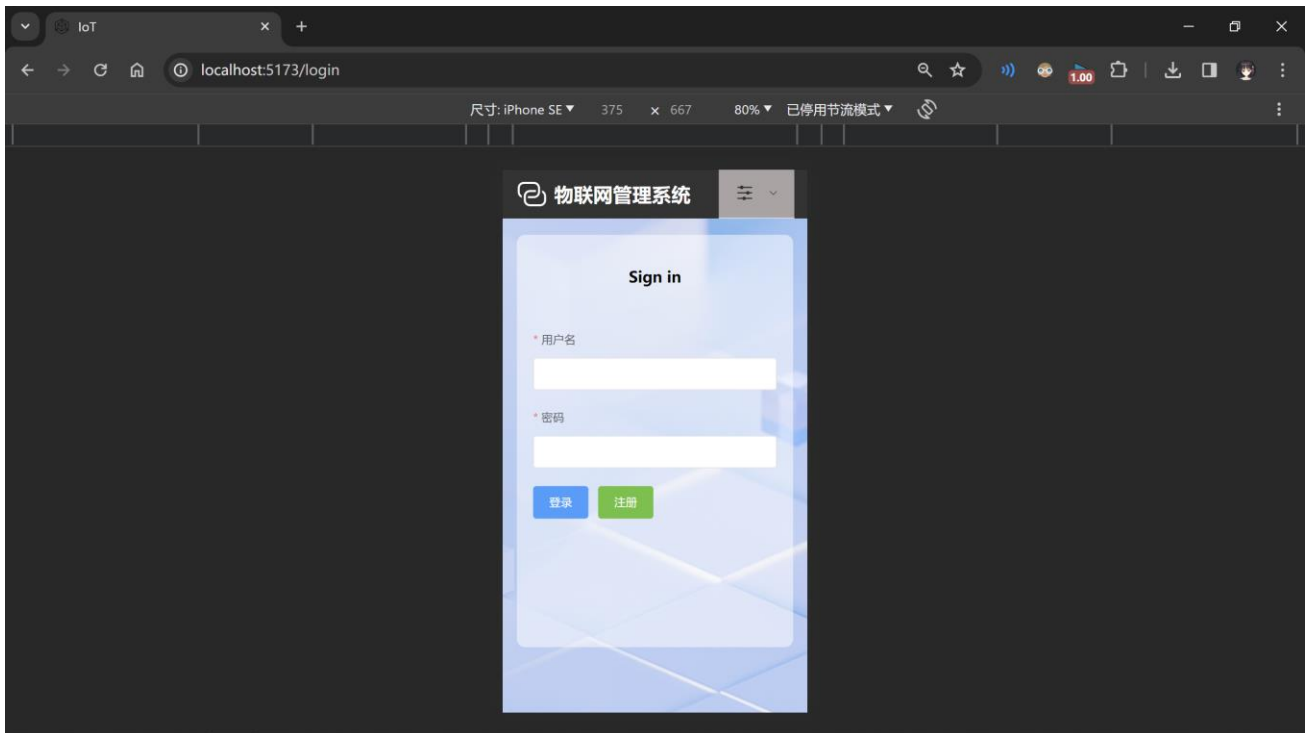
* 设备位置 120.427928,34.162475

设备描述 Description of Device1 owned by User1

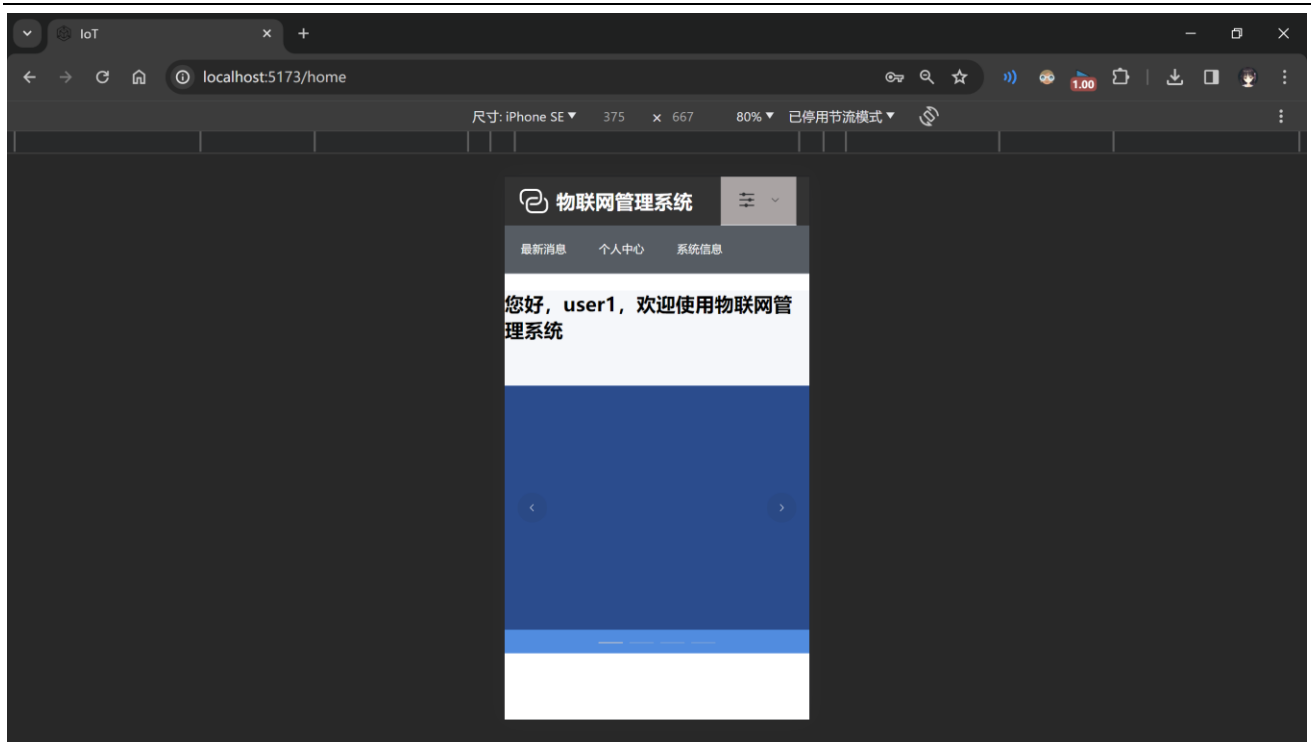
提交 删除

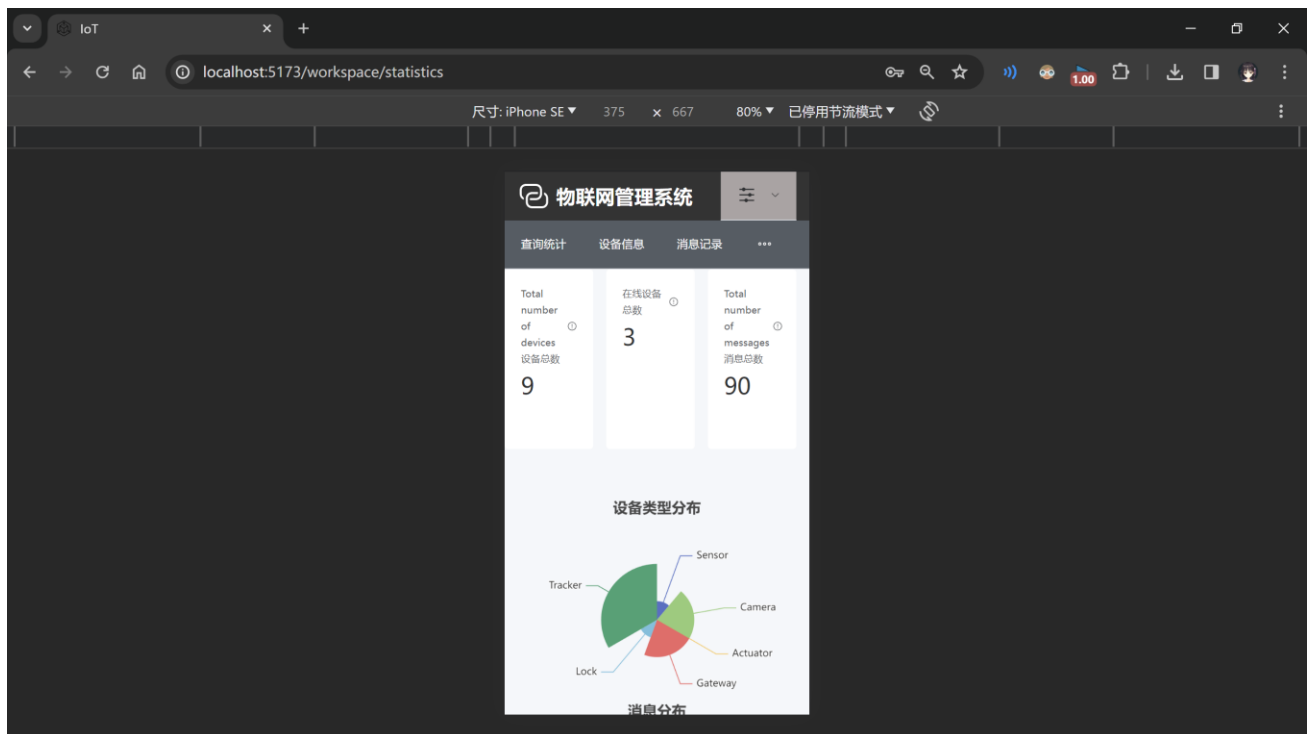
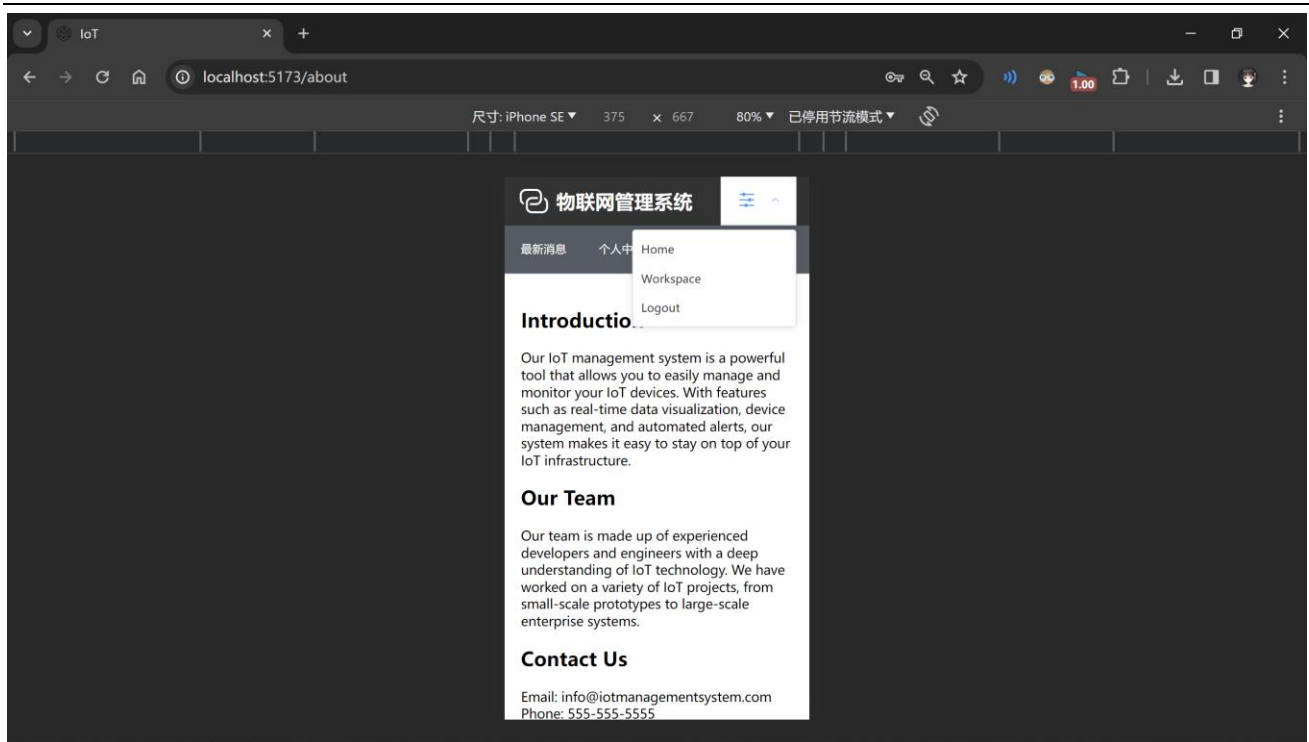
修改成功!

8.5 移动端适配

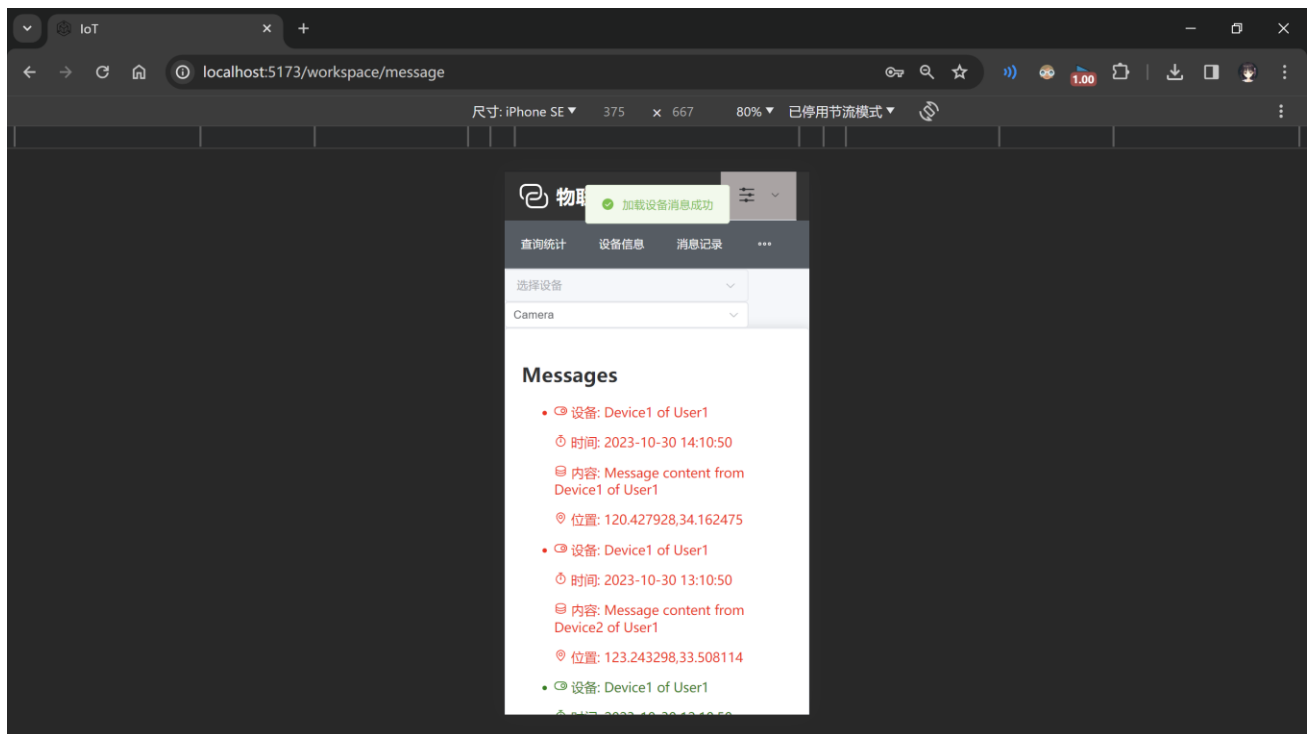
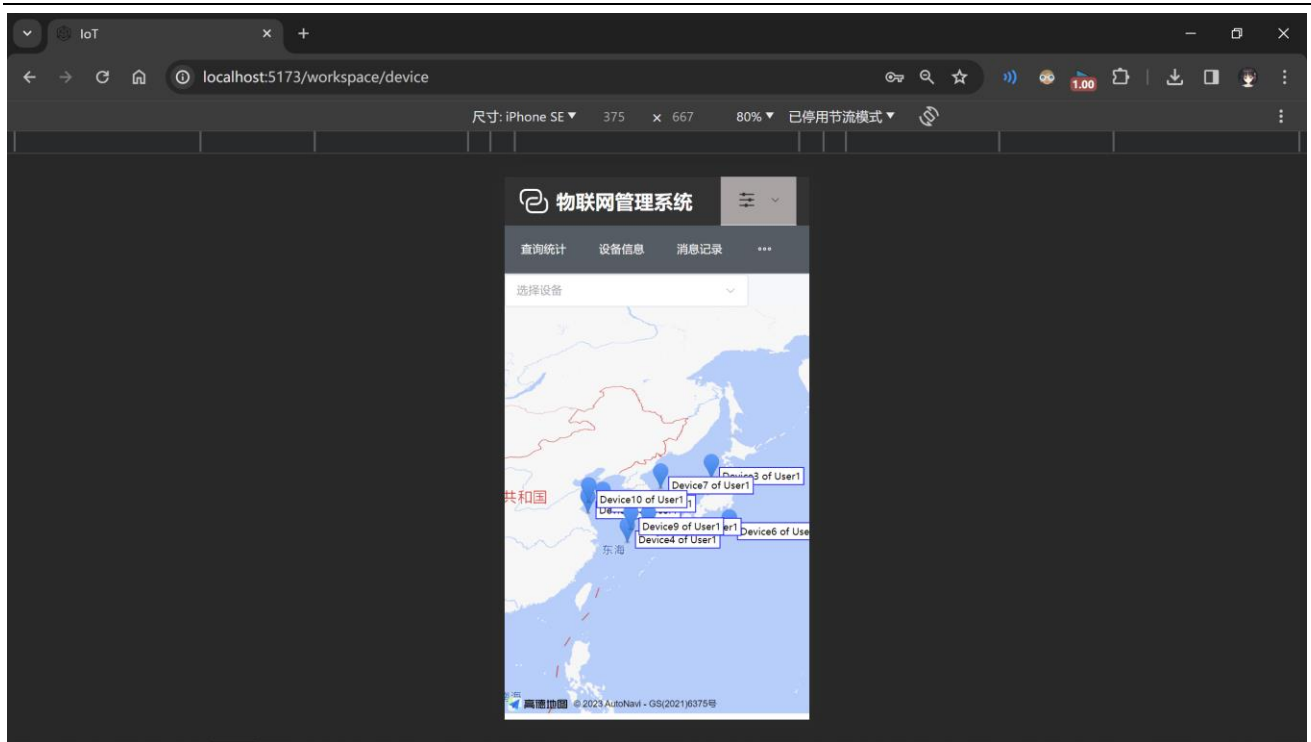


BS2023 物联网管理系统





BS2023 物联网管理系统



9 附录

9.1 项目进度安排

时间段	计划进度
2023.10	完成系统设计与初步框架搭建
2023.11 上中旬	系统前后端代码开发
2023.11 下旬	系统集成与测试及相关文档编写
2023.12 上旬	用户手册等文档编写

9.2 备注

本设计文档内容仅供参考，最终效果以实现代码为准。