

**本科实验报告**

|  |  |
| --- | --- |
| 课程名称： | B/S体系软件设计 |
| 姓 名： | 周炜 |
| 学 院： | 计算机科学与技术学院 |
| 系： | 计算机科学与技术系 |
| 专 业： | 计算机科学与技术 |
| 学 号： | 3210103790 |
| 指导教师： | 胡晓军 |

2023年 10月 10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： 课程大作业

实验项目名称： 物联网设备管理平台

学生姓名： 周炜 专业： 3210103790 学号： 3210103790

同组学生姓名： 指导老师： 胡晓军

实验地点： 实验日期： 年 月 日

目录

[1 背景 3](#_Toc148642553)

[2 系统设计 4](#_Toc148642554)

[2.1 实验要求 4](#_Toc148642555)

[2.2 性能需求 4](#_Toc148642556)

[2.3 系统需求 6](#_Toc148642557)

[2.4 系统结构设计 7](#_Toc148642558)

[2.4.1 前端技术框架 7](#_Toc148642559)

[2.4.2 后端技术框架 8](#_Toc148642560)

[2.4.3 MQTT服务器 8](#_Toc148642561)

[3 数据处理 8](#_Toc148642562)

[3.1 数据建表 8](#_Toc148642563)

[3.1.1 用户信息 8](#_Toc148642564)

[3.1.2 设备信息 9](#_Toc148642565)

[3.1.3 设备消息 10](#_Toc148642566)

[3.2 ER关系图 10](#_Toc148642567)

[4 系统界面设计 11](#_Toc148642568)

[4.1 网站首页 12](#_Toc148642569)

[4.2 问题界面 13](#_Toc148642570)

[4.3 登录界面 14](#_Toc148642571)

[4.4 用户模式 15](#_Toc148642572)

[4.5 设备信息界面 17](#_Toc148642573)

[4.6 管理员模式 20](#_Toc148642574)

[4.7 手机端界面展示 21](#_Toc148642575)

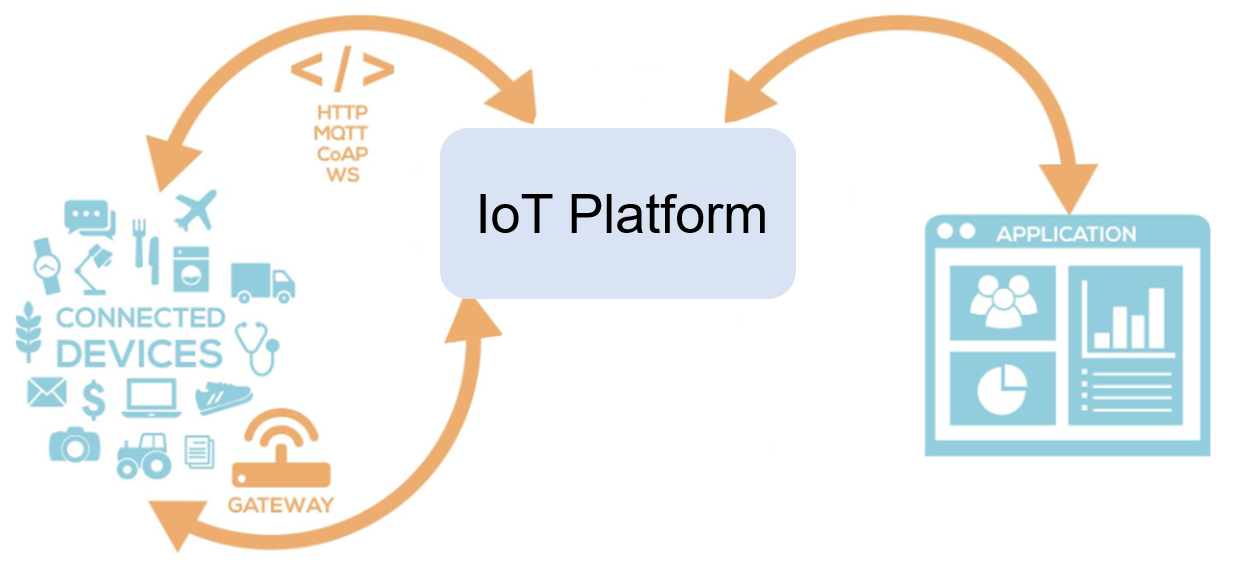
[5 网站部署 23](#_Toc148642576)

[6 时间安排 24](#_Toc148642577)

# 背景

物联网（IoT，互联网物联网）是互联网的延伸和扩展，将各种信息传感设备与网络结合形成一个庞大网络，实现人、机、物在任何时间、任何地点的互联互通。物联网广泛应用于工业、农业、环境、交通、物流和安保等基础设施领域，推动了智能化发展，优化资源分配，提高行业效率。同时，在家居、医疗健康、教育、金融、旅游等领域改善了生活质量。

本项目是2023-2024秋冬学期《B/S体系软件设计》的课程项目，旨在为用户提供友好的界面，适配手机移动端，并提供必要的软件项目文档，以便掌握web应用开发技术和流程。我的物联网应用网站以B/S结构设计，实现数据接收、统计分析，以及用户信息和设备配置的修改。该项目涵盖了web前后端、mqtt服务器和相关文档，由一人独立完成。



# 系统设计

## 实验要求

需要实现的基本功能如下：

1. 搭建一个**mqtt服务器**，能够接收指定的物联网终端模拟器发送的数据。

2. 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。

3. 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、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q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）

增强功能： 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

## 性能需求

**兼容性与多人并发访问**

系统必须兼容主流浏览器，以满足广泛的用户需求。此外，系统应支持多人同时的并发访问，以确保在高负载时仍能提供良好的性能。

**用户操作响应时间**

用户操作的响应时间应在1秒内，这是为了确保用户体验的流畅性。当用户与系统进行交互时，快速响应对于提高用户满意度至关重要。

**及时检测与解决问题**

系统应及时检测与设备的通信中断和数据库连接问题，以避免用户长时间等待。这包括实时监测系统运行状态，识别问题，并采取适当的措施来解决问题。

**页面加载时间**

在一般情况下，页面加载时间应在1秒内完成，即使在高峰时期也应在5秒内完成。这确保了用户能够迅速访问所需的信息，提高了用户的满意度。

**维护与重启**

系统应保证在一周内不超过一次维护与重启。这是为了确保系统的连续性和稳定性，避免对用户造成不便。

**数据有效性与安全性**

除了系统的稳定性，数据的有效性和安全性也是至关重要的。以下是确保数据有效性和安全性的关键措施：

数据输入检查与安全性

在用户输入账号密码时，系统应对数据输入进行有效性检查，确保数据的准确性和完整性。同时，系统应保障数据的安全性，采用加密等措施保护用户的隐私信息。

**可视化呈现**

设备的运动轨迹和统计信息必须以可视化的图标等方式呈现，以确保数据的美观和准确性。用户能够清晰地理解和分析数据，提高了系统的实用性。

**减少出错几率**

系统应通过程序控制出错几率，减少系统因用户人为的错误引起的破坏。开发者应考虑各种可能发生的问题，确保系统尽可能不容易出现错误，从而提高了系统的可靠性。

## 系统需求

**软件层面**

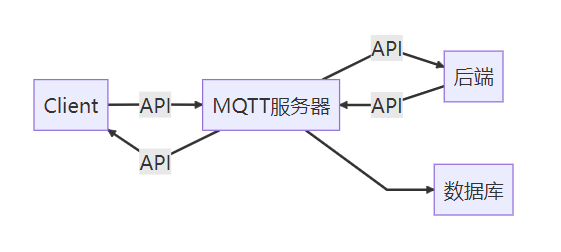
|  |  |
| --- | --- |
| 负载能力 | 系统需要具备足够的负载能力，能够同时满足至少100人次的访问需求。这包括处理用户请求和数据传输的能力 |
| 数据存储能力 | 系统需要具备强大的数据存储能力，以容纳大量的用户数据和设备信息。这包括数据库的设计和优化，以确保数据的高效存储和检索 |
| 网络服务吞吐能力 | 系统需要保证网络服务的吞吐能力，以确保及时响应用户请求，减少延迟。这包括网络架构的设计和优化，以提高数据传输的效率 |
| 数据安全特性 | 系统需要具备强大的数据安全特性，包括用户身份验证、数据加密和访问控制，以保护用户信息和系统数据的安全 |
| 移动终端兼容性 | 系统必须兼容Android和iOS移动终端，以确保用户可以在不同移动平台上使用应用，并获得一致的体验 |
| 客户端浏览器兼容性 | 用户应能够使用常见的浏览器（如IE、Chrome、Firefox、Opera）访问本网站。此外，系统也应支持其他浏览器，以确保用户在各种环境下都能够体验到主要功能 |
| 移动端适配 | 为了提供更好的用户体验，本网站需要在手机浏览器、微信等内置浏览器中友好显示。这需要针对移动端进行样式适配和响应式设计 |

**硬件层面**

|  |  |
| --- | --- |
| CPU性能 | 服务器的CPU主频应大于2GHz，以确保系统能够处理大量的并发请求，并保持高性能 |
| 硬盘容量 | 服务器硬盘容量应大于200GB，以满足数据存储的需求。此外，硬盘转速应大于或等于5400转/分钟，以提高数据读写速度 |
| 其他硬件需求 | 除了上述要求之外，其他硬件应满足正常的使用需求，以确保系统的稳定性和可靠性 |

## 系统结构设计

该项目采用了前后端分离的Web开发技术，并搭建了独立的MQTT服务器，用于接收模拟客户端发送的数据并将其存储在数据库中



|  |  |
| --- | --- |
| 前端 | React + Ant Design +ECharts +OpenStreetMap+yarn+nvm |
| 后端 | Python flask |
| 数据库 | MySQL |
| mqtt服务器 | Mosquitto+Paho |
| 测试工具 | ApiPost |

更为具体的介绍见下：

### 前端技术框架

项目主要使用以下前端技术框架和工具：

React：采用了由Facebook开发的React，这是一个强大的Web开发框架，它使用JavaScript构建用户界面。React采用虚拟DOM技术和声明式设计，可以高效而灵活地开发Web应用。

NVM： 全称node.js version management，是一个nodejs的版本管理工具

Ant Design：使用Ant Design组件库来提供丰富的UI组件，以构建出漂亮的前端界面。

ECharts：ECharts库用于创建交互式图表，提供了数据可视化的功能。

OpenStreetMap：地图服务工具，用于设备位置的可视化。

### 后端技术框架

后端承担以下任务：

**数据库管理：**我们使用MySQL作为关系型数据库来存储项目数据。

**Web服务端：**我们采用Python Flask框架进行Web服务端开发，集成了MQTT信息、MyBatis、Redis、Jackson等中间件，以支持数据库操作和API接口开发。Python Flask是一个轻量级的Web框架，能够简化应用的搭建和开发过程。

### MQTT服务器

项目的MQTT服务器采用Mosquitto，但由于其无法存储消息，因此我们编写了一个Python服务端脚本，利用Paho库接收模拟终端客户端发送的消息，并将其存储到数据库中。

# 数据处理

## 数据建表

### 用户信息

信息表：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 描述 | 备注 |
| id | int | 用户的id | 主键，非空，自增 |
| name | varchar(128) | 用户名 | 非空 |
| password | varchar(128) | 账号密码 | 非空 |
| email | varchar(128) | 用户邮箱 | 非空，并且UNIQUE |
| workunit | varchar(128) | 工作单位 | 非空 |

转换为SQL语句：

CREATE TABLE user\_info

(

    id       INT UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

    email    VARCHAR(128) DEFAULT '' NOT NULL UNIQUE,

    name     VARCHAR(128) DEFAULT '' NOT NULL UNIQUE,

    password VARCHAR(128) DEFAULT '' NOT NULL

);

### 设备信息

信息表：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 描述 | 备注 |
| id | int | 设备的id | 主键，非空，自增 |
| name | varchar(128) | 设备的名称 | 非空，Unique |
| description | varchar(256) | 设备的描述信息 | 非空 |
| userid | int | 设备的用户的id | 非空，foreign key |
| typology | str | 设备的类型 | 非空 |
| timestamp | datetime | 最后发送消息的时间 | 非空 |
| usedId[[1]](#footnote-1) | int | 该消息属于的用户 | 非空 |

转换为SQL语句：

create table device\_message

(

    id        int unsigned auto\_increment   primary key,

    alert     int           default 0                 not null,

    clientId  varchar(128)  default ''                not null,

    info      varchar(128)  default ''                not null,

    lat       float         default 0.0000            not null,

    lng       float         default 0.0000            not null,

    timestamp datetime      default CURRENT\_TIMESTAMP not null on update CURRENT\_TIMESTAMP,

    value     int           default 0                 not null,

    userId    int           default 0                 not null

)charset = utf8;

### 设备消息

信息表：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 描述 | 备注 |
| id | int unsigned | 设备消息的唯一标识符 | 主键，非空，自增 |
| alert | int | 警报标志 | 非空 |
| clientId | varchar(128) | 客户端ID | 默认为空字符串，非空 |
| info | varchar(128) | 消息信息 | 默认为空字符串，非空 |
| timestamp | datetime | 消息时间戳 | 默认为当前时间，非空 |
| lat | float | 纬度坐标 | 默认为0.0000，非空 |
| lng | float | 经度坐标 | 默认为0.0000，非空 |

转换为SQL语句：

create table device\_message(

    id        int unsigned auto\_increment   primary key,

    alert     int           default 0                 not null,

    clientId  varchar(128)  default ''                not null,

info      varchar(128)  default ''                not null,

    timestamp datetime      default CURRENT\_TIMESTAMP not null on update CURRENT\_TIMESTAMP,

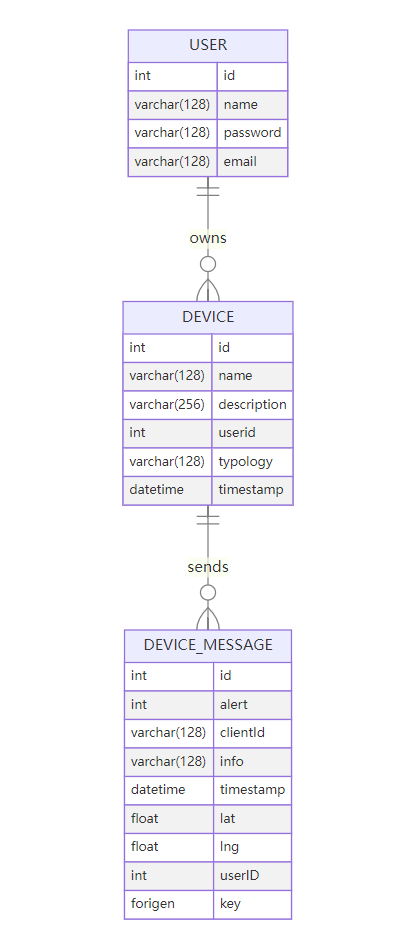
    lat       float         default 0.0000            not null,

    lng       float         default 0.0000            not null,

);

## ER关系图

将上述的三张表格用Mermaid绘制为ER关系图如下（图的字较小，请放大查看）

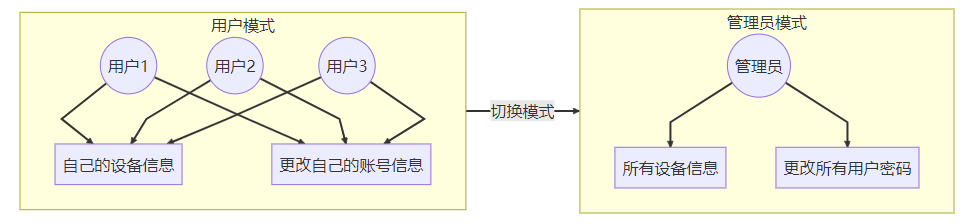


# 系统界面设计

本网站分为用户模式和管理员模式两个模式

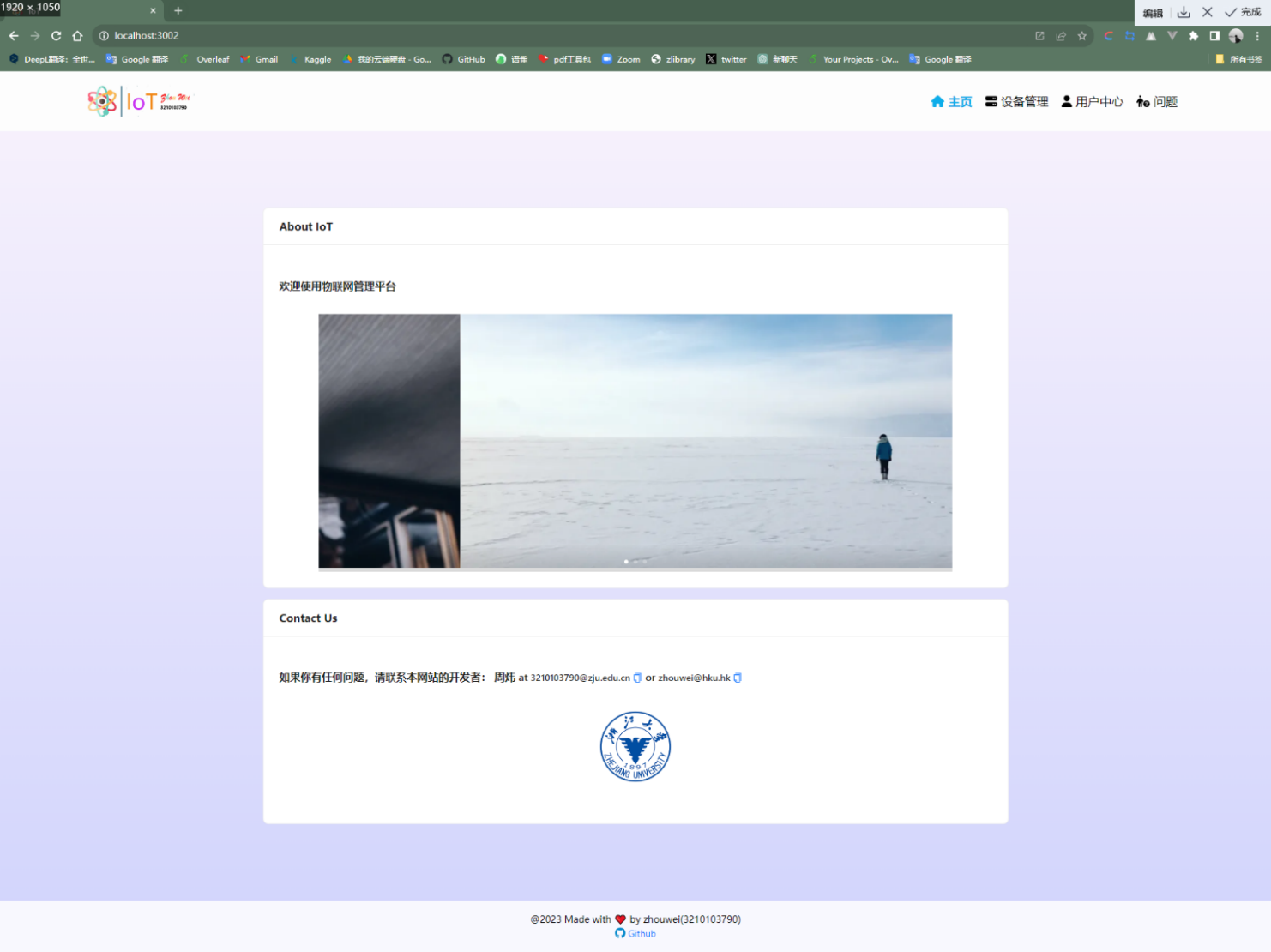
用户模式：用户可以查看到自己的设备相关信息并且可以更改自己的账号信息

管理员模式： 管理员可以查看到所有的设备相关信息并且可以更改已注册用户的密码

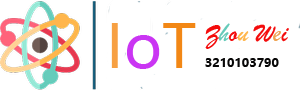


## 网站首页

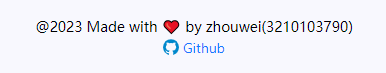
进入该网站后显示的首先是首页的界面，是对本网站的一些介绍和开发者信息（我的信息）



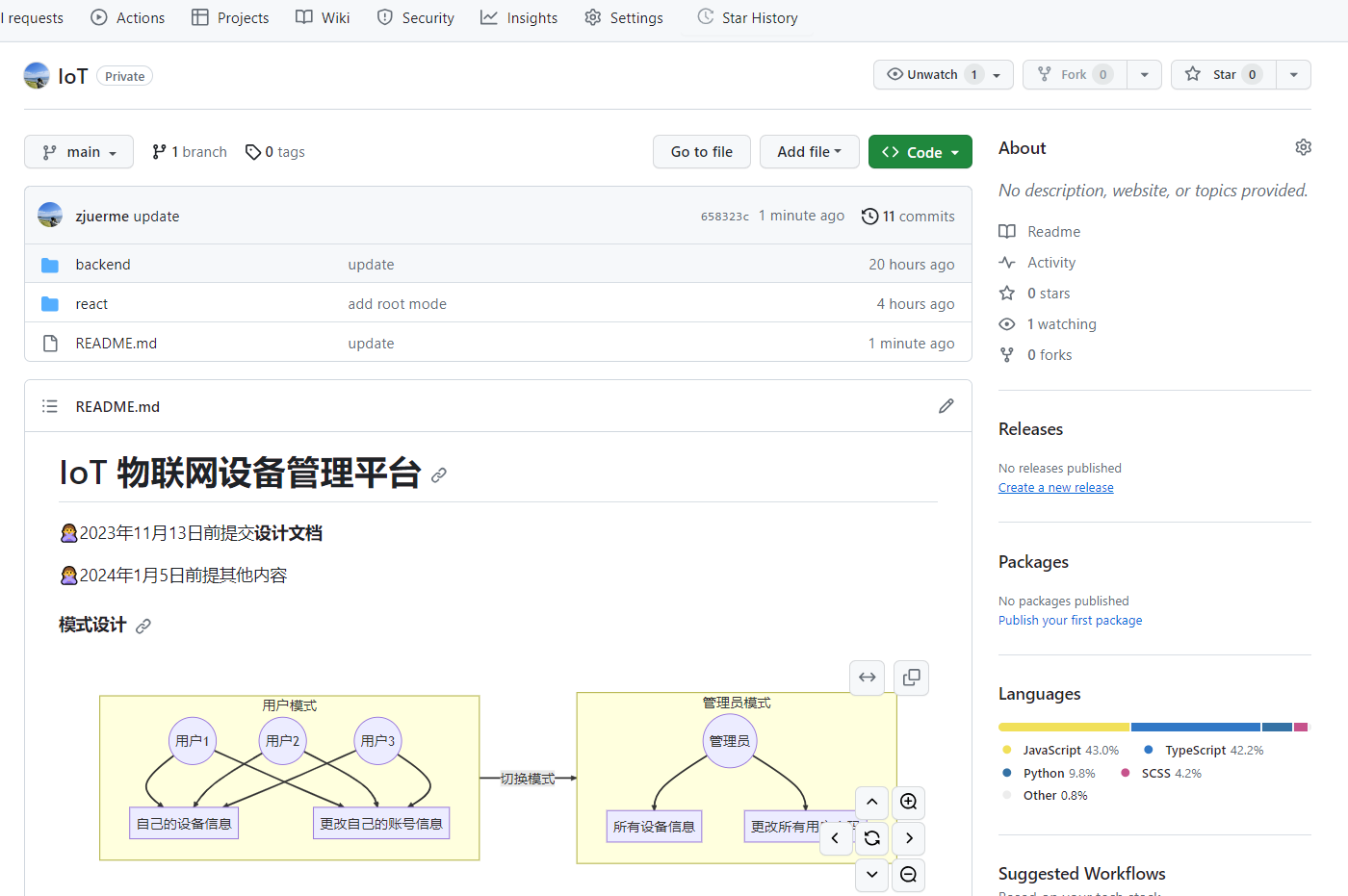
并且我精心设计了该网站的Logo，上面有我的学号和名字



并且在页脚有GitHub链接通往我的Github仓库，其中存储了我的开发过程（该仓库的状态为private）

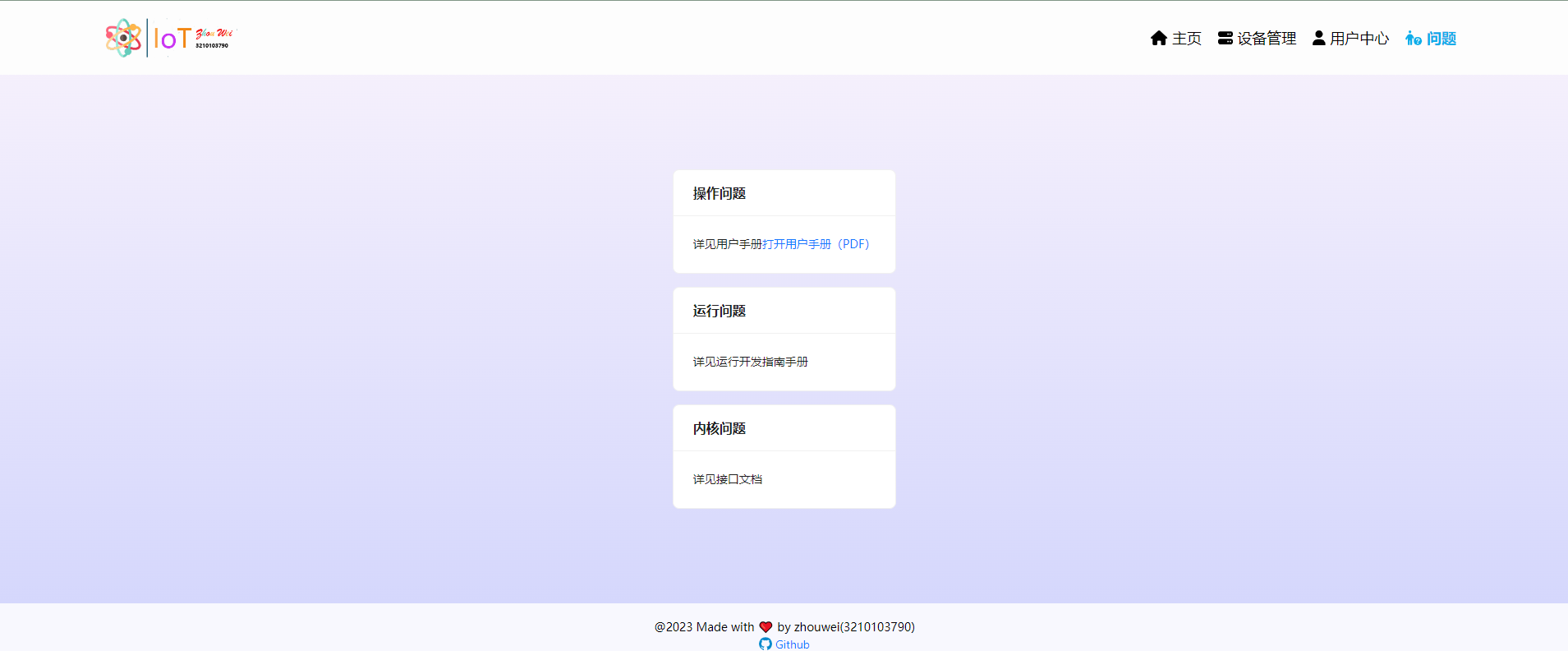


点击后会打开Github仓库



## 问题界面

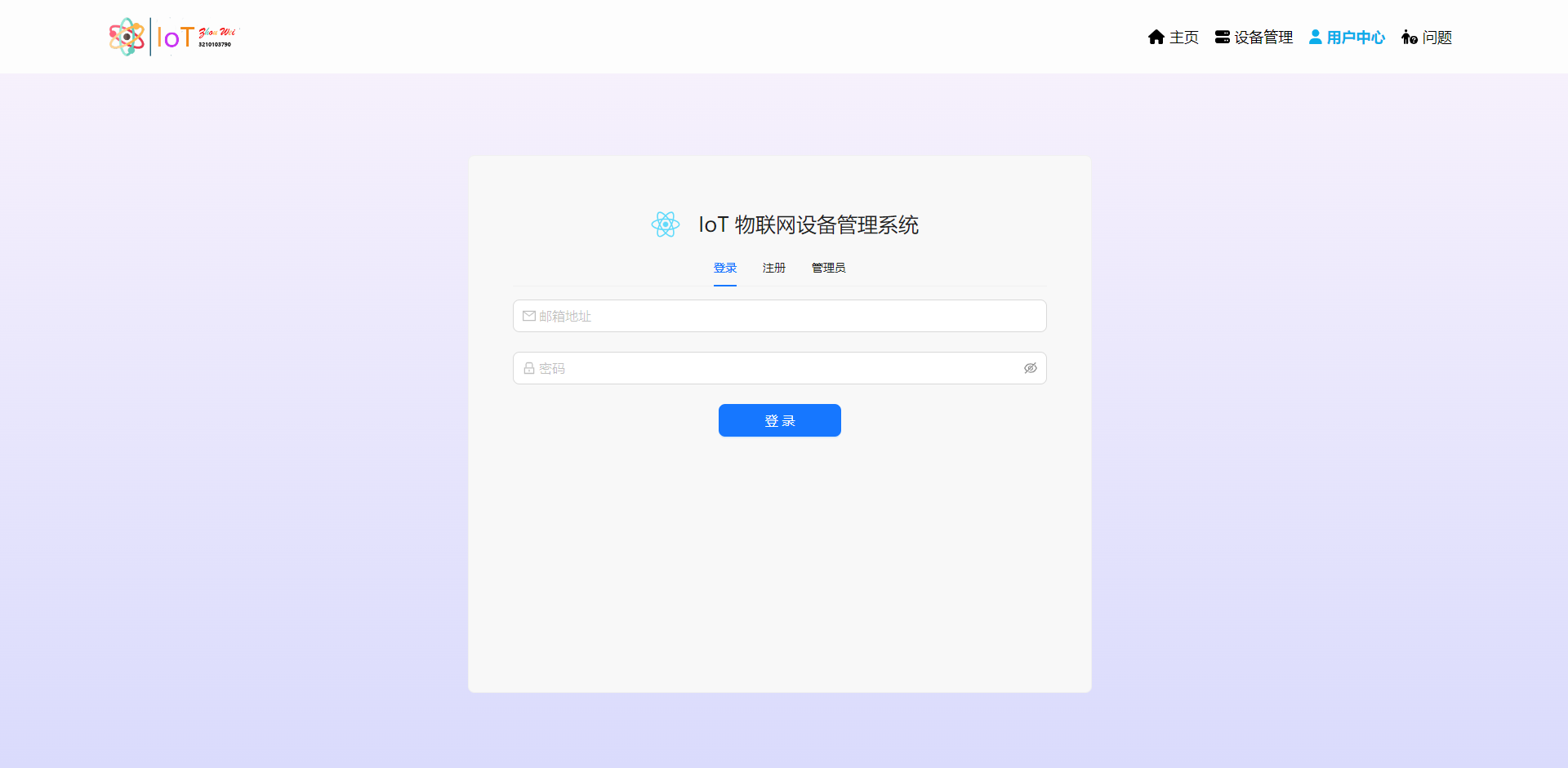
将问题进行了分类并且提出了相关的问题建议，点击所对应的文档会打开对应的PDF文件（由于本网站我目前写设计文档的时候只是大概完成了前端的搭建，后端还未完善，因此用户手册也还没有撰写，故在这里就不展示打开pdf的效果了）



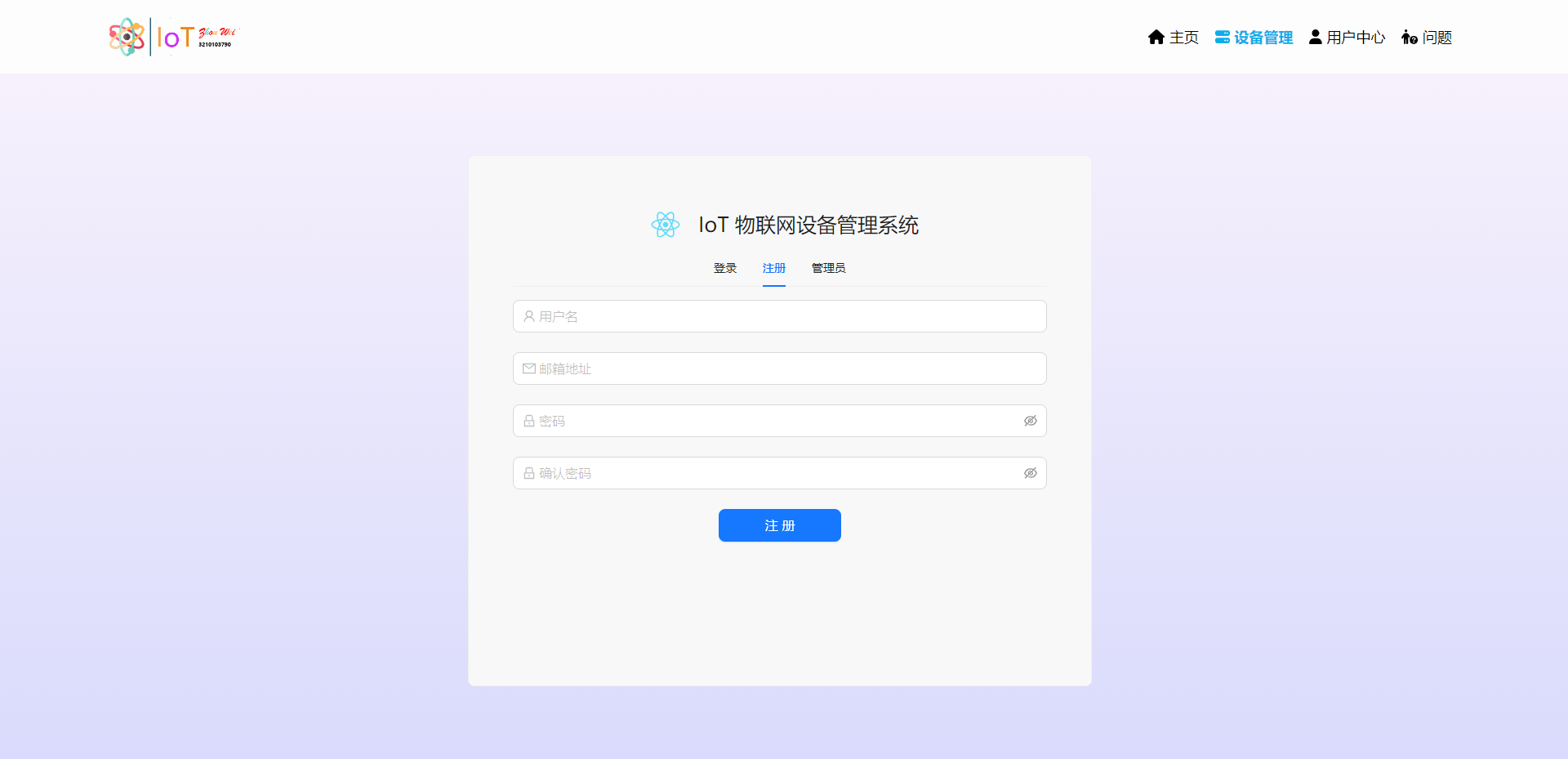
## 登录界面

在未登录时点击用户中心和设备管理会显示登录界面，登录界面分为3个部分组成，分别展示如下：

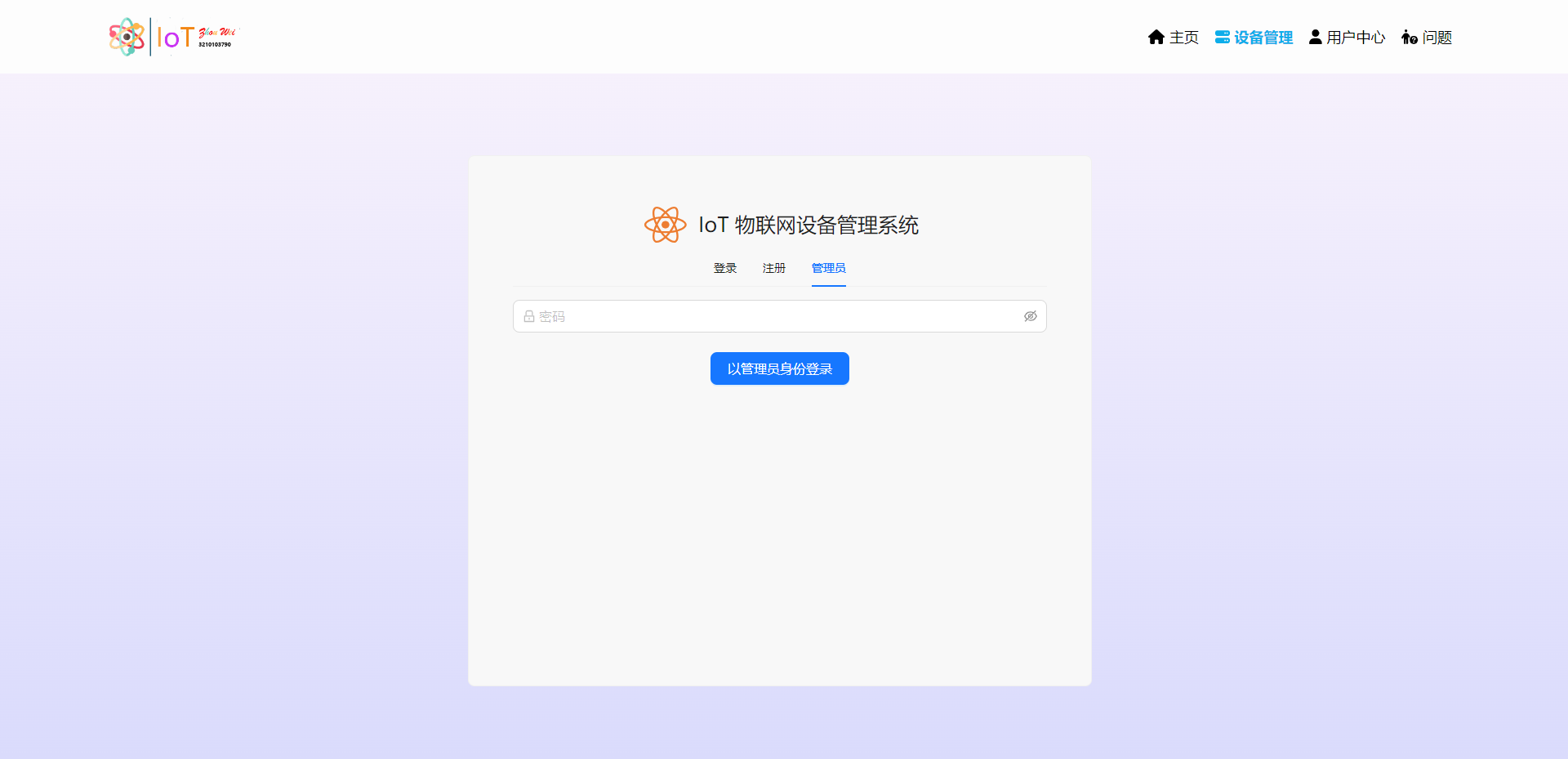
**用户模式登录：**



**用户注册：**

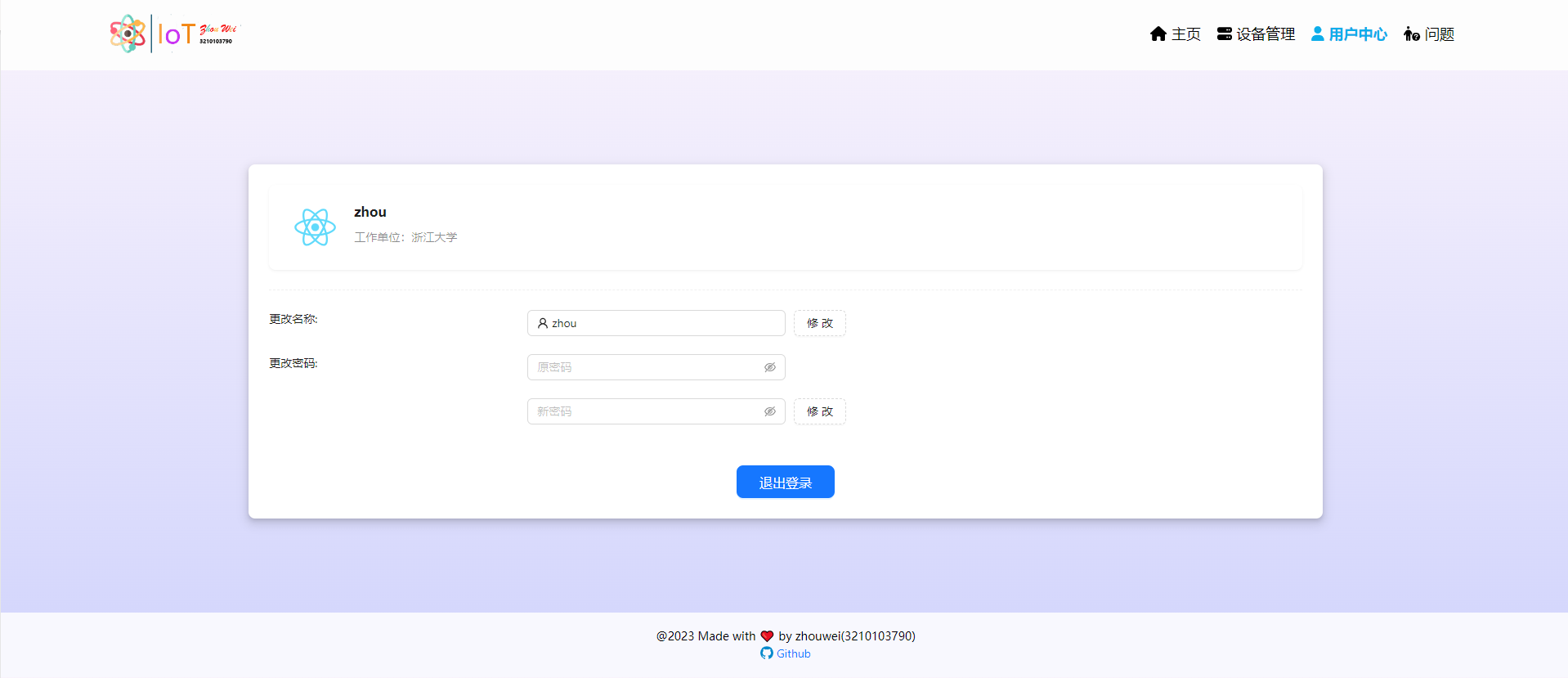


**管理员模式登录：**

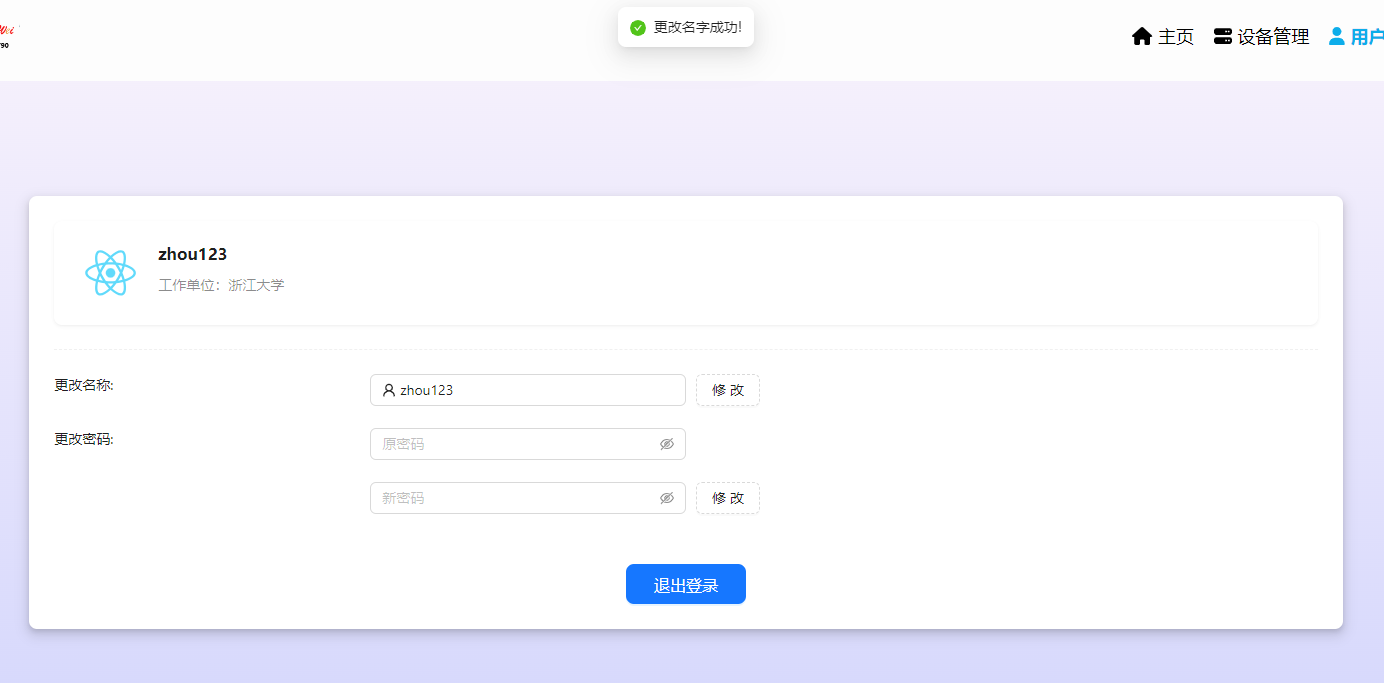


## 用户模式

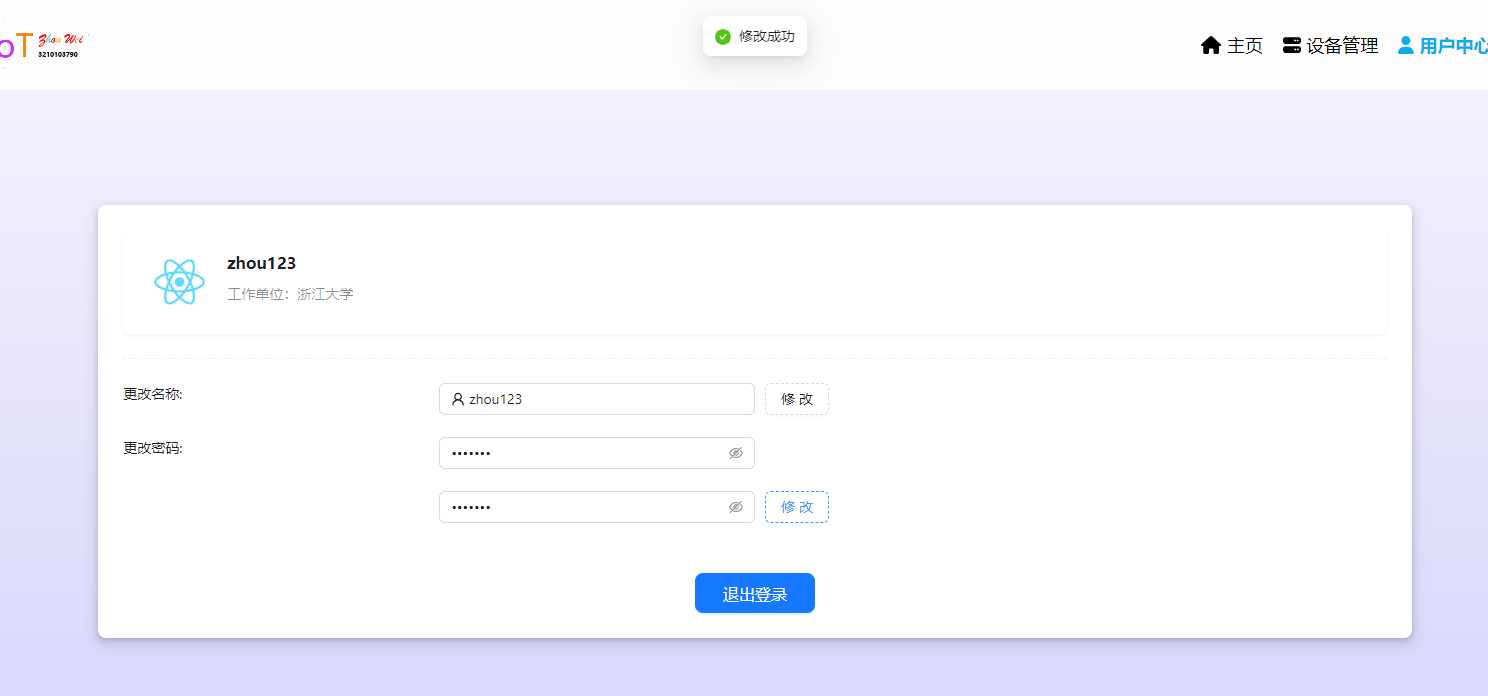
以用户模式登录后的个人界面展示如下



可以修改用户名

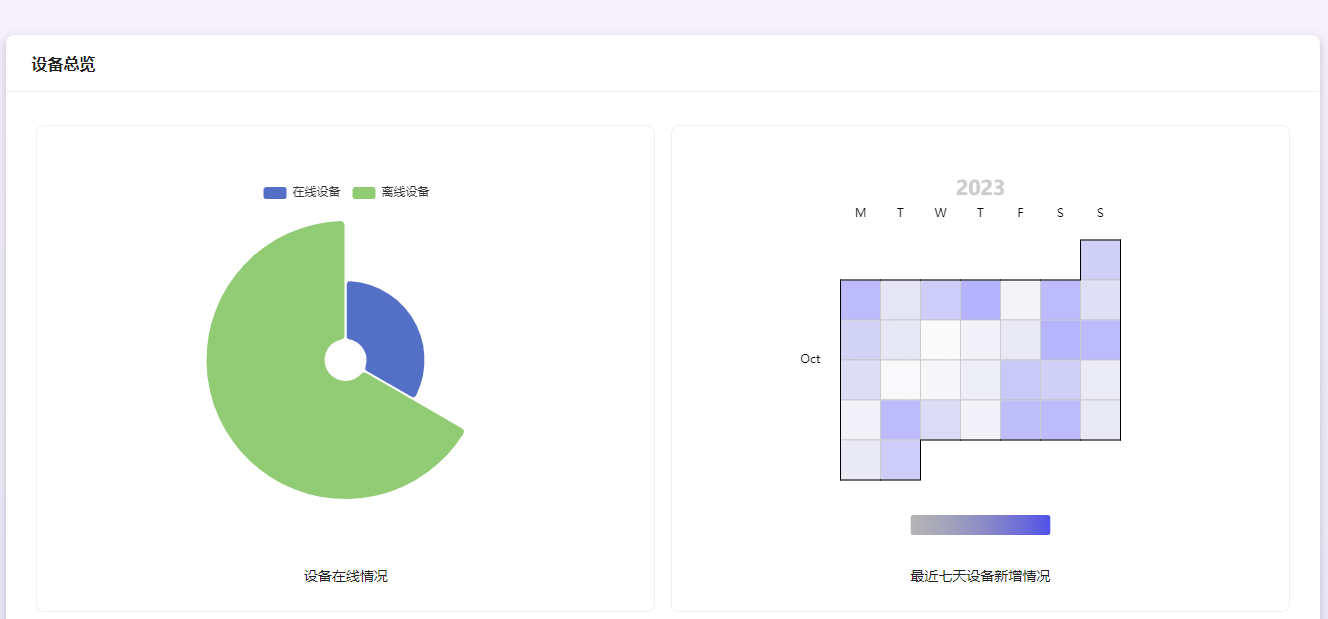


也可以修改密码，并且修改

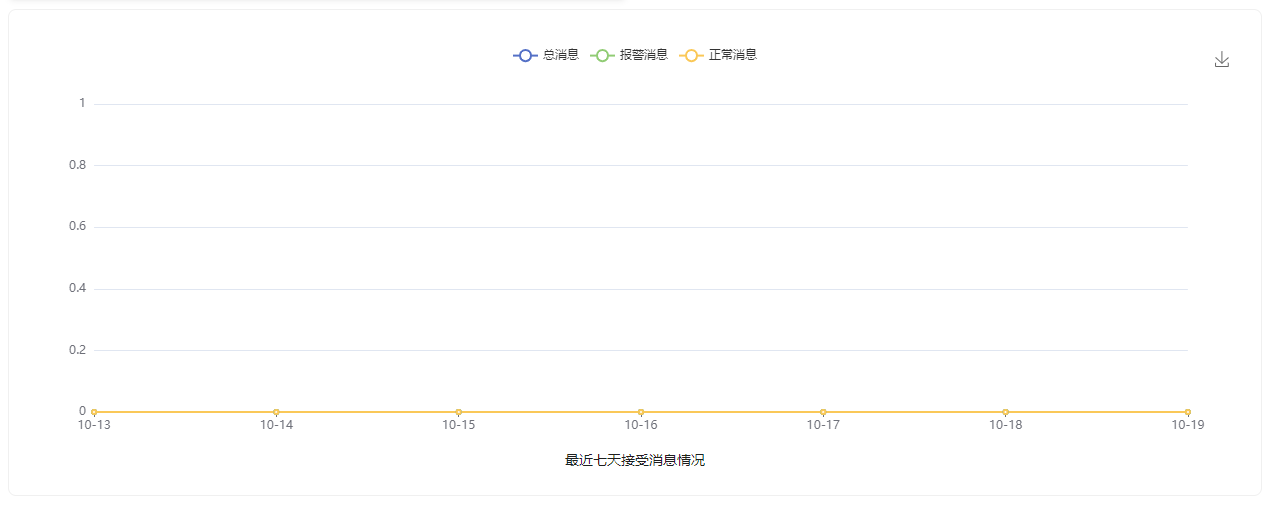


## 设备信息界面

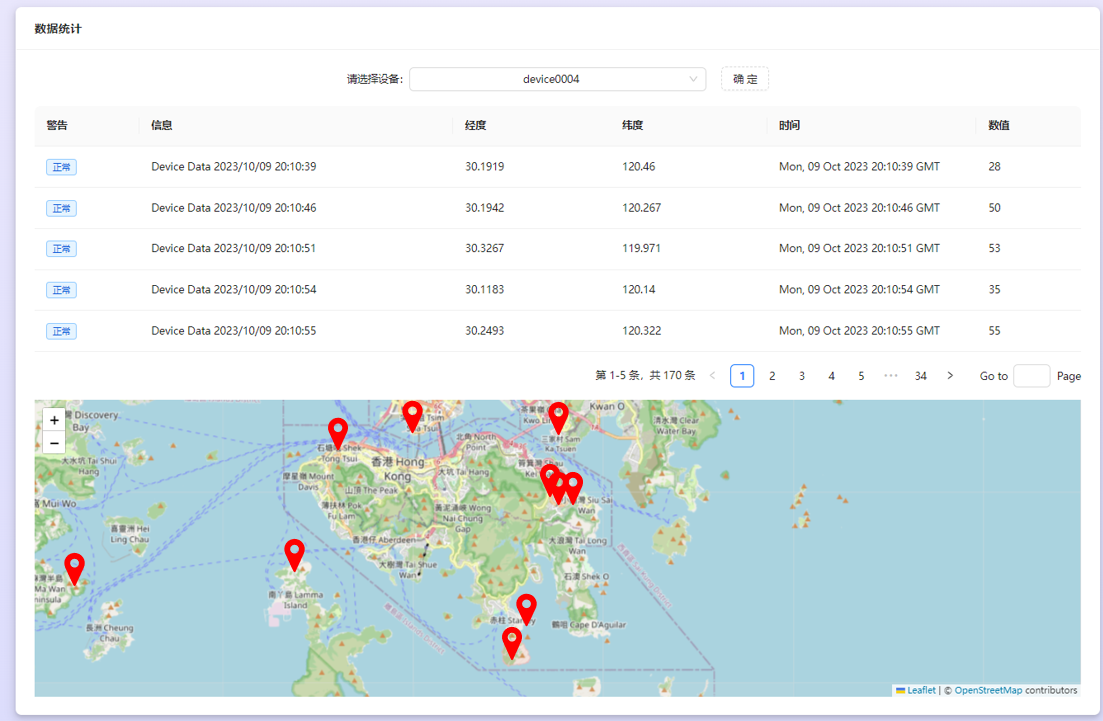
设备总览



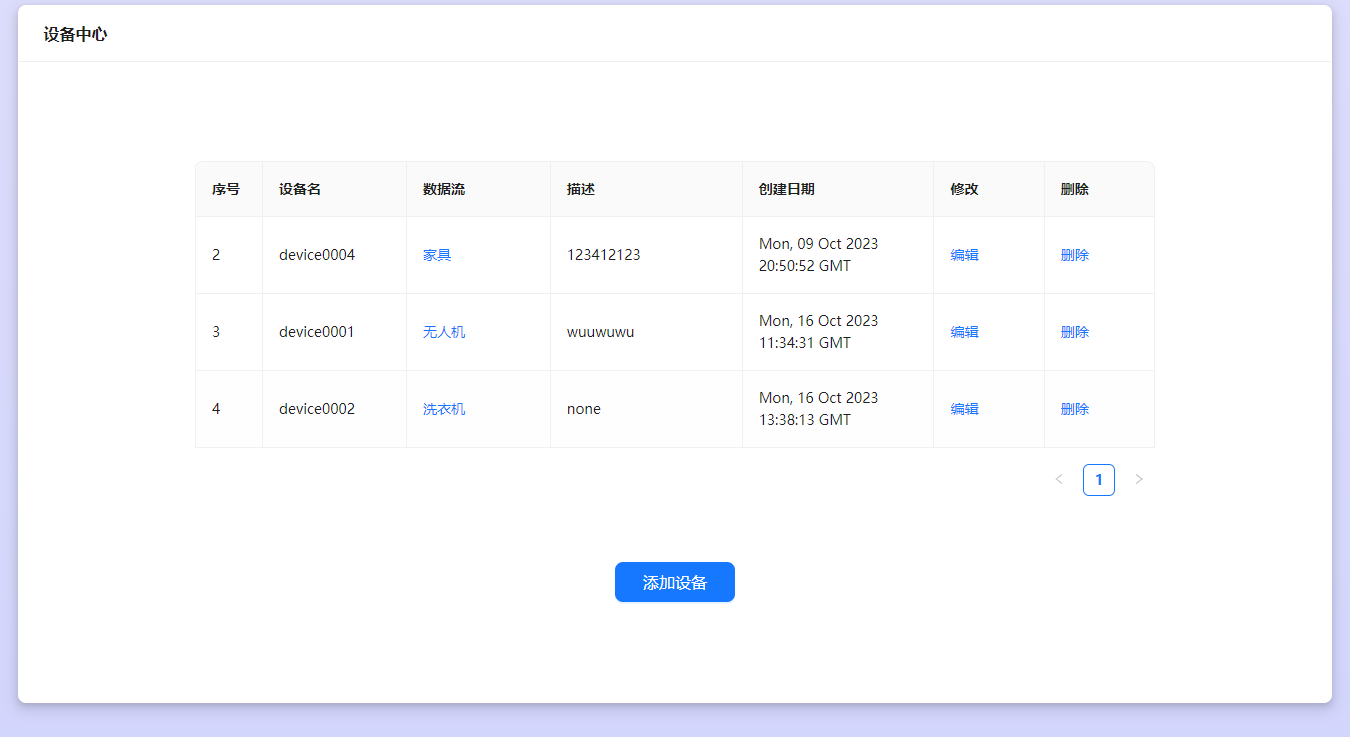
消息显示统计（堆叠图）



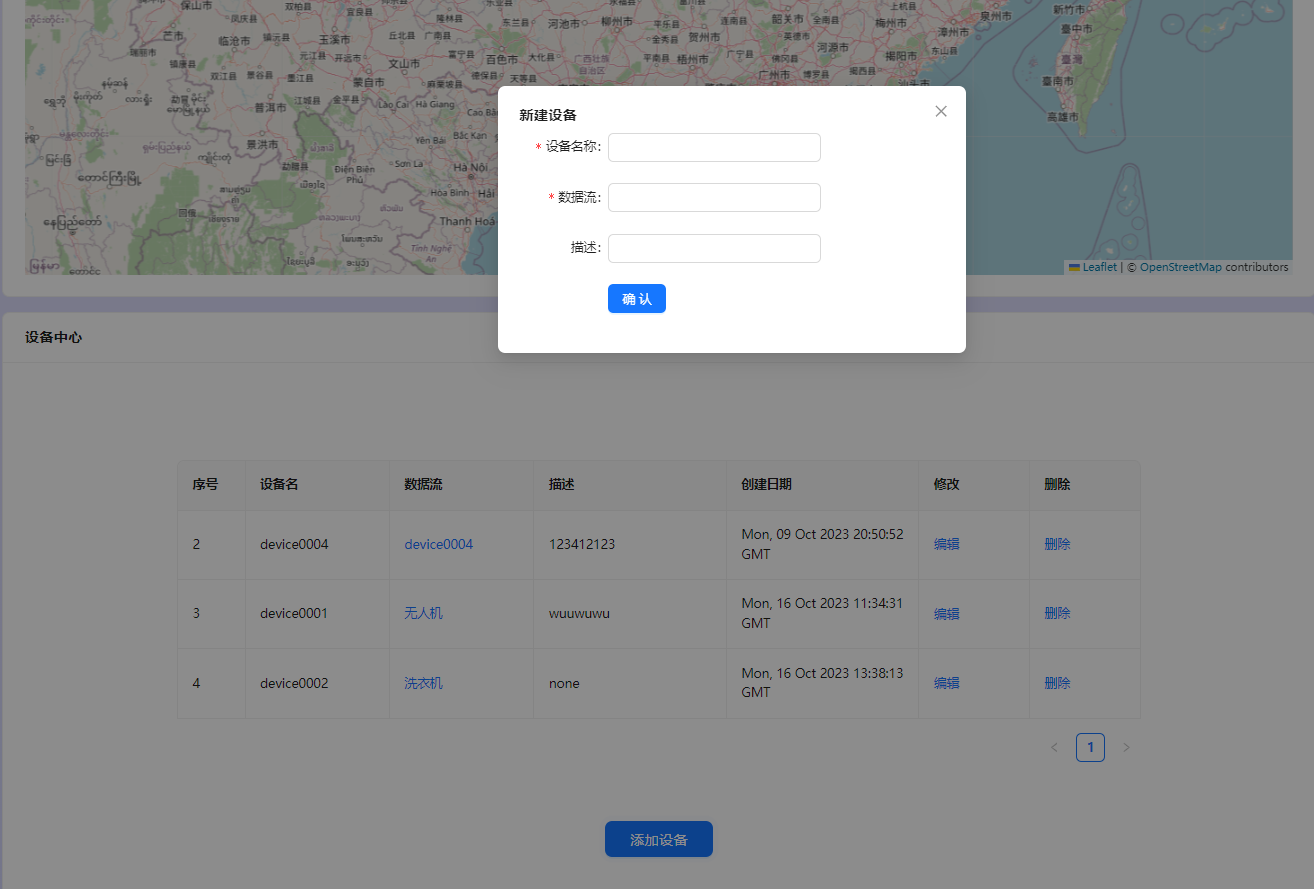
设备消息具体内容展示



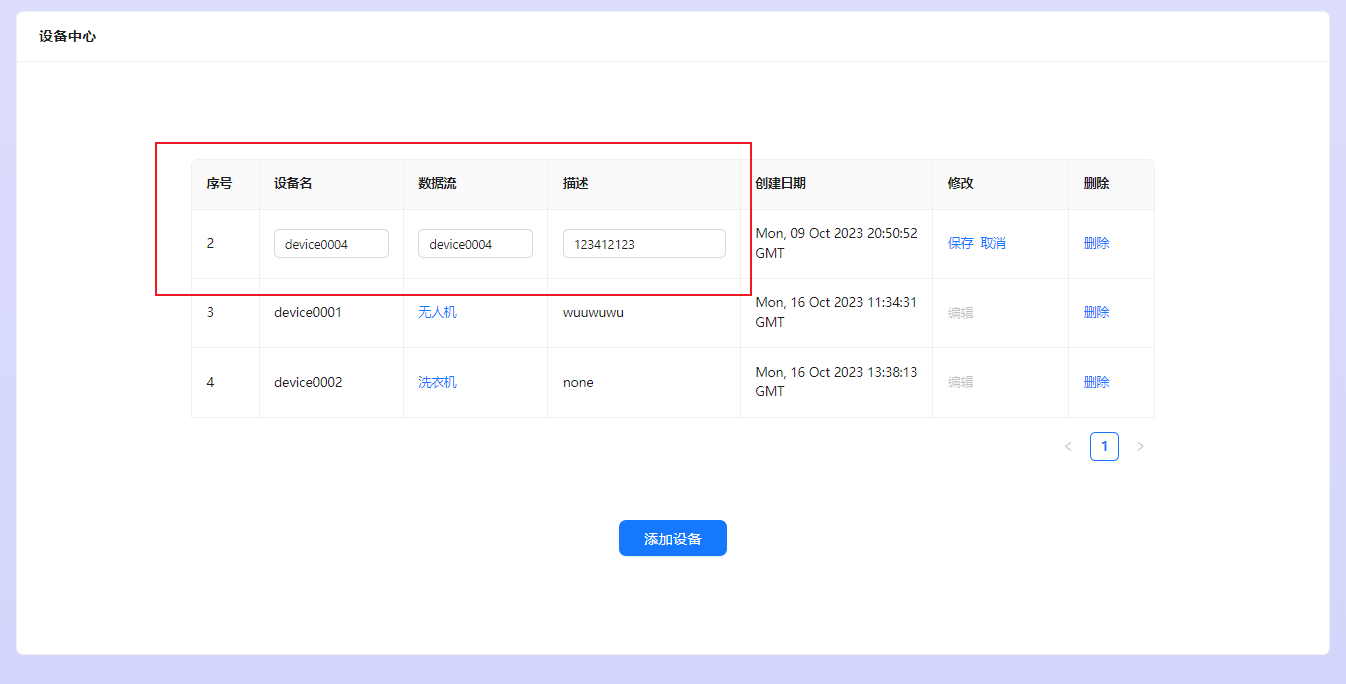
设备中心界面



新增设备子界面

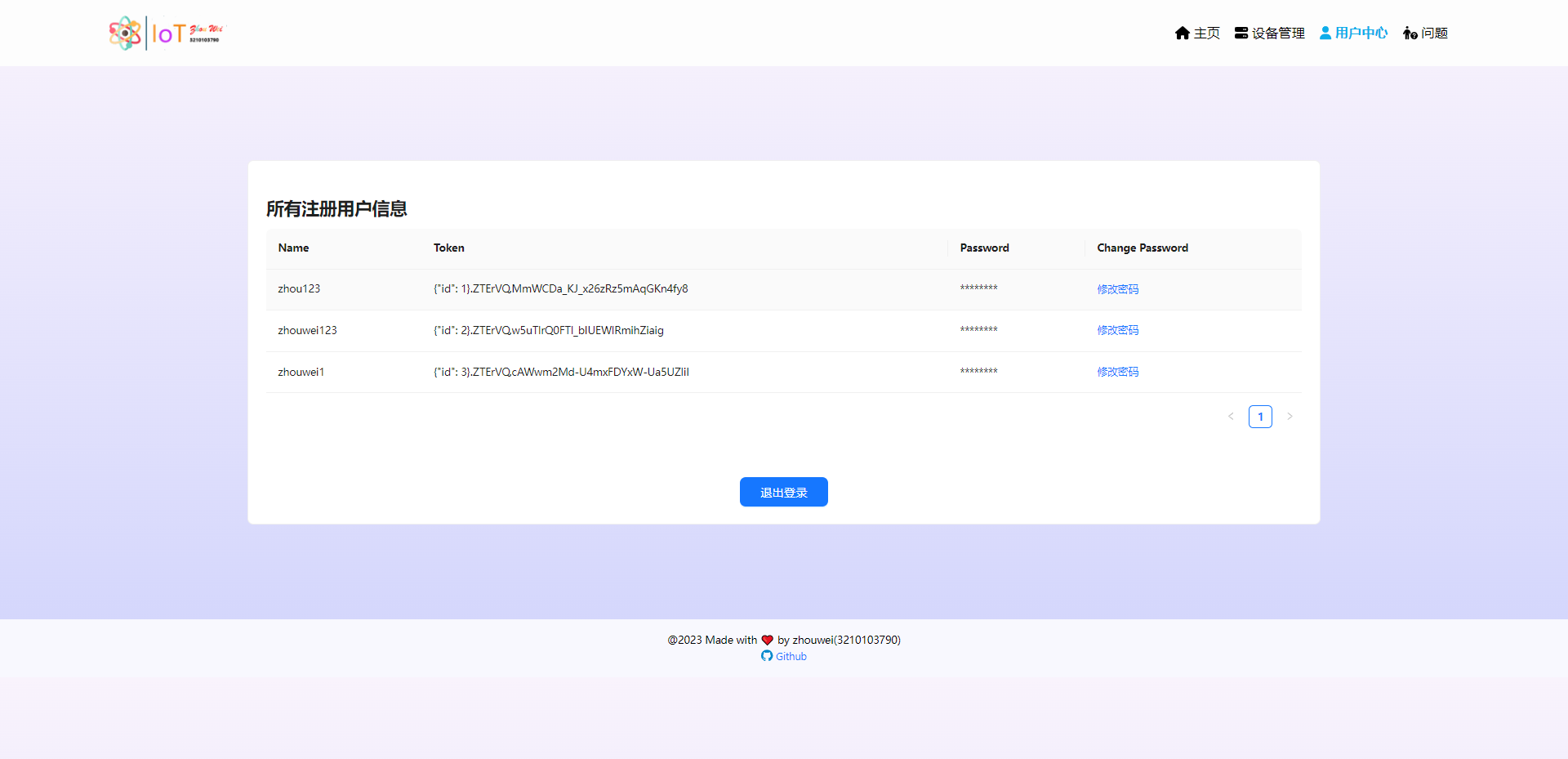


修改设备信息

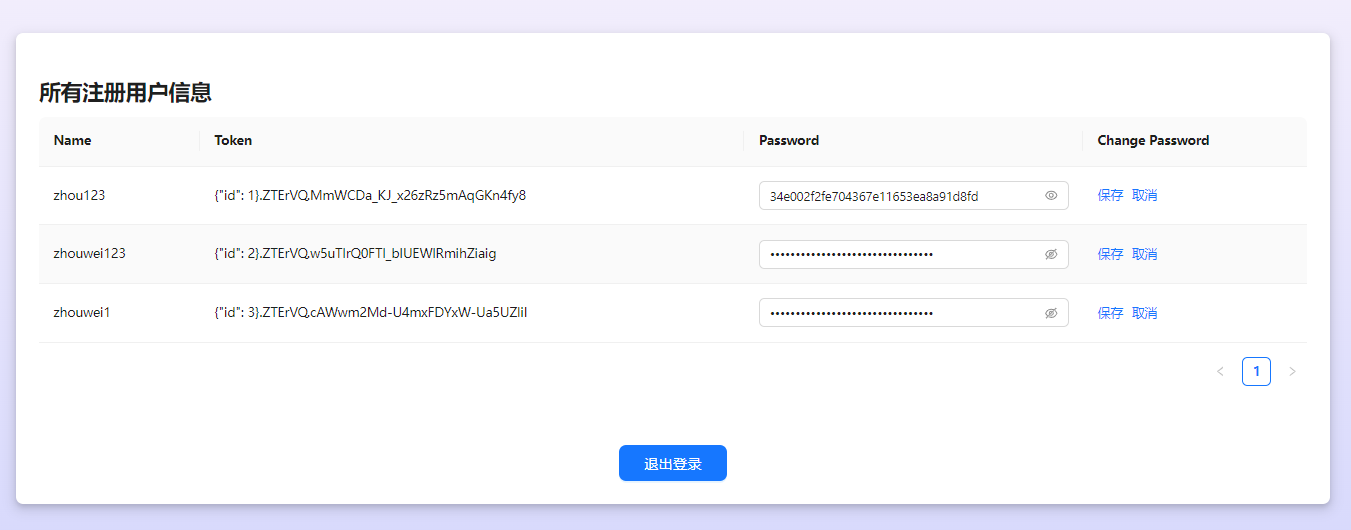


## 管理员模式

当使用管理员模式登录时用户中心显示的内容不再是用户个人的信息，而是所有用户的信息

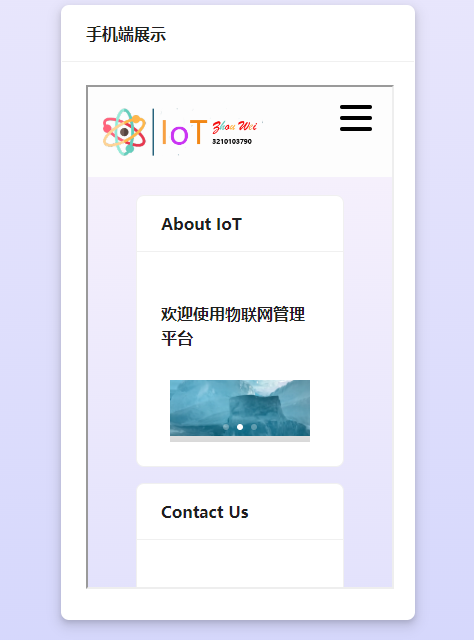


并且点击修改任意用户的密码：



## 手机端界面展示

功能与上述一致不再赘述，只展示效果图







# 网站部署

考虑到提交后教师对于网站相关功能的测试较为困难，因此考虑对网站进行部署，部署的方案目前有2套，视整个网站完成后的情况进行尝试

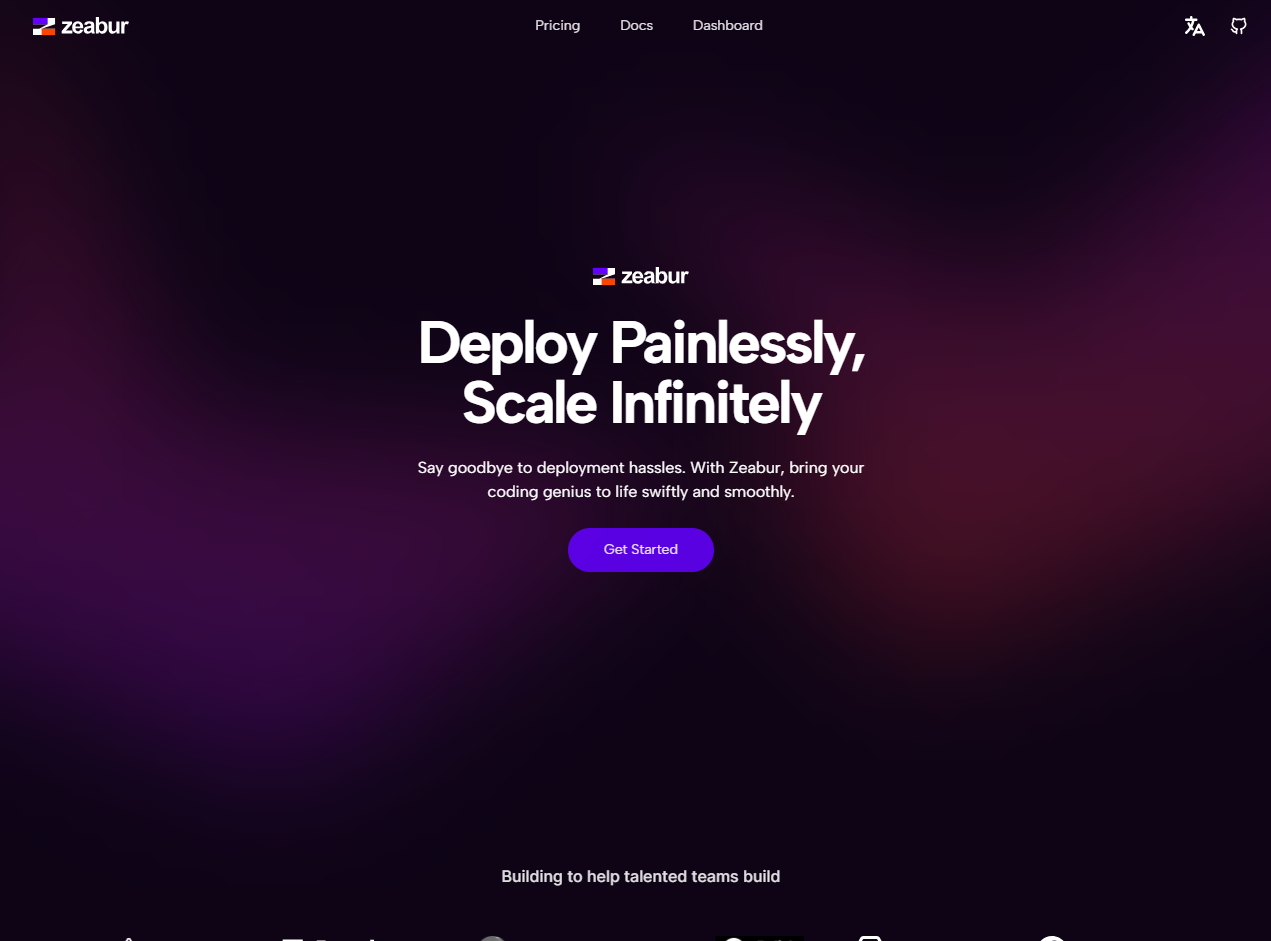
1. **腾讯云学生免费服务器**

相关内容可以见于https://cloud.tencent.com/act/campus

】

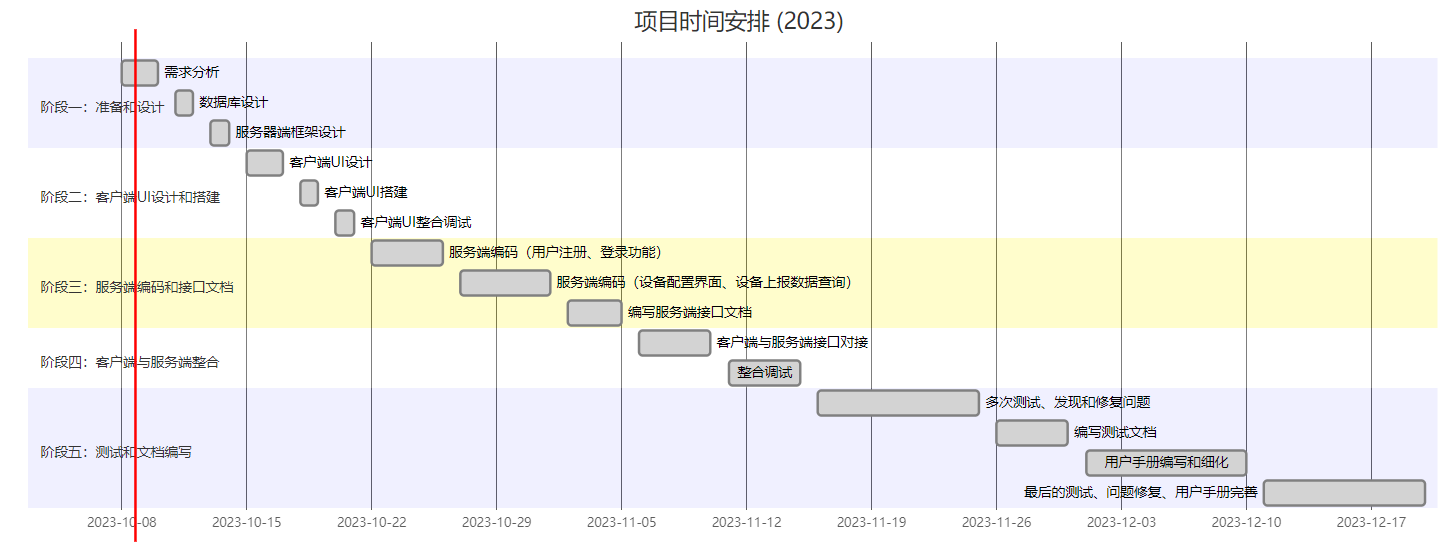
1. **Zeabur**

使用 [Zeabur](https://zeabur.com/home/) 进行一站式、全自动的项目部署，且自带 CI/CD 和 SSL 证书



# 时间安排

（图的字较小，请放大查看）



1. 将消息用uesrId来区分可以，可以解决老师提供的程序消息过密集，并且体现多用户的功能 [↑](#footnote-ref-1)