**目录**

[一、设计概述 1](#_Toc150992226)

[1.1 项目背景 1](#_Toc150992227)

[1.2 技术选型 2](#_Toc150992228)

[1.3 运行环境 2](#_Toc150992229)

[二、需求分析 3](#_Toc150992230)

[2.1 功能性需求 3](#_Toc150992231)

[2.2 非功能性需求 3](#_Toc150992232)

[三、功能模块与层次结构设计 4](#_Toc150992233)

[3.1 用户信息管理模块 4](#_Toc150992234)

[3.2 设备信息管理模块 5](#_Toc150992235)

[3.3 系统整体架构 5](#_Toc150992236)

[3.4 前端架构图 6](#_Toc150992237)

[3.5 后端架构图 7](#_Toc150992238)

[四、类图和ER图 7](#_Toc150992239)

[4.1 类图 7](#_Toc150992240)

[4.2 ER图 8](#_Toc150992241)

[4.3 数据表 8](#_Toc150992242)

[五、系统接口设计 9](#_Toc150992243)

[5.1 用户信息相关接口 9](#_Toc150992244)

[5.2 设备信息相关接口 12](#_Toc150992245)

[5.3 设备消息相关接口 16](#_Toc150992246)

[六、系统界面原型 17](#_Toc150992247)

[6.1 用户登陆页面 18](#_Toc150992248)

[6.2 用户注册页面 18](#_Toc150992249)

[6.3 个人信息页面 19](#_Toc150992250)

[6.4 平台首页 19](#_Toc150992251)

[6.5 设备信息配置页面 20](#_Toc150992252)

[6.6 设备上报数据查询页面 20](#_Toc150992253)

[6.7 设备统计信息总览页面 21](#_Toc150992254)

[6.8 设备轨迹查看页面 21](#_Toc150992255)

[七、预计开发时间轴 21](#_Toc150992256)

[八、附录 22](#_Toc150992257)

[8.1 状态码设计 22](#_Toc150992258)

## 一、设计概述

### 1.1 项目背景

本项目是 2023-2024 春夏学期《B/S 体系软件设计》的课程项目，旨在设计一个物联网设备管理平台。用户在登陆之后可以修改个人信息和物联网设备配置信息，并在可视化界面查看设备信息，包括运动轨迹，设备的统计信息，并对统计信息进行可视化的图表展示，同时需要提供数据上报数据的查询统计的可视化窗口。需要网站界面对用户友好，样式适配手机移动端，可以在手机浏览器和微信等应用内置的浏览器中友好显示，同时也需要提供必要的软件项目文档，使自己了解并掌握一套 web 应用开发技术和开发的总体流程。

### 1.2 技术选型

本项目采用前后端分离的Web开发技术，并在后端编写一系列的Api供前端调用，同时搭建一个独立于前后端的mqtt服务器，用于接收模拟客户端发送的数据并存储在数据库里。整个项目技术选型如下图所示：

* 前端：React全家桶，Ant Design组件库以及Ant Design pro框架，高德地图SDK，npm包管理器，Recharts组件库
* 后端：Java SpringBoot框架，Maven，MyBatis，axios
* 数据库：MySQL
* mqtt服务器：Mosquitto，Paho，Java
* 代码管理工具：Git
* 测试工具：ApiPost，MQTT.fx

### 1.3 运行环境

#### 1.3.1 硬件环境

| Item | Model |
| --- | --- |
| 服务器CPU | 2核及以上 |
| 服务器内存 | 2G及以上 |
| 服务器硬盘 | 40G及以上 |

#### 1.3.2 软件环境

| Software | Version Requirements |
| --- | --- |
| Web服务器操作系统 | Windows7及以上 |
| mqtt服务器操作系统 | Ubuntu 22.04LTS |
| npm | 9.5.1 |
| React | 18.2.0 |
| AntDesign | 5.10.3 |
| 高德地图SDK | 6.4.0 |
| Recharts | 2.9.0 |
| Springboot | 3.1.5 |
| Mybatis | 3.5.11 |
| Axios | 1.x |
| Maven | 3.9 |
| MySQL | 8.0.31 |
| Mosquitto | 5.0.31 |
| Java JDK | 1.8 |
| Paho | 4.3.1 |

## 二、需求分析

### 2.1 功能性需求

本项目主体是一个B/S架构的web应用，主要实现用户对物联网设备的管理与可视化。对于每个用户我们需要实现如下的功能性需求：

* 用户注册，必要信息包括用户名，密码，email等关键信息，需要进行信息验证，用户名和密码必须在六字节以上，email格式验证，用户名和email在系统中唯一
* 用户登录
* 用户信息修改
* 添加/修改设备配置信息，必要信息包括设备ID，设备名称，设备类型等
* 设备上报数据的查询统计，要求提供具体可操作查询窗口，查询进行可视化
* 设备具体信息的可视化查看，要求提供地图界面展示，对设备状态信息进行正常和告警的区分，部分设备的历史数据可以展示成历史轨迹(如设备位置等)
* 设备统计信息查看，包括但不限于设备总量，在线总量，接收的数据量，以及以图标方式可视化

注：该系统所面向的用户是普通用户，其中每个普通用户只能查看自己所持有的一系列设备信息。

### 2.2 非功能性需求

该系统的非功能性需求包括性能需求，输入输出需求，数据管理需求等，具体如下

#### 2.2.1 性能需求

* 系统稳定性，该系统应保证运行稳定，避免出现崩溃，保证在一周内不超过一次维护与重启
* 系统并发性，系统应能保证至少100人的 并发访问
* 系统反馈，当用户进行操作时，系统应该在1s内给出反应，每个页面应在1s内加载完毕，高峰期应在5s内加载完毕
* 系统鲁棒性，当系统检测到非正常情况时，应该能给予用户相应的提示，避免用户长时间等待

#### 2.2.2 输入输出需求

* 用户注册/登录时，应对用户输入的数据进行有效性检查，并确保输入数据安全性和唯一性
* 设备信息修改时，应对输入的设备数据进行格式检查，并确保安全性
* 设备的上报数据需要通过查询的方式，以可视化的图表形式展示，需要保证准确性，美观和清楚
* 设备的统计信息需要以图表的方式可视化展示，并提供地图界面和历史轨迹，需要保证准确性，美观以及清楚

#### 2.2.3 数据管理需求

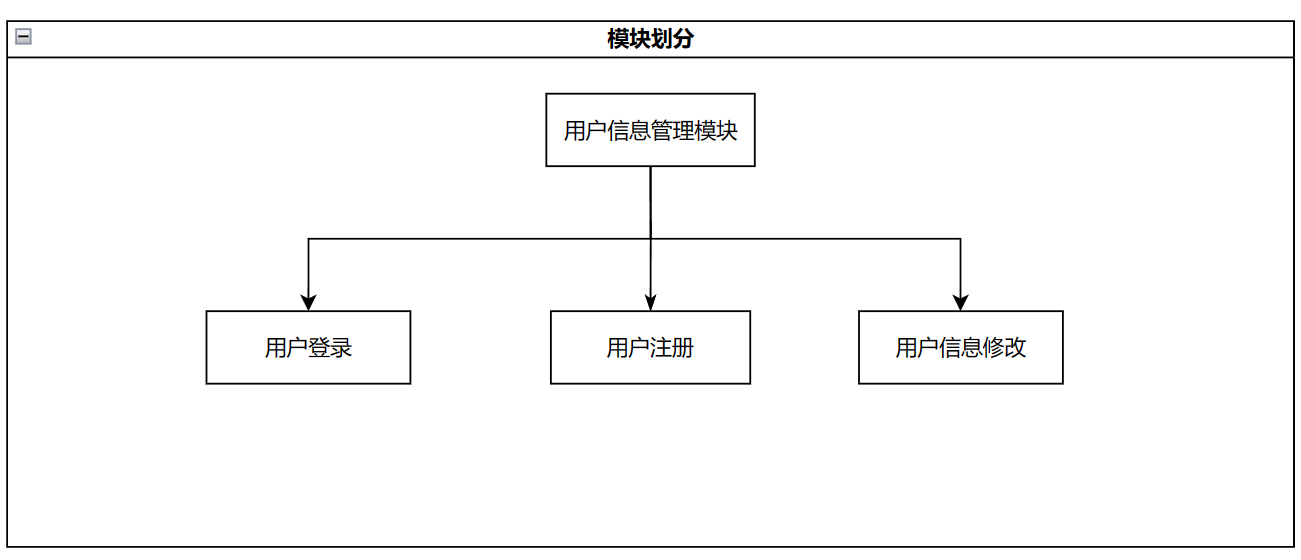
* 系统应维护自身数据的安全性，防止未经授权的用户对本系统进行设置和数据访问，修改
* 系统应提供数据备份和恢复手段，在服务器软件和硬件出现严重故障时，能够根据备份数据恢复正常运行环境
* 系统应在模块与模块之间的交互中，保证数据传输过程中的保密性和安全性

## 三、功能模块与层次结构设计

### 3.1 用户信息管理模块

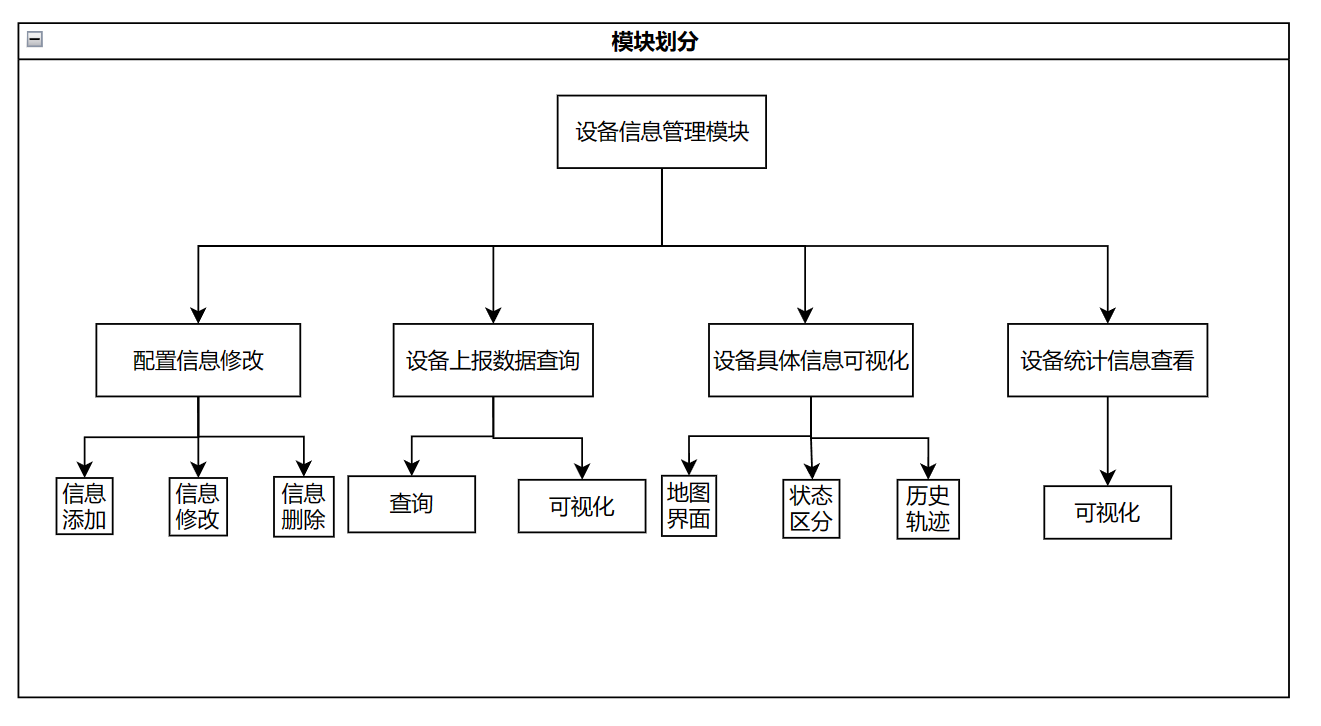
根据用户的功能性需求，我们可以把用户信息管理模块分为如下的三个部分，具体包括用户登录，用户注册和用户信息修改

用户信息管理模块：



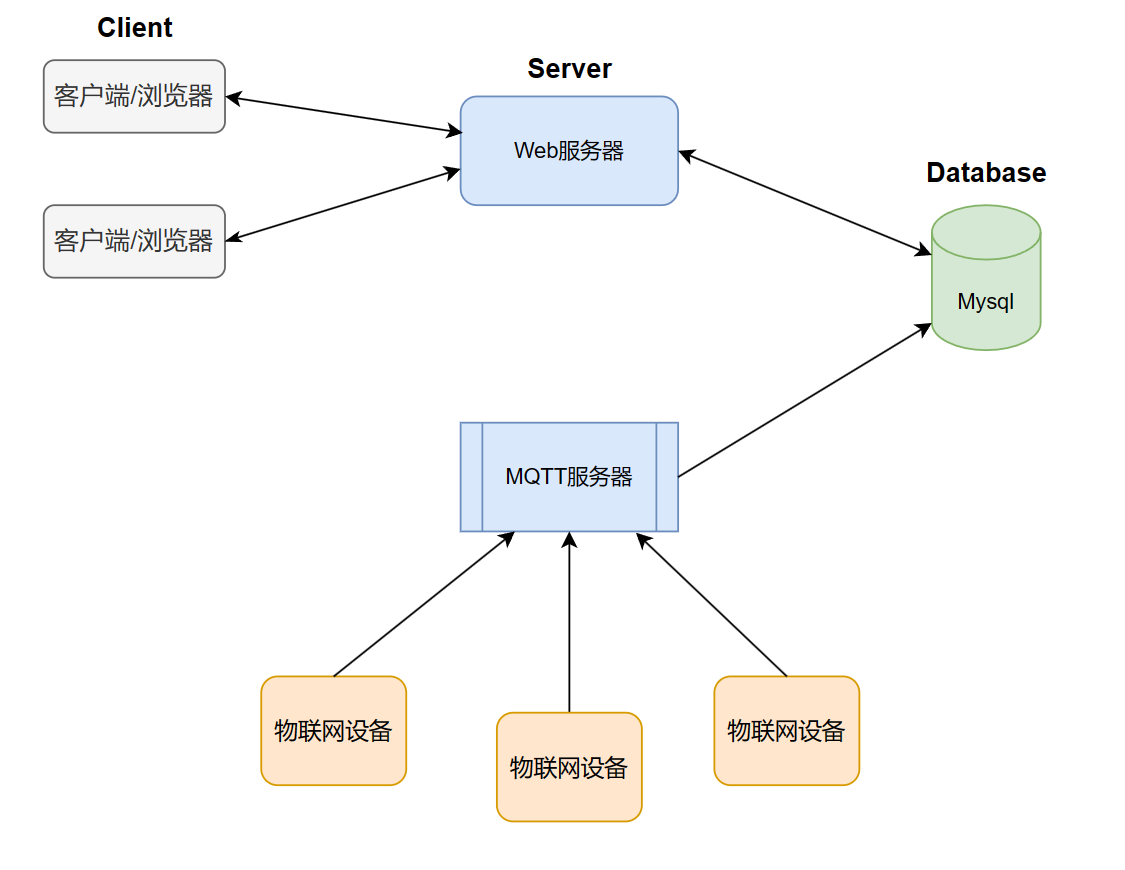
### 3.2 设备信息管理模块

根据功能性需求，我们可以把设备信息管理模块分为如下的四个部分，配置信息修改，设备上报数据查询，设备具体信息可视化以及设备统计信息查看四部分，具体细分如下

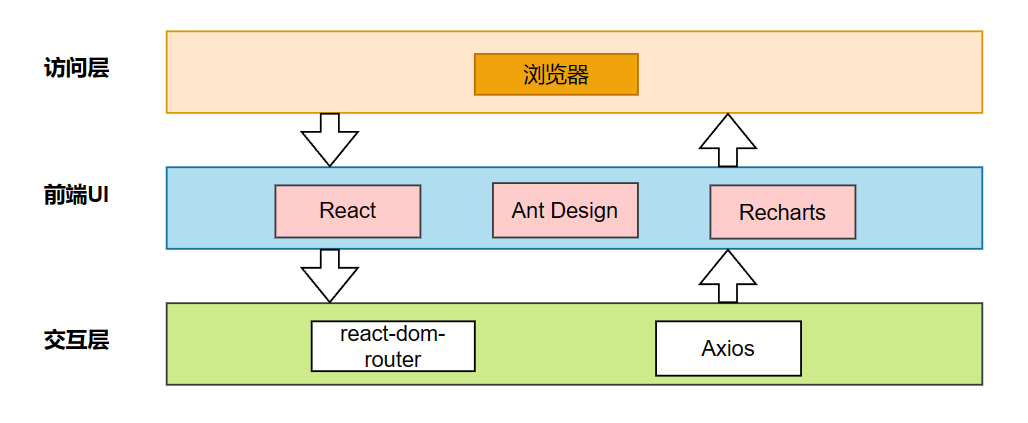


### 3.3 系统整体架构

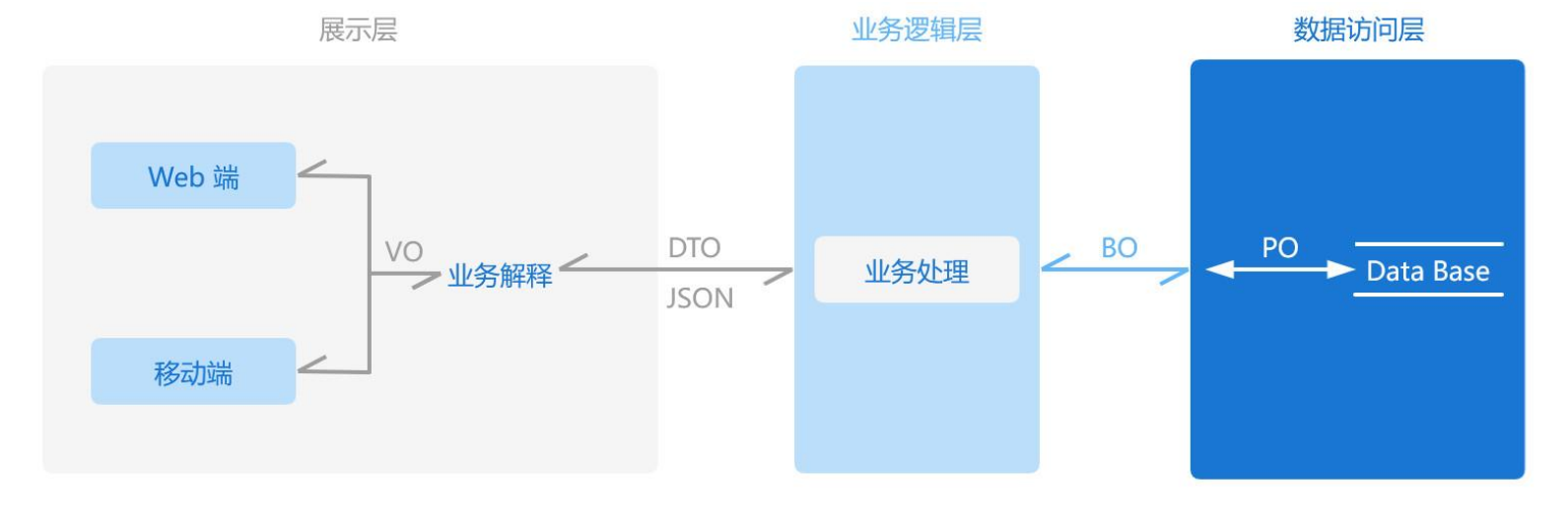
系统的整体架构图如下图所示，其中客户端支持PC和移动端终端进行访问，服务端包括后端服务器和MQTT服务器，其分别和数据库进行交互。其中，后端服务器可以接收前端网页发出的请求并进行处理，与数据库的数据进行相应的交互。MQTT服务器可以接收设备发送的报文并存储到数据库中。



### 3.4 前端架构图

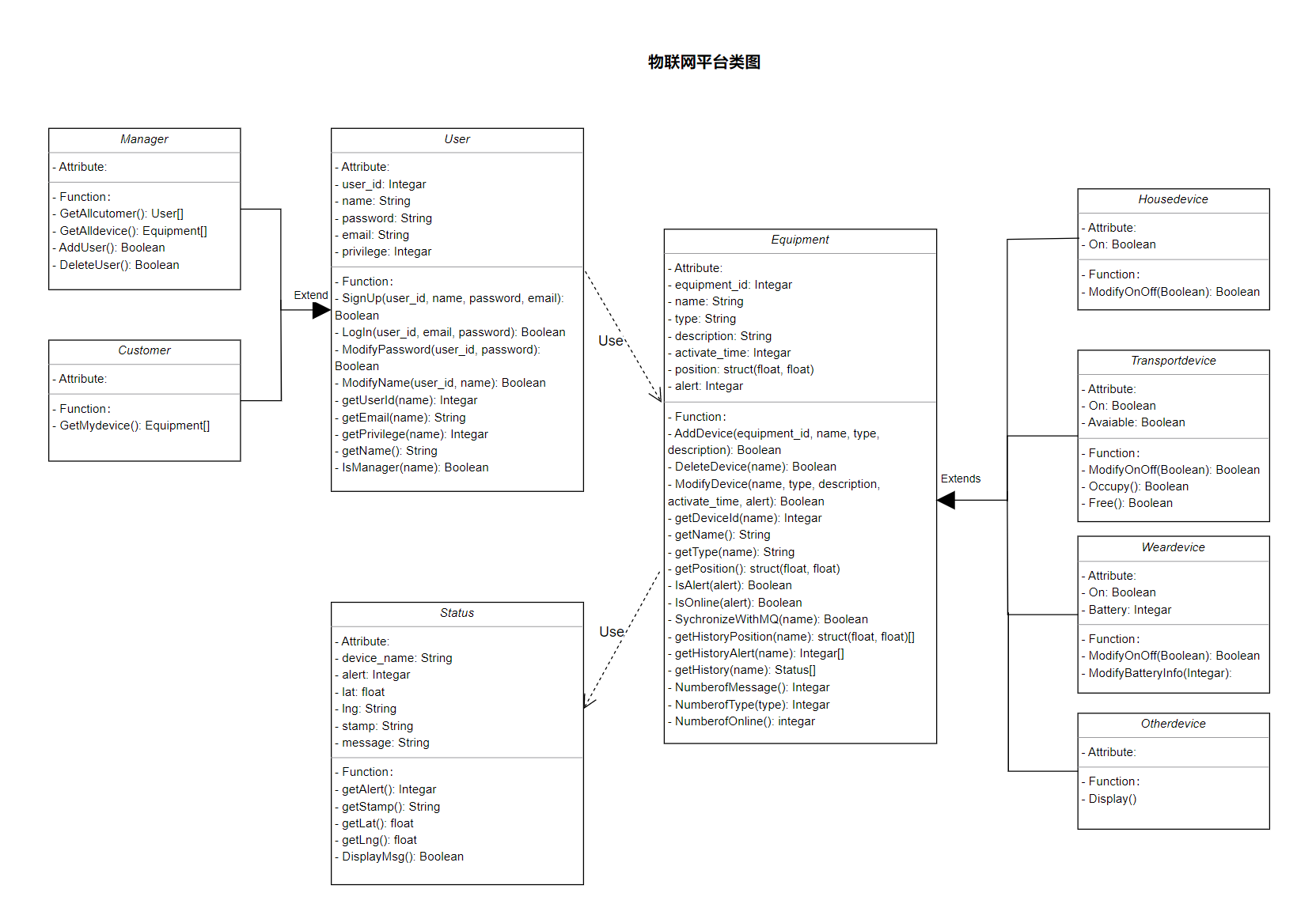


### 3.5 后端架构图

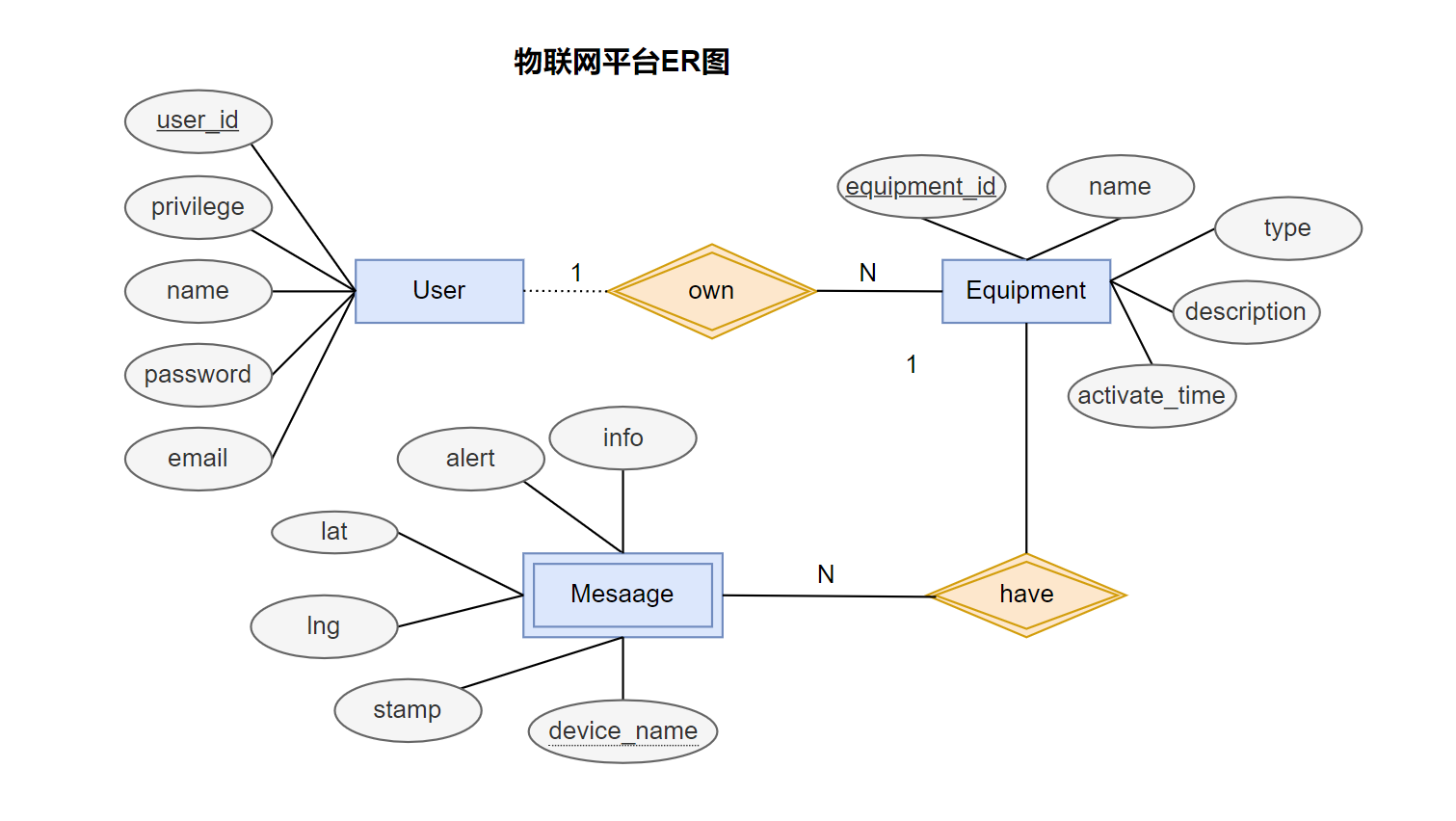


## 四、类图和ER图

### 4.1 类图



### 4.2 ER图



### 4.3 数据表

#### 4.3.1 User表

* 该表用来保存用户的信息

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| user\_id | int | 用户id | PRI, NOT NULL |
| name | varchar(128) | 用户名 | UNIQUE，NOT NULL |
| password | varchar(128) | 用户密码 | NOT NULL |
| email | varchar(128) | 用户邮箱 | NOT NULL, UNIQUE |
| privilege | int | 用户特权级别 | NOT NULL, DEFAULT 1，0代表是管理员，1代表普通用户 |

#### 4.3.2 Equipment表

* 该表用来保存设备的配置信息

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| equipment\_id | int | 设备id | PRI, NOT NULL |
| name | varchar(128) | 设备名 | NOT NULL, UNIQUE |
| type | int | 设备类型 | NOT NULL，DEFAULT 0，目前暂定设备类型包括传感器，智能家居设备，智能穿戴设备，智能交通设备，其他设备 |
| description | varchar(128) | 设备描述信息 | NOT NULL |
| initial\_time | varchar(128) | 设备最初上线时间 | NOT NULL, DEFAULT为设备配置信息的添加时间，当设备第一次向MQTT服务器发送消息时被更改 |
| activate\_time | varchar(128) | 设备最近一次活跃时间 | NOT NULL |
| userid | int | 设备用户id | NOT NULL，FOREIGN KEY |

#### 4.3.3 Message表

* 该表用来保存MQTT服务器读取到的设备的状态信息

| 字段名 | 类型 | 描述 | 备注 |
| --- | --- | --- | --- |
| device\_name | varchar(128) | 设备名称 | NOT NULL，FOREIGN KEY |
| alert | int | 设备在线状态 | NOT NULL，0代表正常online，1代表出现warning，2代表offline |
| info | varchar(128) | 设备上报消息 | NOT NULL，设备发回的消息 |
| lat | numeric | 设备发送消息时的经度 | NOT NULL |
| lng | numeric | 设备发送消息时的纬度 | NOT NULL |
| stamp | varchar(128) | 设备发送消息时的时间戳 | NOT NULL |

## 五、系统接口设计

### 5.1 用户信息相关接口

用户信息相关接口主要利用User表的数据，和用户进行交互，该类接口都以/user进行开头，具体使用的Api如下所示

#### 5.1.1 用户注册

| URL | /user/signup |
| --- | --- |
| 请求方式 | POST |
| 请求格式 | {  "account": 用户名  "password": 密码   "email": 用户邮箱 } |
| 响应格式 | {  "code":响应状态 } |
| 接口备注 | 该接口用于用户进行注册，会对用户输入的用户名，密码以及邮箱进行格式检测。返回的状态码有以下几种可能： 100: 注册成功 200: 注册失败 201:用户名已存在 202：邮箱已被注册 |

#### 5.1.2 用户登录

| URL | /user/login |
| --- | --- |
| 请求方式 | POST |
| 请求格式 | {  "account": 用户名/邮箱   "password": 密码  } |
| 响应格式 | {  "code":响应状态 } |
| 接口备注 | 该接口用于用户进行登录认证，前端会对用户输入的用户名/邮箱，密码进行合法性检测。登陆成功后会生成token返回给用户并保存在浏览器端。返回的状态码有以下几种可能： 100: 操作成功 203: 用户名不存在 204: 密码错误 |

#### 5.1.3 用户退出

| URL | /user/logout |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   "account": 用户名/邮箱  "password": 密码  } |
| 响应格式 | {  "code":响应状态 } |
| 接口备注 | 该接口用于用户进行平台退出，操作前对token进行验证，用户退出后会清除保留在浏览器端的token，返回的状态码有如下几种状态：  100：退出成功  200：退出失败 |

#### 5.1.4 用户修改密码

| URL | /user/change/password |
| --- | --- |
| 请求方式 | POST |
| 请求格式 | {   "user\_id: 用户ID"   old\_password": 旧密码  "new\_password": 新密码  } |
| 响应格式 | {  "code":响应状态 } |
| 接口备注 | 该接口用于用户进行密码修改，操作前对token进行验证，用户在前端输入旧密码以及要修改的新密码，返回的状态码有如下几种状态：  100：退出成功 200：退出失败 |

#### 5.1.5 用户修改用户名

| URL | /user/change/username |
| --- | --- |
| 请求方式 | POST |
| 请求格式 | {   "user\_id: 用户ID"   "new\_username": 新用户名  } |
| 响应格式 | {  "code":响应状态 } |
| 接口备注 | 该接口用于用户进行用户名的修改，操作前对token进行验证，用户在前端输入新用户名后由前端验证合法性，返回的状态码有如下几种状态：  100：退出成功 200：修改失败 201：用户名已存在 |

### 5.2 设备信息相关接口

设备信息相关接口主要利用Equipment表的数据，和用户进行交互，进行设备信息的增改以及静态信息的检索。该类接口都以/equipment进行开头，并都需要对用户token进行验证，具体使用的Api如下所示

#### 5.2.1 设备增加

| URL | /equipment/addconfig |
| --- | --- |
| 请求方式 | POST |
| 请求格式 | {   "equipment\_name": 设备名  "type":设备类型   } |
| 响应格式 | {  "code":响应状态 } |
| 接口备注 | 该接口用于用户进行设备信息的增加，用户在前端输入设备相关信息进行设备信息增加，其中设备ID由数据表自动生成。返回的状态码有如下几种状态：  100：添加成功 200：添加失败 |

#### 5.2.2 设备信息修改

| URL | /equipment/modifyconfig |
| --- | --- |
| 请求方式 | POST |
| 请求格式 | {   "equipment\_id: 设备ID"  "equipment\_name": 设备名  "type": 设备类型   } |
| 响应格式 | {  "code":响应状态 } |
| 接口备注 | 该接口用于用户进行设备信息的修改，用户在前端输入相关信息进行设备信息的修改。由前端验证合法性，返回的状态码有如下几种状态：  100：修改成功 200：修改失败 205：设备不存在 |

#### 5.2.3 设备删除

| URL | /equipment/delete |
| --- | --- |
| 请求方式 | POST |
| 请求格式 | {   "equipment\_id: 设备ID"  } |
| 响应格式 | {  "code":响应状态 } |
| 接口备注 | 该接口用于用户进行设备信息的删除，用户在前端输入设备id并进行设备信息的删除。返回的状态码有如下几种状态：  100：删除成功 200：删除失败 205：设备不存在 |

#### 5.2.4 获取单个设备信息

| URL | /equipment/query/{#id} |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   "equipment\_id": 设备ID  } |
| 响应格式 | 单个Equipment对象+状态码，均为json格式 |
| 接口备注 | 该接口用于用户进行单个设备信息的获取，用户在前端输入设备ID后进行查询，返回的状态码有如下几种状态：  100：查询成功 200：查询失败 205：设备不存在 |

#### 5.2.5 获取用户设备总数

| URL | /equipment/query/sum/all |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   "username": 用户名  "token": 用户token } |
| 响应格式 | {  "code":响应状态  "number": 用户设备总数 } |
| 接口备注 | 该接口用于用户进行设备总数的查询，用户在前端切换到相应界面后自动进行设备总数的查询并用于前端可视化，需要对用户token进行验证。返回的状态码有如下几种状态：  100：查询成功 200：查询失败 |

#### 5.2.5 获取用户设备列表

| URL | /equipment/query/list/all |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   "username": 用户名  "token": 用户token } |
| 响应格式 | Equipment对象数组+状态码，均为json格式 |
| 接口备注 | 该接口用于用户进行设备列表的获取，用户在前端切换到相应界面后自动进行设备列表的获取并用于前端可视化，需要对用户token进行验证。返回的状态码有如下几种状态：  100：查询成功 200：查询失败 |

#### 5.2.6 获取用户特定种类设备数量

| URL | /equipment/query/sum/type |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   "username": 用户名  "token": 用户token  "type": 设备类型 } |
| 响应格式 | 设备数量+状态码，均为json格式 |
| 接口备注 | 该接口用于用户进行特定种类设备数量的获取，需要对用户token进行验证。返回的状态码有如下几种状态：  100：查询成功 200：查询失败 |

#### 5.2.7 获取用户特定种类设备列表

| URL | /equipment/query/list/type |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   "username": 用户名  "token": 用户token  "type": 设备类型 } |
| 响应格式 | Equipment对象数组+状态码，均为json格式 |
| 接口备注 | 该接口用于用户进行特定设备列表的获取，用户在前端切换到相应界面后自动进行设备列表的获取并用于前端可视化，需要对用户token进行验证。返回的状态码有如下几种状态：  100：查询成功 200：查询失败 |

#### 5.2.8 用户在线设备数量获取

| URL | /equipment/query/sum/online |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   "username": 用户名  "token": 用户token  "type": online } |
| 响应格式 | 在线设备数量+状态码，均为json格式 |
| 接口备注 | 该接口用于用户进行特定设备列表的获取，用户在前端切换到相应界面后自动进行设备列表的获取并用于前端可视化，需要对用户token进行验证。返回的状态码有如下几种状态：  100：查询成功 200：查询失败 |

### 5.3 设备消息相关接口

用户信息相关接口主要利用Equipment表和Message的数据，和用户进行交互，进行设备发送消息的检索。该类接口都以/equipment/message进行开头，并都需要对用户token进行验证。请求格式中都会包括对应的用户名和token，在下表中进行省略，具体使用的Api如下所示:

#### 5.3.1 查询所有设备特定时间段接收消息量

| URL | /equipment/message/num/all |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   "start\_time": 开始时间  "end\_time": 结束时间  } |
| 响应格式 | 特定时间段内数量+状态码，均为json格式 |
| 接口备注 | 该接口用于用户进行特定时间段内设备接收消息数量的获取，需要对用户token进行验证。返回的状态码有如下几种状态：  100：查询成功 200：查询失败 |

#### 5.3.2 查询设备历史轨迹

| URL | /equipment/message/path/{#id} |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   "equipment\_id": 设备ID } |
| 响应格式 | 在线设备数量+状态码，均为json格式 |
| 接口备注 | 该接口用于用户进行特定设备历史轨迹的获取，需要对用户token进行验证。返回的状态码有如下几种状态：  100：查询成功 200：查询失败 |

#### 5.3.3 查询单个设备历史消息

| URL | /equipment/message/info/{#id} |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   equipment\_id": 设备ID } |
| 响应格式 | 存有Info信息的String数组+状态码，均为json格式 |
| 接口备注 | 该接口用于用户进行特定设备历史消息的获取，需要对用户token进行验证。返回的状态码有如下几种状态：  100：查询成功 200：查询失败 |

#### 5.3.4 查询单个设备历史消息数量

| URL | /equipment/message/num/{#id} |
| --- | --- |
| 请求方式 | GET |
| 请求格式 | {   equipment\_id": 设备ID } |
| 响应格式 | 单个设备历史消息数量(Integar)+状态码，均为json格式 |
| 接口备注 | 该接口用于用户进行特定设备历史消息数量的获取，需要对用户token进行验证。返回的状态码有如下几种状态：  100：查询成功 200：查询失败 |

## 六、系统界面原型

系统的原型界面如下图所示，**最终成品和界面原型可能略有差异**。由于设计报告完成于最终网站成形之前，因此最终界面细节可能与原型界面具有一定差异。以下原型界面仅供参考。

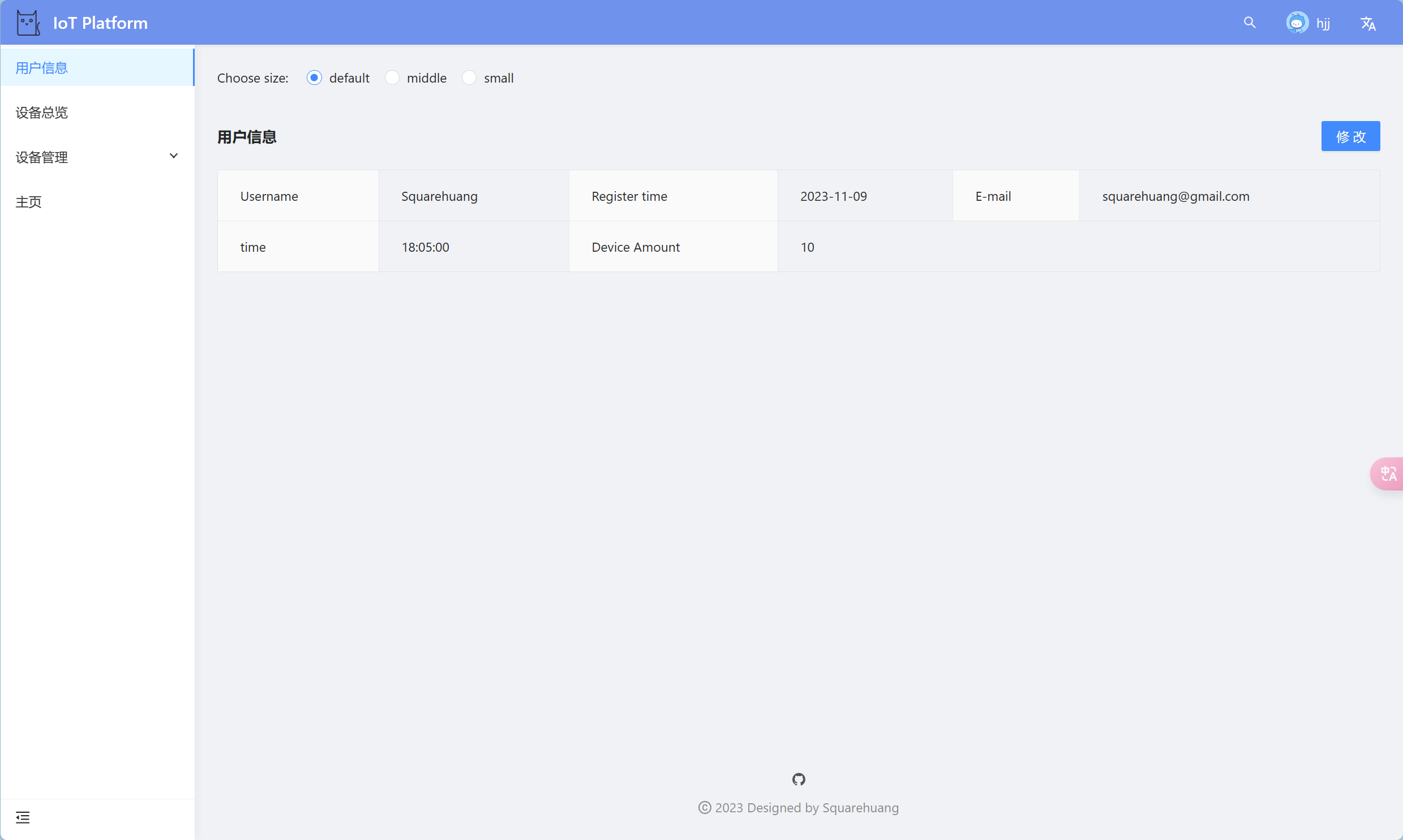
### 6.1 用户登陆页面



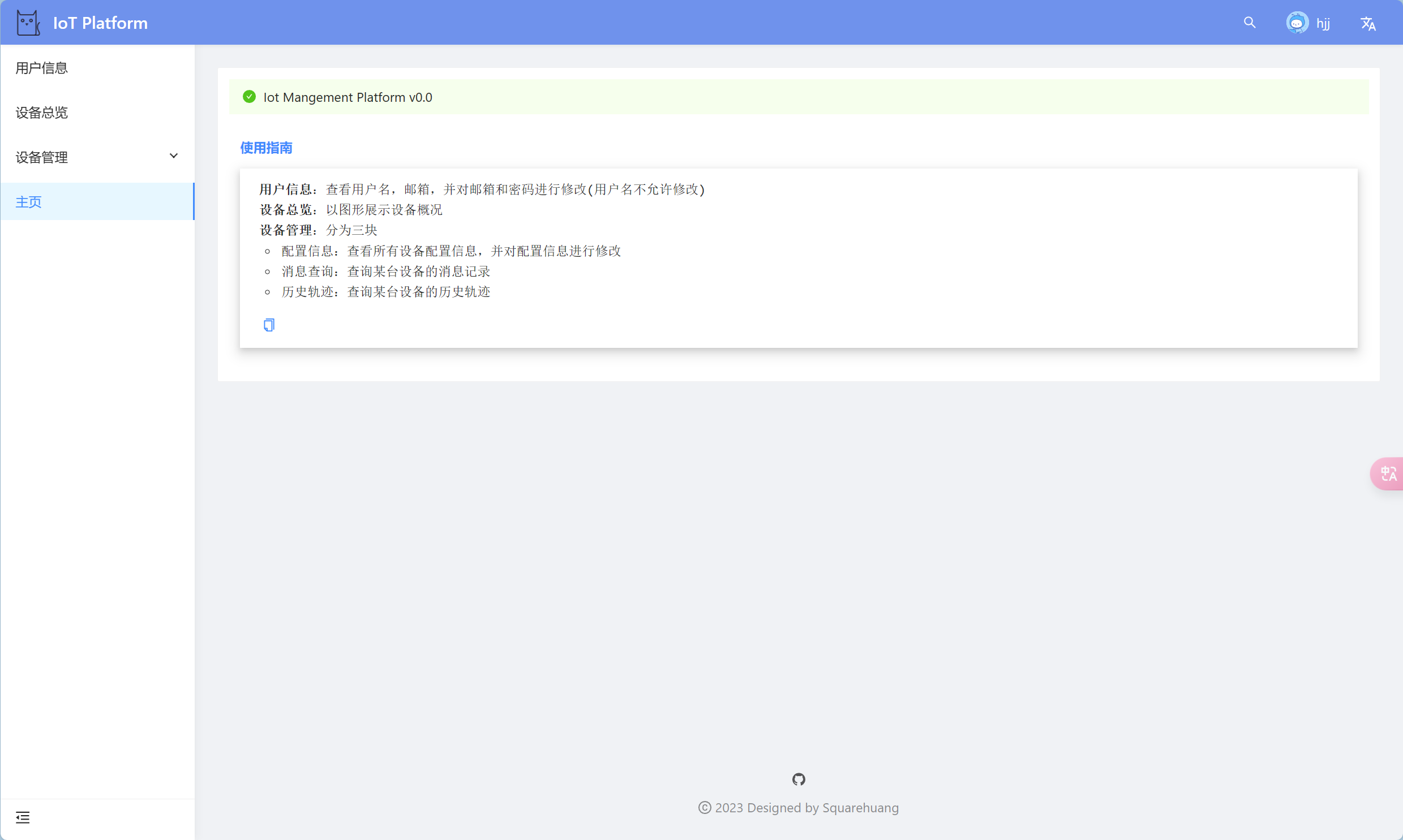
### 6.2 用户注册页面



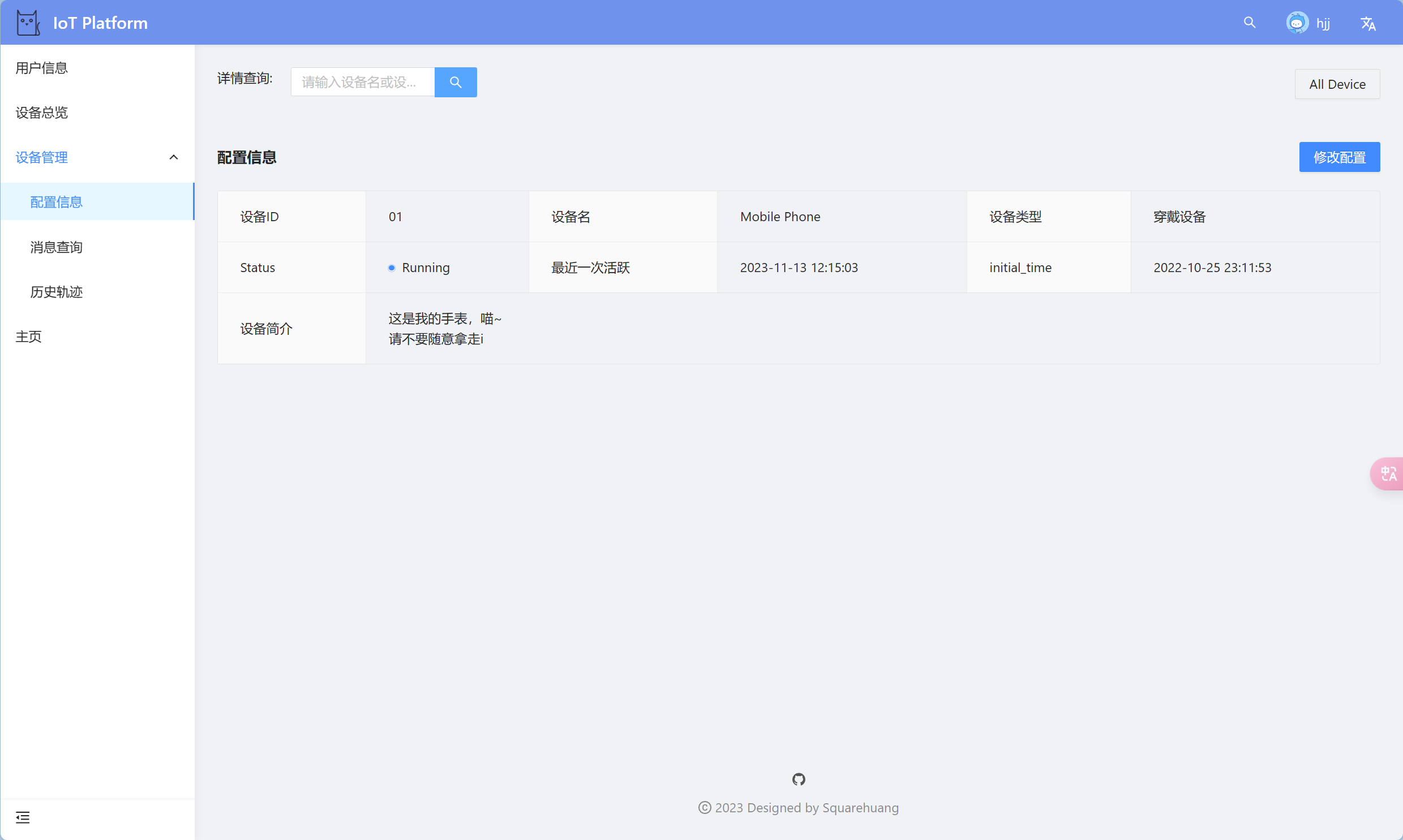
### 6.3 个人信息页面



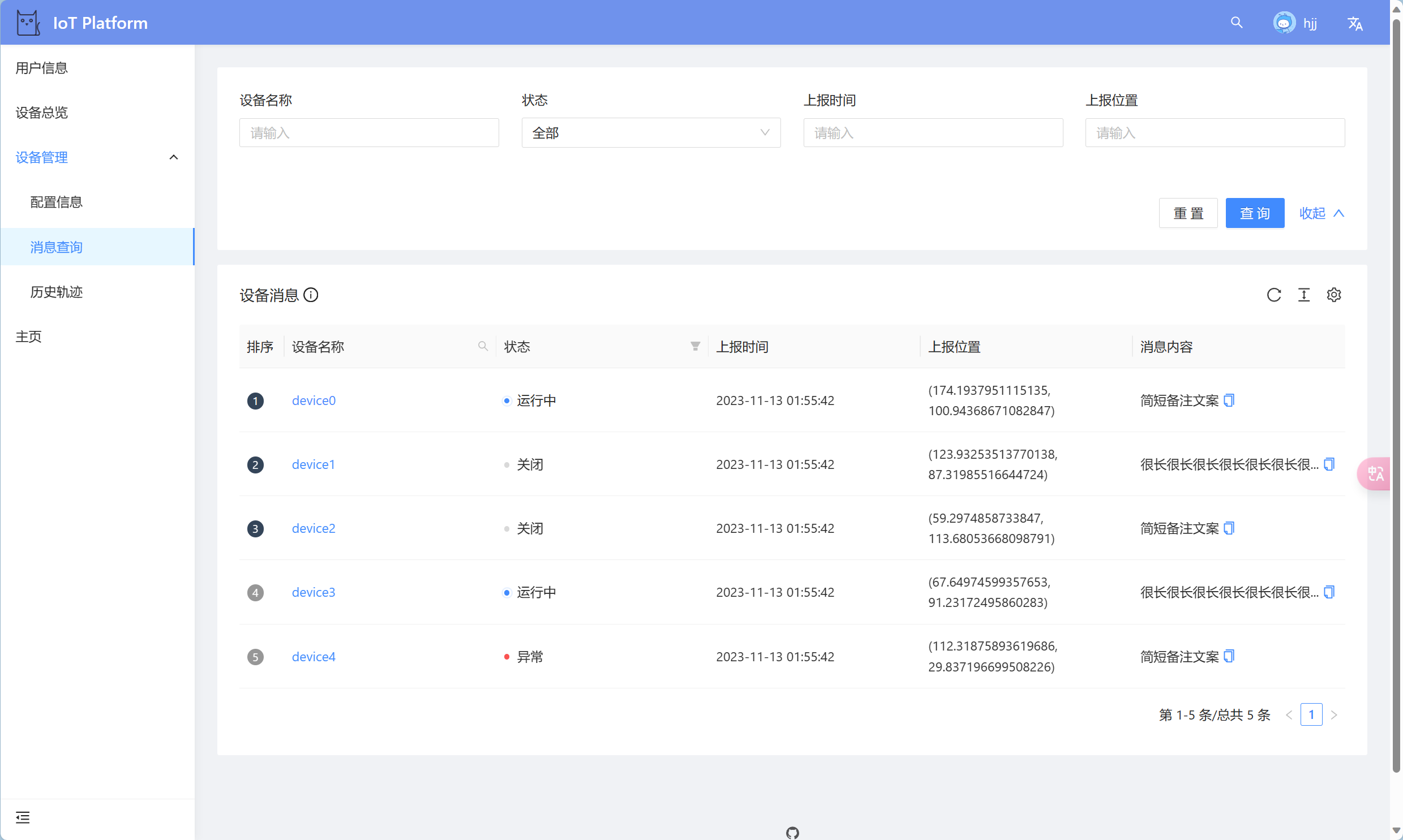
### 6.4 平台首页



### 6.5 设备信息配置页面



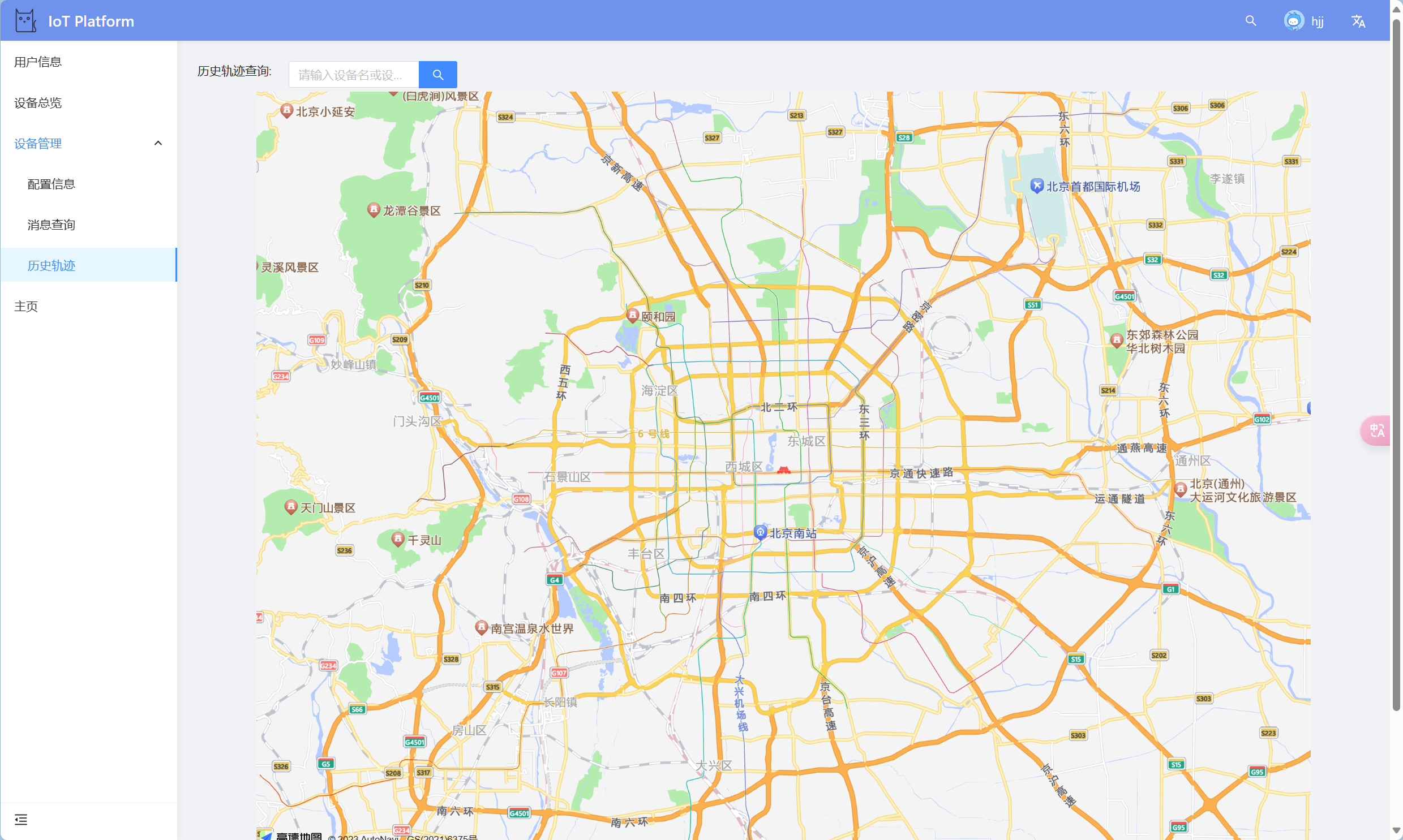
### 6.6 设备上报数据查询页面



### 6.7 设备统计信息总览页面



### 6.8 设备轨迹查看页面



## 七、预计开发时间轴

| 时间段 | 计划进度 |
| --- | --- |
| 2023.10.10—2023.10.17 | 项目需求分析 |
| 2023.10.18—2023.10.25 | 开发技术选型及学习 |
| 2023.10.26—2023.11.03 | 界面原型与接口设计 |
| 2023.11.04—2023.11.11 | 设计文档撰写、数据库建表与实现、mqtt服务器搭建 |
| 2023.11.12—2023.11.25 | 接口文档撰写，服务端代码实现 |
| 2023.11.26—2023.12.10 | 前端代码实现 |
| 2023.12.11—2023.12.25 | 测试文档撰写以及测试 |
| 2023.12.25—2023.01.01 | 完成用户手册并最终上线 |

## 八、附录

### 8.1 状态码设计

| 状态 | 具体含义 |
| --- | --- |
| 100 | 操作成功 |
| 200 | 操作失败 |
| 201 | 用户名已存在 |
| 202 | 邮箱已被注册 |
| 203 | 用户名不存在 |
| 204 | 密码错误 |
| 205 | 设备不存在 |
| 206 | 获取设备最新信息失败 |