

**本科实验报告**

|  |  |
| --- | --- |
| 课程名称： | B/S体系软件设计 |
| 姓 名： | 陈华杰 |
| 学 院： | 计算机学院 |
| 系： | 软件工程 |
| 专 业： | 软件工程 |
| 学 号： | 3210300237 |
| 指导教师： | 胡晓军 |

2023年09月28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： 大作业

实验项目名称： 设计报告《物联网设备管理平台》

学生姓名： 陈华杰 专业： 软件工程 学号： 3210300237

同组学生姓名： 无 指导老师： 胡晓军

实验地点： 无 实验日期：2023年 09月28日

目录

[1 设计概述 4](#_Toc150713062)

[1.1 任务与目标 4](#_Toc150713063)

[1.2 运行环境 4](#_Toc150713064)

[1.2.1 授权服务器 4](#_Toc150713065)

[1.2.2 资源服务器 4](#_Toc150713066)

[1.2.3 客户端服务器（Client Server） 4](#_Toc150713067)

[2 功能模块与层次结构设计 5](#_Toc150713068)

[2.1 模块一 用户信息 5](#_Toc150713069)

[2.2 模块二 资源管理 6](#_Toc150713070)

[2.3 客户端项目层次结构 6](#_Toc150713071)

[2.4 服务端项目层次结构 7](#_Toc150713072)

[2.5 各个服务端负责的功能 7](#_Toc150713073)

[2.5.1 客户端服务端 7](#_Toc150713074)

[2.5.2 授权服务端 7](#_Toc150713075)

[2.5.3 资源服务端 7](#_Toc150713076)

[3 类图 9](#_Toc150713077)

[4 数据字典 10](#_Toc150713078)

[4.1 User 10](#_Toc150713079)

[4.2 Role 10](#_Toc150713080)

[4.3 UserRole 10](#_Toc150713081)

[4.4 LoginHistory 10](#_Toc150713082)

[4.5 Message 11](#_Toc150713083)

[4.6 Device 11](#_Toc150713084)

[4.7 Subscription 11](#_Toc150713085)

[4.8 Topic 11](#_Toc150713086)

[4.9 DeviceCategory 12](#_Toc150713087)

[5 ER图 13](#_Toc150713088)

[6 接口设计 14](#_Toc150713089)

[6.1 前端 14](#_Toc150713090)

[6.2 后端 14](#_Toc150713091)

[7 界面原型 15](#_Toc150713092)

[7.1 登录界面 15](#_Toc150713093)

[7.2 注册界面 15](#_Toc150713094)

[7.3 主页 16](#_Toc150713095)

[7.4 配置界面 17](#_Toc150713096)

[7.4.1 订阅标题管理 17](#_Toc150713097)

[7.4.2 创建设备 18](#_Toc150713098)

[7.4.3 设备类型管理 18](#_Toc150713099)

[7.5 设备上报数据的查询统计界面 19](#_Toc150713100)

[8 附录 Error! Bookmark not defined.](#_Toc150713101)

# 设计概述

## 任务与目标

本项目的任务是实现一个物联网设备管理的网站，其中要实现的功能包括用户注册、登录功能、提供设备配置界面（可以创建或修改设备信息）、提供设备上报数据的查询统计界面、地图界面等。同时前端尽可能地适配多种客户端类型如手机端、微信等应用内置的浏览器。

## 运行环境

本项目采用BS软件设计体系，也就是用户透过浏览器即可以访问前端和后端提供的服务。其中前端主要框架/平台使用为Angular，后端则是Springboot。详细运行环境以下详细列明。

本项目会使用到两个服务器，其中为授权服务器（Authorization Server）+资源服务器（Resource Server）和客户端服务器（Client Server）。三个服务器都会运行在一个主机上，并且是运行在一个Docker Container。

### 授权服务器

* Keycloak

### 资源服务器

* Springboot
* RestAPI
* GraphQL
* MySQL

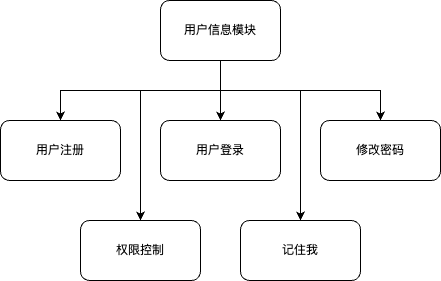
### 客户端服务器（Client Server）

* Angular
* HttpClient
* Angular Material
* Bootstrap
* Nginx

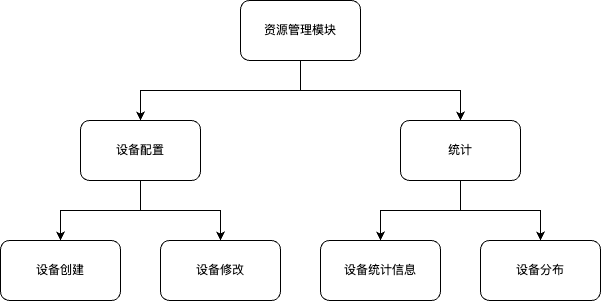
# 功能模块与层次结构设计

此部分首先会先以总体来说明项目的功能模块和结构设计，之后再分为三个部分详细说明，其中这三个部分都代表着整个系统的子系统，也就是授权服务器、资源服务器和客户端服务器。而这三个服务器其实都会运行在同一台主机上。接下来将对三个服务器分别详细说明。

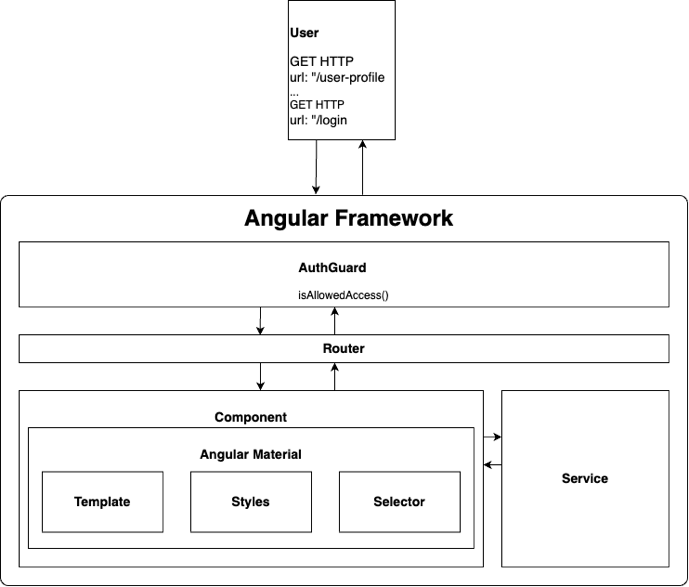
## 模块一 用户信息



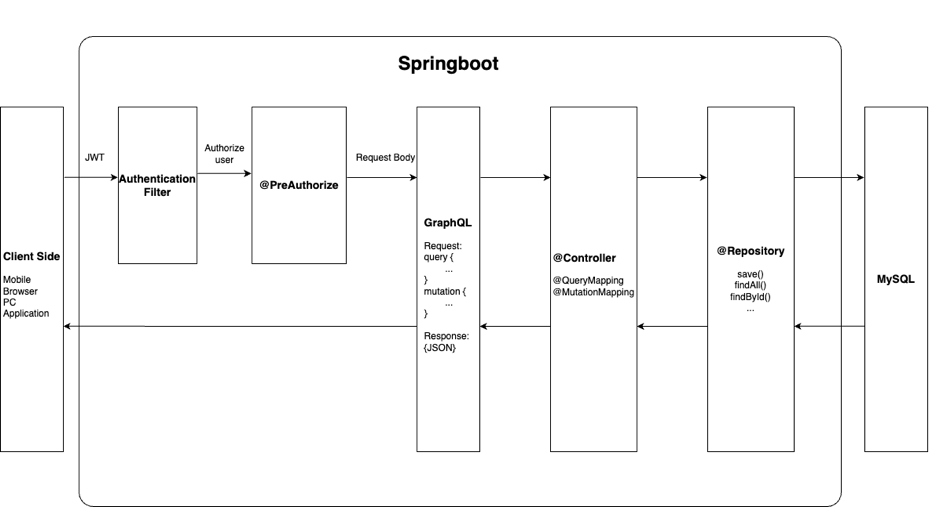
## 模块二 资源管理



## 客户端项目层次结构



## 服务端项目层次结构



## 各个服务端负责的功能

### 客户端服务端

客户端服务端通常是指应用程序的客户端部分，负责与用户互动并发送请求到服务器端。它的功能包括：

* **用户界面**：客户端服务端通常包括用户界面，使用户能够与应用程序进行交互。
* **发送请求**：客户端服务端负责构建和发送请求到授权服务端以获取授权令牌或到资源服务端以访问受保护的资源。
* **接收响应**：客户端服务端接收来自服务器端的响应，然后根据响应的数据来更新用户界面或采取其他必要的操作。

### 授权服务端

授权服务端负责处理用户身份验证和授权请求，以确保只有合法的客户端能够访问资源。其功能包括：

* **用户身份验证**：验证用户的身份，通常使用用户名和密码、令牌或其他身份验证凭证。
* **发放授权令牌**：一旦用户被验证，授权服务端生成并分发访问令牌或授权码给客户端，以允许客户端访问资源。
* **授权决策**：授权服务端根据用户的权限和请求的范围来决定是否授予客户端访问资源的权限。

### 资源服务端

资源服务端负责存储和管理受保护的资源，并根据令牌的有效性来控制对这些资源的访问。其功能包括：

* **资源管理**：资源服务端维护受保护资源的存储，这些资源可能包括用户数据、文件、API端点等。
* **访问控制**：资源服务端根据收到的令牌来验证客户端的访问权限，只允许合法令牌的请求访问受保护资源。
* **响应请求**：资源服务端处理客户端请求，提供请求的资源或执行相应的操作。

# 类图



# 数据字典

## User

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 描述 | 备注 |
| id | int | 用户id | - |
| username | String | 用户名 | - |
| email | String | 邮箱地址 | - |
| password | String | 哈希密码 | - |
| firstName | String | 姓氏 | - |
| lastName | String | 名 | - |
| phoneNumber | String | 手机号 | - |

## Role

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 描述 | 备注 |
| id | int | 身份id | - |
| name | String | 身份标识 | - |

## UserRole

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 描述 | 备注 |
| id | int | 用户身份id |  |
| userId | int | 用户id | - |
| roleId | int | 身份id | - |

## LoginHistory

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 描述 | 备注 |
| id | int | 登录id |  |
| time | Timestamp | 时间戳 |  |
| ipSrc | String | ip来源 |  |
| type | Enum | 登录/登出 |  |
| userId | int | 用户id |  |

## Message

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 描述 | 备注 |
| id | int | 消息id |  |
| time | Timestamp | 时间戳 |  |
| src | String | message来源ip |  |
| dest | String | message发送ip |  |
| srcDeviceId | int | 来源设备id |  |
| destDeviceId | int | 目标设备id |  |
| type | Enum | 消息类型 |  |
| topicId | int | 标题id |  |
| data | String | 数据 |  |
| coordinate | String | 设备坐标 |  |
| state | Enum | 设备状态 |  |

## Device

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 描述 | 备注 |
| id | int | 设备id |  |
| name | String | 设备名字 |  |
| model | String | 设备款式 |  |
| brand | String | 设备品牌 |  |
| serialNumber | String | 设备序列号 |  |
| deviceCategoryId | int | 设备类别id |  |
| online | Enum | 设备状态 |  |

## Subscription

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 描述 | 备注 |
| id | int | 订阅id |  |
| deviceId | int | 设备id |  |
| categoryId | int | 类别id |  |

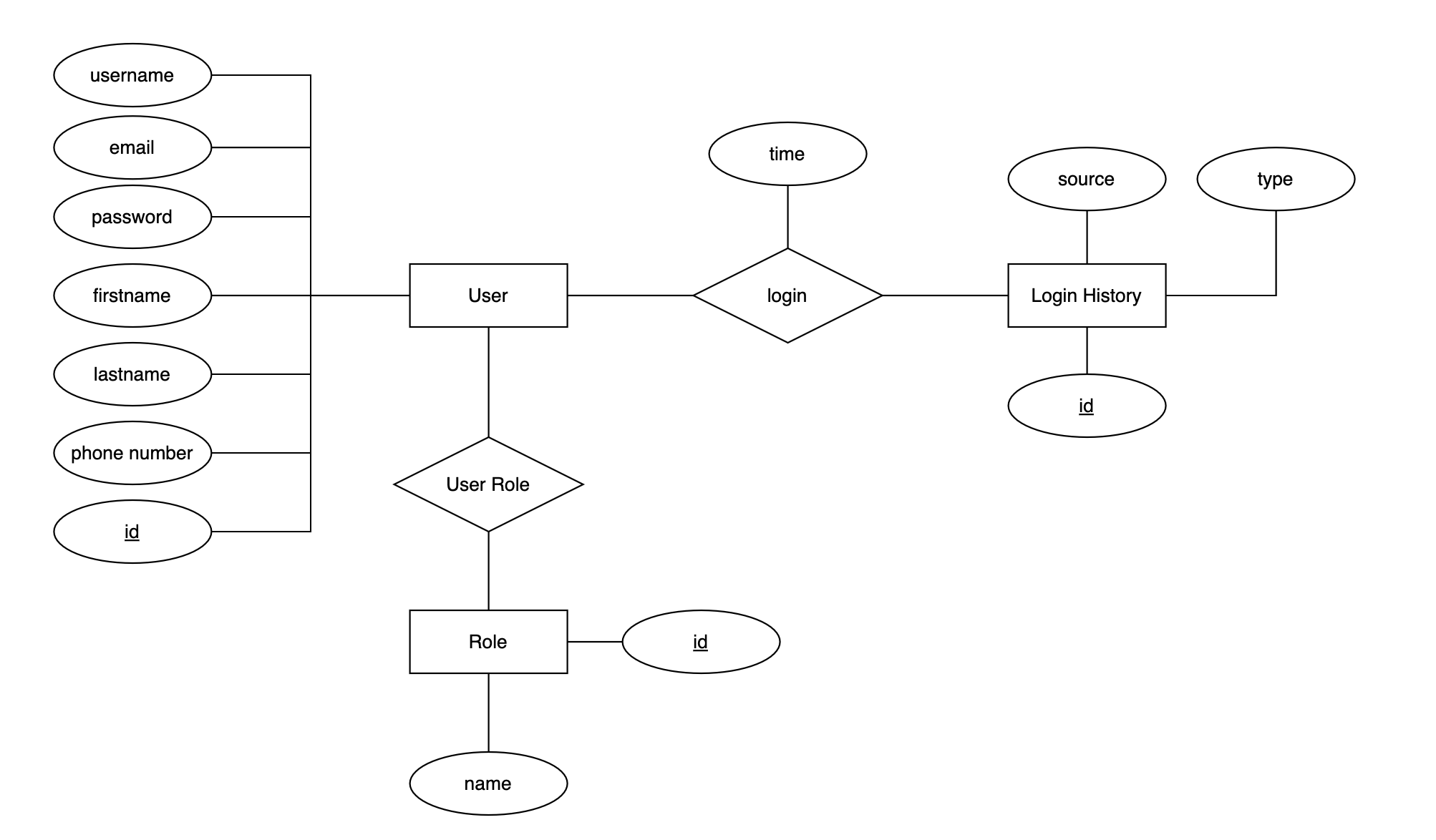
## Topic

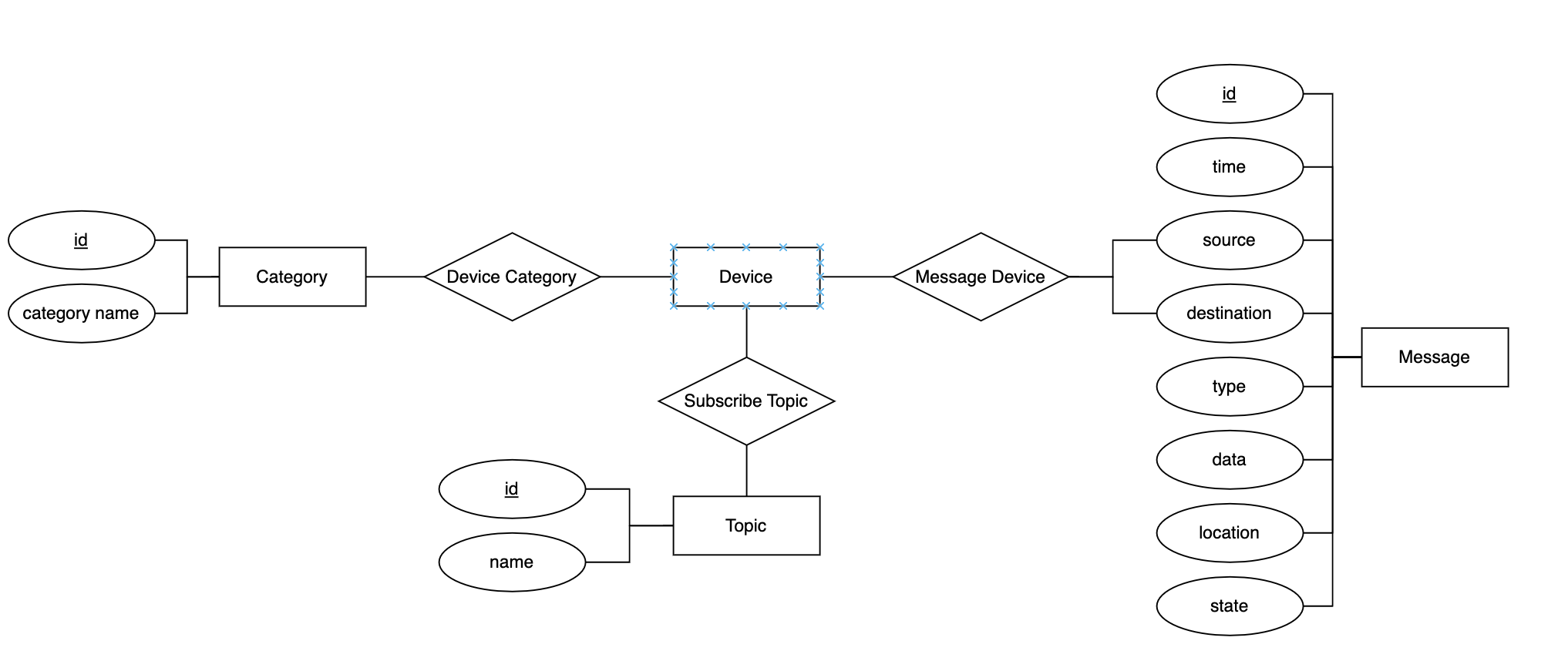
|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 描述 | 备注 |
| id | int | 主题id |  |
| topic | String | 主题 |  |

## DeviceCategory

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 描述 | 备注 |
| id | int | 设备类别id |  |
| categoryName | String | 类别名 |  |

# ER图





# 接口设计

## 前端

前端的接口设计主要采用HttpClient工具请求后端API。基于验证与授权机制的考量，因此前端每次在请求后端接口的时候，都会在请求头Header中加一条Authorization，它的value为用户在登录时候获取的JWT （JSON Web Token，Access Token），这样后端在处理业务逻辑之前会先对用户的请求进行验证与授权。

## 后端

此项目采用GraphQL作为后端接口设计的主要框架。GraphQL 是一个用于 API 的查询语言，是一个使用基于类型系统来执行查询的服务端运行时（类型系统由你的数据定义）。GraphQL 并没有和任何特定数据库或者存储引擎绑定，而是依靠你现有的代码和数据支撑。

一个 GraphQL 服务是通过定义类型和类型上的字段来创建的，然后给每个类型上的每个字段提供解析函数。例如，一个 GraphQL 服务告诉我们当前登录用户是 me，这个用户的名称可能像这样：

type Query {

me: User

}

type User {

id: ID

name: String

}

因此，只要利用查询语句如：

{

me {

name

}

}

会产生这样的JSON结果：

{

"me": {

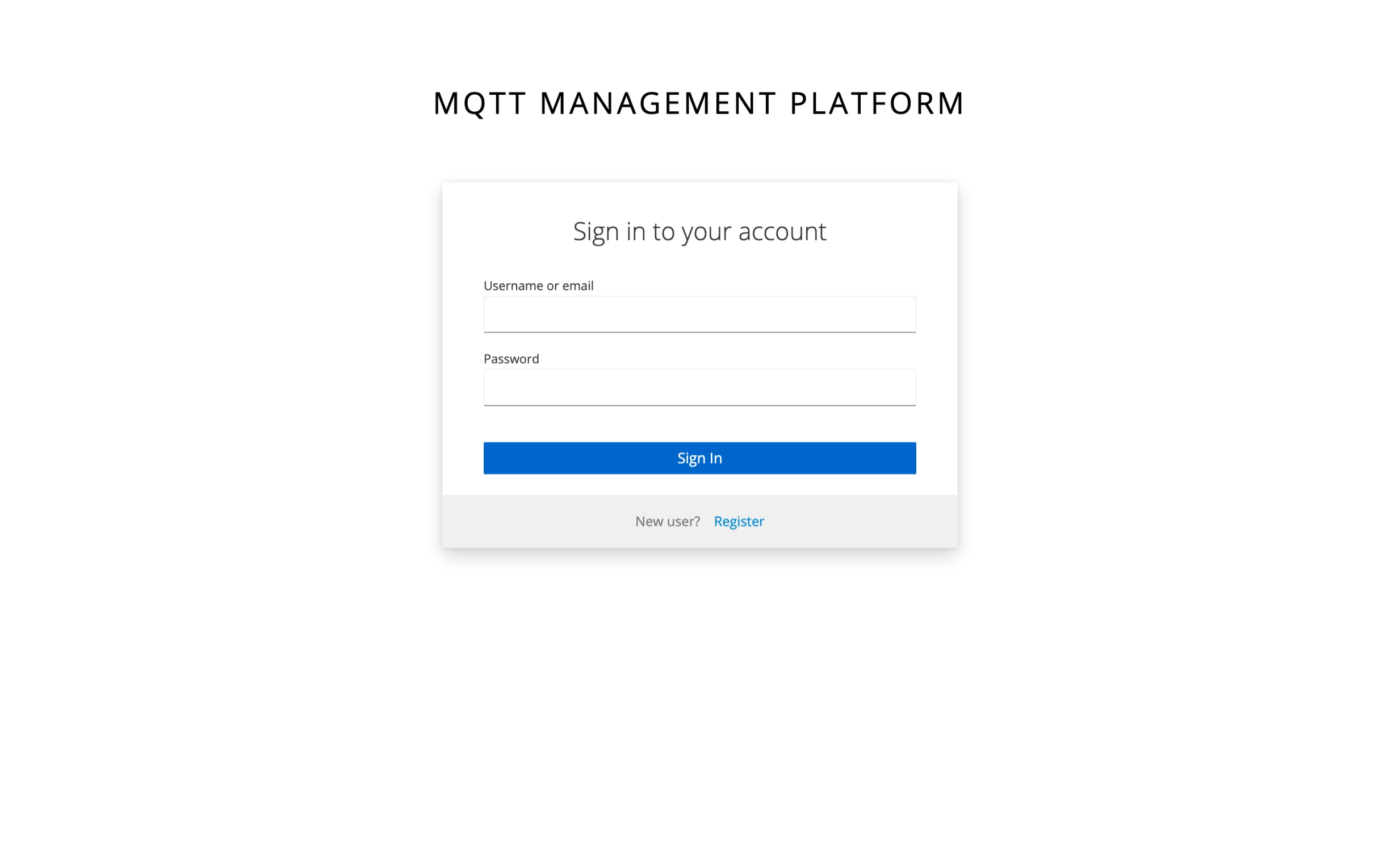
"name": "Luke Skywalker"

}

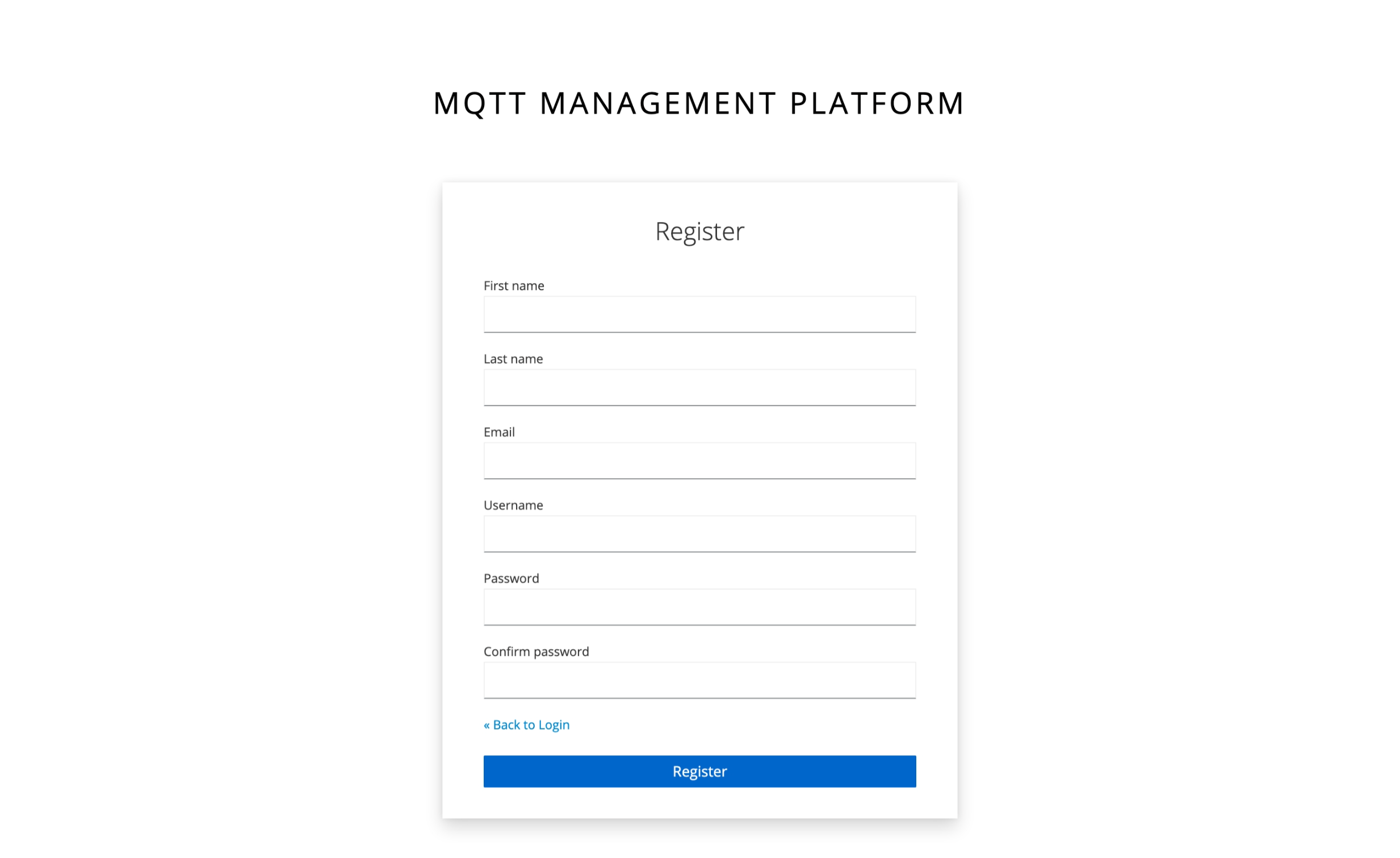
}

# 界面原型

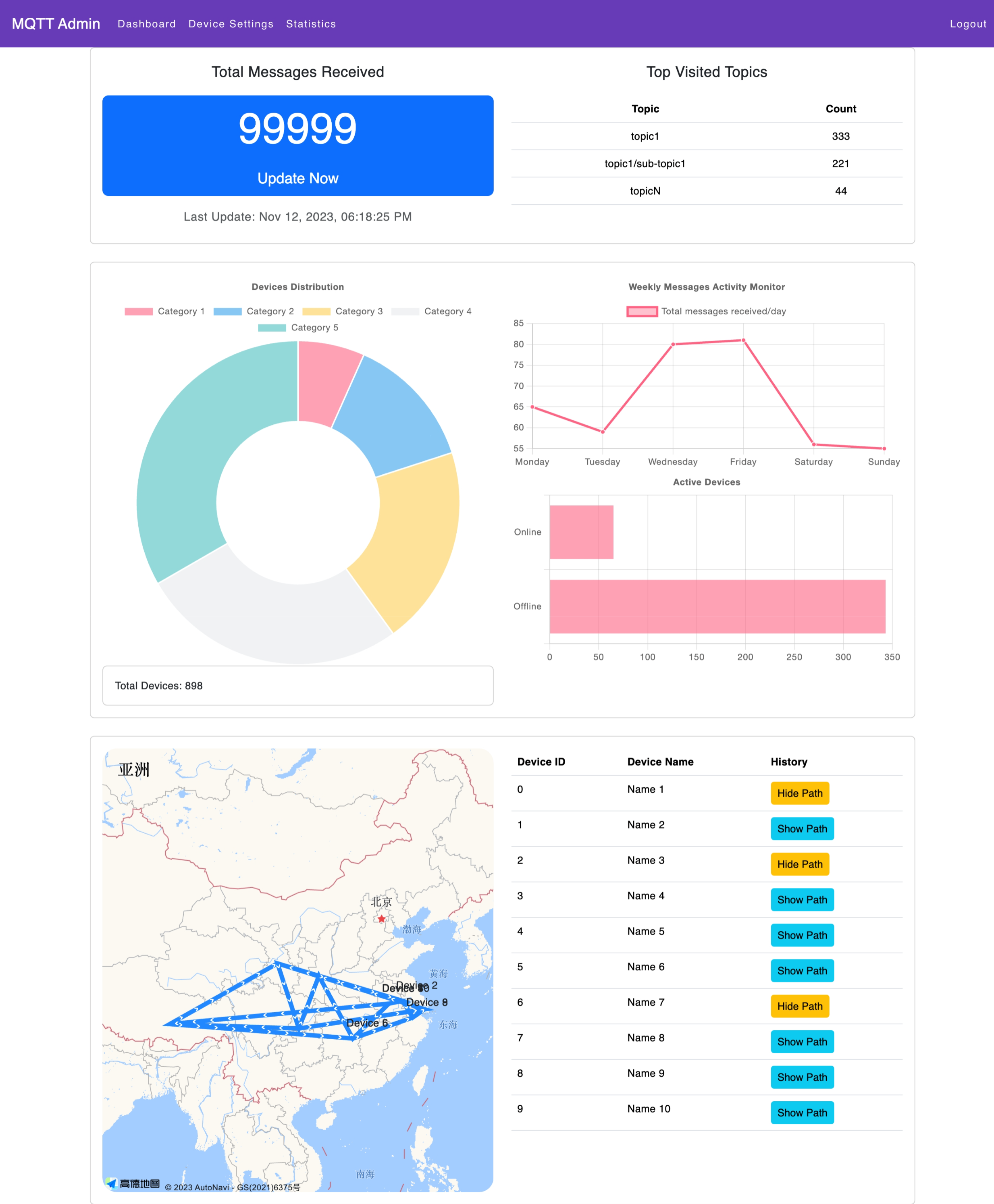
## 登录界面



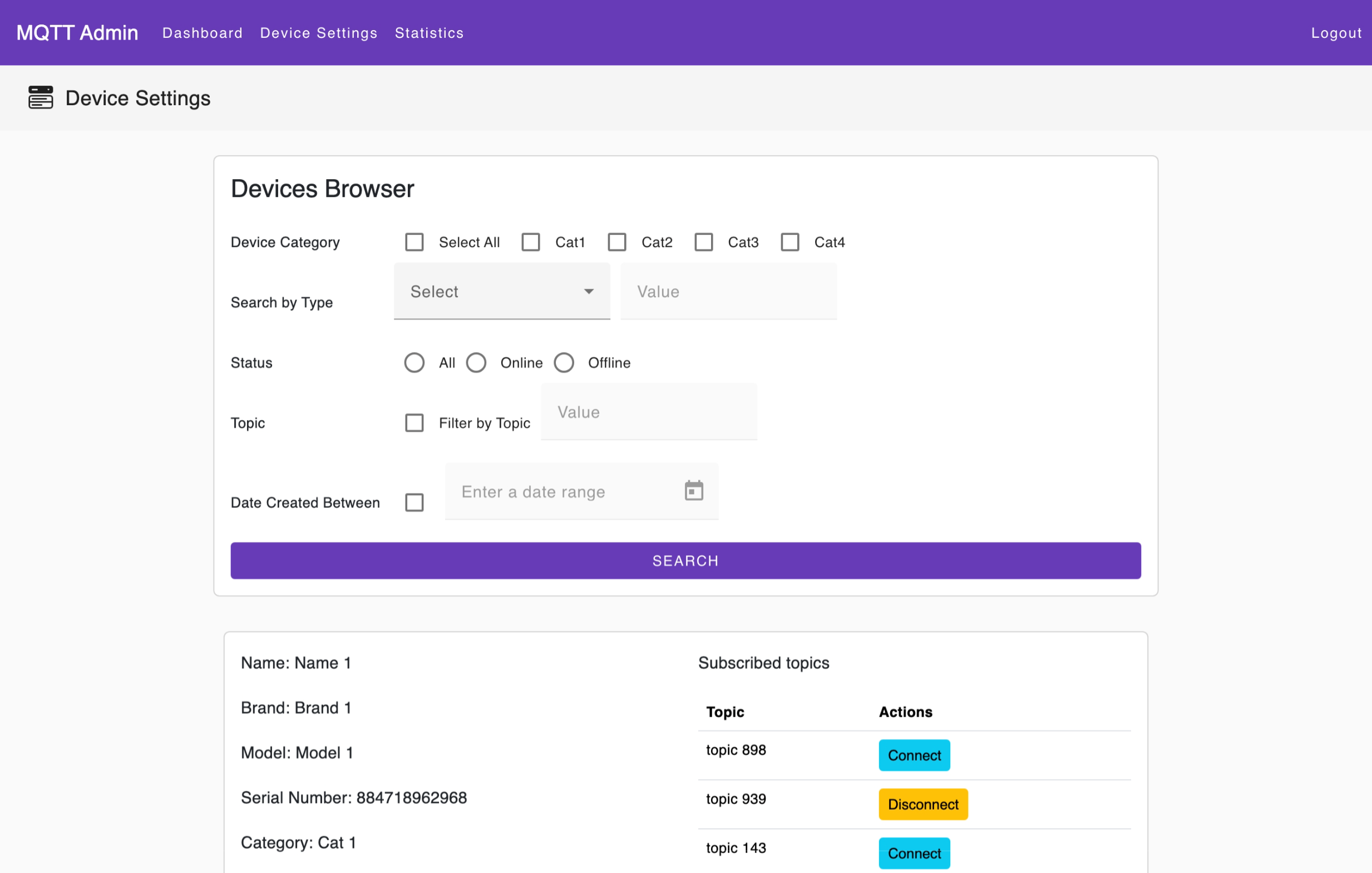
## 注册界面



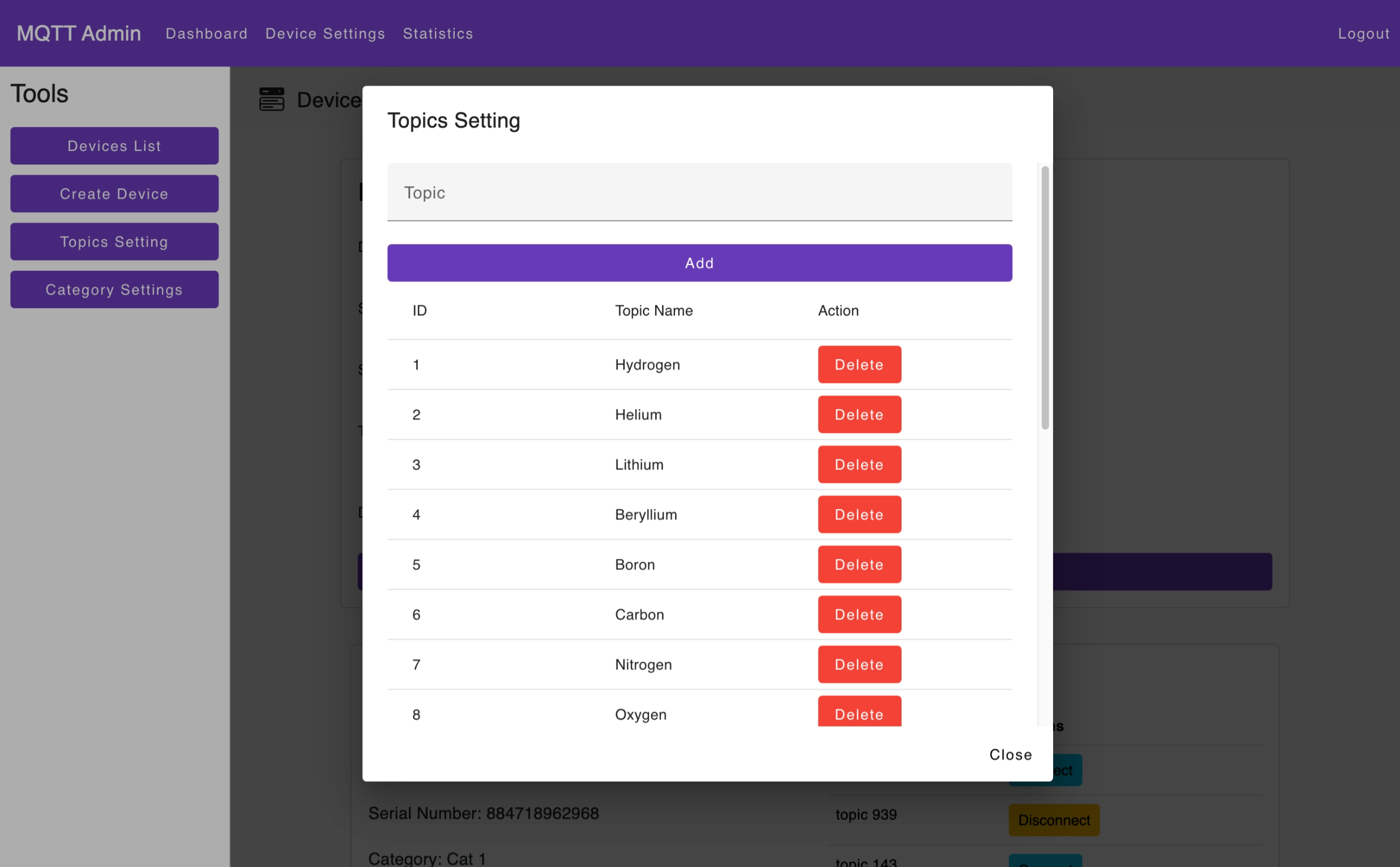
## 主页



## 配置界面



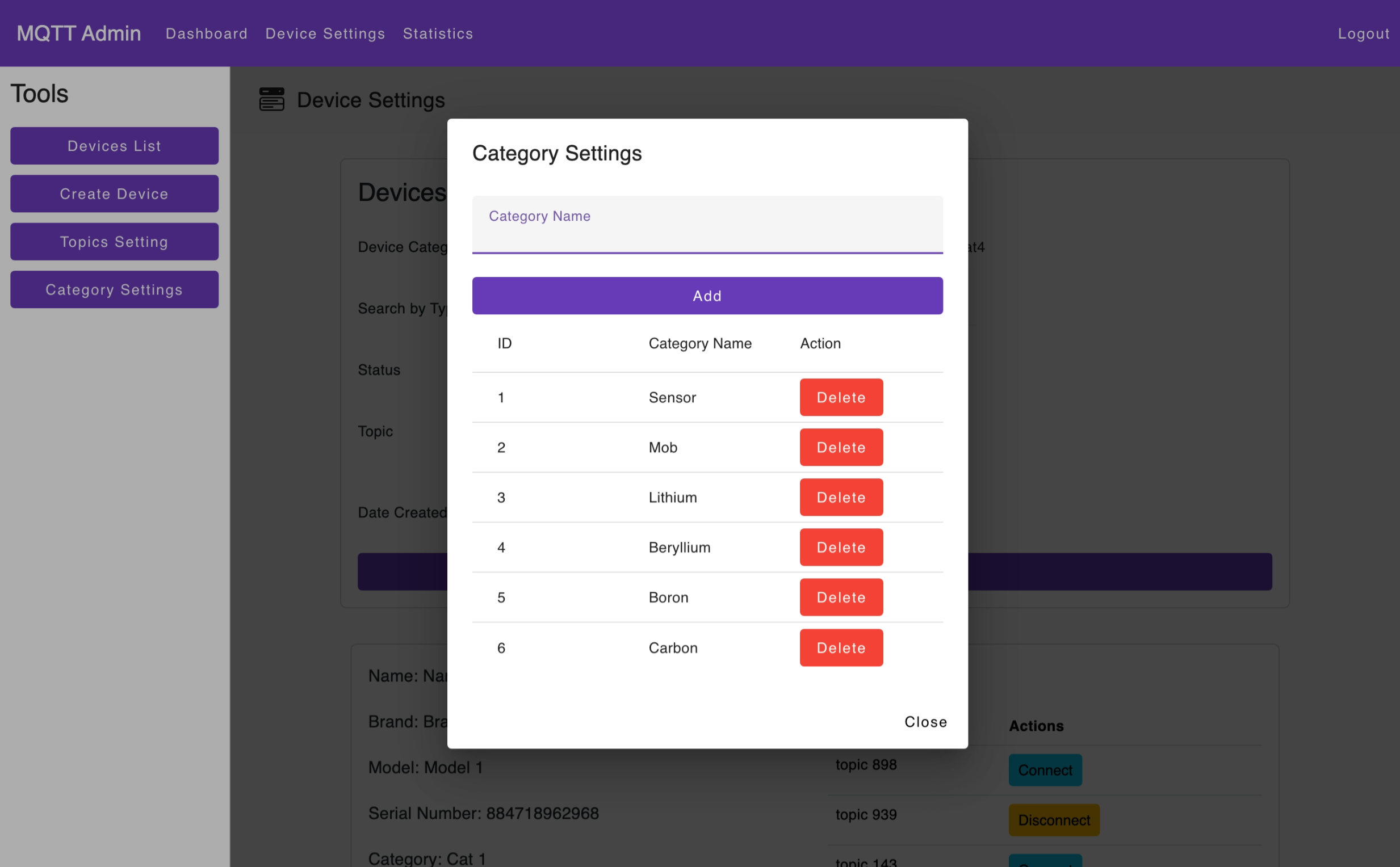
### 订阅标题管理



### 创建设备



### 设备类型管理



## 设备上报数据的查询统计界面

