浙江大学

BS 体系软件设计

物联网应用网站
设计报告

姓名：_____

学院：_____

专业：_____

学号：_____

指导教师：_____

May 6, 2021

Contents

1 引言

MQTT（Message Queuing Telemetry Transport，消息队列遥测传输协议），是一种基于发布/订阅（publish/subscribe）模式的”轻量级”通讯协议，该协议构建于 TCP/IP 协议上，由 IBM 在 1999 年发布。MQTT 最大优点在于，可以以极少的代码和有限的带宽，为连接远程设备提供实时可靠的消息服务。作为一种低开销、低带宽占用的即时通讯协议，使其在物联网、小型设备、移动应用等方面有较广泛的应用。

MQTT 协议是轻量、简单、开放和易于实现的，这些特点使它适用范围非常广泛。在很多情况下，包括受限的环境中，如：机器与机器（M2M）通信和物联网（IoT）。其在，通过卫星链路通信传感器、偶尔拨号的医疗设备、智能家居、及一些小型化设备中已广泛使用。

本次开发的是一个基于 MQTT 协议的物联网应用平台，用户可以在平台上注册，登陆，然后管理自己（订阅）的 MQTT 设备，进行查看设备状态，配置设备等操作。

平台采用 Browser/Server 架构，前后端分别实现。

1.1 目标

(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，并可以展示历史轨迹。

(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）。

1.2 设计说明

本平台前端采用 React 框架，使用了 Ant Design 组件库，后端采用 Java Spring Boot 框架。

2 总体设计

总体架构图：

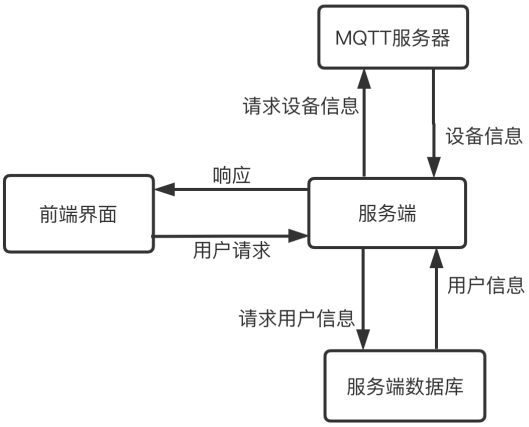

Figure 1: **总体架构图**

用户通过操作浏览器网页对服务器发出请求：

- 如果是关于用户信息的请求，比如注册，登陆，服务端会请求数据库
- 如果是关于 MQTT 设备，服务端会请求 MQTT 服务器

2.1 前端总体设计

前端主要分为 4 个界面，登陆/注册界面，设备管理界面，地图界面，历史数据界面。

- 用户在未登陆之前只能打开登陆/注册界面，选择登陆/注册；
- 用户登陆/注册后会来到设备管理界面；
- 用户可以在设备管理界面点击某个设备对其进行配置；
- 用户可以在侧边栏切换不同的信息显示模式，包括普通的陈列，地图，历史数据；

2.2 后端总体设计

后端主要分为两个部分，数据库和 MQTT 服务。

数据库中会存储用户的 ID，密码，邮箱等个人信息，还有用户订阅的 MQTT 设备。MQTT 服务主要用于从 MQTT 服务器上订阅，接收消息，并返回给前端。

2.3 流程图设计

用户登陆：

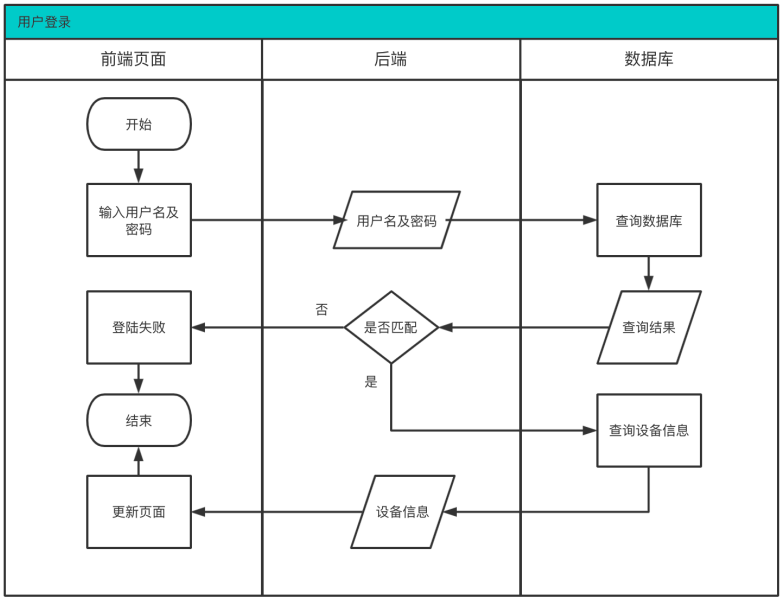

Figure 2: **用户登陆流程图**

用户注册：

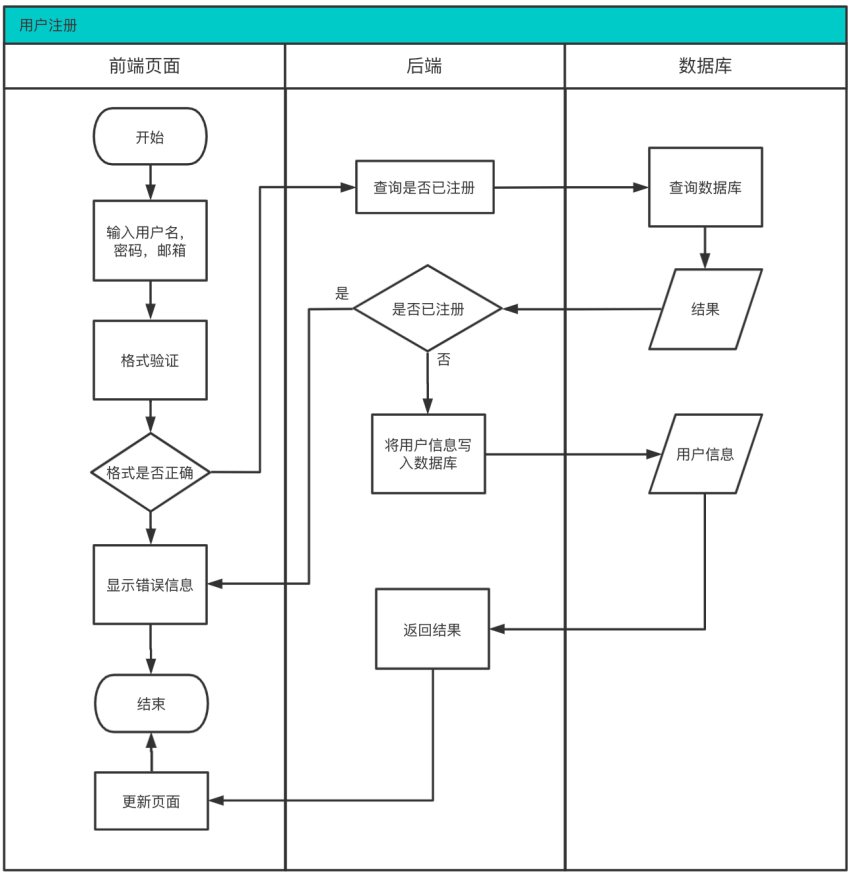

Figure 3: **用户注册流程图**

用户添加设备：

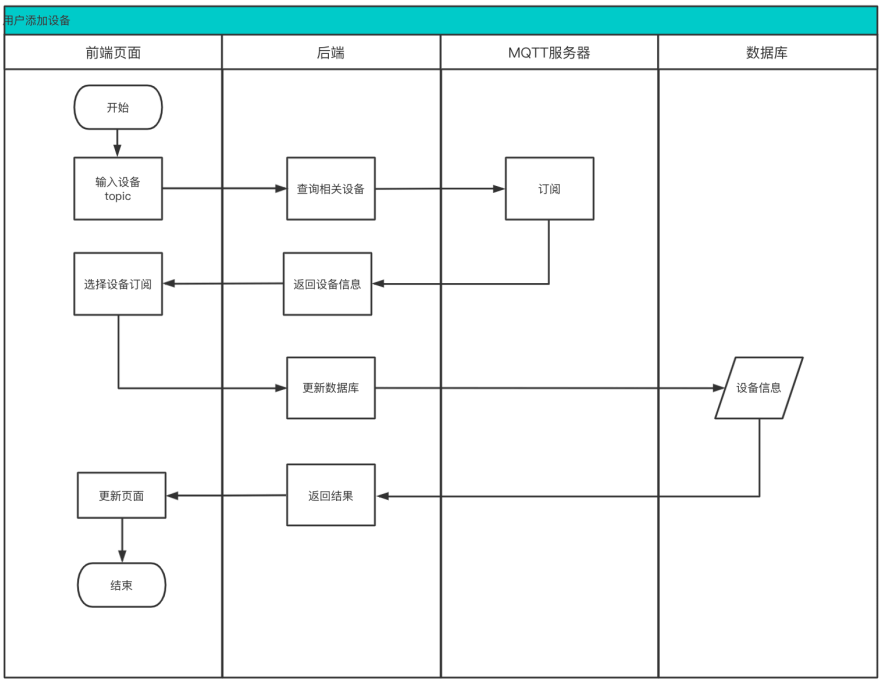

Figure 4: **用户添加设备流程图**

用户接收设备信息：

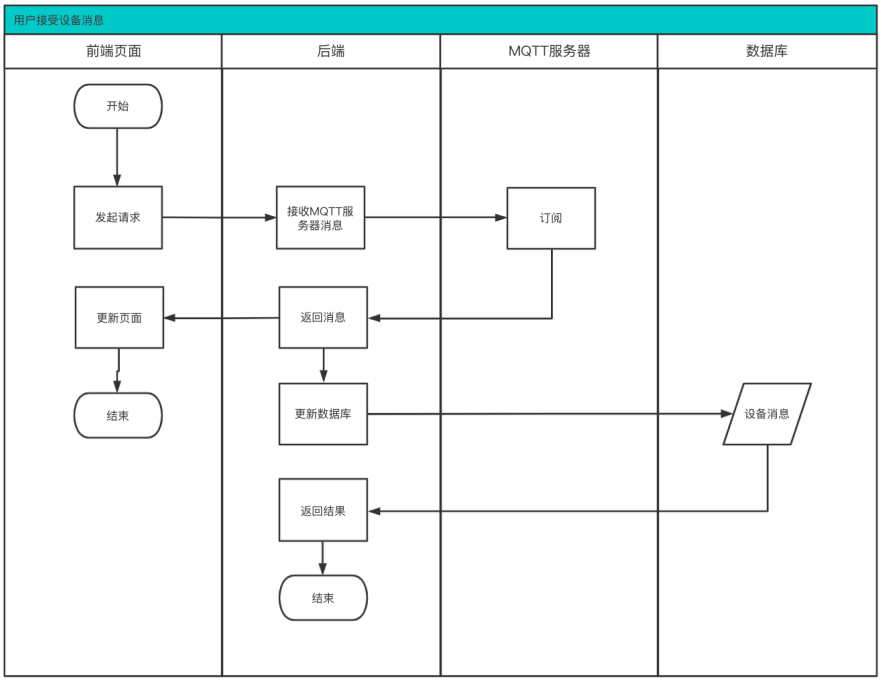

Figure 5: **用户接收设备信息流程图**

用户修改设备配置：

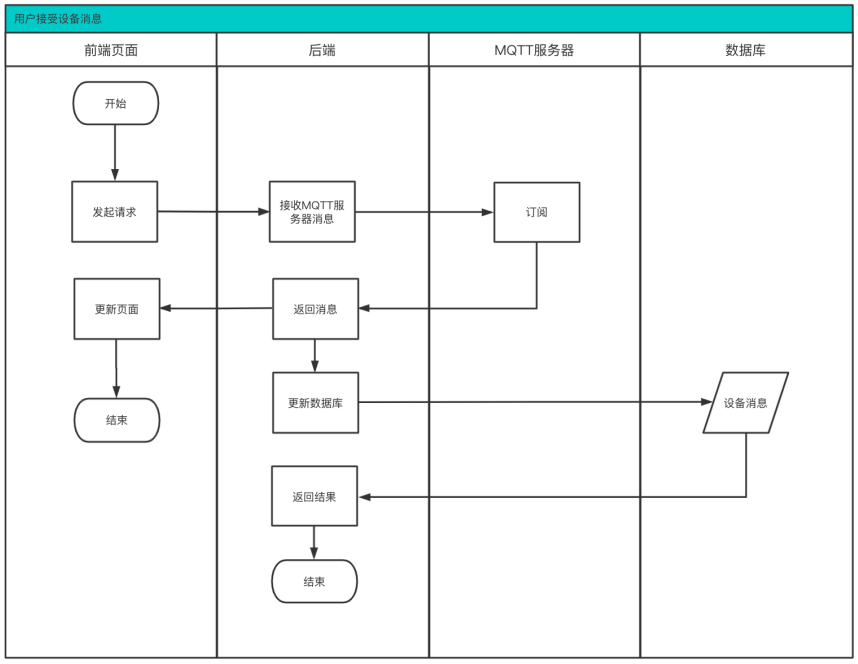

Figure 6: **用户修改设备配置流程图**

3 详细设计

3.1 前端详细设计

主要页面 UI 概念:

登陆/注册界面:

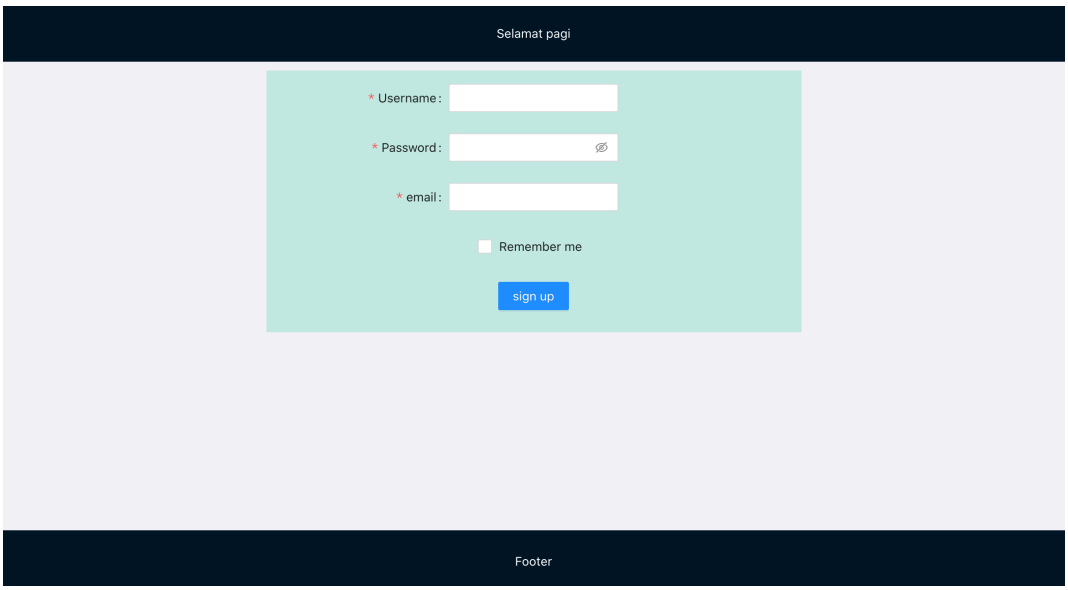

Figure 7: **注册界面（大致框架）**

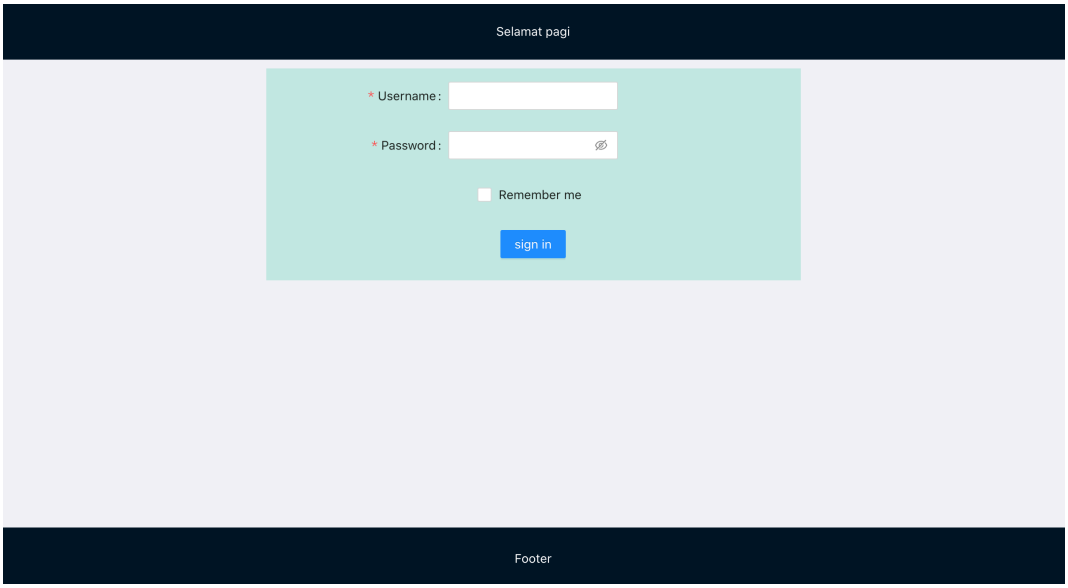

Figure 8: **登陆界面（大致框架）**

设备管理界面：

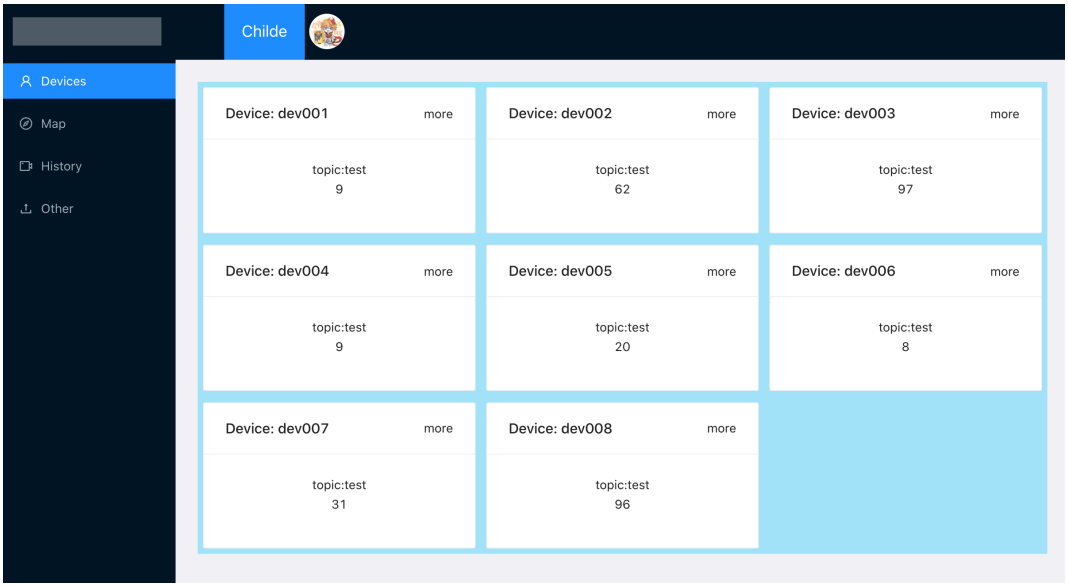

Figure 9: **设备管理界面（大致框架）**

地图显示：

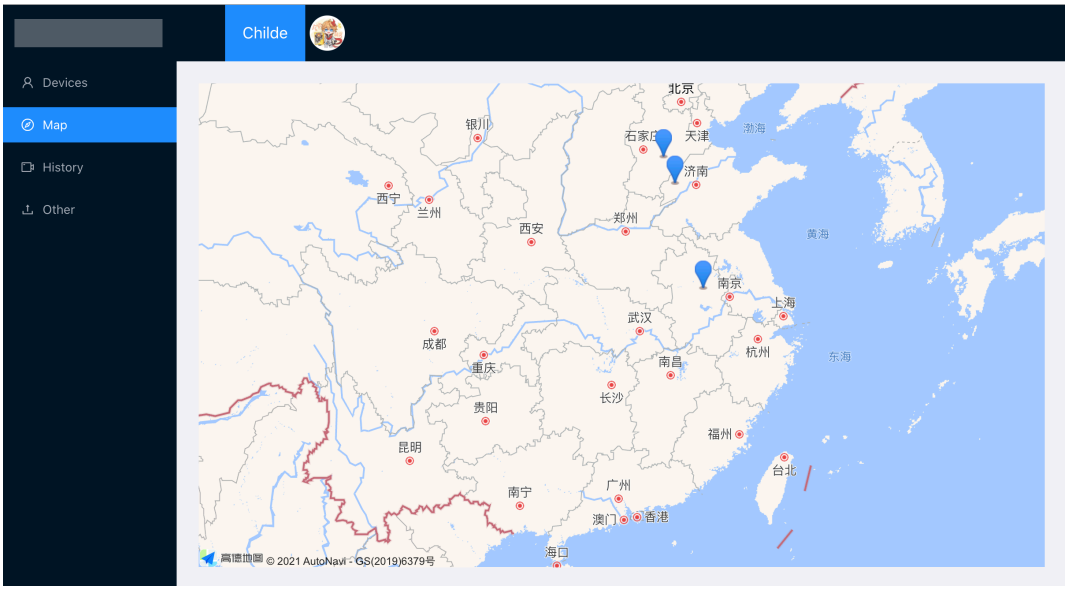

Figure 10: **地图界面（大致框架）**

历史数据界面：

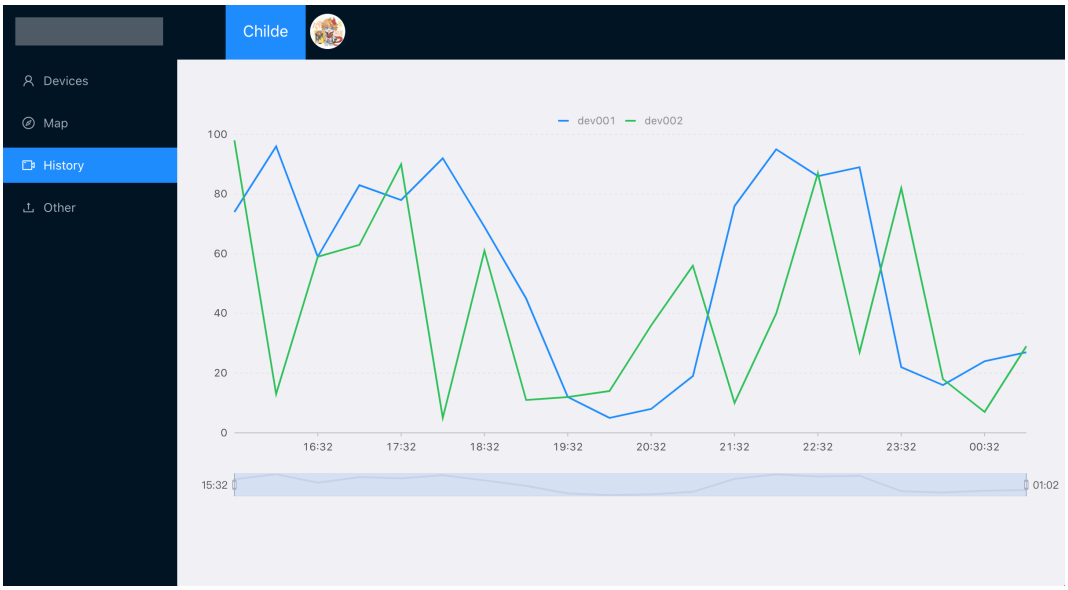

Figure 11: **历史数据界面（大致框架）**

3.2 后端详细设计

MQTT 服务

MQTT 服务的实现应用了 Eclipse 提供的 Paho Java 客户端库。

Paho Java 客户端提供了两个 API，MqttAsyncClient 提供了一个完全异步的 API，其中活动的完成是通过注册的回调通知的。MqttClient 是 MqttAsyncClient 周围的同步包装器，其中函数与应用程序同步显示。

每个 MQTT 设备都有自己的 topic，多个 MQTT 设备可能有相同的 topic，而 Paho Java 客户端提供了按 topic 订阅设备消息的方式。

后端会根据用户输入的 topic 去向 MQTT 服务器订阅消息。

后端以 topic 为单位接收消息，每个 topic 可能有多个设备，后端会过滤出哪些被用户添加的设备的消息然后返回给用户。

如果用户要添加某一设备，需要先输入设备的 topic，后端会把 topic 相关的所有设备都列出来供用户选择，如果 topic 已经被订阅，则直接从该 topic 相关的未添加的设备中选择，如果该 topic 还未被订阅，则订阅该 topic。

3.3 数据库设计

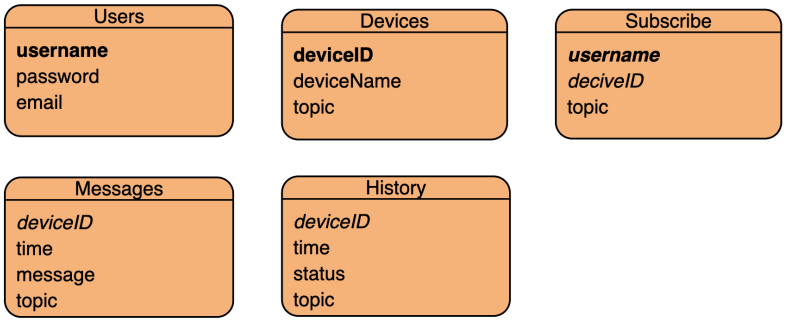

Figure 12: **数据库 *Tables***

其中，

- Users 存储了用户信息，包括用户名，密码，邮件等信息，其中用户名为主键；
- Devices 存储了设备信息，包括 ID，由用户定义的名字，相关的 topic 等，其中 ID 为主键；
- Messages 存储了设备的所有消息，包括 ID，时间，对应的消息，相关的 topic，没有主键；
- History 存储了设备的状态信息，包括 ID，状态，其中状态包括地理位置等信息，没有主键；
- 只有被用户添加的设备才会被计入数据库；

4 关键技术

4.1 React

React 是一个为数据提供渲染为 HTML 视图的开源 JavaScript 库。React 视图通常采用包含以自定义 HTML 标记规定的其他组件的组件渲染。React 为程序员提供了一种子组件不能直接影响外层组件（"data flows down"）的模型，数据改变时对 HTML 文档的有效更新，和现代单页应用中组件之间干净的分离。

它由 Facebook、Instagram 和一个由个人开发者和企业组成的社区维护。根据 JavaScript 分析服务 Libscore，React 目前正在被 Netflix、Imgur、Bleacher Report、Feedly、Airbnb、SeatGeek、

HelloSign 等很多网站的主页使用。

在本项目中，React 主要用于前端开发，配合 Ant Design 组件库为前端页面提供简洁高效的样式。

4.2 Spring

Spring 框架是 Java 平台的一个开源的全栈应用程序框架和控制反转容器实现，一般被直接称为 Spring。该框架的一些核心功能理论上可用于任何 Java 应用，但 Spring 还为基于 Java 企业版平台构建的 Web 应用提供了大量的拓展支持。Spring 没有直接实现任何的编程模型，但它已经在 Java 社区中广为流行，基本上完全代替了企业级 JavaBeans（EJB）模型。

Spring 框架以 Apache License 2.0 开源许可协议的形式发布，该框架最初由 Rod Johnson 以及 Juergen Hoeller 等人开发。

在本项目中，Spring 主要用于后端，即服务端开发，包括与前端交互，访问数据库，与 MQTT 服务器交互。

4.3 Mosquitto

Eclipse Mosquitto 是实现 MQTT 协议版本 5.0、3.1.1 和 3.1 的开源消息代理（经 EPL/EDL 许可），也就是一个 MQTT 服务器。Mosquitto 比较轻巧，适合在从低功耗单板计算机到完整服务器的所有设备上使用。

MQTT 协议提供了使用发布/订阅模型执行消息传递的轻量级方法，这使其适用于物联网消息传递，例如具有低功率传感器或移动设备（例如电话，嵌入式计算机或微控制器）的消息传递。

在本项目中，Mosquitto 主要用于提供 mqtt 服务，与后端相对独立。