



网络编程技术

IOT设备管理系统

程序说明书

班级：2014211314

姓名：叶文霆

学号：2014211519

一. 实现功能

实现的基本功能有：

- 用户管理功能（即创建与登录用户）
- 设备信息的查看
- 设备的添加、删除、更新功能
- 模拟设备向服务器上报状态

实现的扩展功能有：

- 查看设备的历史数据（温度、开关）
- 设备数据的同步刷新
- 设备开关的控制
- 设置报警阈值，当温度超过阈值将报警
- 生成报警日志（支持 excel, csv, pdf）

实现的额外功能有：

- 设备信息的实时搜索、排序、分页
- 用户修改密码
- 多用户访问（为每个用户维护单独的设备列表与报警阈值）
- 验证提交数据有效性
- 无效URL请求返回404页面

前端：

- 使用Xenon Admin模板进行网页设计
- 使用jQuery DataTable插件控制表格的搜索、排序、分页、Ajax刷新与日志导出功能
- 使用Toastr插件控制报警与通知的显示
- 使用jQuery Validate检验表单数据

后端：

- 使用Django Admin进行用户管理

二. 系统设计

1. 设备管理

1.1 模型建立

设备模型一共有7个字段：

字段	字段类型	默认值
SN	字符串	-
设备名	字符串	-
最近修改	时间	timezone.now
警告阈值	整型数	-
用户	外键 (User)	-
开关	布尔型	TRUE
温度	整型数	0

表1 DEVICE模型

对上述字段进行补充说明：

- 不存在默认值的字段，在新增或更新数据时，必须提供准确值；
- 用户字段存储设备所属的用户，用于多用户管理；
- 开关默认打开；
- 最近修改默认为新建设备时的时间，温度为0，当设备发来第一条数据后将会覆盖初始状态。

1.2 新增设备

新增设备的流程如图1所示：

1. 用户填写表单。
2. jQuery Validate对数据进行验证，要求：
 - SN必填，且长度不超过10；
 - 设备名必填，且长度不超过20；
 - 报警阈值必填，且是一个整数
3. 用户端打包好数据字典用Post发送给URL： /devices/add/

4. 服务器端收到请求，并从request中取到userid，在Device表中创建一个新设备，同时在DeviceLog表中创建设备的初始记录，将结果返回一个字符串给客户端。
5. 客户端收到返回的json数据，分情况处理：
 - 字符串是“success”，则显示成功，并刷新数据表。
 - 字符串是“existed”，则说明该设备已存在于用户设备表中，显示错误信息。
 - 字符串是“invalid”，则说明提交表单信息无效，显示错误信息。

客户端和服务器的流程图如图1所示。

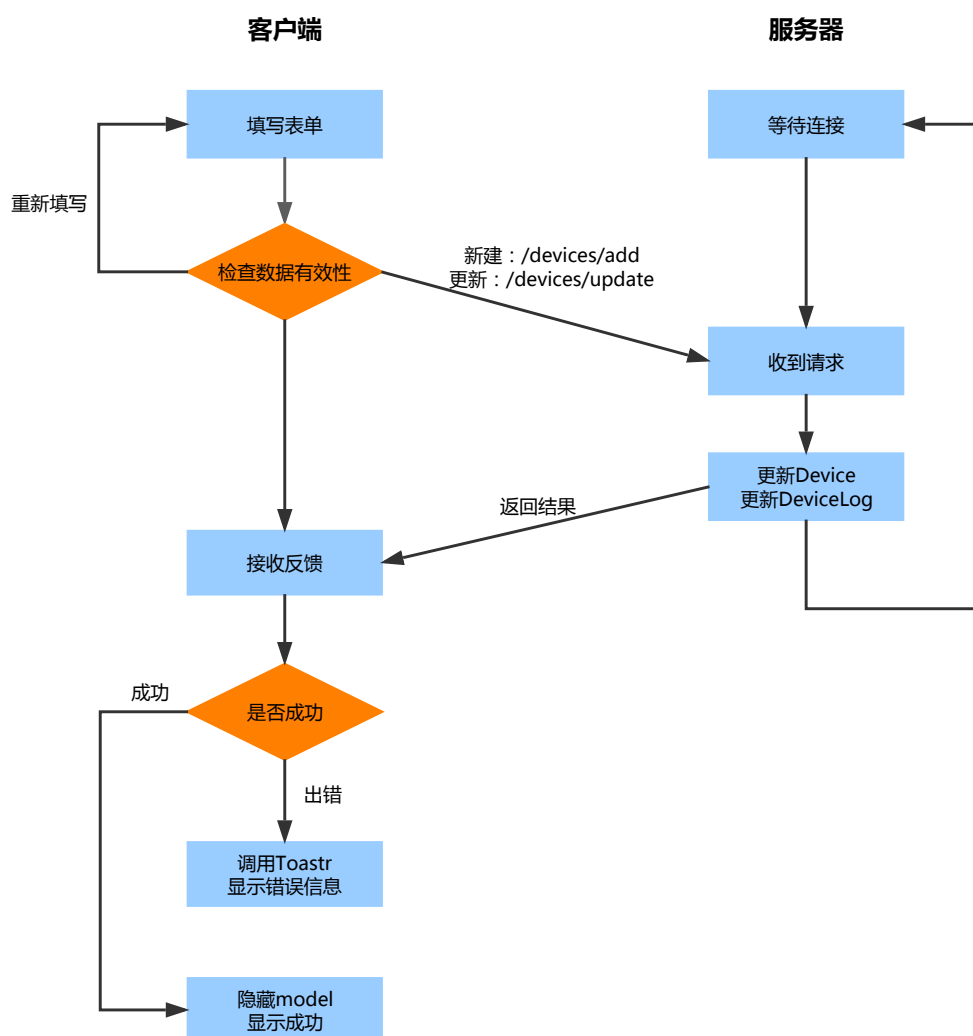


图1 修改/新建设备流程图

1.3 更新设备

更新设备的流程如图1所示：

1. 用户点击设备项旁的更新按钮，填写表单。
2. jQuery Validate对数据进行验证，要求：
 - 设备名必填，且长度不超过10；

- 报警阈值必填，且是一个整数
3. 用户端连同设备的ID号，打包好数据字典向URL: /devices/update/ 发送Post请求
 4. 服务器收到请求，并从request中取到userid，首先判断该用户的设备表下是否有该设备，若存在再更新该设备的信息。同时在DeviceLog表中记录修改，将结果返回一个字符串给客户端
 5. 客户端收到返回的json数据，分情况处理：
 - 字符串是"success"，则显示成功，并刷新数据表。
 - 字符串是"nonexisted"，则说明该设备已不存在，显示错误信息。
 - 字符串是"invalid"，则说明提交表单信息无效，显示错误信息。

为提高重用性，更新设备和新增设备使用同一个表单和提交函数，而使用变量save_method的值来区分，使其将表单提交给不同的URL。

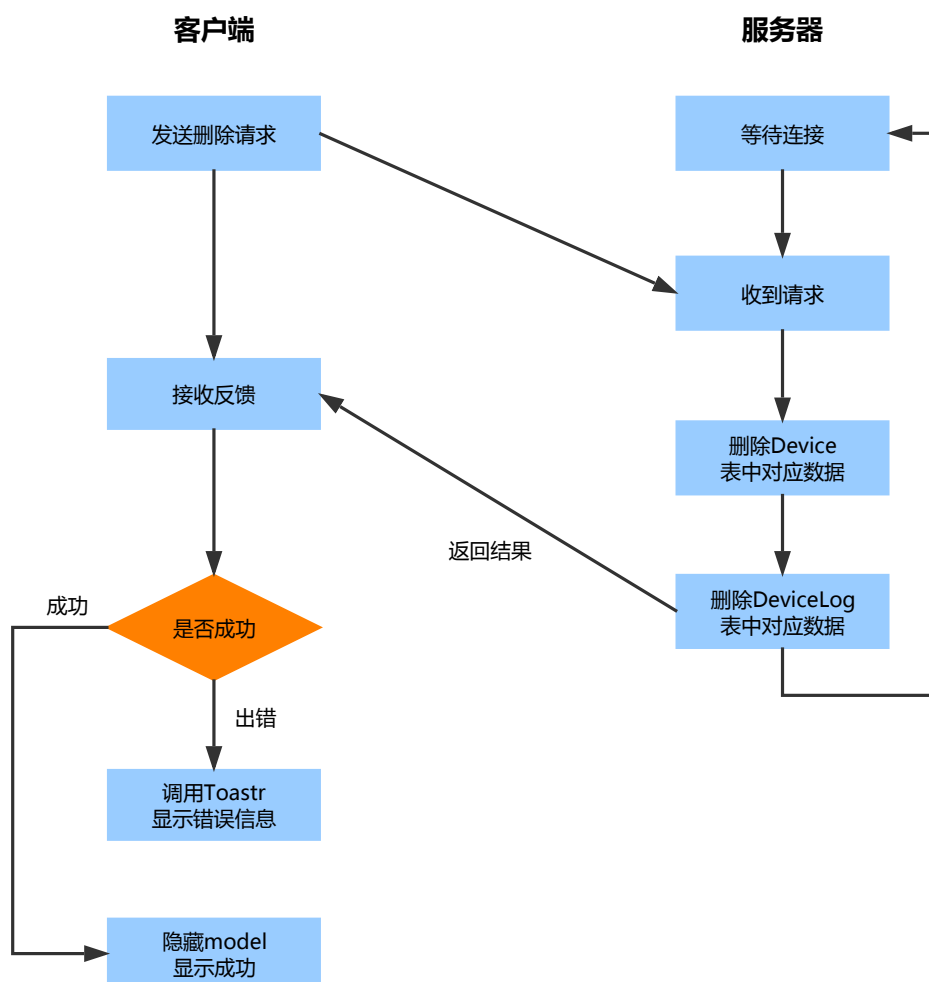


图2 删除设备流程图

1.4 删除设备

删除设备的流程如图2所示：

1. 用户点击设备项旁的删除按钮，触发jQuery事件

2. 客户端向服务器端URL: /devices/delete 发送设备的id号
3. 服务器端收到请求，删除该用户设备表中的对应设备，返回处理结果
4. 用户端接收服务器反馈，分情况处理：
 - 字符串是“success”，则显示成功，并刷新数据表。
 - 字符串是“nonexisted”，则说明该设备已不存在，显示错误信息。

流程图如图2所示。

2. 日志管理

2.1 模型建立

日志模型用于保存设备的历史状态，按照项目需求其应有以下字段：

字段	字段类型	默认值
SN	字符串	-
时间	时间	timezone.now
温度	整型数	-
开关	布尔型	-

表2 DEVICELOG模型

对上述字段进行补充说明：

- 每次设备状态，包括温度、开关状态的变化，都新建一个表项
- 时间每次使用当前时间点作为默认值保存

2.2 查看记录

查看记录有两种方式：

- 单独查看某一设备的记录
- 查看所有记录（支持搜索、排序）

流程如下：

- 客户端DataTable服务器发送Ajax请求：如果单独查看某一设备，还需要用Get方式提交SN号
- 服务器收到客户端的请求，从DeviceLog表格中用userid筛选出该用户的日志表，如果是单独查看则还需要用SN筛选。将处理好的数据数组回传给客户端
- 客户端收到Json数据后显示。

流程图如图3。

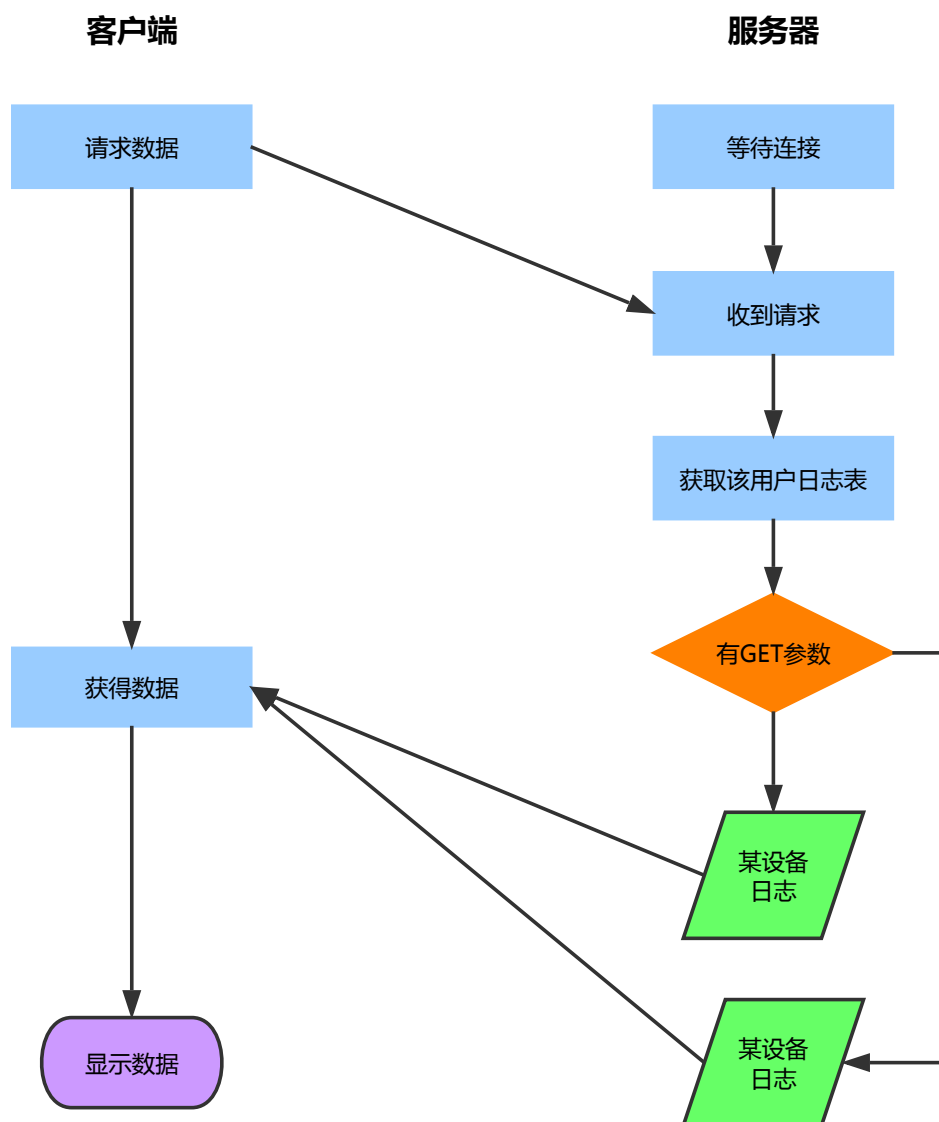


图3 查看日志流程图

2.2 查看告警记录

和查看记录方式相同，只是对温度再做一次筛选，故在此不再赘述。

3. 开关&设备模拟

3.1 设备模拟

首先读取配置文件，获得设备的设备号和温度范围（均匀分布）。获得数据后创建多个线程，每个线程对应一个设备，每隔固定时间S1向服务器用POST方式发送数据，包括：

- SN：表示设备号
- Temperature：表示设备温度（如设备休眠则不需要发送此字段）

每次服务器根据请求中的设备号和温度，返回一个包含两个表项的字典：

- add：表示添加是否成功。

- is_open：表示设备开关。is_open=1代表服务器打开设备，is_open=0表示关闭。

3.2 开关模拟

考虑到不可能真的终止设备程序运行（服务器不能命令设备端运行程序），我所采用的做法是让设备进入休眠。

当设备收到服务器的关闭指令，便进入休眠状态。进入休眠的设备虽然不更新温度，但仍然每隔固定向服务器发送设备号以或许服务器对设备的控制。如果发现服务器回送一个is_open字段为True的Json文件，则重新将设备设为工作状态。

3.3 工作流程

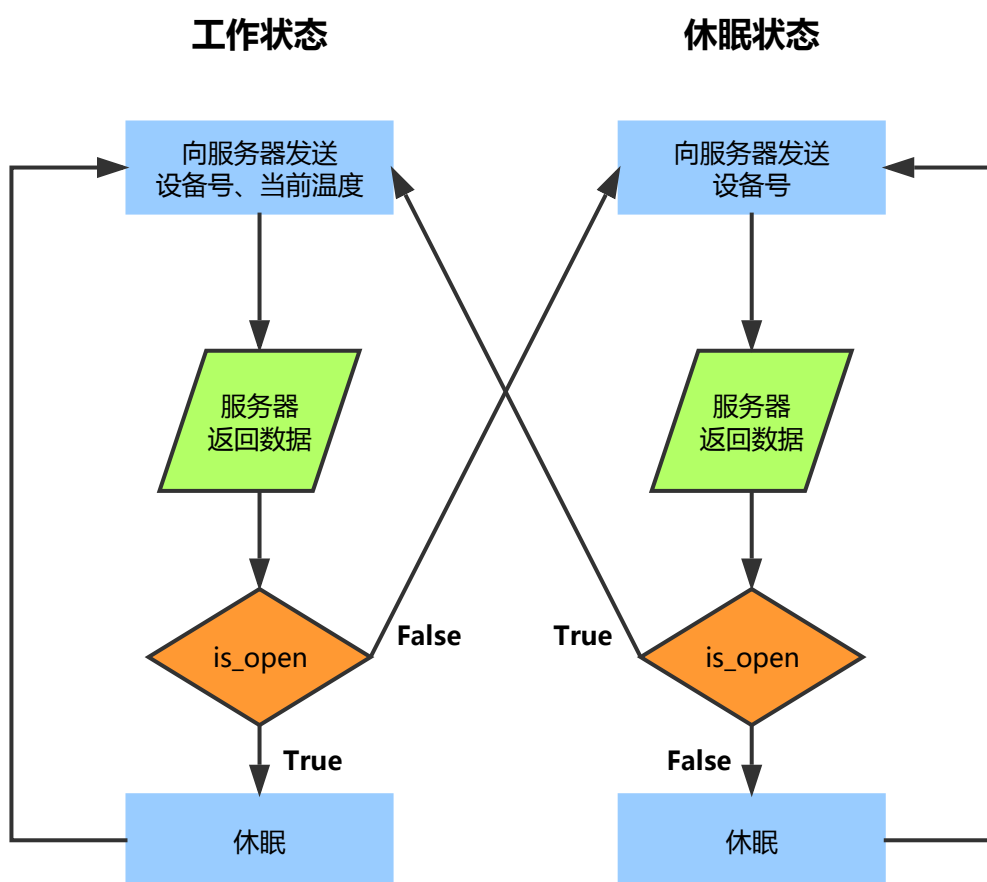


图4 模拟设备工作流程图

三. 实现&测试

1. 用户管理

1.1 用户登录&注册

访问网站地址，将会自动判断是否登陆过，如果没有登陆过将会需要进行用户登录。如果没有用户则可以点击下面的链接进行注册。同时如果用户验证失败，会给出相应提示框，光标会自动选中密码框。

如果用户没有登录，则可以点击下方按钮注册。

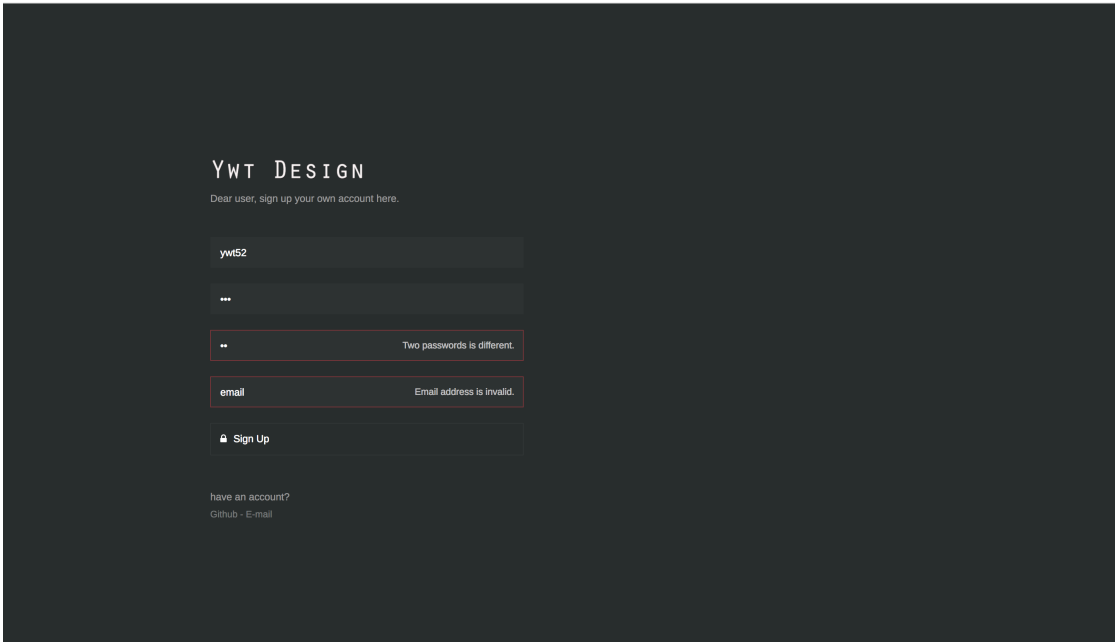


图5 注册数据检查

在用户注册页面中，页面会动态检查用户输入数据的有效性，如图5所示。

1.2 修改密码

登陆后，在导航栏的左侧点击修改密码，按照相应提示即可修改密码，此处页面同样会检查数据的有效性。

2. 设备查看

登陆成功后，将会进入设备查看主页面。

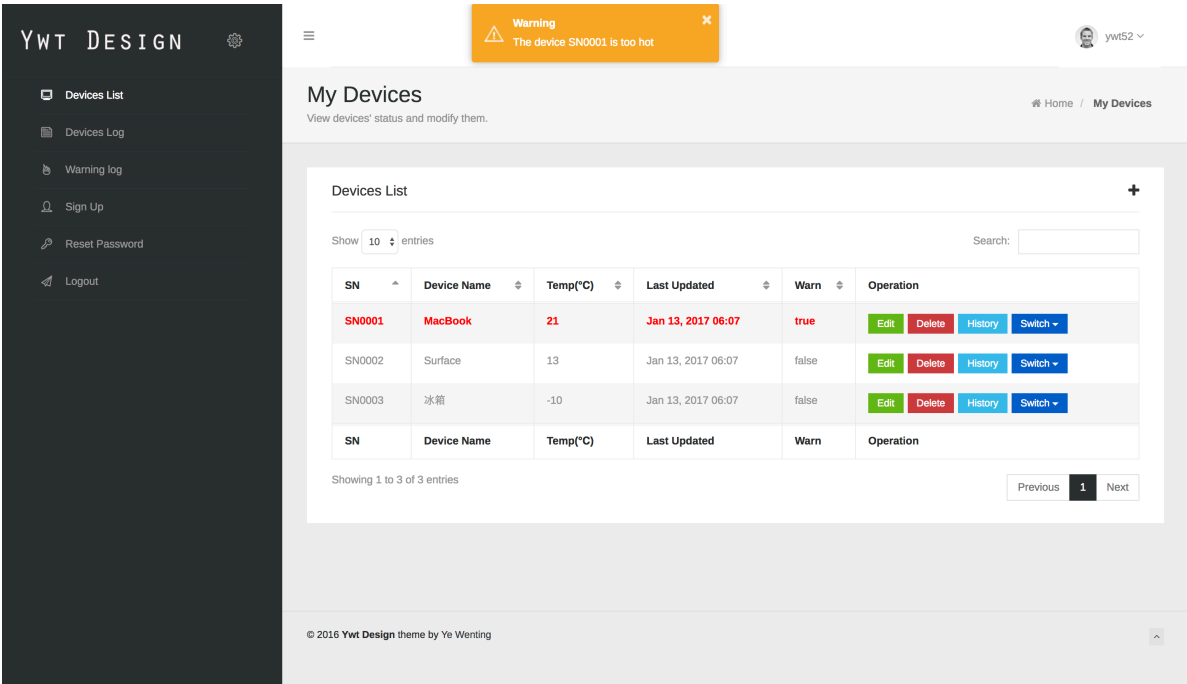


图6 设备查看页面

主页面分为侧边栏，导航栏和内容。侧边栏包含了网站其他功能的链接；导航栏可以修改面板的查看方式，也可以看到当前登陆用户。而内容包含了设备列表。设备列表支持以下功能：

- 显示设备。显示当前用户的设备列表
- 提示报警。对于温度高于阈值的设备，在列表上予以高亮显示，同时在页面上方显示能够延迟消失的提示框。
- 支持排序。通过点击每个表项的表头即可排序，默认是以SN字典序排序
- 支持查找。在右侧搜索框中输入SN或设备名，下方列表就会动态显示对应的查找结果。
- 支持分页。点击右下角的页号进入不同的页，左上角可以设置不同每页的显示数。
- 自动刷新。默认情况下本页面每隔5s向服务器发起请求，刷新数据。

3. 设备管理

3.1 新增设备

点击表格左上角的“+”可以新增设备，输入SN号，设备名和告警阈值即可创建。

此处页面会检查输入SN号长度和阈值类型是否符合要求，提交表单后如果该用户已经存在有对应设备号的设备，则会提示失败。

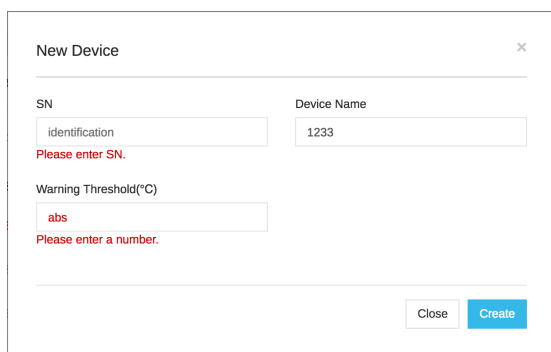


图7 新增设备数据检查

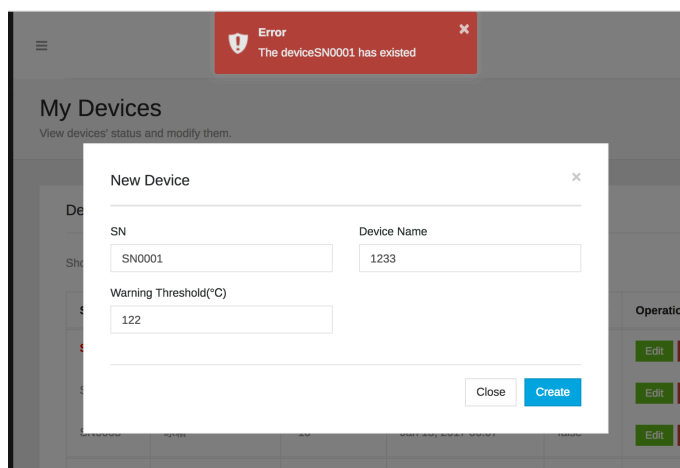


图8 设备查重检测

3.2 删除设备

点击设备右边的删除按钮，即可删除对应设备。

3.3 更新设备

点击设备右边的更新按钮，如图可以观察到设备号是无法编辑的，输入新的设备名和报警阈值即可更新。

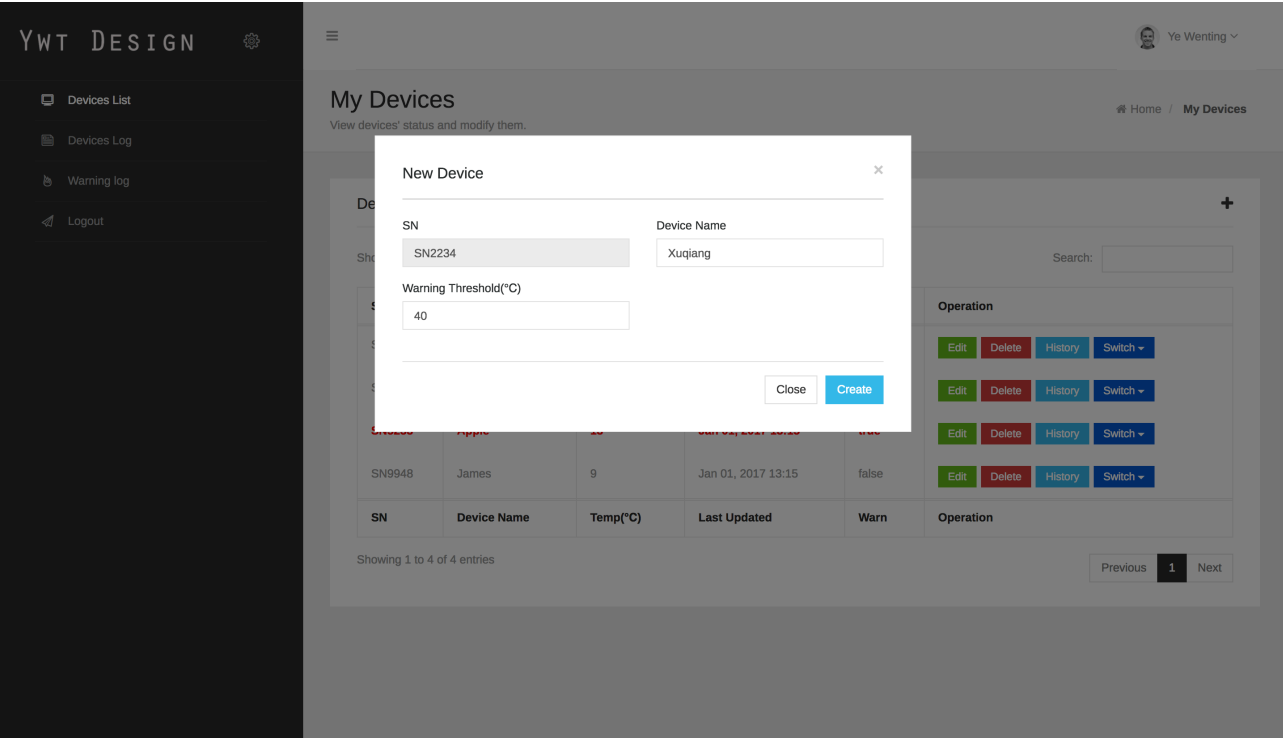


图9 更新设备页面

4. 告警提示

当设备当前温度超过用户事先设定的阈值时，这一行表项将会加粗标红，同时会有警告窗弹出，如图6所示。

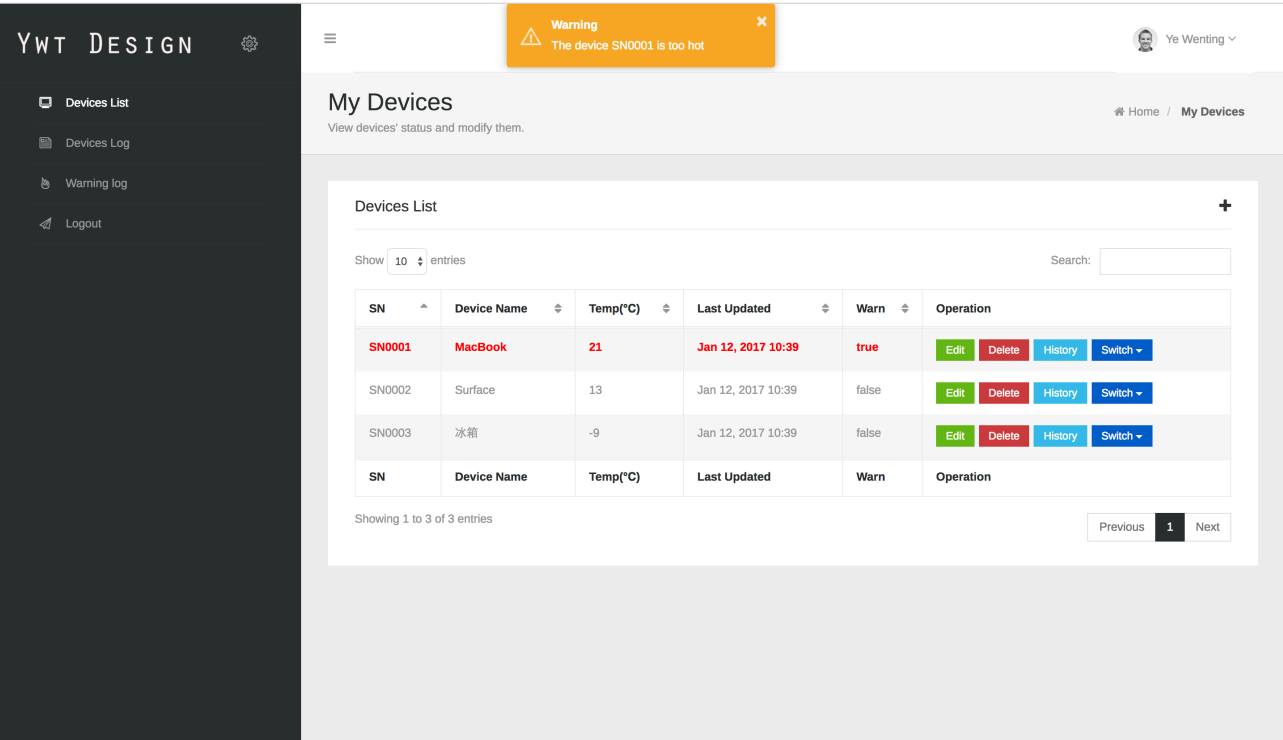


图10 设备报警

5. 开关控制

1. 点击设备最右边的开关，选择关闭设备，页面提示关闭成功，可以看到关闭的设备用绿色标出。
2. 客户端可以观察到该设备的“最近更新”字段不再刷新，从模拟设备程序可以观察到该设备已经被关闭，如图7。
3. 再次打开设备，页面提示打开成功。
4. 此时设备又开始向服务器发送数据，如图8。

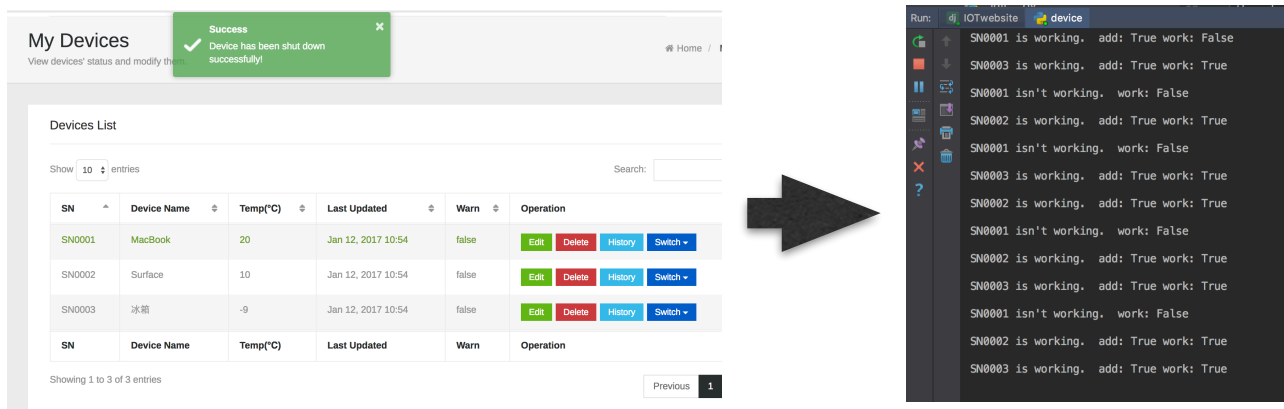


图11 关闭设备测试

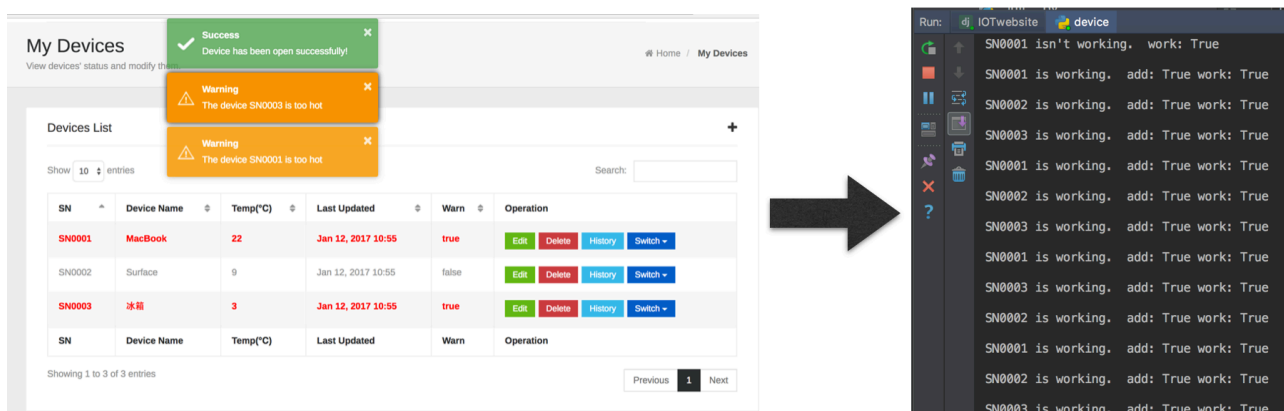


图12 重新打开设备

6. 查看日志

通过点击页面上的查看日志，或者点击设备旁边的历史按钮，均可以查看设备的工作日志。

7. 日志导出

在查看日志或报警日志页面下，点击表格左上方的按钮即可导出表格，图9是以excel表格为例。

The screenshot displays the 'Devices Log' interface on the left and an Excel spreadsheet on the right, connected by a large black arrow pointing from the interface to the spreadsheet.

Devices Log Interface:

- Header: **Devices Log** (View devices' history log easily.)
- Buttons: Copy, **Excel** (highlighted with a red circle), CSV, PDF.
- Table:

SN	Device Name	Time
SN0001	MacBook	Jan 12, 2017 10:16:05
SN0001	MacBook	Jan 12, 2017 10:30:49
SN0001	MacBook	Jan 12, 2017 10:30:59

Excel Spreadsheet:

- Worksheet: **G71**
- Columns: A (SN), B (Device Name), C (Time), D (Temperature), E (State), F (G), H, I, J, K.
- Rows: 1 to 30, showing device activity logs with timestamps and states (Active, Inactive, Error).

图13 导出EXCEL表格

8. 多用户测试

3.1 设备列表的独立性

我们新建一个Test用户并登陆，如图10。

登陆进去后是一个新的表格，同时在左上角显示有用户名。

我们可以对这个设备表进行正常的添加删除和查看日志等操作，退出后再重新登陆也能够保存其数据，证明本系统支持多用户维护多个设备表。

Test

My Devices

View devices' status and modify them.

Home / My Devices

Devices List

Show 10 entries

Search:

SN	Device Name	Temp(°C)	Last Updated	Warn	Operation
No data available in table					

Showing 0 to 0 of 0 entries

Previous

Next

图14 多用户测试

3.1 单一设备在不同用户间的同步

1. 首先，在ywt52账户下有一个SN0001的设备号，其有26条设备记录。
2. 在新建的Test账户下，新增一个设备号为SN0001的设备，查看其设备号，可以观察到之前的26条记录也显示在了日志列表中。

由此可以证明同一设备可以在不同账号中同步。

四. 问题&解决

Q：更新设备后原有的排序、搜索功能失效？

A：模板中的排序搜索功能是基于DataTable插件完成的，所以如果使用JavaScript语言直接对HTML元素进行重载，只是改变了页面上的显示，而没有重新对新表格元素进行Hash和排序的功能。故而点击相应按钮时会显示空白。

S：使用jQuery DataTable中的dt.ajax.reload()函数

Q：重新加载后的button元素事件绑定失败？

A：原因在于JavaScript中的事件绑定是一次性的，只绑定执行这段代码时的对应元素，而如果是新创建元素则需要重新绑定。

S：不绑定button，而绑定整个table。具体代码如下：

```
$('#device-table').on('click', '.edit-button', ...)
```

该语句的意思是每当ID为device-table的元素被点击时，检查点击的元素是否带有edit-button类，如果是则执行后续的函数。由于table在重载元素时不被修改，所以能保证绑定的正确性。

Q：导入最新的DataTable.js文件后，复选框和model显示失败？

A：原因在于最近的DataTable里面默认导入了jQuery，而与原本导入的jQuery产生冲突，导致错误的产生。

S：取消DataTable里自动导入的jQuery选项。

五. 工作日志

Dec 25, 2016

- 完成静态页面

Dec 26, 2016

- 完成新增设备逻辑

Dec 29, 2016

- 完成删除设备
- 完成Ajax原生刷新
- 发现Ajax刷新以后，DataTable的排序、检索、分页会失效

Dec 30, 2016

- 删除reload_table(), 重写DataTable, 完成增加删除效果

Dec 31, 2016

- 完成dt.ajax.reload()重新加载页面;
- 完成edit功能和add功能合写;
- 使用最新的toastr.js, 优化提示框
- 修改id作为后端Filter的关键字
- 修改数据结构: 新建device的时候只输入名字SN和阈值, 其他默认
- 优化日期显示样式
- 完成定时刷新

Jan 1, 2017

- 重新设计DeviceLog数据
- 完成基于DataTable设计Log.html
- 完成查询某一个设备的记录
- 完成模拟物联网设备发包
- 完成log.html实时刷新
- 完成某一个设备的报警历史的查询
- 完成开关
- 完成日志数据导出
- 增加404页面

Jan 6, 2017

- 优化导出文档按钮样式 (靠左)

Jan 8, 2017

- 解决edit的窗口无法弹出的bug
- 删除无用按钮
- 完成日志页面的重设计

附录

1. 模型关系图

