**网络编程作业报告**

物联网设备Web管理

平台的设计与实现

1. **需求分析及实现功能**
2. 基本的用户管理功能：
3. 用户可以登录和注册。
4. 用户可以更改自己的密码。
5. 用户可以通过注册的邮箱收到警告邮件
6. 用户可以添加删除需要检测管理的设备：
7. 用户可以通过SN号查询设备
8. 用户可以查看设备列表及设备的当前状态：
9. 模拟物联网设备向服务器周期性上报状态：
10. 可以查看设备的历史数据：
11. 前端界面能及时显示设备状态变更情况：
12. 可以控制模拟设备的开和关：
13. 有告警功能，如果超过预先设定的警报值则会向邮箱发送邮件。
14. 所有表格及其数据均为Ajax刷新

**二、开发说明**

1. 服务器开发

次大作业，后端服务器采用python 3.5进行开发，采用Django服务器框架进行设计；

1. 前端web开发

端web使用bootstrap框架进行构建，采用javascript进行前端数据和操作的预处理，建立层次化的网页处理体系；

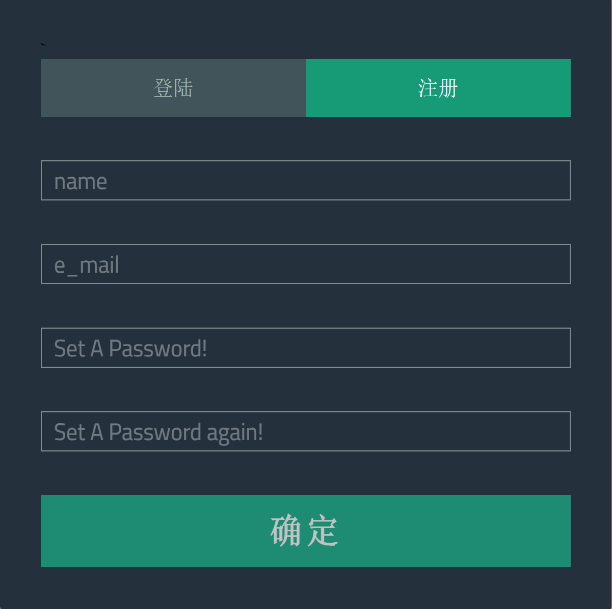
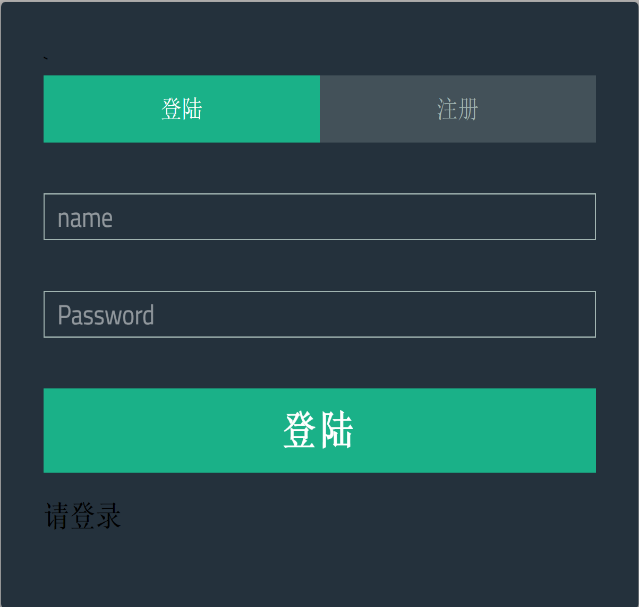
3. 开发环境

本次大作业，使用pycharmIDE进行开发，在windows操作系统和ubuntu操作系统中共同调试完成，支持网站在手机、ipad等移动端的自适应。

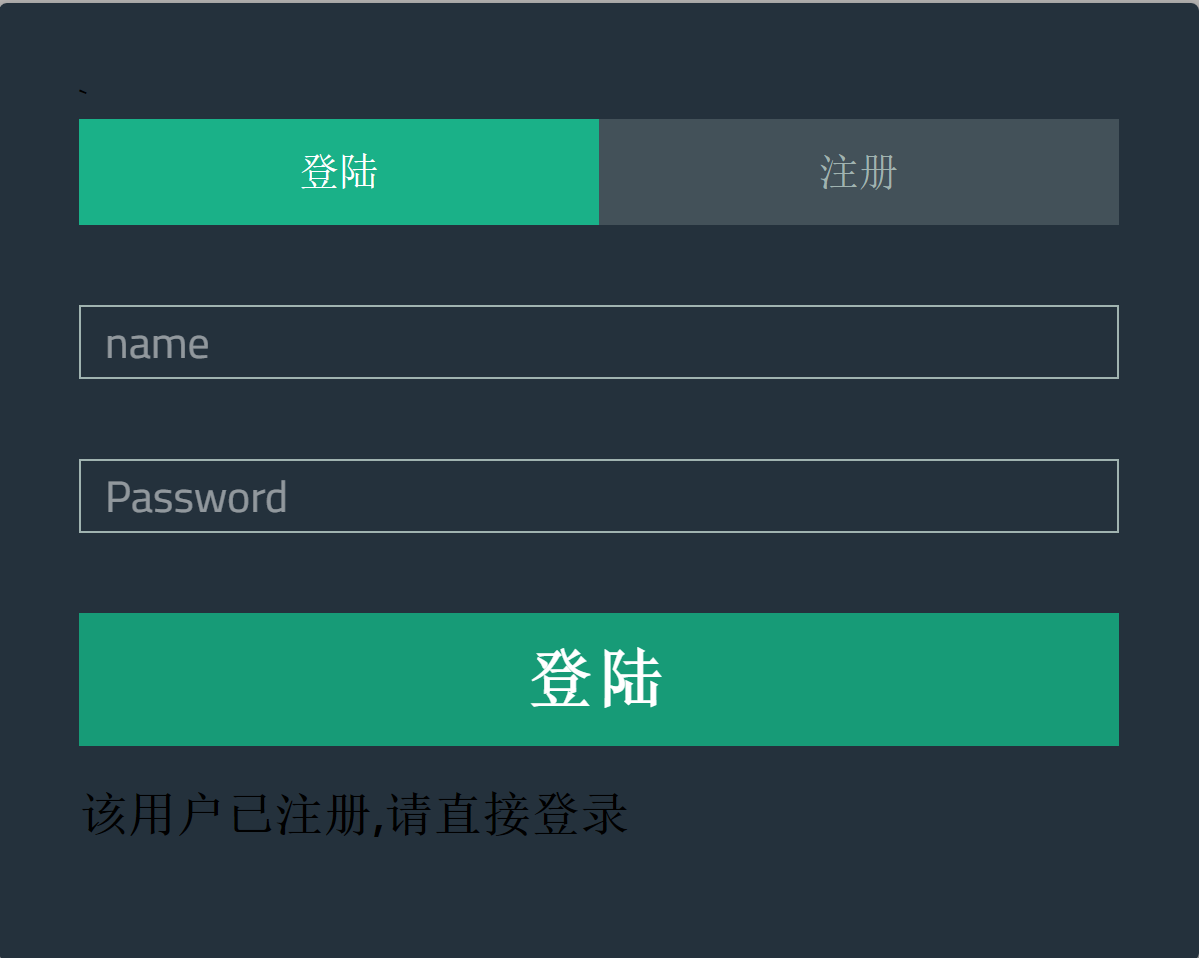
**三、设计**

1.管理平台

* 用户通过login界面登入管理平台。如果用户没有注册账号，可以通过注册页面注册账户。



* 如果注册的账号已经存在，会直接跳转到登录界面，并提示该账号已经存在，请直接登录



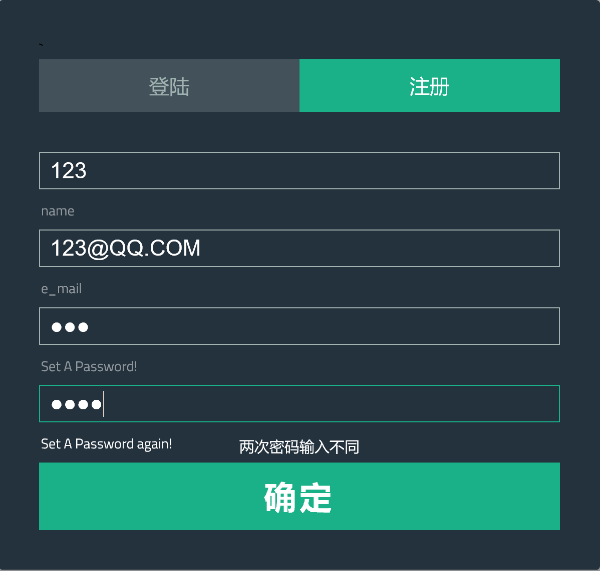
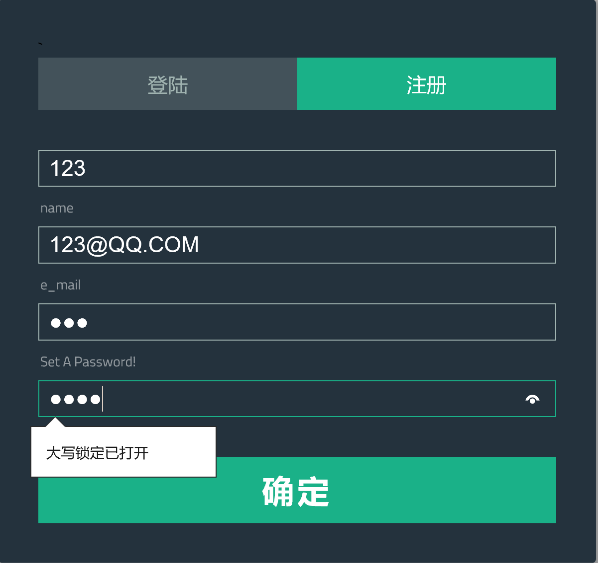
* 如果登录未注册的账号，会跳转到注册界面，提示用户注册。



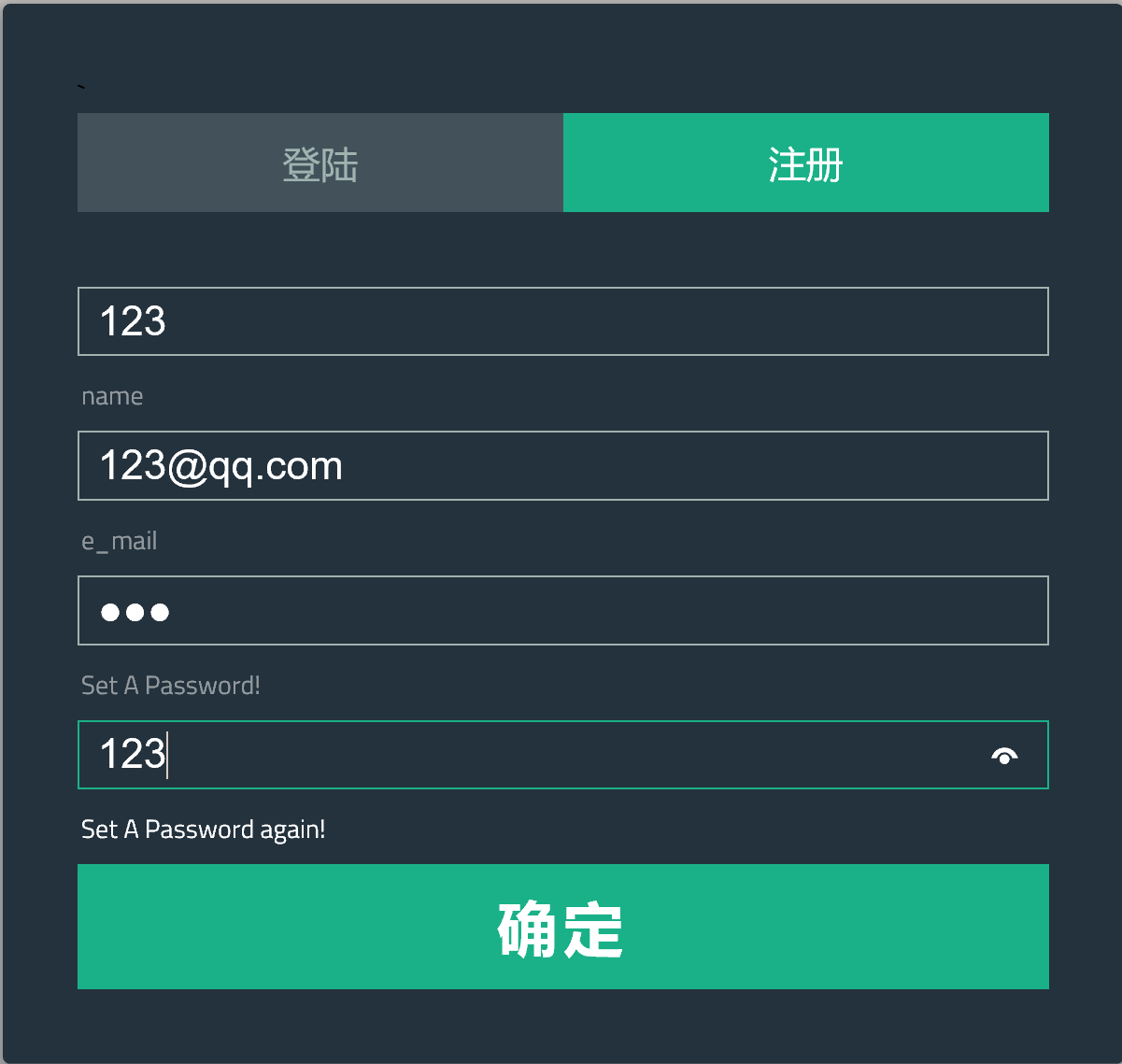
* 注册时会有java script数据验证



* 密码字段



可查看你写入的密码



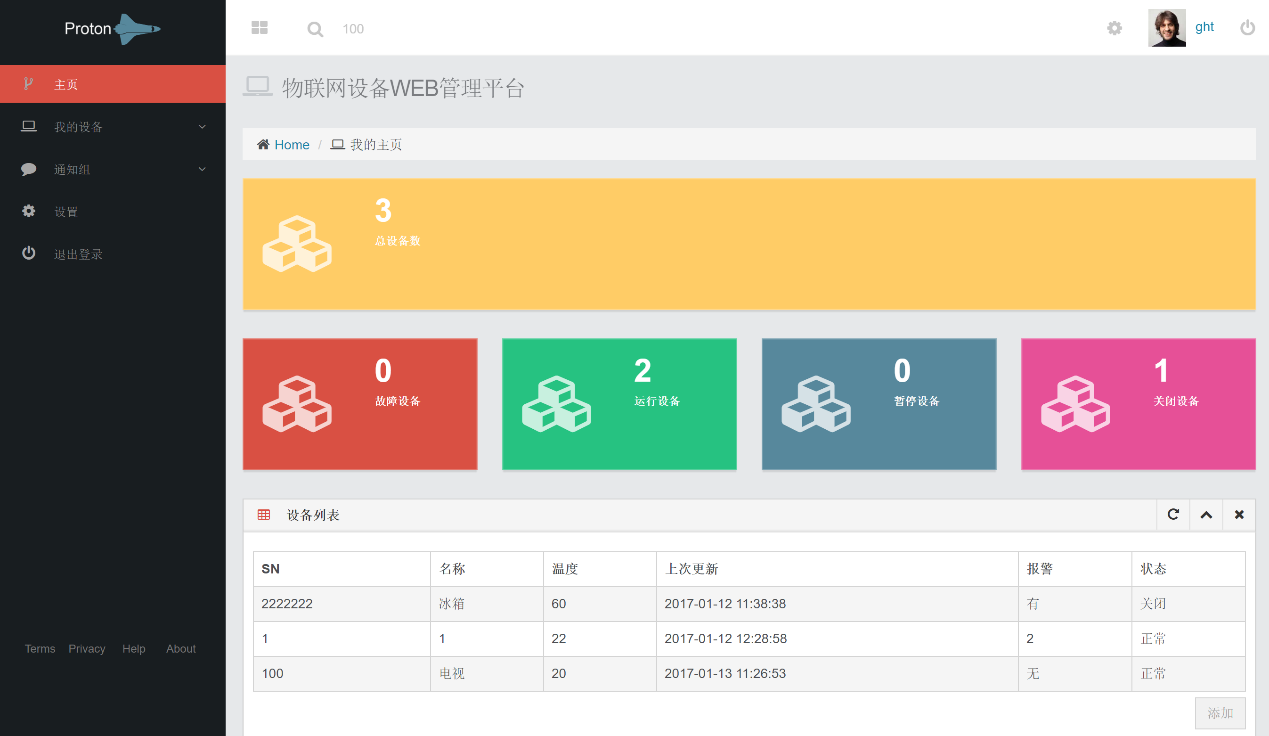
* 通过登录进入主页，主页可以显示：

总设备数，故障设备数，运行设备数，暂停设备数和关闭设备数，

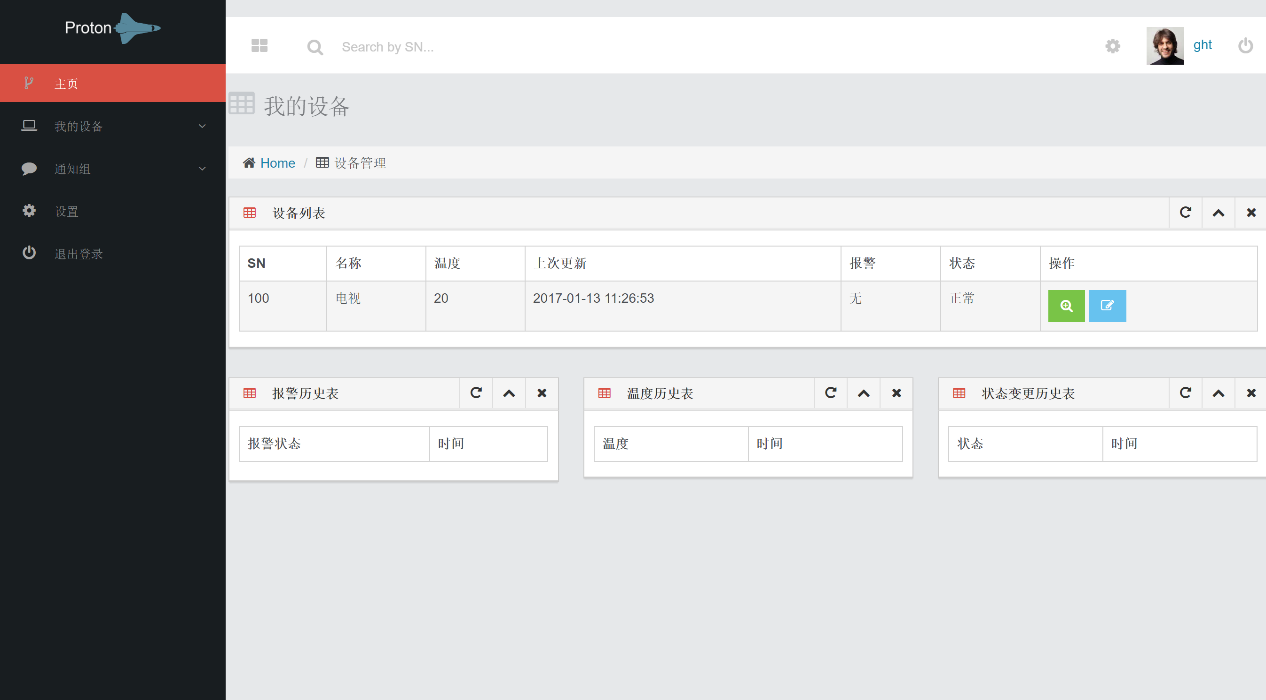
设备列表。设备列表由编号，名称，温度，上次更新时间，报警和状态组成。

这些全部是ajax刷新，当设备改变时会自动刷新。

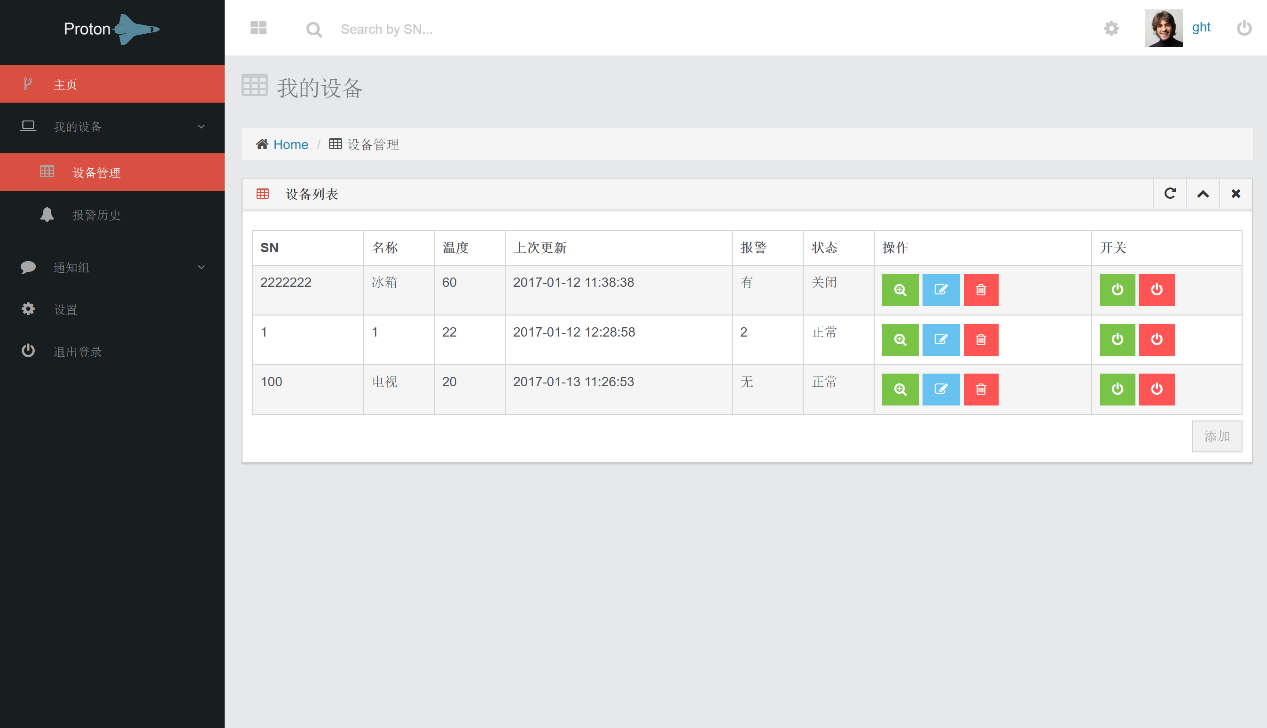
* 在search表单填写 SN 号可以进行查询设备

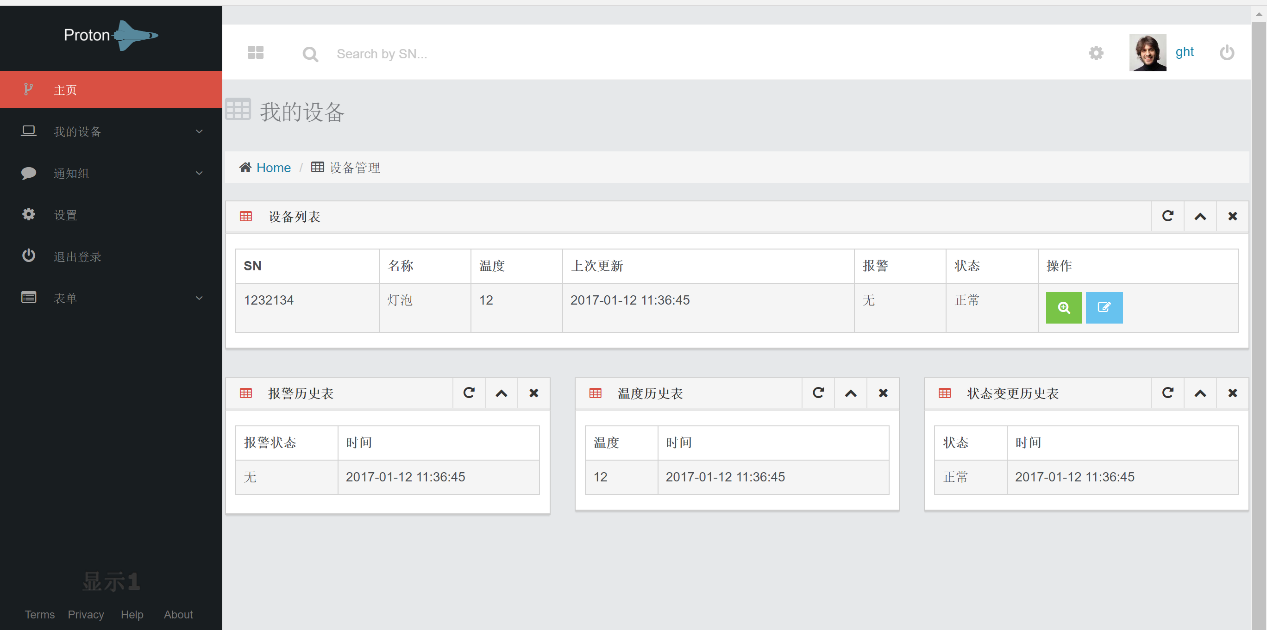


进入设备100 的详细界面



* 通过点击设备管理可以进入设备管理页面，在设备管理页可以：
* 查看设备详细信息
* 修改设备信息
* 删除设备
* 开关设备

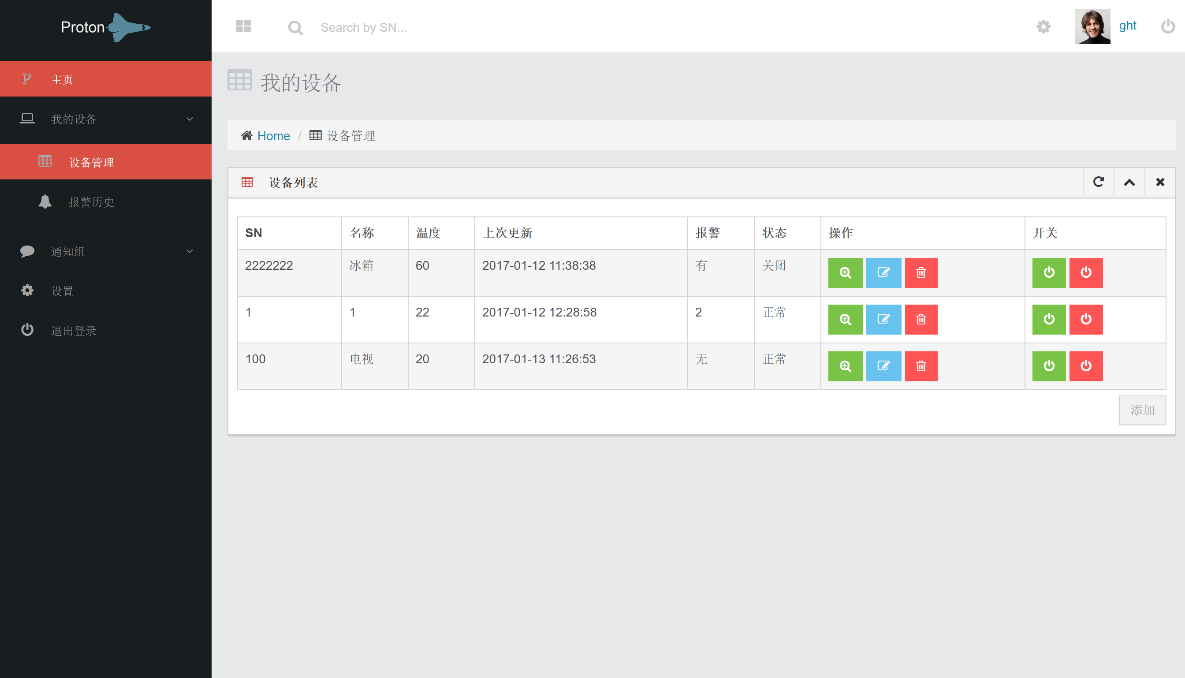


* 设备添加
* 设备详细查看，
* 设备修改：修改时 SN号不可以变动，禁止了用户改动，有javascript输入校验



* 设备删除：

删除前：



删除后：

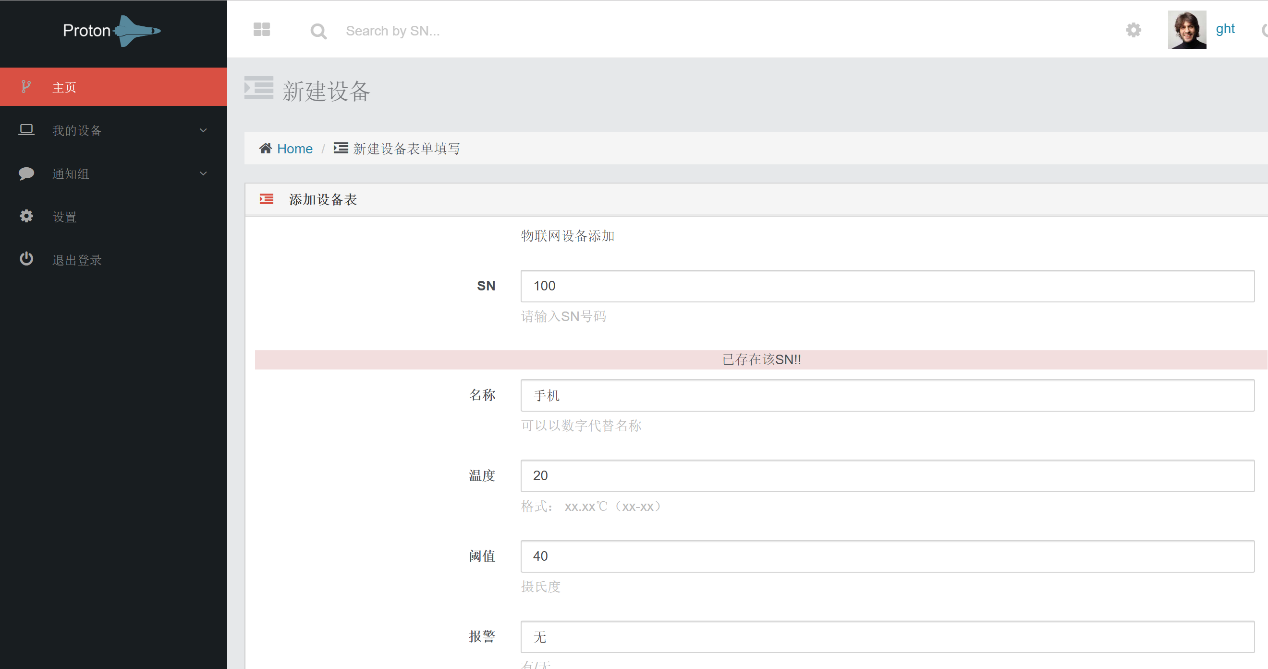


* 设备开关：

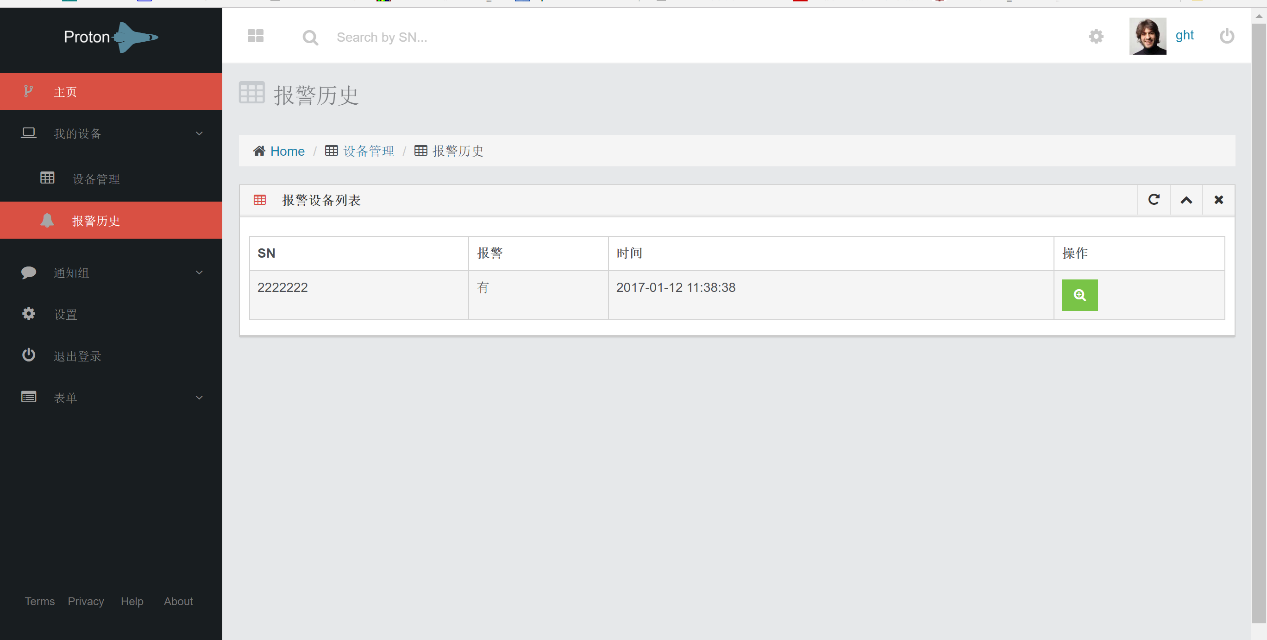




* 设备添加：设备SN是唯一的，当SN重复时，会有报警，有java script数据校验

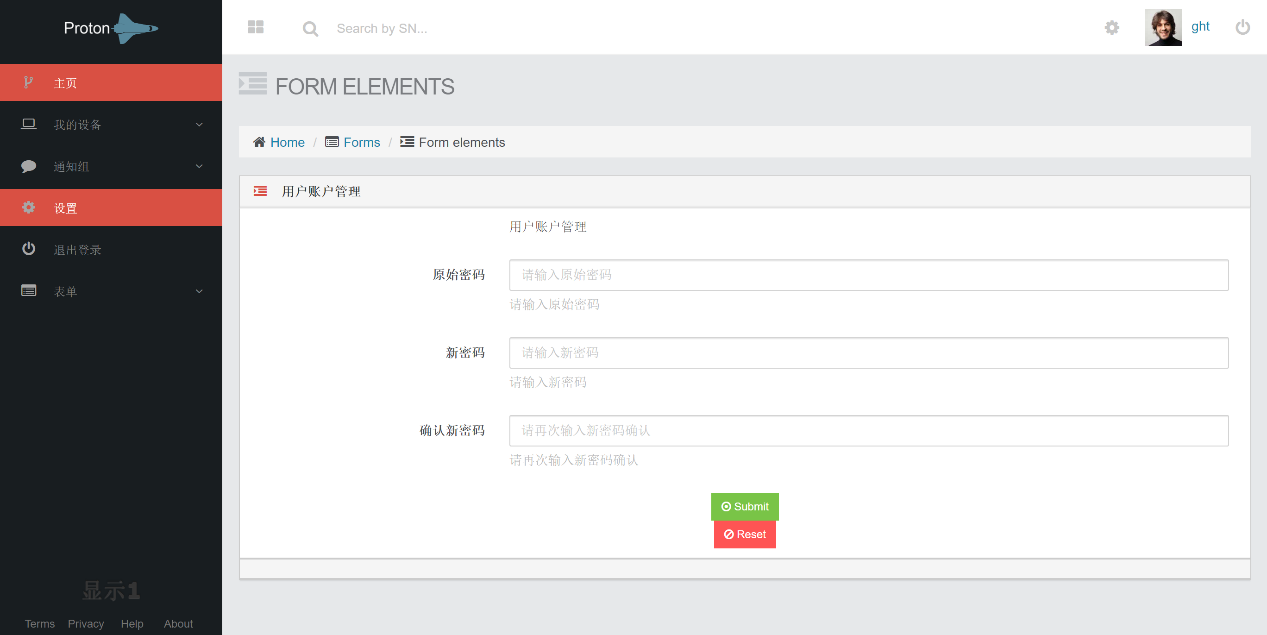


* 通过在主页点击报警历史可以进入报警历史页面，在本页可以查看报警的历史消息。



* 在主页点击设置可以进入账户设置页面，在本页可以更改账户密码。





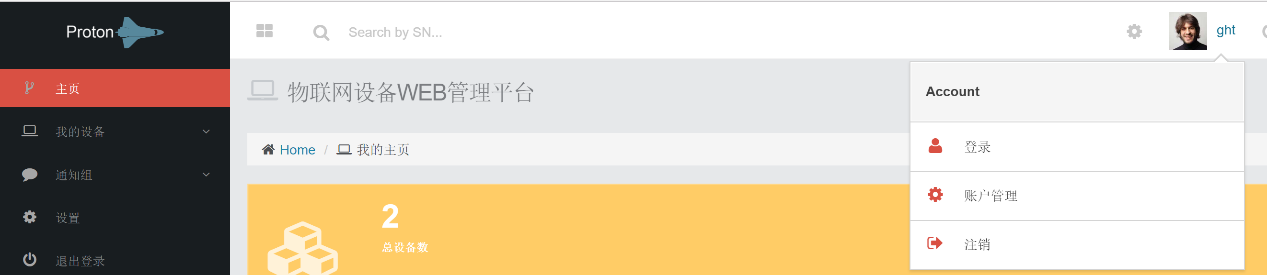
两次新密码输入不同时，会有提示：

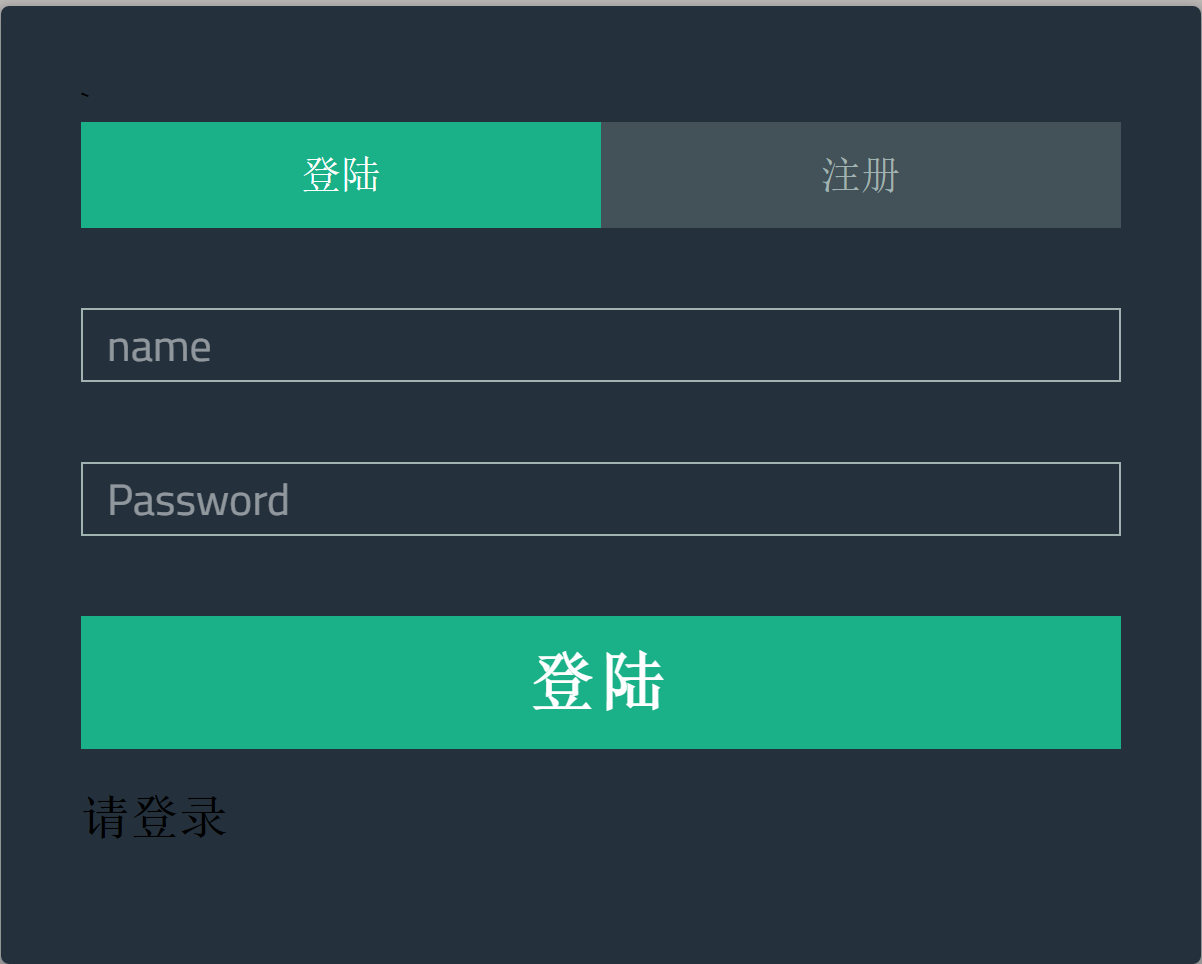


会检查原始密码输入是否正确否则不会修改：



* 点击退出登录会回到登入 页面。





通过Ajax实现页面数据的动态更新。

2. 设备模拟

* 前提说明：

本次大作业中，我小组实现了对网页的动态刷新，即对于用打开的网页维持固定时间的数据查询更新。为了更加直观地并且真实地模拟物联网内设备运行的基本情况，我小组采用pyhon脚本进行设备的模拟，脚本在启动后，定时使用urllib库函数访问服务器视图函数接口，获取当前活动设备的基本情况，在模拟端进行设备运行的模拟，通过json的形式返回给服务器端进行处理。

* 实现过程

1. 脚本说明



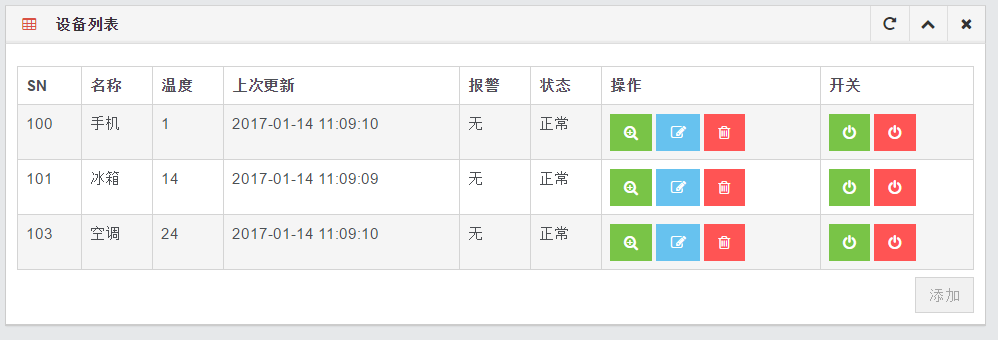
此脚本实现设备的实时模拟，脚本置入到Django服务器中，在服务器后，使用windows 命令行启动，即可完成设备远端模拟。

1. 模拟结果

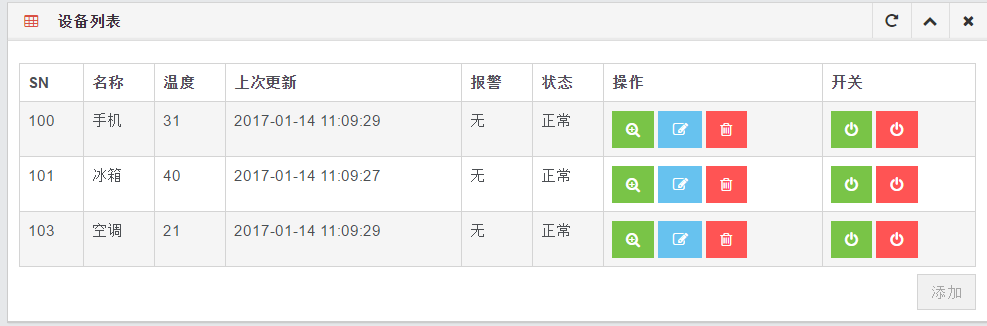
在python脚本启动后，脚本会通过访问特定url来启动对应的视图函数，以此来获取当前处于正常工作状态的设备，并将他们的设备列表信息返回给脚本端，脚本端会更新在脚本的设备列表，依据设备的limit项进行设备的温度模拟，通过json的形式返回给服务器端，运行过程如下图：



下图是启动设备模拟之前的设备维持状态：



下图是启动设备模拟后，设备的温度状态会进行动态模拟，前端会使用AJAX进行动态更新，更新结果如下图：



3.日志维护

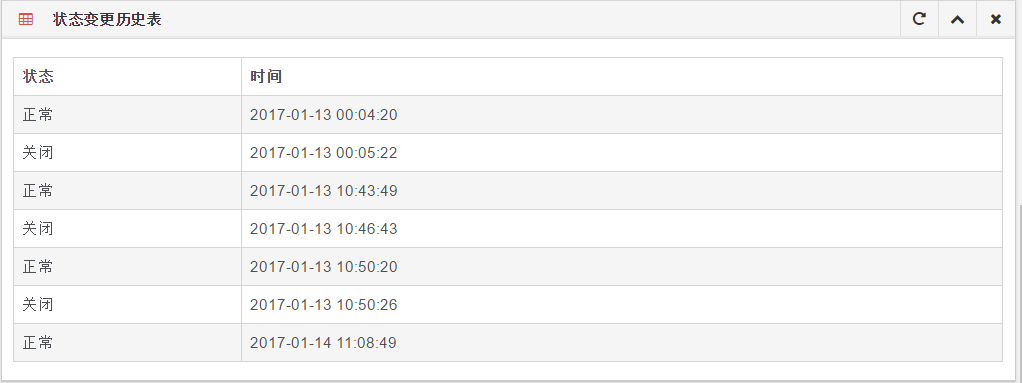
在设备模拟的同时，在服务器数据库维护着设备的状态项，用于进行建立设备状态日志。在前端，设置专门的页面查看设备的日志的维护情况。状态日志分为三个项，分别如下：

1. 设备状态日志，用于记录设备状态变更的情况；

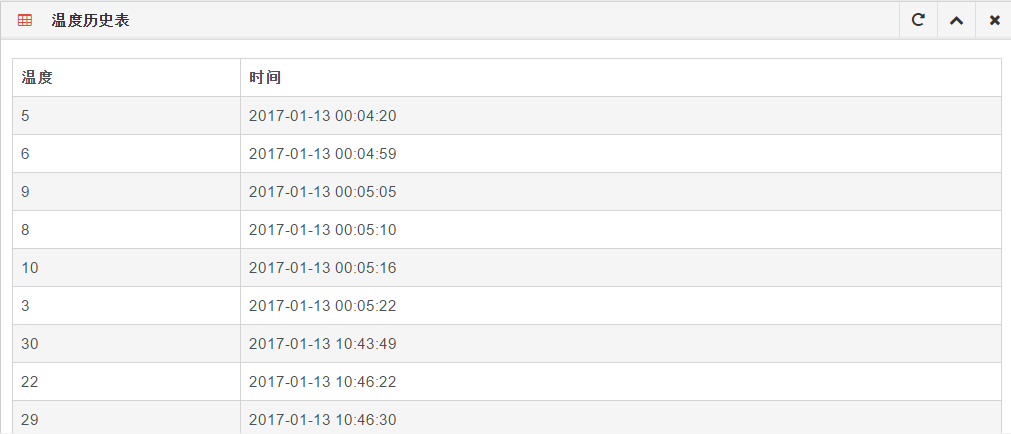
2. 设备温度状态，用于记录当前设备的温度变化情况；

3. 设备告警日志，用于记录出现告警情况的设备的基本情况；

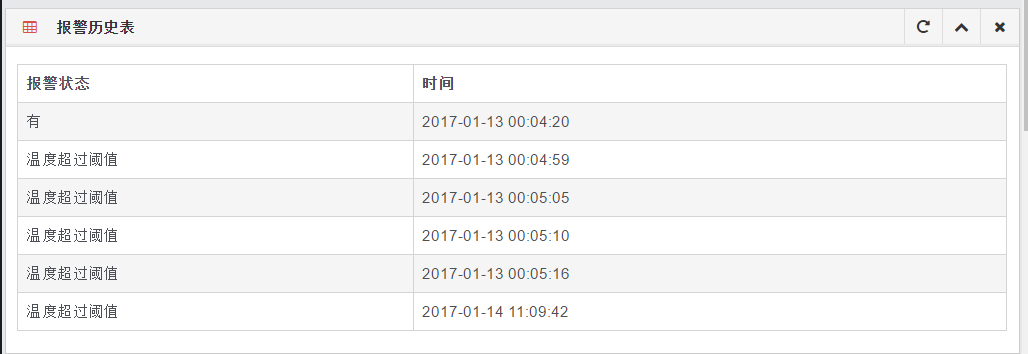
* 设备状态日志



* 设备温度日志



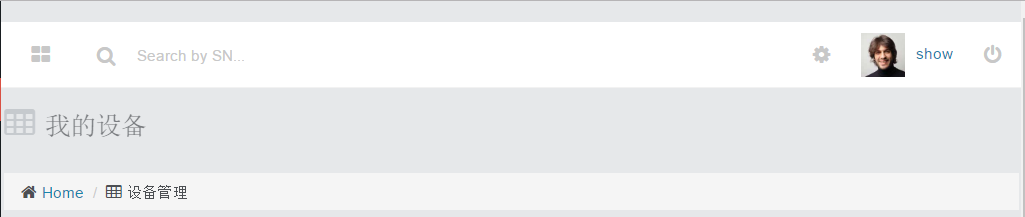
* 设备报警日志



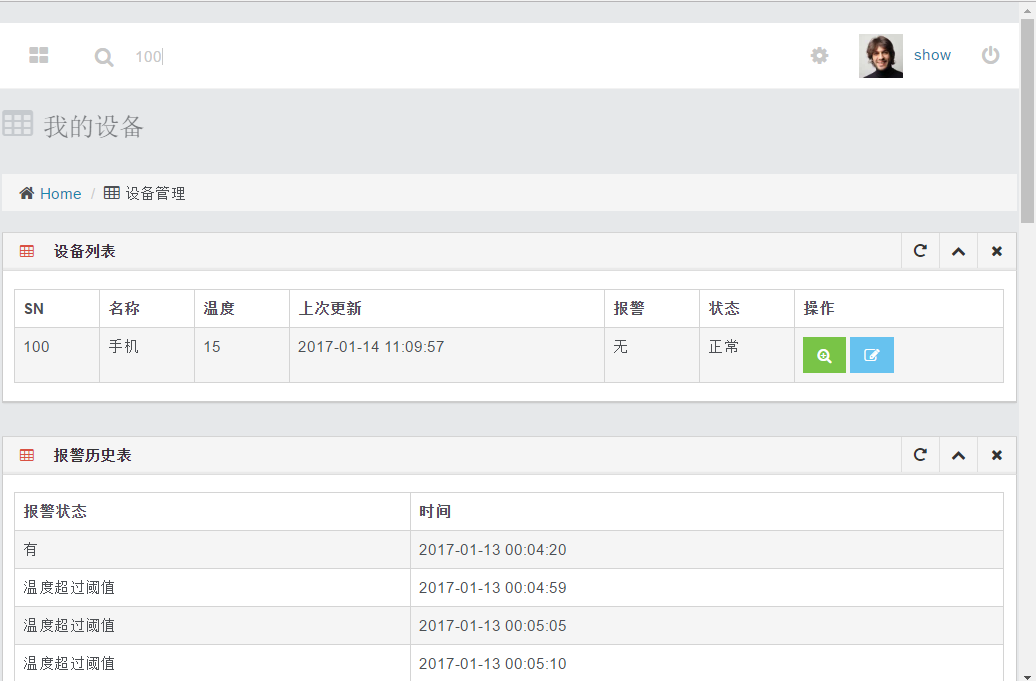
4.智能化查询

作为智能化的物联网管理系统，如果出现多项设备的话，进行设备的查询会是一件非常麻烦的事情，所以我们在网站中置入了搜索模块，用于进行设备智能化的搜索，用户只需要输入需要查询设备的SN值，就可以查询到设备的基本情况：

* 搜索模块位置



* 搜索结果

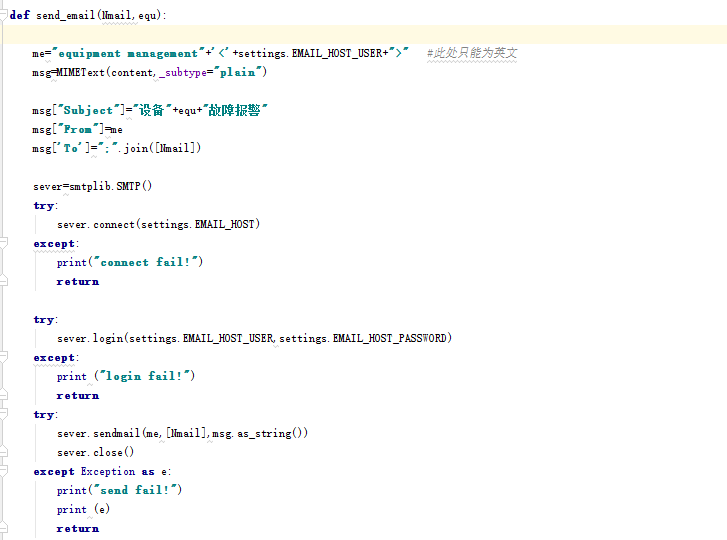


在搜索结果页面上，我们可以查看设备的基本情况，以及设备对应维护的状态日志。

5.报警提醒

在实际的设备管理中，如果设备出现报警的情况，管理系统理应向用户发出报警信息，并对设备进行紧急处理，这样就可以极大化地保证安全性，在我小组设计中，我们充分考虑到了这种情况，在服务器中内置邮件模块，用户在注册的时候会输入联系邮箱。当设备出现温度超过阈值的情况，服务器端会紧急终止设备的运行，并且向用户邮箱了发送邮箱，提醒用户某设备出现运行错误，需要进行紧急处理。

代码实现如下：



在本次实验中，使用北邮注册的官方邮箱（因为这玩意没有垃圾邮件检测），置入到服务器中，作为邮件的发送端。

发送结果如下：



在邮件会提醒设备号，以及设备出现的告警原因，方便用户对设备进行查询，并进行及时处理。

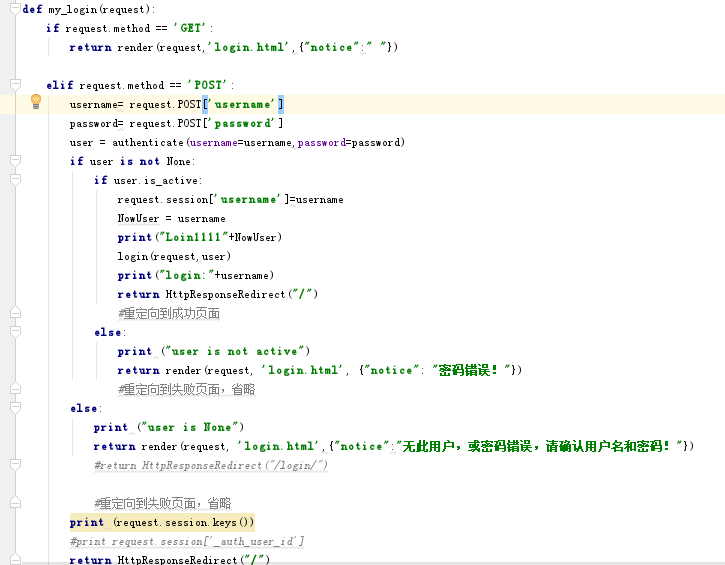
**四、实现**

1. 管理平台

****

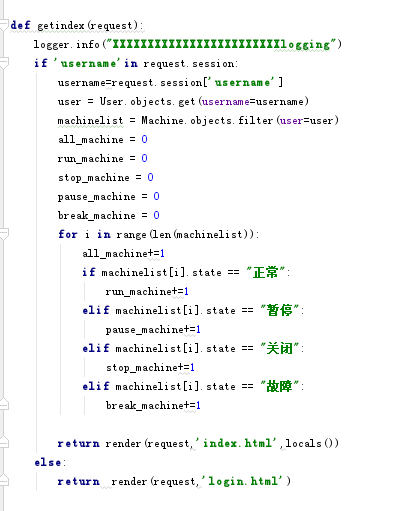
以上为全部的url，本实现抽取了主要功能的实现，Ajax动00态更新的一个页面，查看历史记录的实现，控制模拟设备的开的实现，着重展示和介绍。

登录页面（my\_login）：



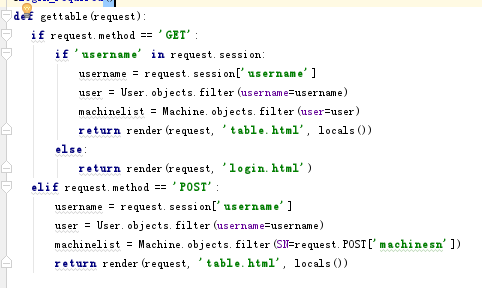
如果浏览器发送GET那么服务器就把登录页面发送给浏览器。如果浏览器发送POST那么就认为用户在进行登入操作，所以检验用户用户名和密码是否正确，如果错误给出提示并重新登入，如果成功则跳转进管理平台。

主页面（getindex）：



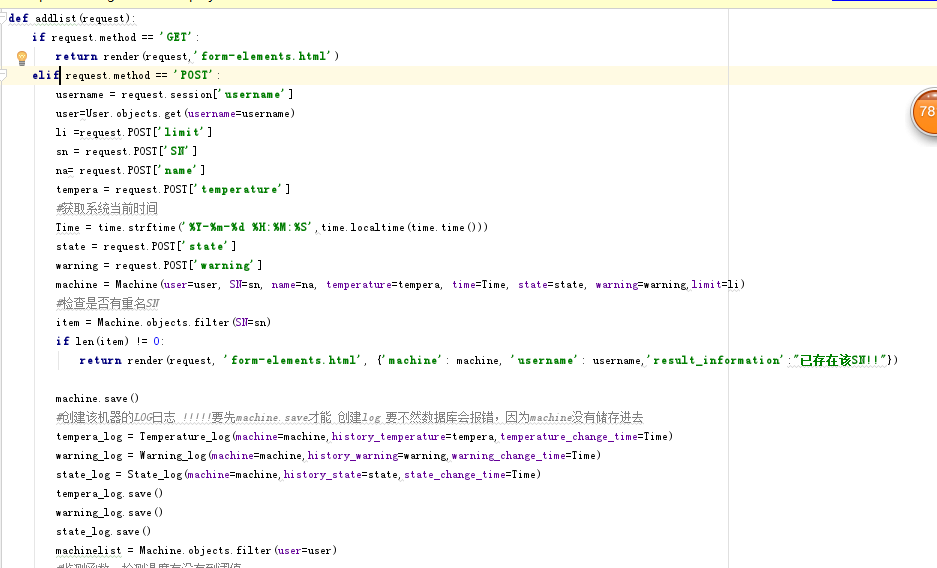
登入后读取用户名并根据用户名查询数据库，依次更新用户的名下设备的状态数量。然后将网页返回给浏览器。

设备管理页面（gettable）：



如果服务器收到GET那么就从session域取出username作为key取得machinelist并返回给浏览器，在此加入了对session域的检验，如果为空则返回登录页面。如果服务器收到POST，那么就认定这是一个要按sn查询的请求，那么就返回满足要求的机器信息。

添加设备（addlist）：

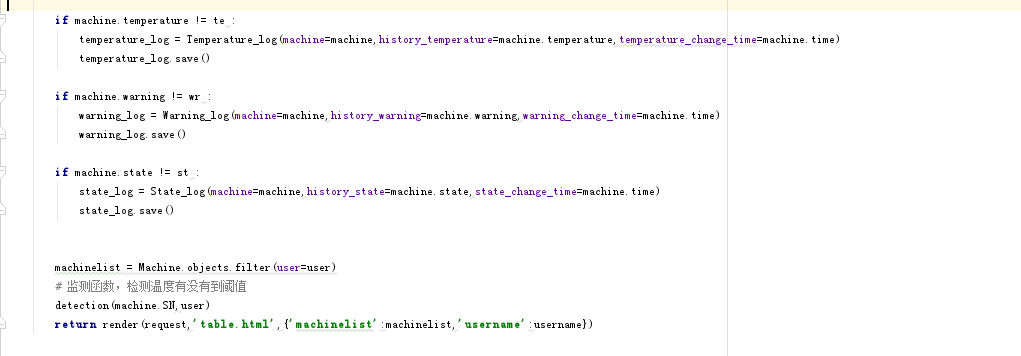




如果收到GET就返回相应的页面。如果收到POST请求，就认定为这是一个上交表单的请求，那么按相应的信息内容依次提取并保存在数据库中，在保存在数据库中的过程中顺便进行初步的检错。

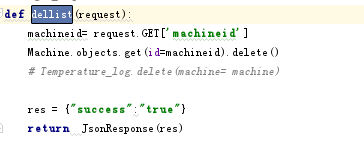
更新设备（updatelist）：





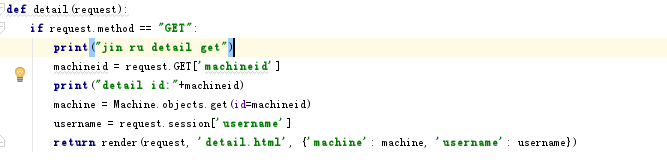
当收到GET按收到的信息查找相应的机器并返回给浏览器。如果收到POST那么这是一个提交的更新设备信息的请求，依次按提交内容更新数据库的原信息，最后检测阈值。

删除设备（dellist）：



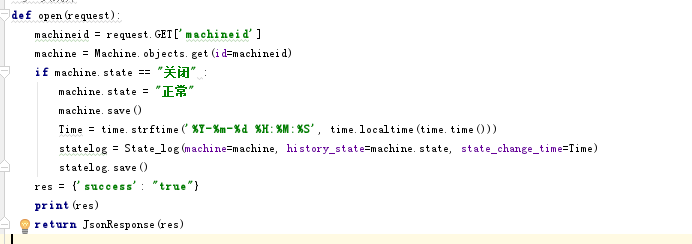
按收到的机器id删除设备，然后返回删除成功的信息。

设备详细信息（detail）：



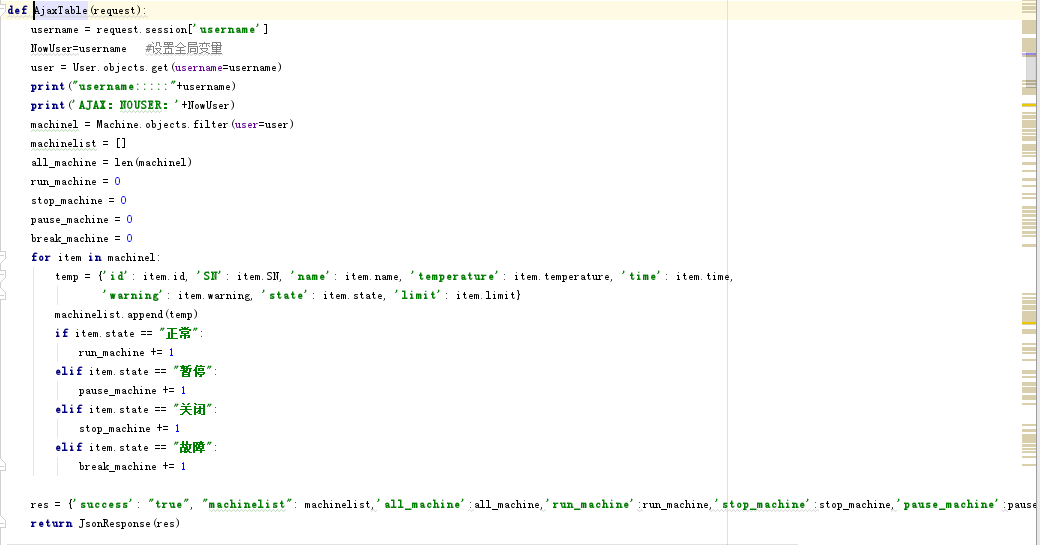
在收到相应的请求后从数据库查询相应的数据返回给浏览器

开设备（open）：



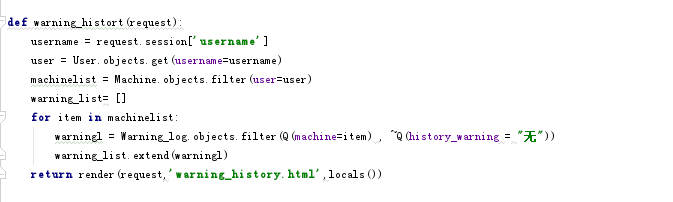
当收到开设备命令后，按machineid查询相应设备，调整设备状态并保存。然后更新时间，写入日志，保存日志。

Ajax动态更新主页（Ajaxtable）：



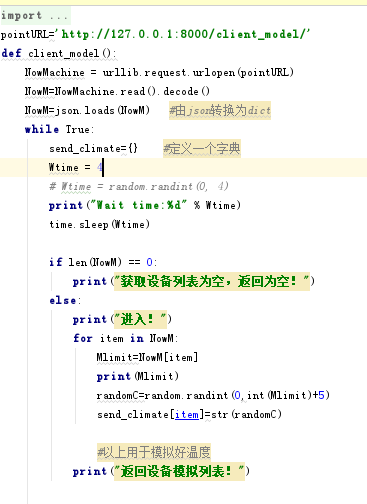
Ajax动态更新由前段没4秒请求一次。服务器按用户名查找到所有机器的信息列表并和计算出的每种机器的数量传回浏览器。

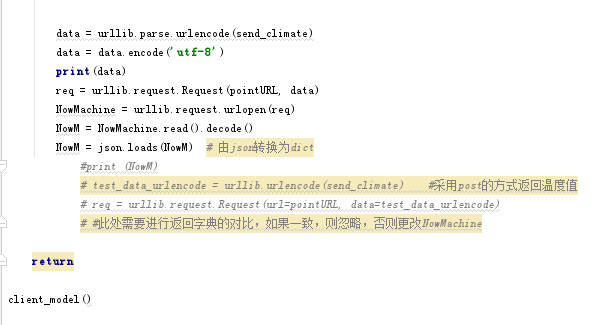
报警历史（warning\_histort）：



按用户名查询相应的warning信息，然后按字典形式存入warninglist的表单，回传给浏览器。

1. 实物模拟：





先从网站读取设备的信息列表，然后依次以从零到比当前值高五度的随机值更新设备的温度，然后上传给网站。

**五、测试**

1. 检测所有按钮，看链接是否完整。

经过测试，在正确的操作下所有的按钮都可以跳转到正确的网页。但是如果进行非规范输入会出现跳转到空网页，网站发生错误的情况，具体错误及解决记录在问题分析中。

1. 对所有输入框进行写入，看网站是否可正常运行。

经过测试，在正确的操作下所有的写入都可以正确的被服务器接受和被浏览器显示。对于全部关键信息的写入，可以规避非规范的内容。对于其他描述信息，可以保证网页的工作正常，但会出现显示非规范的信息的情况。

1. 打开网页不进行任何操作，检查ajax是否可以正常更新。

经过测试，ajax可以正常工作。在测试时间范围（大于三个小时）内没有出现网页崩溃的情况。

1. 加入设备模拟程序，并打开网页不进行任何操作，检查设备模拟程序是否正常运行。

经过测试，设备模拟程序再正常环境下可以正常运行，在测试时间范围（大于三个小时）内没有出现程序崩溃的情况。但是如果有因为输入错误而产生的设备条目，则模拟程序无法正常运行，会报错后退出。

**六、问题分析**

1.邮件模块开发经验

在邮件模块开发当中，我们对于163邮箱的垃圾邮件处理机制没有进行充分的理解，在163邮箱中设置垃圾邮件的发送检测，如果发送的邮件有垃圾邮件的嫌疑，163会关闭客户端邮箱的授权。在实际的实验中，我们没有充分考虑到这个问题，因此出现了邮箱莫名被屏蔽的情况，耗费了的开发时间。当我们使用北邮邮箱后，问题解决。

2.设备模拟开发经验

在设计设备模拟模块，我们首先使用在Django置入多线程进行模拟，当服务器启动后，设备会自动启动模拟。但在实际操作，置入多线程造成了不必要的麻烦，尤其是逻辑错误，造成了网站出现跳转错误。因此，我们使用了python脚本进行视图函数的定期访问，以此模拟设备的运行状态。

3.日志模块开发经验

在设备进行操作，我们通过数据库进行设备状态的记录，但在数据库格式设计的时候，我们对设备状态的考虑不足，造成了状态不能有效地记录设备状态的变化情况，出现了不必要的麻烦，造成设备的状态记录不明。在调整了数据库设计格式后，错误解决。