郑州轻工业大学

课 程 报 告

**课程名称： 《物联网企业项目实践》**

课题题目： 智能楼宇安全监视系统设计

姓 名： 刘广城

院 （系）： 计算机与通信工程学院

专业班级： 计算机科学与技术1801班

学 号： 541807010119

指导教师： 张涛、陈燕

时间： 2021-2022(1)

目录

[1 需求说明 1](#_Toc89511912)

[1.1 项目背景 1](#_Toc89511913)

[1.2 需求分析 1](#_Toc89511914)

[2 开发环境 2](#_Toc89511915)

[2.1 C#语言 2](#_Toc89511916)

[2.2 OneNET云平台 2](#_Toc89511917)

[2.3 MySQL数据库 2](#_Toc89511918)

[3 系统总体设计 3](#_Toc89511919)

[3.1 系统功能设计 3](#_Toc89511920)

[3.2 系统总体实现 3](#_Toc89511921)

[3.3 系统示意图 4](#_Toc89511922)

[4 数据库设计 6](#_Toc89511923)

[4.1 基础表 7](#_Toc89511924)

[4.2 关系图 9](#_Toc89511925)

[5 客户端设计 10](#_Toc89511926)

[5.1 客户端总体设计 10](#_Toc89511927)

[5.1.1 监控流程图 10](#_Toc89511928)

[5.1.2 程序流程图 11](#_Toc89511929)

[5.2 功能设计 12](#_Toc89511930)

[5.2.1 登录功能 12](#_Toc89511931)

[5.2.2 主菜单功能 18](#_Toc89511932)

[5.2.3 设备信息查询功能 22](#_Toc89511933)

[5.2.4 其他功能 27](#_Toc89511934)

[5.3 核心功能实现 27](#_Toc89511935)

[5.3.1 连接OneNET云平台 27](#_Toc89511936)

[5.3.2 连接MySQL数据库 29](#_Toc89511937)

[5.3.3 内网穿透 31](#_Toc89511938)

[5.4 系统功能测试 32](#_Toc89511939)

[5.4.1 登录功能 32](#_Toc89511940)

[5.4.2 注册与注销账号功能 33](#_Toc89511941)

[5.4.3 主菜单功能 35](#_Toc89511942)

[5.4.4 添加设备功能 36](#_Toc89511943)

[5.4.5 设备信息查询功能 38](#_Toc89511944)

[5.4.6 设备历史记录查询功能 39](#_Toc89511945)

[结束语 42](#_Toc89511946)

[参考文献 43](#_Toc89511947)

[附 录 44](#_Toc89511948)

# 需求说明

## 项目背景

楼宇建筑常发生漏水的的部位主要有屋面漏水、卫生间漏水、消防管道破损漏水等，一旦发生漏水，将影响人们正常的生活或工作，造成不必要的损失，严重时可能还会发生触电等安全事故。遇上连续的大雨天气，排水不通畅不及时，漏水现象会更加严重。电梯一旦漏水，可能导致电梯发生故障，损坏电梯，严重时可能发生触电等安全事故。

漏水检测系统能够监控智能楼宇内如中央空调系统、消防水管、生活用水水管等有水源区域的漏水情况，一旦发生漏水，能及时准确地检测到并通过监控系统发出报警通知工作人员，同时可联动排水设备，控制阀门关闭；如果漏水隐患分布范围较广，还需进行定位，监测漏水的具体位置，以便迅速找到泄漏点。

## 需求分析

市场上常见的智能楼宇安全监视系统一般包括数据系统、报警系统、对讲系统和IBMS集成管理系统。

数据系统实现智能楼宇自动化控制的首要条件是要充分掌握建筑的总体情况,从建筑楼宇区域范围内进行动态控制,这样做的目的在于发挥建筑楼宇自动化控制功能。数字系统是智能楼宇监控的重要因素,其主要宗旨是通过数字信号之间的数据转换,形成了不同的层面, 并通过图像处理后转变成为数字信号,以信号传输的方式达到监控的目的。

报警系统是通过对建筑楼宇工作范围内的异常情况,通过信号方式传输和反馈到建筑监控中心,并提醒安全保卫人员对异常情况采取有效的处理措施,以恢复建筑楼宇的正常工作。报警系统主要由监控中心控制,如发现异常状况,在第一时间作出报警提示,这样有利于技术人员及时做出回应。

对讲系统实际上是沟通交流系统,主要是方便楼宇安保管理人员之间的沟通与交流, 该系统主要是采用通信系统对信号进行传输操作,其操作程序较为简单、方便。例如,监控人员通过监控中心计算机平台监控建筑楼宇安全和故障隐患,如发现出现类似的隐患，可采用对讲系统及时汇报给安全管理人员。

IBMS集成管理系统利用建筑的信息网络,采用一体化集成的手段,把构成整个建筑的弱电子系统单独分离的设备、功能和信息集成为一个统一、协调的综合网络系统,并通过该系统把这些分散、复杂而庞大的各类设备和系统进行充分的信息、资源、任务共享,从而方便地在统一的界面上实现对各子系统全局的监视和管理,这样不仅能有效控制和降低管理营运成本,而且有效地控制智能楼宇的突发事件,并能将灾害损失降到最低。

# 开发环境

## C#语言

C#是一种最新的、面向对象的编程语言。它使得程序员可以快速地编写各种基于Microsoft .NET平台的应用程序，Microsoft .NET提供了一系列的工具和服务来最大程度地开发利用计算与通讯领域。

正是由于C#面向对象的卓越设计，使它成为构建各类组件的理想之选——无论是高级的商业对象还是系统级的应用程序。使用简单的C#语言结构，这些组件可以方便的转化为XML 网络服务，从而使它们可以由任何语言在任何操作系统上通过Internet进行调用。

最重要的是，C#使得C++程序员可以高效的开发程序，而绝不损失C/C++原有的强大的功能。因为这种继承关系，C#与C/C++具有极大的相似性，熟悉类似语言的开发者可以很快地转向C#。

## OneNET云平台

OneNET是由中国移动打造的PaaS物联网开放平台。平台能够帮助开发者轻松实现设备接入与设备连接，快速完成产品开发部署，为智能硬件、智能家居产品提供完善的物联网解决方案。

OneNET致力于开发者的体验，逐步提升云服务体量，着手用户运营，深化运维管理和云端大数据分析，协同产业上下游，长期发展以“大连接、云平台、轻应用、大数据”为架构的平台级服务，打造用户导向的物联网生态环境。作为“云管端”核心布局的OneNET秉承中国移动的发展理念。

## MySQL数据库

MySQL是一个关系型数据库管理系统，由瑞典MySQL AB 公司开发，属于 Oracle 旗下产品。MySQL 是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL是最好的 RDBMS (Relational Database Management System，关系数据库管理系统) 应用软件之一。

MySQL是一种关系型数据库管理系统，关系数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。

MySQL所使用的 SQL 语言是用于访问数据库的最常用标准化语言。MySQL 软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库。

# 系统总体设计

## 系统功能设计

系统包含六部分功能，分别为实时监控、及时报警、紧急避险、历史记录、移动端、客户端。详细功能描述如下：

1. 实时监控：监测楼宇安全状态，包括火灾监测，可燃气体泄漏检测，漏水检测；
2. 及时报警：能够对异常情况做出判断，并自动或手动（通过设备按键或者手机远程操控）发出报警；
3. 紧急避险：对险情能够做出及时处理：可燃气体泄漏时，启动通风系统，火灾时发出警报与淋喷，漏水时进行排水作业；
4. 历史记录：设置数据库对进一个月的监测数据进行存储，能够调出历史曲线。
5. 移动端APP：设计手机APP软件，用户能够通过软件了解楼宇实时安全状态，并通过手动操作发出警报或者设备操控。
6. 客户端：用户通过客户端接收现场发出的警报，软件界面美观简洁，操作方便。

## 系统总体实现

由STM32模块控制的硬件部分，各传感器分别监测各个监测点的数据，并将数据进行显示并判断是否需要发出警报，再通过WIFI模块传到云平台，同时，移动端APP和客户端软件可从云平台上传、下载数据，当硬件在线时，还可通过云平台下达针对硬件的各种指令。



图 3‑1系统总体设计图

硬件组成包括有：以太网适配器、zigbee控制板、人体红外模块、DHT11湿度模块、火焰传感器模块、常闭型震动传感器模块、一氧化碳MQ-7模块、液化气天然气MQ-5模块、红外测温传感器模块GY-906 MLX90614ESF、MQ-135模块空气质量有害气体检测传感器报警模块、ESP8266串口WIFI 无线模块、GP2Y1010AU0F灰尘传感器检测灰尘PM2.5、Lenovo/联想MIIX310-101CR平板电脑win10英寸PC平板二合一笔记本。

## 系统示意图

该系统包括各种传感器、监控摄像头和中央控制器。传感器可监测各种数据，监控摄像头用于判断是否有入侵者以及记录日常情况，中央控制器可集中控制整个系统。



图 3‑2传感器布置图

1. 系统传感器
2. 温湿度传感器：用于监测当前环境下的温度、湿度；
3. 人体红外摄像头：用于判断是否有入侵者；
4. 气体传感器：用于判断是否有有害气体泄露，如天然气、一氧化碳以及各种烟雾等；
5. 玻璃破碎传感器：用于判断玻璃是否破碎；
6. 监控摄像头：用于监控门外访客的身份，以及是否有入侵者试图通过窗户进入室内。
7. 中央控制器功能
8. 能够进行Wifi通信；
9. 与传感器和其他控制器的通信；

能根据传感器的数据进行相应处理，或发出指令要求控制器对设备进行控制。包括但不限于：

1. 与气体传感器的交互（感知危险、鸣笛发出警报，通知用户处理）
2. 与红外传感器的交互（感知危险、鸣笛发出警报，通知用户处理）
3. 与烟感传感器的交互（感知危险、鸣笛发出警报，通知用户处理）
4. 与监控摄像头的交互（感知危险、鸣笛发出警报，通知用户处理）
5. 与门禁系统的交互（感知危险、鸣笛发出警报，通知用户处理）
6. 查看监控视频；
7. 查看系统日志；
8. 进行智能判断，能够根据多个传感器的数据进行智能判断。例如，采用多值逻辑方法，例如3种传感器中有2种判断发生入侵，则报警并进行响应；
9. 与云服务通信，能够在感知到危险的时候，例如，天然气泄漏，将日志文件上传到云端服务器；或感知到入侵后，自动控制监控摄像头，进行拍照、摄像，并将文件上传到云端服务器。

# 数据库设计

首先需要明确，需要设计两个基本表：

一个用户账号表user\_tb，用来来保存用户账号的相关信息。

一个设备信息表device\_tb，用来保存设备的相关信息。

其次，根据功能来分析，还需要另外两个衍生表。

一个是历史记录表record\_tb，用来保留设备历史记录的相关信息，实现查询历史记录功能。

一个是用户设备表user\_device\_tb，用来在用户账号和设备之间建立联系。

主键的确定：

根据账号唯一性，可以确定user\_tb的主键为uid。

根据设备唯一性，可以确定device\_tb的主键为device\_id。

由于同一台设备可能需要查询不同种类传感器的不同时间段的历史数据，因此record\_tb的主键由device\_id和device\_type和at共同构成。也就是说同一台设备的同一传感器在同一时刻的数据是唯一的。

由于一个账号可以拥有多台设备，一台设备也可以被多个账号所拥有，因此user\_device\_tb的主键由uid和device\_id和device\_type共同构成。也就是说同一账号只能拥有同一个设备的一个同一类型的传感器。

## 基础表

1. user\_tb表

用于记录用户账号uid和密码pwd的表。

表 4‑1 用户账号密码表user\_tb

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **是否空值** | **约束** | **备注** |
| uid | Char(16) | 否 | 主键 | 用户账号 |
| pwd | Char(16) | 否 |  | 用户密码 |

表 4‑2 user\_tb数据示例表

|  |  |
| --- | --- |
| **用户账号(uid)** | 用户密码(pwd) |
| 541807010119 | lgc2621690255 |

数据来源：用户自定义

1. device\_tb表

用于记录设备的设备ID、及设备url和设备api\_key的表。

表 4‑3设备详情表device\_tb

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **是否空值** | **约束** | **备注** |
| device\_id | Char(16) | 否 | 主键 | 设备号 |
| device\_name | Char(32) | 否 |  | 设备名称 |
| device\_type | Char(16) | 否 |  | 设备种类 |
| url | Char(64) | 否 |  | API地址 |
| api\_key | Char(32) | 否 |  | APIKey |

表 4‑4 device\_tb示例数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **设备号(device\_id)** | 设备名称(device\_name) | 设备种类(device\_type) | API地址(url) | APIKey(api\_key) |
| 804632405 | 默认传感器 | 温度 | http://api.heclouds.com/devices/804632405 | U7UU2Hp4ns2V2Ug55BgrHGinQYA= |

数据来源：OneNET平台

1. record\_tb表

用于记录设备的历史记录的表，形式为设备种类device\_type + 监测值value + 监测时间at。

表 4‑5历史数据表record\_tb

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **是否空值** | **约束** | **备注** |
| device\_id | Char(16) | 否 | 主键，device\_tb外键 | 设备号 |
| device\_type | Char(16) | 否 | 主键 | 设备种类 |
| value | Char(16) | 否 |  | 监测值 |
| at | Datetime | 否 | 主键 | 监测时间 |

表 4‑6 record\_tb示例数据表

|  |  |  |  |
| --- | --- | --- | --- |
| **设备号(device\_id)** | **设备种类(device\_type)** | 监测值(value) | **监测时间(at)** |
| 804632405 | 温度 | 28.000000 | 2021-11-06 09:35:51 |

数据来源：OneNET平台

1. user\_device\_tb表

用于记录用户所拥有的设备的表，通过用户账号uid和设备号device\_id来建立用户与设备的联系。

表 4‑7用户设备表user\_device\_tb

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **是否空值** | **约束** | **备注** |
| uid | Char(16) | 否 | 主键，user\_tb外键 | 用户账号 |
| device\_id | Char(16) | 否 | 主键，device\_tb外键 | 设备号 |
| device\_type | Char(16) | 否 | 主键 | 设备种类 |
| device\_name | Char(32) | 否 |  | 设备名称 |

表 4‑8 user\_device\_tb示例数据表

|  |  |  |  |
| --- | --- | --- | --- |
| **用户账号(uid)** | **设备号(device\_id)** | **设备种类(device\_type)** | 设备名称(device\_name) |
| 541807010119 | 804632405 | 温度 | 温度传感器 |

数据来源：用户自定义

## 关系图

user\_tb表和device\_tb表为独立表。不依赖于任何其他表。

record\_tb表依赖于device\_tb表，通过device\_id外键和device\_tb表相连接。

user\_device\_tb表依赖于user\_tb表和device\_tb表，通过uid外键和user\_tb表相连接、通过device\_id外键与device\_tb表相连接。

**user\_tb**

uid

pwd

**device\_tb**

device\_id

device\_name

device\_type

url

api\_key

**user\_device\_tb**

uid

device\_type

device\_name

**record\_tb**

device\_id

value

at

device\_type

device\_id

图 4‑1基础表关系图

# 客户端设计

## 客户端总体设计

### 监控流程图

在用户账号登入成功后，执行此流程图，从云平台获取数据，判断获取的数据是否超出预警值，如超出预警值，向设备发送报警指令，启动避险装置，并将数据写入数据库，结束此流程；如未超出预警值，则直接将数据写入数据库，并结束流程。

开始

从云平台获取数据

是否超过预警值

报警

数据写入数据库

启动避险装置

结束

是

否

图 5‑1监控流程图

### 程序流程图

在软件程序运行时执行此流程图，先进行监控流程的预处理(详见监控流程图)，然后进入程序主菜单，通过主菜单，可以选择诸如“火灾监控”、“可燃气体”、“漏水检测”等传感器功能的使用，传感器功能均具备“发出警报”、“查询历史记录”等通用功能，共享一套模板，查询历史记录的过程将与数据库进行互动。在未选择退出系统之前，程序会一直停留在主菜单界面并等待用户进行相关操作，如选择退出系统，则流程结束。

开始

监控流程

主菜单

数据库

结束

退出系统

可燃气体

发出警报

控制设备

历史记录

漏水检测

发出警报

控制设备

历史记录

火灾监控

发出警报

控制设备

历史记录

是

否

图 5‑2程序流程图

## 功能设计

### 登录功能

在点击用户登录按钮后执行以下流程图，用户输入账号和密码，系统判断账号、密码是否为空，如账号或密码为空时，则提示用户输入账号或密码。当二者均不为空时，判断账号是否存在，如账号不存在，则提示用户进行注册账号功能；如账号存在，则检验用户账号与密码是否匹配，如账号密码匹配，则进入主菜单功能，此登录流程结束。如账号密码不匹配，则提示用户重新输入密码。当用户不再尝试登录操作时，此登录流程结束。



图 5‑3用户登录流程图

#### 游客模式

游客模式的功能与正式用户的功能相同，主要区别为游客模式下无需访问数据库，游客的用户数据将在退出本次登录之后重置。在系统下方的提示条将会显示当前的用户状态。

#### 账号注册与注销

以下流程图为注册/注销功能输入预处理流程图，当注册账号或注销账号按钮被按下之后，首先执行以下输入预处理流程图，判断输入的账号、密码以及二次密码是否为空，如三者皆不为空，则进入下一流程，否则一直循环等待三者输入或整体流程结束。



图 5‑4输入预处理流程图

下图为注册功能流程图，当用户按下注册账号按钮时，执行本流程图。先进行账号密码的输入预处理(详见输入预处理流程图)，当该流程合法后，判断注册账号是否存在，如已存在，则提示账号已被注册，重新进入注册账号状态；如注册账号不存在，则判断两次输入密码是否相同，如相同，则成功注册账号，此注册账号流程结束。如两次密码输入不相同，则重新进入账号注册流程，直到执行退出系统操作。



图 5‑5注册账号流程图

下图为注销功能流程图，当用户按下注销账号按钮时，执行本流程图。先进行账号密码的输入预处理(详见输入预处理流程图)，当该流程合法后，判断注销账号是否存在，如不存在，则提示注销账号不存在，重新进入注销账号状态；如注销账号存在，则判断两次输入密码是否相同，如相同，则成功注销账号，此注销账号流程结束。如两次密码输入不相同，则重新进入账号注销流程，直到执行退出系统操作。



图 5‑6注销账号流程图

#### 测试页

测试页主要用于开发人员进行程序调试，通过测试页，开发人员可以直观的看到从oneNET平台传回来的数据以及经过程序处理之后的数据。

/datastreams功能：点击按钮，自动在url后面添加一个/dastreams后缀。

ping功能：通过输入形如”http://api.heclouds.com/devices/804632405/datastreams”的url和与之配套的apikey可以查询到对应设备数据流，respond为oneNET平台返回的数据，solved为系统抓取到的第一个数据结点。

get功能：通过输入形如”http://api.heclouds.com/devices/804632405”的url和与之配套的apikey可以查询到对应设备历史记录的url参数格式，respond为oneNET平台返回的数据，solved为系统处理返回的查询对应设备历史的url。

post功能：通过输入形如”http://api.heclouds.com/devices/804632405”的url和与之配套的apikey可以向对应设备发送”LEDON”或”LEDOFF”的命令，此功能无respond内容，solve为oneNET平台返回的提示信息。

#### 连接数据库

通过该功能，输入软件开发人员所提供的数据库端口号即可进行远程访问数据库服务，使用用户模式进行程序操作，不输入亦可以通过游客模式体验软件的全部功能。

### 主菜单功能

下图为主菜单功能流程图，当以游客模式/用户模式进入主菜单时执行该流程图。首先，系统进行用户模式的判断，如为游客模式，则初始化时，以系统原始数据进行系统初始化；如为用户账号登入，则以数据库数据进行系统初始化，初始化数据之后即进入主菜单功能，此功能下，可进行查看设备、添加设备、删除设备三项子功能，功能流程图见相应流程图。当选择退出登录功能时，结束本次流程。



图 5‑7主菜单功能流程图

#### 查看设备

下图为查看设备流程图，在点击“查看设备”按钮后执行。首先判断是否选中传感器，如未选中，则提示未选中设备，否则，进入查看设备界面，界面有修改设备信息、查询最近一次数据、发出警报、查询历史记录四大功能，点击返回主菜单时或修改了设备信息之后，本流程结束。



图 5‑8查看设备流程图

#### 添加设备

下图为添加设备流程图，当在添加设备界面点击“添加设备”按钮时执行该流程图。首先判断指示灯是否为绿色，如果指示灯为红色，说明设备不可用，则提示用户输入新增设备的相关信息，判断设备信息是否合法，如不合法，则重复此步骤，如设备信息合法，则指示灯变绿，进行下一步骤。如果指示灯为绿色，说明设备可用，判断系统当前处于用户模式还是游客模式，如为游客模式，则将新增设备写入系统数据，结束本次添加流程；如为用户模式，则判断设备是否已存在于数据库，如存在则判断账号是否已拥有该设备，如已拥有，则提示账号已拥有该设备，结束本次流程。如未拥有，则将数据写入数据库并提示添加成功，结束本次流程。



图 5‑9添加设备流程图

#### 删除设备

下图为删除设备流程图，在点击删除设备按钮之后执行。首先，判断是否选中传感器，如未选中，则提示未选中设备，重复该流程，如选中，则需要用户确认是否删除，如确认删除，则根据系统模式选择用户模式或游客模式对应删除设备，提示删除成功，如选择否，则取消删除，本次流程结束。



图 5‑10删除设备流程图

### 设备信息查询功能

本页面具有以下几项功能：

查看设备信息：可通过在账户主菜单界面选中想要查看的设备进行信息查看，分为设备基本信息和设备监测数据两部分。

修改设备信息：此功能支持对设备名称进行修改。

查询最近数据：此功能将从OneNET云平台查询最近的一次监测数据并显示，同时会显示设备的在线情况。

打开/关闭警报：此功能可在线控制设备LED灯亮或灭。

查询历史记录：此功能支持对设备进行历史记录信息的查询，提供oneNET云平台和本次查询两种不同的数据存储方式查询。

#### 修改设备信息

通过此功能能够对设备的名称进行修改。

下图为修改设备信息流程图，当“确认修改”按钮按下时执行。首先用户输入修改后的设备信息，系统判断输入信息是否合法，如不合法，则恢复设备原始信息，并重复该流程；如合法，则判断当前处于何种模式，分别根据游客模式或用户模式针对数据进行修改保留，然后提示成功修改设备信息，本次修改流程结束。

图 5‑11修改设备信息流程图

#### 查询最近数据

此功能将从OneNET云平台查询最近的一次监测数据并显示，同时会显示设备的在线情况。

下图为查询最近数据流程图，在点击“查询数据”按钮后执行，首先获取设备的url和api-key，调用OneNetConnect类的get函数，获取http的回应respond内容，然后从respond内容中提取有用信息并显示到窗体上，此流程结束。



图 5‑12查询最近一次数据流程图

#### 打开/关闭警报

通过此功能，能够对在线的设备发出指令（控制设备的LED灯开或关）。

下图为发出警报流程图，在点击“打开警报”按钮后执行，首先判断设备的在线情况，若离线，则提示设备未在线；如在线，则获取设备的url和api-key，调用OneNetConnect类的post函数，观察设备的响应情况，此流程结束。



图 5‑13 发出警报流程图

#### 历史记录

通过此功能能够查询对应设备的历史记录，可以通过监测时间筛选指定数量的记录，同时支持从数据库及云平台两种不同途径进行查询（区别在于云平台数据时间精确到了毫秒，而数据库仅精确到秒）。

下图为查询历史记录流程图，在点击“确认查询”按钮后执行。首先输入筛选条件，系统会根据输入的筛选条件自动进行合法化处理。处理完筛选条件之后，判断历史数据的来源平台，如果选择oneNET云平台，那么会自动获取设备和历史记录查询有关的url和设备对应api-key，通过调用OneNetConnect类的get函数，获取http的响应内容respond，然后根据respond处理出对应的来自云平台的数据，加载并显示，此流程结束。如果选择本地存储，那么会判断是游客模式还是用户模式，如果是游客模式，系统会随机生成1~10组今年之内的数据加载并显示，然后结束此流程；如果是用户模式，则会查询数据库中对应的数据信息，加载并显示，最后结束本流程。



图 5‑14查询历史记录流程图

### 其他功能

#### 界面大小自定义

界面可以进行自由拉伸，控件位置及大小会随之变化。

#### 用户设备列表

设备会根据种类的不同而进行分类显示，用户可灵活选取需要操作的设备，并支持同时查看多个设备信息。

#### 账号系统

通过数据库存储账号及设备信息，并通过natapp实现内网穿透，提供数据库的远程联网访问。

游客模式：游客模式与用户模式功能相同，界面相同，不同之处在于游客模式下无需账号，无需访问数据库，用户数据会在退出游客模式后重置。

用户模式：用户模式下，用户可通过注册的账号登陆系统，并永久保存用户个人数据信息。注册账号：该模式下，用户可注册任意不存在的账号，账号由数字、英文字母组成，不超过16个字符。注销账号：该模式下，用户可将不再使用的账号进行注销，需提供与账号对应的密码。

#### 完备的提示功能

系统具备完善的提示功能，能够针对各种误操作、错误操作进行提醒，及时告知用户如何正确使用软件。

## 核心功能实现

### 连接OneNET云平台

通过http协议与OneNET平台进行通信，封装一个OneNetConnect类来提供get及post方法。提供对应url和api-key即可针对特定设备进行查询数据、发送指令等操作。

逻辑流程图

下图为OneNetConnect类get函数流程图。通过调用get函数，传入oneNET平台提供的url和对应api-key作为参数，即可针对特定设备进行get查询。首先判断url是否以http://或https://开头，若不是，则提示url格式错误，结束本流程，若是，则判断url是否有回应，如有，则说明url和api-key是可用的，如没有回应，则说明url和api-key是不可用的，提示url请求回应失败，结束本流程。



图 5‑15 OneNetConnect类Get函数流程图

下图为OneNetConnect类post函数流程图。通过调用Post函数，传入oneNET平台提供的url和对应api-key以及需要post的内容主体content即可通过oneNET平台向指定设备发送命令，设置好报文头以及http主体，即可获取到来自http的响应respond，返回respond内容，结束本流程。



图 5‑16 OneNetConnect类Post函数流程图

### 连接MySQL数据库

通过系统库提供的对应包与MySQL数据库进行通信，封装一个MySQLConnect类来提供query及update方法。提供相应sql语句即可针对指定数据库进行增删改查操作。

逻辑流程图

下图为MySQLConnect类的ExecuteQuery函数流程图，通过传入对应的sql语句作为参数，完成针对数据库的查询操作。首先判断是否能连接到数据库，如不能，则返回空数据，结束本流程；如能够连接到数据库，则返回查询到的二维表，然后结束本流程。



图 5‑17 MySQLConnect类ExecuteQuery函数流程图

下图为MySQLConnect类的ExecuteUpdate函数流程图，通过传入对应的sql语句作为参数，完成针对数据库的增删改操作。首先判断是否能连接到数据库，如不能，则返回空数据，结束本流程；如能够连接到数据库，则返回数据库中受影响的行数，然后结束本流程。



图 5‑18 MySQLConnect类的ExecuteUpdate函数流程图

### 内网穿透

通过第三方软件natapp实现内网穿透技术，将内网ip映射到公网，使本地数据库能够变为服务端数据库，支持数据库的远程访问，实现账号异地登录。

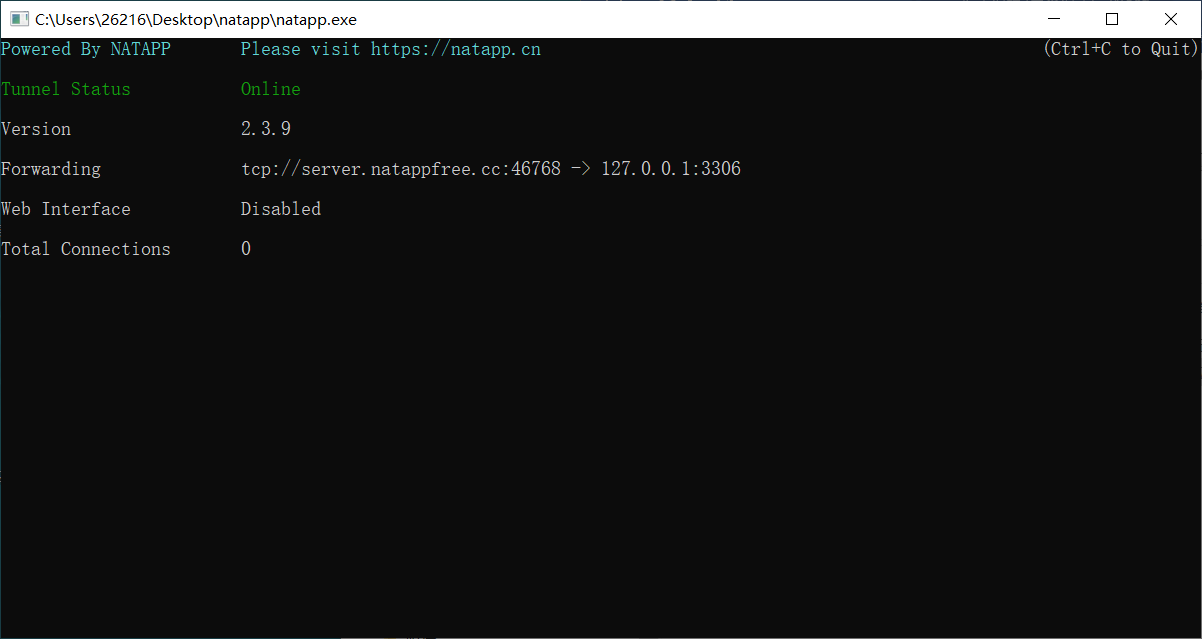


图 5‑19 natapp运行图

## 系统功能测试

### 登录功能

下图为用户登录界面的截图，由截图可以看到，登录界面主要提供的功能有登录功能、游客模式、账号注册与注销功能、测试页功能、连接数据库功能和退出系统功能。

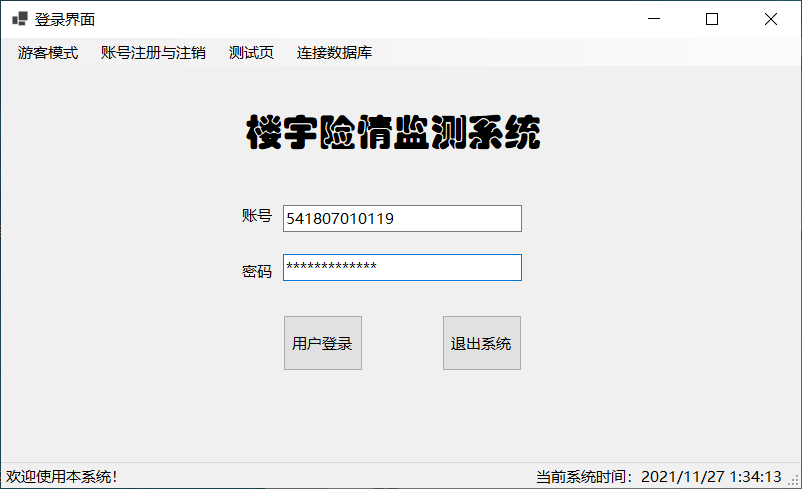


图 5‑20登录界面图

下表为登录功能测试表。

表 5‑1登录功能测试表

|  |  |  |
| --- | --- | --- |
| 测试输入 | 测试输出 | 是否异常 |
| 空账号 | 提示输入账号 | 否 |
| 空密码 | 提示输入密码 | 否 |
| 不存在的账号 | 提示账号不存在 | 否 |
| 不匹配的密码 | 提示密码错误 | 否 |
| 数据库端口号不正确 | 提示数据库错误 | 否 |
| 合法的登录操作 | 进入用户模式主菜单 | 否 |
| 点击游客模式 | 进入游客模式主菜单 | 否 |
| 点击账号注册与注销 | 进入账号注册与注销界面 | 否 |
| 点击测试页 | 进入测试页界面 | 否 |
| 点击连接数据库 | 提示输入数据库端口号 | 否 |

### 注册与注销账号功能

下图为账号注册与账号注销界面截图，由图中可以看到三个对应的输入框以及对应提示



图 5‑21注册与注销界面截图

下表为注册与注销账号功能测试表。

表 5‑2注册与注销账号功能测试表

|  |  |  |
| --- | --- | --- |
| 测试输入 | 测试输出 | 是否异常 |
| 空账号 | 提示输入账号 | 否 |
| 空密码 | 提示输入密码 | 否 |
| 空的二次密码 | 提示再次输入密码 | 否 |
| 两次输入密码不同 | 提示两次密码不相同 | 否 |
| 未设置数据库端口号 | 提示数据库连接错误 | 否 |
| 注册已存在的账号 | 提示账号已被注册 | 否 |
| 注册合法账号 | 提示注册成功 | 否 |
| 注销不存在的账号 | 提示账号不存在 | 否 |
| 注销的账号与密码不匹配 | 提示账号密码不匹配 | 否 |
| 注销二次操作取消 | 无提示，即取消注销 | 否 |
| 注销二次操作确认 | 提示注销成功 | 否 |

### 主菜单功能

下图为游客模式视角下的主菜单界面截图，由图中可以看到查看设备、添加设备、删除设备、退出登录四大功能。

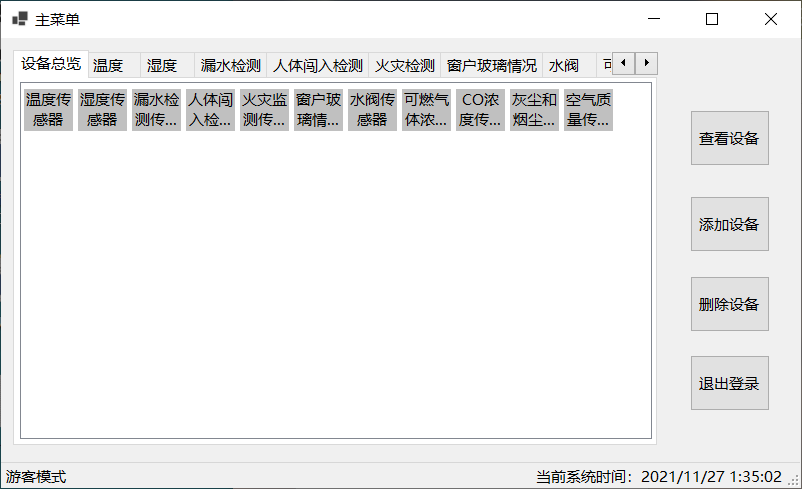


图 5‑22主菜单界面截图

下表为主菜单功能测试表。

表 5‑3主菜单功能测试表

|  |  |  |
| --- | --- | --- |
| 测试输入 | 测试输出 | 是否异常 |
| 点击添加设备 | 进入添加设备界面 | 否 |
| 点击退出登录 | 返回登录界面 | 否 |
| 空列表点击查看设备 | 提示未选中传感器 | 否 |
| 未选中设备点击查看设备 | 提示未选中传感器 | 否 |
| 选中设备点击查看设备 | 进入查看设备信息界面 | 否 |
| 空列表点击删除设备 | 提示未选中传感器 | 否 |
| 未选中设备点击删除设备 | 提示未选中传感器 | 否 |
| 删除设备二次操作取消 | 无提示，即取消删除 | 否 |
| 删除设备二次操作确认 | 提示设备已删除 | 否 |
| 分别在用户模式下和游客模式下进行相同的操作 | 二者输出一致 | 否 |

### 添加设备功能

下图为添加设备界面截图，由图可看到需要填写的设备信息，以及检测设备是否可用的指示灯及按钮。提供设备来源于OneNet云平台正确的url及对应的api-key即可为账户新增一个设备。注意数据流名称应与OneNet云平台的数据流名称一致，以便正确查询数据。



图 5‑23添加设备界面截图

下表为添加设备功能测试表。

表 5‑4添加设备功能测试表

|  |  |  |
| --- | --- | --- |
| 测试输入 | 测试输出 | 是否异常 |
| 直接点击添加设备 | 提示先检测设备是否可用 | 否 |
| 空设备名称 | 提示输入设备名称 | 否 |
| 空设备url | 提示输入设备url | 否 |
| 空设备api-key | 提示输入设备api-key | 否 |
| 空数据流名称 | 无提示 | 否 |
| 不正确的url格式 | 提示url须以http://或https://开头 | 否 |
| 不匹配的url和api-key | 提示检查url和api-key是否正确 | 否 |
| 合法的url和api-key | 指示灯变绿 | 否 |
| 用户模式下，添加同一设备同一种类传感器 | 提示重复添加 | 否 |
| 用户模式下，添加不同设备同一种类传感器 | 提示添加成功 | 否 |
| 用户模式下，添加同一设备不同种类传感器 | 提示添加成功 | 否 |
| 用户模式下，添加不同设备不同种类传感器 | 提示添加成功 | 否 |
| 游客模式下，添加同一设备同一种类传感器 | 提示添加成功，经测试，并未重复添加，只起到了修改设备信息的功能 | 否 |
| 游客模式下，添加不同设备同一种类传感器 | 提示添加成功 | 否 |
| 游客模式下，添加同一设备不同种类传感器 | 提示添加成功 | 否 |
| 游客模式下，添加不同设备不同种类传感器 | 提示添加成功 | 否 |

### 设备信息查询功能

查看设备信息：可通过在账户主菜单界面选中想要查看的设备进行信息查看，分为设备基本信息和设备监测数据两部分。

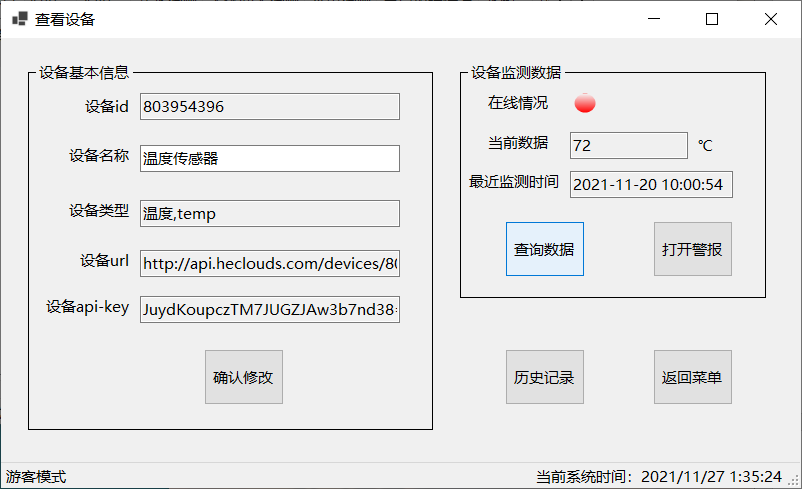


图 5‑24设备信息界面图

下表为查看设备信息功能测试表。

表 5‑5查看设备信息功能测试表

|  |  |  |
| --- | --- | --- |
| 测试输入 | 测试输出 | 是否异常 |
| 直接点击确认修改按钮 | 提示修改成功，返回主菜单 | 否 |
| 空设备名称 | 提示设备名称不能为空，恢复成为原设备名称 | 否 |
| 合法设备名称 | 提示修改成功，返回主菜单 | 否 |
| 点击查询数据 | 显示设备的在线情况、最近数据、数据种类的单位和监测时间 | 否 |
| 设备无数据时 | 显示设备的在线情况、数据显示no data、单位显示unknow、监测时间显示no data | 否 |
| 设备在线时 | 显示在线情况的指示灯为绿色 | 否 |
| 设备离线时，点击打开警报 | 提示设备未在线 | 否 |
| 设备在线时，点击打开警报 | 打开警报变为关闭警报。若观察设备，可看到设备LED灯亮 | 否 |
| 设备在线时，点击关闭警报 | 关闭警报变为打开警报。若观察设备，可看到设备LED灯灭 | 否 |
| 设备无LEDON和LEDOFF指令时 | 按钮文字仍然改变，但设备不会响应命令 | 否 |

### 设备历史记录查询功能

通过此功能能够查询对应设备的历史记录，可以通过监测时间筛选指定数量的记录，同时支持从数据库及云平台两种不同途径进行查询（区别在于云平台数据时间精确到了毫秒，而数据库仅精确到秒）。

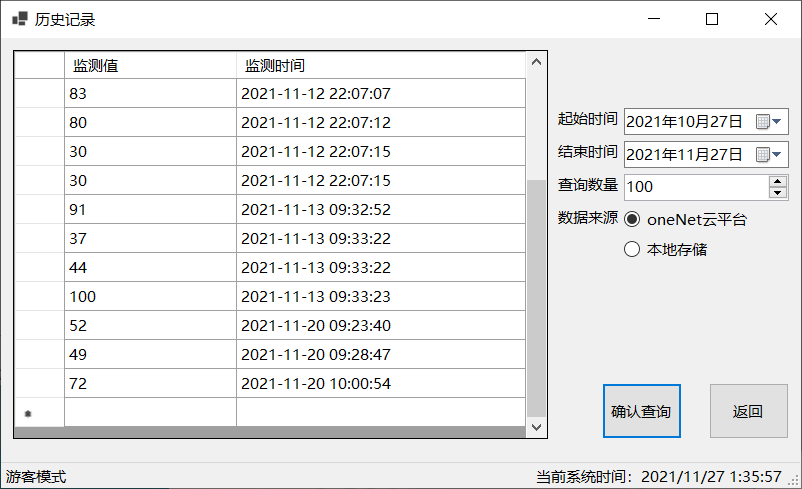


图 5‑25设备查询历史记录界面示意图

下表为设备历史记录功能测试表。

表 5‑6设备历史记录功能测试表

|  |  |  |
| --- | --- | --- |
| 测试输入 | 测试输出 | 是否异常 |
| 起始时间晚于结束时间 | 起始时间和结束时间数据互换 | 否 |
| 查询数量不在[1,6000]范围内 | 小于1时值为1，大于6000时值为6000，其他值不变 | 否 |
| 数据来源oneNET平台 | 数据监测时间格式为xxxx-xx-xx xx:xx:xx | 否 |
| 数据来源本地存储 | 数据监测时间格式为xxxx/xx/xx xx:xx | 否 |
| 游客模式和用户模式仅数据来源不同，且为oneNET平台 | 两者同一设备同一种类传感器数据相同 | 否 |
| 游客模式和用户模式仅数据来源不同，且为本地存储 | 两者同一设备同一种类传感器数据不同。  用户模式下数据相对稳定，游客模式下数据均随机 | 否 |
| 点击表中监测值 | 数据按值大小升序/降序排序 | 否 |
| 点击表中检测时间 | 数据按监测时间升序/降序排序 | 否 |
| 无网络连接时，点击查询oneNET平台数据 | 提示url地址错误 | 否 |
| 无网络连接时，用户模式下点击查询本地存储数据 | 提示数据库错误 | 否 |

结束语

通过这次的物联网企业项目实践课程，我受益匪浅。首先，我对硬件的组成以及相关原理有了一个大致的了解，通过与物联网专业同学之间的接触、沟通，我了解到一个完善的监测设备需要很多传感器分工工作，各司其职，同时，在使用esp8266 wifi模块时又需要协调工作，统一调度。其次，我对各种网络通信协议，如http协议、mqtt协议等也有了一定的了解，项目需要实现客户端与OneNET云平台之间的通信，其中就涉及到了http协议的相关内容，get、post操作需要按照协议进行代码编写才能正常运行。除此之外，我数据库方面的知识也得到了提升，通过设计并实现账号表、设备表、历史记录表和用户账号表，并将它们关联起来，通过客户端实现对mysql数据库的增删改查操作，我的整体设计能力得到了锻炼。本次课程虽然任务量比较大，但由于每周老师都有分配好一个个小目标，一周一周做下来工作量其实也不大。在项目实现过程中也有过困难，如解决如何与OneNET云平台之间正确通信、NetCore如何与MySQL进行连接等问题，最终在老师的帮助下以及网上查询资料的过程中一一解决了难题，尤其是当看到自己写的客户端代码能够控制硬件部分的LED灯亮灭的时候，还是很激动的，比起以往的纯代码编写界面、实现逻辑功能等更有成就感。

参考文献

1. lnwin521. C# 基于OneNet 的物联网数据通信[EB/OL], https://blog.csdn.net/lnwin521/article/details/84549606, 2018-11-26/2018-11-26.
2. Maze-Mozo. Asp.Net Core 连接MySQL [EB/OL], <https://blog.csdn.net/zhongyanfu0/article/details/71063773?spm=1001.2014.3001.5501>, 2017-05-01/2017-05-01.
3. A-C-K. VS连接SQL Server数据库，增删改查详细教程（C#代码）[EB/OL], https://blog.csdn.net/kiss\_\_soul/article/details/80316788, 2018-05-15/2018-05-15.
4. Jack Tian. 推荐几款实现内网穿透的实用工具[EB/OL], https://blog.csdn.net/jake\_tian/article/details/99962157, 2018-10-23/2018-10-23.
5. 开发小牛牛. natapp的使用[EB/OL], https://blog.csdn.net/baby901019/article/details/79157867, 2018-01-25/2018-01-25.
6. 匿名劉. WinForm(C#)自定义控件之——RoundButton(圆形按钮)[EB/OL], https://blog.csdn.net/qq\_36074218/article/details/102947227, 2019-11-07/2019-11-07.
7. 天黑不熄灯. C#控件大小随窗体大小等比例变化[EB/OL], https://www.cnblogs.com/sakalaka/p/5135230.html, 2016-01-16/2016-01-16.

附 录

附录1 OneNetConnect类源代码

using System;

using System.Collections.Generic;

using System.Collections.Specialized;

using System.Data;

using System.IO;

using System.Net;

using System.Reflection;

using System.Text;

using System.Windows.Forms;

namespace BuildingRiskDetectionSystem

{

/// <summary>

/// 一个连接云平台OneNet的类

/// </summary>

class OneNetConnect

{

/// <summary>

/// bool 判断url地址是否可用

/// </summary>

/// <param name="url"></param>

/// <returns></returns>

private bool IsAvaliableUrl(string url)

{

string http = url.Substring(0, url.Length < 7 ? url.Length : 7);

string https = url.Substring(0, url.Length < 8 ? url.Length : 8);

try

{

if (http != "http://" && https != "https://")

{

MessageBox.Show("url必须以http://或https://开头", "错误信息", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else return true;

}

catch

{

MessageBox.Show("url必须以http://或https://开头", "错误信息", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

return false;

}

/// <summary>

/// HttpWebResponse 获取http回应

/// </summary>

/// <param name="url"></param>

/// <param name="api\_key"></param>

/// <returns></returns>

private HttpWebResponse GetConnect(string url, string api\_key)

{

if (!IsAvaliableUrl(url)) return null;

//string url = "http://api.heclouds.com/devices/804632405/datastreams";//设备地址

try

{

HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);

request.Method = "GET";

//SetHeaderValue(request.Headers, "api-key", "U7UU2Hp4ns2V2Ug55BgrHGinQYA=");//设备API地址和 首部参数

SetHeaderValue(request.Headers, "api-key", api\_key);//设备API地址和 首部参数

request.Host = "api.heclouds.com";

request.ProtocolVersion = new Version(1, 1);

request.ContentType = "text/html;charset=UTF-8";

HttpWebResponse response = (HttpWebResponse)request.GetResponse();

if (response.StatusCode == HttpStatusCode.OK)

{

//MessageBox.Show("url请求回应成功！", "连接信息");

return response;

}

else

{

MessageBox.Show("url请求回应失败！", "错误信息", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

catch

{

MessageBox.Show("url请求回应失败！", "错误信息", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

return null;

}

/// <summary>

/// 传入url、api\_key，从oneNet获取数据

/// </summary>

/// <param name="url"></param>

/// <param name="api\_key"></param>

/// <returns></returns>

public string Get(string url, string api\_key)

{

HttpWebResponse response = GetConnect(url, api\_key);

if (response == null) return string.Empty;

Stream myResponseStream = response.GetResponseStream();

StreamReader myStreamReader = new StreamReader(myResponseStream, Encoding.GetEncoding("utf-8"));

string retString = myStreamReader.ReadToEnd();

myStreamReader.Close();

myResponseStream.Close();

return retString;

}

/// <summary>

/// 传入url、api\_key、content，向oneNet发送数据

/// </summary>

/// <param name="url"></param>

/// <param name="api\_key"></param>

/// <param name="content">post内容主体</param>

/// <returns></returns>

public string Post(string url, string api\_key, string content)

{

//string url = "http://api.heclouds.com/cmds?device\_id=843418893";

HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);

request.Method = "POST";

SetHeaderValue(request.Headers, "api-key", api\_key);//设备API地址和 首部参数

request.Host = "api.heclouds.com";

request.ProtocolVersion = new Version(1, 1);

//string Cod = "LEDOFF";

byte[] data = Encoding.UTF8.GetBytes(content);

request.ContentLength = data.Length;

using (Stream reqStream = request.GetRequestStream())

{

reqStream.Write(data, 0, data.Length);

reqStream.Close();

}

HttpWebResponse resp = (HttpWebResponse)request.GetResponse();

Stream stream = resp.GetResponseStream();

//获取响应内容

using (StreamReader reader = new StreamReader(stream, Encoding.UTF8))

{

return reader.ReadToEnd();

}

// return result;

}

/// <summary>

/// 设置报文头

/// </summary>

/// <param name="header"></param>

/// <param name="name"></param>

/// <param name="value"></param>

private void SetHeaderValue(WebHeaderCollection header, string name, string value)

{

var property = typeof(WebHeaderCollection).GetProperty("InnerCollection", BindingFlags.Instance | BindingFlags.NonPublic);

if (property != null)

{

var collection = property.GetValue(header, null) as NameValueCollection;

collection[name] = value;

}

//throw new NotImplementedException();

}

/// <summary>

/// 获取errno码

/// </summary>

/// <param name="oneNetStr"></param>

/// <returns></returns>

private int GetErrno(string oneNetStr)

{

/\*

\*{

\* "errno": 3,

\* "error": "auth failed: key: K=AoA5YDyoXu=5Ng7wDaivtyWlg"

\* }

\*/

string[] vs = oneNetStr.Split("{\"errno\":");

int errno = -1;

for(int i = 0; i < vs[1].Length; ++i)

{

if (vs[1][i] >= '0' && vs[1][i] <= '9')

{

if (errno == -1) errno = vs[1][i] - '0';

else errno = errno \* 10 + vs[1][i] - '0';

}

else break;

}

return errno;

}

/// <summary>

/// 判断url、api\_key是否可用

/// </summary>

/// <param name="url"></param>

/// <param name="api\_key"></param>

/// <returns></returns>

public bool IsAvailable(string url, string api\_key)

{

string oneNetStr = Get(url, api\_key);

if (oneNetStr == string.Empty) return false;

if (GetErrno(oneNetStr) != 0) return false;

return true;

}

/// <summary>

/// 获取历史记录表

/// </summary>

/// <param name="oneNetStr"></param>

/// <returns></returns>

public DataTable GetRecordDataTable(string oneNetStr)

{

/\*

\* {

\* "errno":0,

\* "data":{

\* "cursor":

\* "465310\_803954396\_1635843751361",

\* "count":3,

\* "datastreams":

\* [{"datapoints":

\* [

\* {"at":"2021-11-02 17:02:31.361","value": 50},

\* {"at":"2021-11-02 16:53:08.074","value":"80\r"},

\* {"at":"2021-11-02 16:53:41.808","value":"70\r"}

\* ],

\* "id":"temp"}]},

\* "error":"succ"

\* }

\*/

DataTable dataTable = new DataTable();

dataTable.Columns.Add("at", typeof(String));

dataTable.Columns.Add("value", typeof(String));

Stack<int> vs = new Stack<int>();

for(int i = 0; i < oneNetStr.Length; ++i)

{

if(oneNetStr[i] == '[')

{

vs.Push(i);

}

else if(oneNetStr[i] == ']')

{

string[] dataPoints = oneNetStr.Substring(vs.Peek() + 1, i - vs.Peek() - 1).Split("},");

foreach(string Str in dataPoints)

{

//{"at":"2021-11-02 17:02:31.361","value":50

//{"at":"2021-11-02 16:53:08.074","value":"80\r"

//{"at":"2021-11-02 16:53:41.808","value":"70\r"}

if (!Str.Contains("at") || !Str.Contains("value")) continue;

string at = Str.Substring(7, 19);

string value = string.Empty;

int idx = "{\"at\":\"2021-11-02 16:53:41.808\",\"value\":".Length;

if (Str[idx] == '"') ++idx;

while (idx < Str.Length && Str[idx] != '"' && Str[idx] != '}')

{

if (Str[idx] == '\\') value += '\\';

value += Str[idx];

++idx;

}

dataTable.Rows.Add(at, value);

}

break;

}

}

return dataTable;

}

}

}

附录2 MySQLConnect类源代码

using System;

using System.Collections.Generic;

using System.Text;

using System.Data;

using MySql.Data.MySqlClient;

using System.Windows.Forms;

namespace BuildingRiskDetectionSystem

{

/// <summary>

/// 一个连接MySQL数据库的类

/// </summary>

public class MySQLConnect

{

/// <summary>

/// 数据库地址

/// </summary>

private static string connetStr =

"data source=127.0.0.1;database=building\_risk\_detection\_system\_db;user id=root;password=lgc2621690255;";

//"Server=server.natappfree.cc;Port=38390;Stmt=;Database=building\_risk\_detection\_system\_db;Uid=root;Pwd=lgc2621690255;";

/// <summary>

/// 连接数据库，不成功时抛出异常

/// </summary>

/// <returns></returns>

private MySqlConnection Connect()

{

MySqlConnection conn = new MySqlConnection(connetStr);

try

{

conn.Open();

if (conn.State == ConnectionState.Open)

{

return conn;

//MessageBox.Show("连接成功！", "连接信息");

}

else

{

MessageBox.Show("数据库连接失败！", "错误信息", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

catch

{

MessageBox.Show("数据库连接失败！", "错误信息", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

conn.Close();

}

return null;

}

public static void SetMySqlAdress(string port)

{

connetStr = "Server = server.natappfree.cc; Port =" + port + ";Stmt=;Database=building\_risk\_detection\_system\_db;Uid=root;Pwd=lgc2621690255;";

}

/// <summary>

/// 执行查询操作 传入sql语句 返回查询结果

/// </summary>

/// <param name="sqlStr">sql语句</param>

/// <returns>查询结果</returns>

public DataTable ExecuteQuery(string sqlStr)

{

MySqlConnection conn = Connect(); //连接数据库

DataTable dataTable = new DataTable();

if (conn == null) return dataTable;

conn.Open(); //打开数据库连接

MySqlCommand cmd = new MySqlCommand(sqlStr, conn);

MySqlDataReader dr = cmd.ExecuteReader();

dataTable.Load(dr);

conn.Close();

return dataTable;

}

/// <summary>

/// 执行增删改操作 传入sql语句 返回受影响的行数

/// </summary>

/// <param name="sqlStr">sql语句</param>

/// <returns>受影响的行数</returns>

public int ExecuteUpdate(string sqlStr)

{

MySqlConnection conn = Connect(); //连接数据库

if (conn == null) return 0;

conn.Open(); //打开数据库连接

MySqlCommand cmd = new MySqlCommand(sqlStr, conn);

int count = cmd.ExecuteNonQuery(); //返回受影响的行数

conn.Close();

return count;

}

}

}