# IOT大作业说明文档

组号: 12

成员: 181250066-李镔达 181250052-黄雨晨 181250090-刘育麟 181250178-殷德林

## 基础部分

## 作业基础:

第二次作业3tasks,即温度传感器控制蜂鸣器在超过一定温度报警

### 代码:

在文件夹tempAlarmer内

- docker-compose文件负责在本地启动edgex服务
- serialCommu.py用于与Arduino uno进行串口通信从而获取DHT11传感器读取的温度
- requirements.txt内规定了python安装需要的工具包
- deviceCreation:edgex的基本配置,包含了用于向edgex注册设备和设备服务的自动化执行脚本(.py)以及相关 配置文件(.yaml)
- app.py用于接收edgex规则引擎执行的命令并将结果发送给Arduino uno
- 3tasks,内含Arduino代码,用于感知温度,通信以及控制蜂鸣器

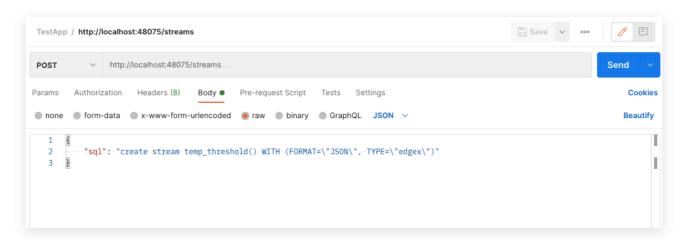
## 基本逻辑:

- 1. Arduino温度传感任务通过温度传感器感知温度放进消息队列
- 2. 通信任务串口通知本地
- 3. 本地发送到edgex
- 4. edgex规则引擎进行判断并传送判断结果
- 5. 本地将结果写回串口
- 6. 通信任务从串口度结果并将控制指令放进另一个消息队列
- 7. 蜂鸣器任务从第二个消息队列读取控制指令进行控制

## 配置和执行步骤:

- 1. 运行docker-compose文件启动edgex服务
- 2. 分别运行deviceCreation内的createAlarmDevice.py和createTempSensor.py
- 3. 配置规则引擎

创建数据流



#### 规则1:小于等于28度时向服务发送判断结果:low

```
POST
             http://localhost:48075/rules
                                                                                                                                Send
Params
        Authorization Headers (8)
                                   Body 

                                              Pre-request Script
                                                                Tests
                                                                       Settings
                                                                                                                                     Cookies
        ● form-data ■ x-www-form-urlencoded ■ raw ■ binary ■ GraphQL JSON ∨
                                                                                                                                    Beautify
none
   1
           "id": "temp_rule2",
   2
   3
           "sql": "SELECT temperature FROM temp_threshold WHERE temperature <= 28",
   4
           "actions": [
   5
                     ...<mark>"url":</mark> http://host.docker.internal:5000/api/v1/device/47df48a8-c8b7-4b6d-896e-e3635e8dab41/judgeTemp",
   8
                       "method": "put",
   9
                       "retryInterval": -1,
                      "dataTemplate": "{\"judge\":\"low\"}",
  10
  11
                       "sendSingle": true
  12
  13
  14
  15
                   "log":{}
  16
  17
  18
       3
```

## 规则2:超过28度时向服务发送判断结果:high

```
POST
            http://localhost:48075/rules
                                                                                                                           Send
Params
        Authorization Headers (8) Body ● Pre-request Script Tests Settings
                                                                                                                               Cookies
none
       Beautify
   1
   2
           "id": "temp_rule1",
           "sql": \ "SELECT \cdot temperature \cdot FROM \cdot temp\_threshold \cdot WHERE \cdot temperature \cdot > \cdot 28",
   3
   4
           "actions": [
                    ···<mark>"url":</mark>:"http://host.docker.internal:5000/api/v1/device/47df48a8-c8b7-4b6d-896e-e3635e8dab41/judgeTemp",
                     "method": "put",
                      "retryInterval": -1,
                      "dataTemplate": "{\"judge\":\"high\"}",
  10
                      "sendSingle": true
  11
  12
  13
             . . },
  14
                  "log":{}
  15
  16
  17
  18
```

#### 4. 3tasks的代码上板运行

5. 运行serialCommu.py和app.py 运行时serialCommu.py会滚动输出读取的温度以及规则引擎判定的结果:low/high app.py则会滚动输出执行的命令

### 结果视频:

edgex3tasks.mp4

在温度超过28度时,蜂鸣器会发出滴滴响声进行报警

## bonus部分

## 作业基础:

atlas200DK基本实验

## 技术说明:

face\_recognition人脸识别库

shell脚本定时任务

python定时任务

edgex redis存储设备

## 执行步骤:

- 1. 使用shell脚本调用main应用,进行5秒一次的拍照。
- 2. shell脚本在每次拍完照之后将文件传到服务器。
- 3. 服务器有一个python人脸识别的脚本一直运行,侦测传过来的照片并计算是不是有人脸。
- 4. 如果有人脸,则查看样本库有没有对应的人脸。
- 5. 有对应的人脸的话,则将识别到的人脸存到edgex中。

#### 结果视频:

bonus.MP4

预先将拍好的人脸照片存入人脸识别库当中

通过脚本实现定时拍照,视频中的同学正在使用摄像头进行拍摄

拍好后本地人脸识别脚本执行判断,found face以后会对拍摄到的人脸进行识别,识别出结果后打印include:识别的人名