import RPi.GPIO as GPIO

import sys

import dht11

from PyQt5.QtWidgets import \*

from PyQt5 import uic

from PyQt5.QtCore import QThread, pyqtSignal, QStringListModel

from PyQt5.QtGui import QImage, QPixmap

import time

form\_class = uic.loadUiType("./test01.ui")[0]

# GPIO 초기화

GPIO.cleanup()

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

# 핀 설정

LED\_RED = 19

LED\_BLUE = 13

LED\_GREEN = 6

DHT\_PIN = 17

PIEZO\_PIN = 18  # buzzer

# 센서 및 부저 설정

dht\_sensor = dht11.DHT11(pin=DHT\_PIN)

GPIO.setup(PIEZO\_PIN, GPIO.OUT)

Buzz = GPIO.PWM(PIEZO\_PIN, 440)

# LED 핀 설정

GPIO.setup(LED\_RED, GPIO.OUT)

GPIO.setup(LED\_BLUE, GPIO.OUT)

GPIO.setup(LED\_GREEN, GPIO.OUT)

class DHTSensorReader(QThread):

    update\_signal = pyqtSignal(list)

    def \_\_init\_\_(self, dht\_sensor):

        super().\_\_init\_\_()

        self.dht\_sensor = dht\_sensor

        self.running = True

    def run(self):

        while self.running:

            result = self.dht\_sensor.read()

            if result.is\_valid():

                temperature = f"Temperature: {result.temperature:.1f}"

                humidity = f"Humidity: {result.humidity:.1f}%"

                self.update\_signal.emit([temperature, humidity])

            else:

                self.update\_signal.emit(["Failed to get reading. Try again!"])

            time.sleep(2)

    def stop(self):

        self.running = False

        self.wait()

class WindowClass(QMainWindow, form\_class):

   def \_\_init\_\_(self):

      super().\_\_init\_\_()

      self.setupUi(self)

      self.Btn\_ON.clicked.connect(self.btnOnFunction)

      self.Btn\_OFF.clicked.connect(self.btnOffFunction)

      self.Btn\_RED.clicked.connect(self.btnRedFunction)

      self.Btn\_BLUE.clicked.connect(self.btnBlueFunction)

      self.Btn\_GREEN.clicked.connect(self.btnGreenFunction)

      self.Btn\_WHITE.clicked.connect(self.btnWhiteFunction)

      self.sensor\_reader = DHTSensorReader(dht\_sensor)

      self.sensor\_reader.update\_signal.connect(self.update\_list\_view)

      # QListView 설정

      self.model = QStringListModel()

      self.listView.setModel(self.model)

      self.sensor\_reader = None

      self.is\_running = False

      self.Stbtn.clicked.connect(self.start\_clicked)

      self.Spbtn.clicked.connect(self.stop\_clicked)

   def btnOnFunction(self):

      print("LED가 활성화 되었습니다")

   def update\_list\_view(self, data):

      self.model.setStringList(data)

      if len(data) > 1 and "Humidity" in data[1]:

         humidity = float(data[1].split(':')[1].strip('%'))

         if humidity < 50:

            Buzz.stop()

         else:

            Buzz.start(50)

   def btnOffFunction(self):

      GPIO.output(LED\_RED, True)

      GPIO.output(LED\_BLUE, True)

      GPIO.output(LED\_GREEN, True)

      print("LED가 종료되었습니다")

   def btnRedFunction(self):

      GPIO.output(LED\_RED, False)

      GPIO.output(LED\_BLUE, True)

      GPIO.output(LED\_GREEN, True)

      print("빨간불이 켜졌습니다")

   def btnBlueFunction(self):

      GPIO.output(LED\_RED, True)

      GPIO.output(LED\_BLUE, False)

      GPIO.output(LED\_GREEN, True)

      print("파란불이 켜졌습니다")

   def btnGreenFunction(self):

      GPIO.output(LED\_RED, True)

      GPIO.output(LED\_BLUE, True)

      GPIO.output(LED\_GREEN, False)

      print("초록불이 켜졌습니다")

   def start\_clicked(self):

      if not self.is\_running:

         self.is\_running = True

         self.sensor\_reader = DHTSensorReader(dht\_sensor)

         self.sensor\_reader.update\_signal.connect(self.update\_list\_view)

         self.sensor\_reader.start()

         print("start")

   def stop\_clicked(self):

      if self.is\_running:

         self.is\_running = False

         self.sensor\_reader.stop()

         Buzz.stop()

         print("stop")

   def btnWhiteFunction(self):

      GPIO.output(LED\_RED, False)

      GPIO.output(LED\_BLUE, False)

      GPIO.output(LED\_GREEN, False)

      print("흰색불이 켜졌습니다")

   def closeEvent(self, event):

      if self.sensor\_reader and self.sensor\_reader.isRunning():

         self.sensor\_reader.stop()

         self.sensor\_reader.wait()

      GPIO.cleanup()

      event.accept()

if \_\_name\_\_ == "\_\_main\_\_":

   app = QApplication(sys.argv)

   myWindow = WindowClass()

   myWindow.show()

   app.exec\_()