11/5/2023

Outline of IOT 25-

Use a Python Programs to:  
Write GrovePi Sensor Data to MariaDB  
Write index.html webpage with 25 most recent sensor readings

Create two crontab entries that read sensor on even minutes, write webpage on odd minutes:

For creating new crontab entry, open Terminal window, enter:  
sudo nano /etc/crontab  
Use arrows to edit  
Refer to: <https://crontab.guru/examples.html>

A computer screen shot of text

Description automatically generated

A screenshot of a computer

Description automatically generated

Code:

Python program to read sensor data:

# PgP 10/31/2023

# insertSensorReading.py

# read GrovePi+ sensors-light, potentiometer, sound, ultrasonic, DHT

# Test of WSU COB Somsen 301 GrovePi+ system

# send readings to mariaDB

# separate program sensorswebpage.py writes sensor values to web page

#!/usr/bin/env python

# insert.py, inserts record into maria database 'sensor'

# table-GroveSensors

# fields- temperature(F), humidity(%), distance(inches), light(0 to 1023), sound(0 to 1023), potentiometer(0 to 1023) and timestamp

# PgP 11/5/2023

import mariadb # avoid mysql connectors which can generate 'UTF-8 unsupported' error

from grovepi import \*

import grovepi

import datetime

# Assign ports for all sensors and actuators

light\_sensor = 0 # Analog Port for Light sensor

potentiometer = 1 # Analog Port for Rotary Angle sensor

sound\_sensor = 2 # Analog Port for Sound sensor

ultrasonic\_ranger = 2 # Digital Port for Ultrasonic Ranger sensor

buzzer\_pin=3 # Digital Port for Buzzer actuator

dht\_sensor\_port = 5 # Digital Port for DHT sensor

dht\_sensor\_type = 0 # use 0 for the blue-colored sensor and 1 for the white-colored sensor

# setup Mariadb Connection

mydb = mariadb.connect( # establish database connection

host="192.168.1.208", # replace with RPi IP address

user="user",

password="user",

database="sensor"

)

mycursor = mydb.cursor()

try:

# Ultrasonic ranger

distance = (ultrasonicRead(ultrasonic\_ranger))

inches =str(int(distance / 2.54))

print("Ultrasonic sensor indicates distance to object is ", inches, " inches")

# Tenperature and Humidity sensor

[temp,hum] = dht(dht\_sensor\_port,dht\_sensor\_type)

temperature = str(int(temp \* 9 / 5 + 32))

humidity = str(int(hum))

print("Temperature is ", temperature, "degrees Fahrenheit, Humidity is ", humidity, " percent")

# Light Sensor

light=str(grovepi.analogRead(light\_sensor))

print("Light level is ", light, " out of a 1023 maximum")

# Potentiometer, dial

adc\_ref = 5 #analog to digital converter reference voltage

grove\_vcc = 5 # grove interface voltage

full\_angle = 300 # potentiometer rotates through 300 degrees

# Read sensor value from potentiometer

sensor\_value = grovepi.analogRead(potentiometer)

# Calculate voltage

voltage = round((float)(sensor\_value) \* adc\_ref / 1023, 2)

# Calculate rotation in degrees (0 to 300)

degrees = round((voltage \* full\_angle) / grove\_vcc, 2)

timestamp = datetime.datetime.now()

print("Potentiometer setting = %d voltage = %.2f degrees = %.1f " %(sensor\_value, voltage, degrees))

# Microphone

sound\_value = grovepi.analogRead(sound\_sensor)

soundValue = str(int(sound\_value))

print ("Background sound level is ", soundValue, " out of a 1023 maximum")

# write readings to database

sql = "INSERT INTO GroveSensors(Humidity, Temperature\_F, Distance\_Inch, Light, Sound, Potentiometer, Timestamp) VALUES (%s, %s, %s, %s, %s, %s, %s)"

val = (temperature, humidity, inches, light, soundValue, degrees, timestamp)

mycursor.execute(sql, val) # insert sensor readings and time into database

mydb.commit()

except (IOError,TypeError) as e:

print("Error")

Python program to create sensor web page, index.html:

# PgP 11/5/2023 create webpage with table to display last 25 sensor readings

# this does all environment sensors

import mariadb

# Database configuration

db\_config = {

"host": "192.168.1.208", # replace with RPi IP address

"user": "user",

"password": "user",

"database": "sensor",

}

try:

# Connect to the MariaDB database

connection = mariadb.connect(\*\*db\_config)

cursor = connection.cursor()

# Execute a query to retrieve the data

# query = "SELECT recordid, temperature, humidity, timestamp FROM dht"

query = "SELECT \* FROM GroveSensors ORDER BY recordid DESC LIMIT 25;"

cursor.execute(query)

# Fetch all the rows from the query

data = cursor.fetchall()

# Close the cursor and the connection

cursor.close()

connection.close()

# Generate an HTML file with a table

with open("/var/www/html/index.html", "w") as html\_file:

#with open("/home/pi/public\_html/index.html", "w") as html\_file:

html\_file.write("<meta http-equiv='refresh' content='50'>")

html\_file.write("<html><head><title>Grove Sensor Table</title></head><body>")

html\_file.write("<h1>Grove Sensors Table, 25 most recent records</h1>")

html\_file.write("<table border='1'><tr><th>Record ID</th><th>Relative Humidity</th><th>Temperature, F</th><th>Distance(inch)</th><th>Light</th><th>Sound</th><th>Potentiometer</th><th>Timestamp</th></tr>")

for row in data:

html\_file.write(f"<tr><td>{row[0]}</td><td>{row[1]}</td><td>{row[2]}</td><td>{row[3]}</td> <td>{row[4]}</td><td>{row[5]}</td><td>{row[6]}</td><td>{row[7]}</td> </tr>")

html\_file.write("</table></body></html")

print("HTML file created: index.html")

except mariadb.Error as e:

print(f"Error: {e}")

Here is a refactored version of sensor readings, suggested by ChatGPT:

import mariadb

import grovepi

import datetime

# Define constants

DB\_CONFIG = {

"host": "192.168.1.208",

"user": "user",

"password": "user",

"database": "sensor",

}

SENSOR\_CONFIG = {

"dht\_sensor\_port": 5,

"dht\_sensor\_type": 0,

"light\_sensor": 0,

"potentiometer": 1,

"sound\_sensor": 2,

"ultrasonic\_ranger": 2,

}

def connect\_to\_database():

# Connect to the MariaDB database

try:

mydb = mariadb.connect(\*\*DB\_CONFIG)

return mydb, mydb.cursor()

except mariadb.Error as e:

print(f"Database connection error: {e}")

return None, None

def read\_sensor\_values(sensor\_config):

try:

# Read sensor values here

# Return values as a dictionary

return {

"temperature": 0,

"humidity": 0,

"distance": 0,

"light": 0,

"sound": 0,

"potentiometer": 0,

"timestamp": datetime.datetime.now(),

}

except (IOError, TypeError) as e:

print(f"Sensor reading error: {e}")

return None

def insert\_sensor\_data(cursor, data):

if data is not None:

try:

sql = "INSERT INTO GroveSensors (Humidity, Temperature\_F, Distance\_Inch, Light, Sound, Potentiometer, Timestamp) VALUES (%s, %s, %s, %s, %s, %s, %s)"

val = (

data["temperature"],

data["humidity"],

data["distance"],

data["light"],

data["sound"],

data["potentiometer"],

data["timestamp"],

)

cursor.execute(sql, val)

except mariadb.Error as e:

print(f"Database insertion error: {e}")

def main():

mydb, mycursor = connect\_to\_database()

if mydb and mycursor:

data = read\_sensor\_values(SENSOR\_CONFIG)

insert\_sensor\_data(mycursor, data)

mydb.commit()

mydb.close()

else:

print("Database connection failed.")

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

main()