



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROJECT NAME: ENVIRONMENTAL MONITORING

TEAM NAME : Proj_224786_Team_3

TEAM MEMBERS:

HARIKEERTHI.S(113321104025)

HARITHA.D(113321104027)

HEMALATHA.S(113321104028)

JANANI.M.A(113321104030)

DEFINITION:

The Environmental Monitoring using IOT Project aims to deploy a network of Internet of things(IOT) devices to collect real-time data about various environmental parameters.

These parameters typically include temperature, humidity, air quality, pollution levels, soil moisture, and more depending on the specific objectives of the project.

The collected data is then transmitted to a central database or cloud platform for analysis and visualization, enabling stakeholders to make informed decisions regarding environmental conservation, resource management, and public health.



OBJECTIVE:

1.Data Storage and Management:

Implement a database or cloud-based storage solution to securely store the incoming environmental data. Ensure data integrity, availability and scalability.

2.Data Analysis and Visualization:

Develop the analytical algorithms and visualization tools to process and Present the collected data in a user-friendly format.

3.Environmental Impact Assessment:

Use the collected data to assess the impact of human activities on the environment.



PROJECT:

This guide covers building a small, internet-enabled environmental monitor which can track a range of data such from temperature to UV-level to the amount of total-volatile-organic-compounds present in the air.

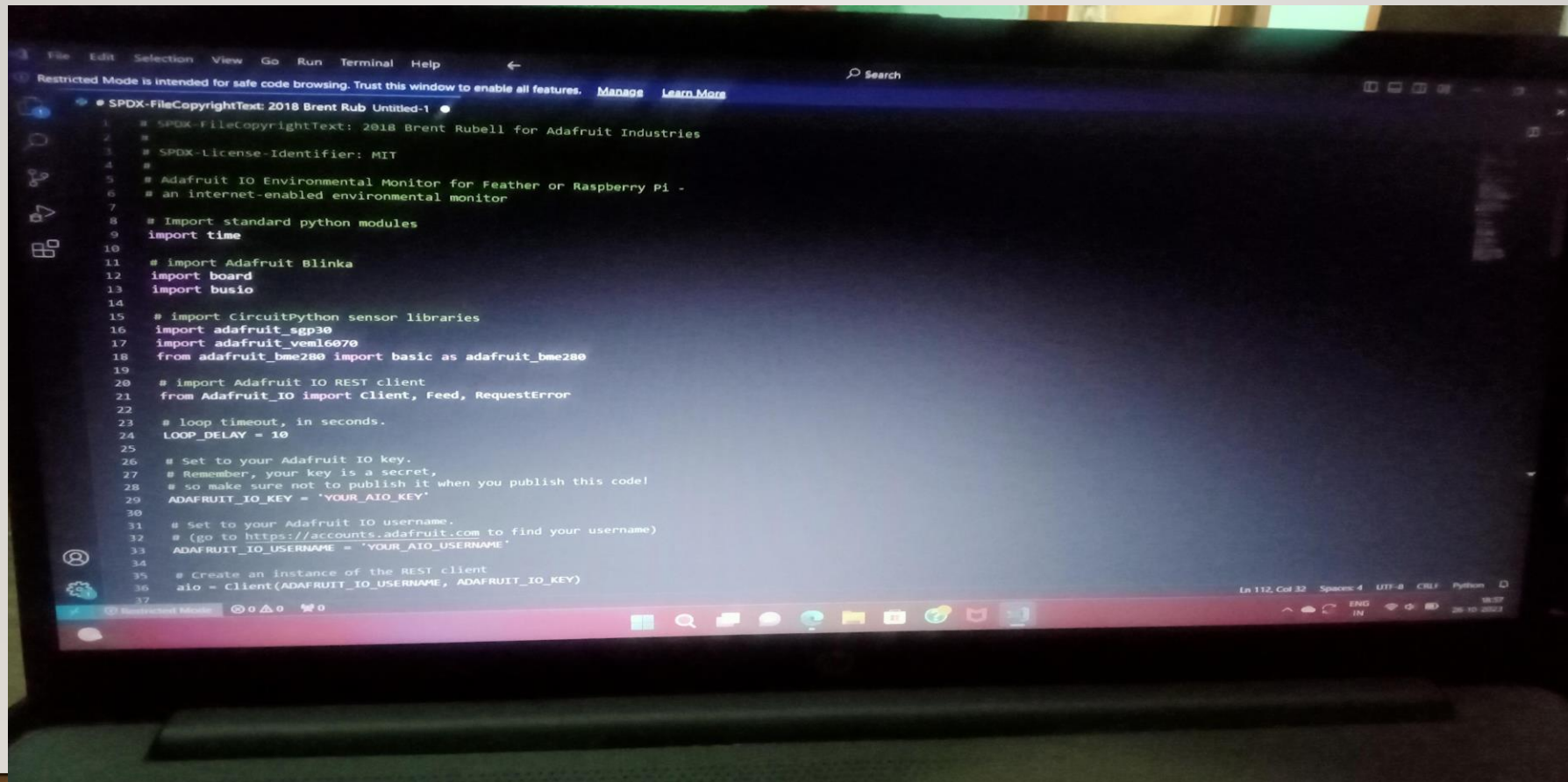
Build it, place it in a location you'd like to log , and then monitor it from anywhere in the world using our internet of things platform – AdafruitIO.



REQUIREMENTS:

- Temperature sensors.
- Air quality sensors.
- Gas sensors.
- Humidity sensors.
- Water quality sensors.
- Radiation sensors.
- Research and policy Support.

PYTHON CODE:



```
1  # SPDX-FileCopyrightText: 2018 Brent Rubell for Adafruit Industries
2  #
3  # SPDX-License-Identifier: MIT
4  #
5  # Adafruit IO Environmental Monitor for Feather or Raspberry Pi -
6  # an internet-enabled environmental monitor
7  #
8  # Import standard python modules
9  import time
10
11 # Import Adafruit Blinka
12 import board
13 import busio
14
15 # Import CircuitPython sensor libraries
16 import adafruit_sgp30
17 import adafruit_veml6070
18 from adafruit_bme280 import basic as adafruit_bme280
19
20 # Import Adafruit IO REST client
21 from Adafruit_IO import Client, Feed, RequestError
22
23 # loop timeout, in seconds.
24 LOOP_DELAY = 10
25
26 # Set to your Adafruit IO key.
27 # Remember, your key is a secret,
28 # so make sure not to publish it when you publish this code!
29 ADAFRUIT_IO_KEY = 'YOUR_AIO_KEY'
30
31 # Set to your Adafruit IO username.
32 # (go to https://accounts.adafruit.com to find your username)
33 ADAFRUIT_IO_USERNAME = 'YOUR_AIO_USERNAME'
34
35 # Create an instance of the REST client
36 aio = Client(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)
```

```
File Edit Selection View Go Run Terminal Help
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
• SPDX-FileCopyrightText: 2018 Brent Rub Untitled-1 •
23 # loop timeout, in seconds.
24 LOOP_DELAY = 10
25
26 # Set to your Adafruit IO key.
27 # Remember, your key is a secret,
28 # so make sure not to publish it when you publish this code!
29 ADAFRUIT_IO_KEY = 'YOUR_AIO_KEY'
30
31 # Set to your Adafruit IO username.
32 # (go to https://accounts.adafruit.com to find your username)
33 ADAFRUIT_IO_USERNAME = 'YOUR_AIO_USERNAME'
34
35 # Create an instance of the REST client
36 aio = Client(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)
37
38 try: # if we already have the feeds, assign them.
39     tvoc_feed = aio.feeds('tvoc')
40     eCO2_feed = aio.feeds('eco2')
41     uv_feed = aio.feeds('uv')
42     temperature_feed = aio.feeds('temperature')
43     humidity_feed = aio.feeds('humidity')
44     pressure_feed = aio.feeds('pressure')
45     altitude_feed = aio.feeds('altitude')
46 except RequestError: # if we don't, create and assign them.
47     tvoc_feed = aio.create_feed(Feed(name='tvoc'))
48     eCO2_feed = aio.create_feed(Feed(name='eco2'))
49     uv_feed = aio.create_feed(Feed(name='uv'))
50     temperature_feed = aio.create_feed(Feed(name='temperature'))
51     humidity_feed = aio.create_feed(Feed(name='humidity'))
52     pressure_feed = aio.create_feed(Feed(name='pressure'))
53     altitude_feed = aio.create_feed(Feed(name='altitude'))
54
55 # Create busio I2C
56 i2c = busio.I2C(board.SCL, board.SDA)
57 # Create VEML6070 object.
58 uv = adafruit_veml6070.VEML6070(i2c)
```



```
temperature_feed = aio.create_feed(Feed(name='temperature'))
humidity_feed = aio.create_feed(Feed(name='humidity'))
pressure_feed = aio.create_feed(Feed(name='pressure'))
altitude_feed = aio.create_feed(Feed(name='altitude'))

# Create busio I2C
i2c = busio.I2C(board.SCL, board.SDA)
# Create VEML6070 object.
uv = adafruit_veml6070.VEML6070(i2c)
# Create BME280 object.
bme280 = adafruit_bme280.Adafruit_BME280_I2C(i2c)
bme280.sea_level_pressure = 1013.25
# Create SGP30 object using I2C.
sgp30 = adafruit_sgp30.Adafruit_SGP30(i2c)
sgp30.iaq_init()
sgp30.set_iaq_baseline(0x8973, 0x8aee)

# Sample VEML6070
def sample_VEML():
    for _ in range(10):
        uv_raw = uv.uv_raw
    return uv_raw

while True:
    print('Reading sensors...')
    # Read SGP30.
    ecO2_data = sgp30.ecO2
    tvoc_data = sgp30.TVOC

    # Read VEML6070.
    uv_data = sample_VEML()

    # Read BME280.
    temp_data = bme280.temperature
    # Convert temperature (C->F)
    temp_data = int(temp_data) * 1.8 + 32
```

Ln 112, Col 32 Spaces: 4 UTF-8 CRLF Python 3.8.10 10:58 20-10-2023


```
File Edit Selection View Go Run Terminal Help
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
# SPDX-FileCopyrightText: 2018 Brent Rub Untitled-1
90 temp_data = bme280.temperature
91 # convert temperature (C->F)
92 temp_data = int(temp_data) * 1.8 + 32
93 humid_data = bme280.humidity
94 pressure_data = bme280.pressure
95 alt_data = bme280.altitude
96
97 print('sending data to adafruit io...')
98 # Send SGP30 Data to Adafruit IO.
99 print('eCO2:', eCO2_data)
100 aio.send(eCO2_feed.key, eCO2_data)
101 print('tvoc:', tvoc_data)
102 aio.send(tvoc_feed.key, tvoc_data)
103 time.sleep(2)
104 # Send VEML6070 Data to Adafruit IO.
105 print('UV Level: ', uv_data)
106 aio.send(uv_feed.key, uv_data)
107 time.sleep(2)
108 # Send BME280 Data to Adafruit IO.
109 print('Temperature: %0.1f C' % temp_data)
110 aio.send(temperature_feed.key, temp_data)
111 print("Humidity: %0.1f %" % humid_data)
112 aio.send(humidity_feed.key, int(humid_data))
113 time.sleep(2)
114 print("Pressure: %0.1f hPa" % pressure_data)
115 aio.send(pressure_feed.key, int(pressure_data))
116 print("Altitude = %0.2f meters" % alt_data)
117 aio.send(altitude_feed.key, int(alt_data))
118 # avoid timeout from adafruit io
119 time.sleep([LOOP_DELAY * 60])
```

CONCLUSION:

The IOT based Environmental Monitoring System has been designed and implemented. The environmental parameters successfully transmitted via ESP 8266 Wi-Fi module. The density of the gases in the remote located area viewed through the Thing speak web server. This project will protect the people from the pollutant gases. It is more useful for the industries to control the air pollution in the surrounding area for the workers safety .



