Code of wokwi

import network

import time

import urandom

from umqtt.simple import MQTTClient

# ThingSpeak MQTT broker details

mqtt\_client\_id = "OSMAGzMLLh8cPC4wDB0tOgU"

mqtt\_user = "OSMAGzMLLh8cPC4wDB0tOgU"

mqtt\_password = "ene6Wle7359cO+fx6gagtfww"

mqtt\_server = "mqtt3.thingspeak.com"

mqtt\_port = 1883

mqtt\_topic\_temperature = "channels/486827/publish/fields/field1"

mqtt\_topic\_humidity = "channels/486827/publish/fields/field2"

mqtt\_topic\_co2 = "channels/486827/publish/fields/field3"

# Wi-Fi details

WIFI\_SSID = "Wokwi-GUEST"

WIFI\_PASSWORD = ""

# Historical data storage

historical\_data = []

# Function to generate random sensor values

def generate\_sensor\_data():

    temperature = urandom.uniform(-50, 50)

    humidity = urandom.uniform(0, 100)

    # Ensure CO2 value is within the acceptable range (300 to 2000 ppm)

    co2 = urandom.uniform(300, 2000)

    return temperature, humidity, co2

# Function to publish data to ThingSpeak

def publish\_to\_thingspeak(temperature, humidity, co2):

    client = MQTTClient(mqtt\_client\_id, mqtt\_server, user=mqtt\_user, password=mqtt\_password)

    client.connect()

    client.publish(mqtt\_topic\_temperature, str(temperature))

    client.publish(mqtt\_topic\_humidity, str(humidity))

    client.publish(mqtt\_topic\_co2, str(co2))

    client.disconnect()

# Connect to Wi-Fi

sta\_if = network.WLAN(network.STA\_IF)

sta\_if.active(True)

sta\_if.connect(WIFI\_SSID, WIFI\_PASSWORD)

# Wait for Wi-Fi connection

while not sta\_if.isconnected():

    pass

print("Connected to Wi-Fi")

# Main loop to generate and publish sensor data

while True:

    temperature, humidity, co2 = generate\_sensor\_data()

    publish\_to\_thingspeak(temperature, humidity, co2)

print("Published: Temperature={:.2f}C, Humidity={:.2f}%, CO2={:.2f}ppm".format(temperature, humidity, co2))

    historical\_data.append((temperature, humidity, co2))  # Store historical data

    if len(historical\_data) > 720:  # Approximately 5 hours with data every 5 seconds

        historical\_data.pop(0)  # Remove oldest data point if exceeds 5 hours

    time.sleep(5)  # Adjust the delay as needed (Reduced to 5 seconds for faster data entry)

def get\_last\_five\_hours\_data(sensor\_type):

    # Assuming sensor\_type is one of 'temperature', 'humidity', or 'co2'

    sensor\_index = {'temperature': 0, 'humidity': 1, 'co2': 2}[sensor\_type]

    last\_five\_hours\_data = [(data[sensor\_index],) for data in historical\_data[-720:]]

    return last\_five\_hours\_data

# Example usage to retrieve last five hours data for temperature

last\_five\_hours\_temperature = get\_last\_five\_hours\_data('temperature')

print("Last five hours temperature data:", last\_five\_hours\_temperature)

MATLAB Code:

% Set your ThingSpeak channel ID and read API key

channelID = 2486827 ;

readAPIKey = 'DRMUN5G1XVMKR8NN';

% Get the current time and time five hours ago

currentTime = datetime('now', 'TimeZone', 'UTC');

fiveHoursAgo = currentTime - hours(5);

% Set up the ThingSpeak URL for fetching data

url = sprintf('https://api.thingspeak.com/channels/%d/feeds.json?api\_key=%s&start=%s&end=%s', ...

channelID, readAPIKey, datestr(fiveHoursAgo, 'yyyy-mm-ddTHH:MM:SSZ'), ...

datestr(currentTime, 'yyyy-mm-ddTHH:MM:SSZ'));

% Fetch data from ThingSpeak

data = webread(url);

% Extract sensor data

if ~isempty(data.feeds)

sensorData = [data.feeds.field1]; % Assuming the sensor data is in Field 1

timestamps = datetime({data.feeds.created\_at}, 'InputFormat', 'yyyy-MM-dd''T''HH:mm:ss''Z''', 'TimeZone', 'UTC');

% Display sensor data

disp('Sensor Data:');

disp(sensorData);

disp('Timestamps:');

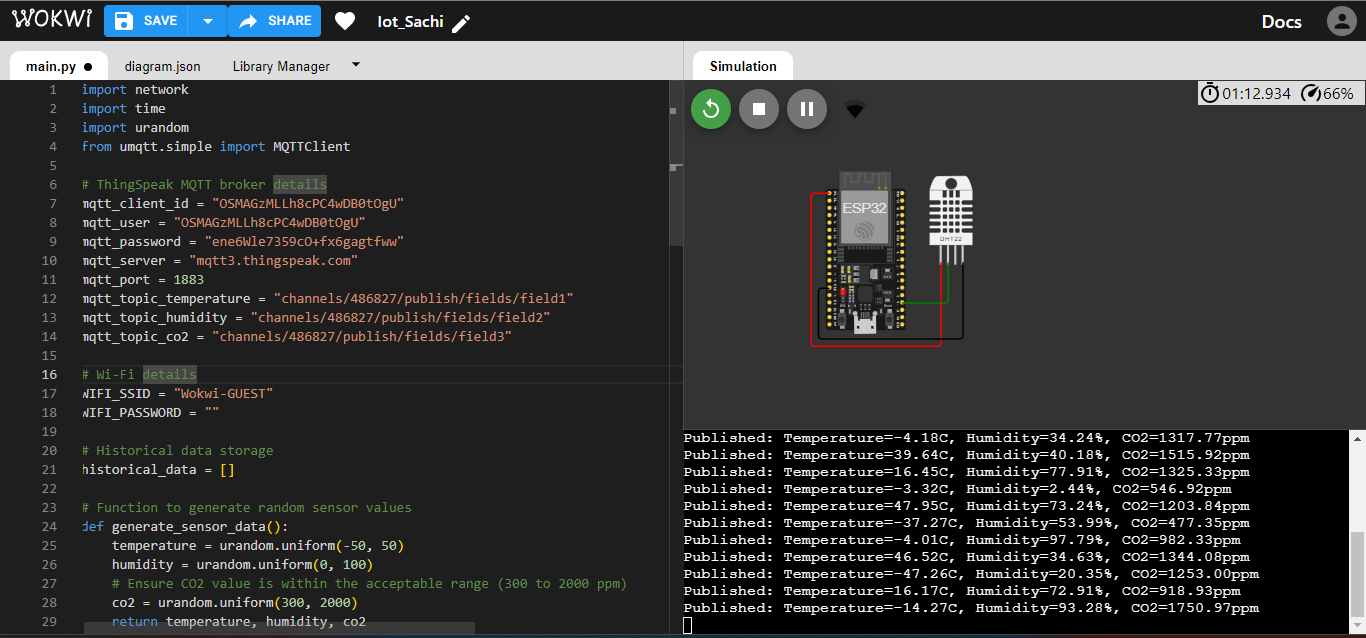
disp(timestamps);

else

disp('No data found in the specified time range.');

end

Wokwi Screenshots



MATLAB CODE

% Set your ThingSpeak channel ID and read API key

channelID = 2486827 ;

readAPIKey = 'DRMUN5G1XVMKR8NN';

% Get the current time and time five hours ago

currentTime = datetime('now', 'TimeZone', 'UTC');

fiveHoursAgo = currentTime - hours(5);

% Set up the ThingSpeak URL for fetching data

url = sprintf('https://api.thingspeak.com/channels/%d/feeds.json?api\_key=%s&start=%s&end=%s', ...

channelID, readAPIKey, datestr(fiveHoursAgo, 'yyyy-mm-ddTHH:MM:SSZ'), ...

datestr(currentTime, 'yyyy-mm-ddTHH:MM:SSZ'));

% Fetch data from ThingSpeak

data = webread(url);

% Extract sensor data

if ~isempty(data.feeds)

sensorData = [data.feeds.field1]; % Assuming the sensor data is in Field 1

timestamps = datetime({data.feeds.created\_at}, 'InputFormat', 'yyyy-MM-dd''T''HH:mm:ss''Z''', 'TimeZone', 'UTC');

% Display sensor data

disp('Sensor Data:');

disp(sensorData);

disp('Timestamps:');

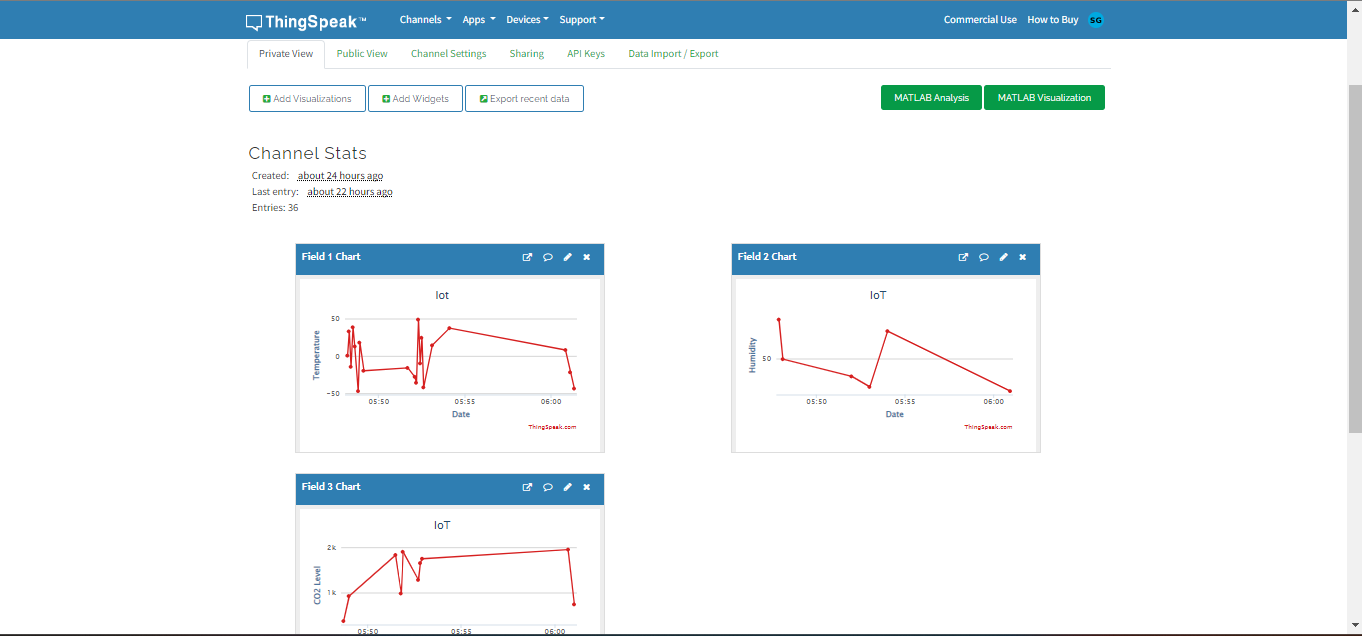
disp(timestamps);

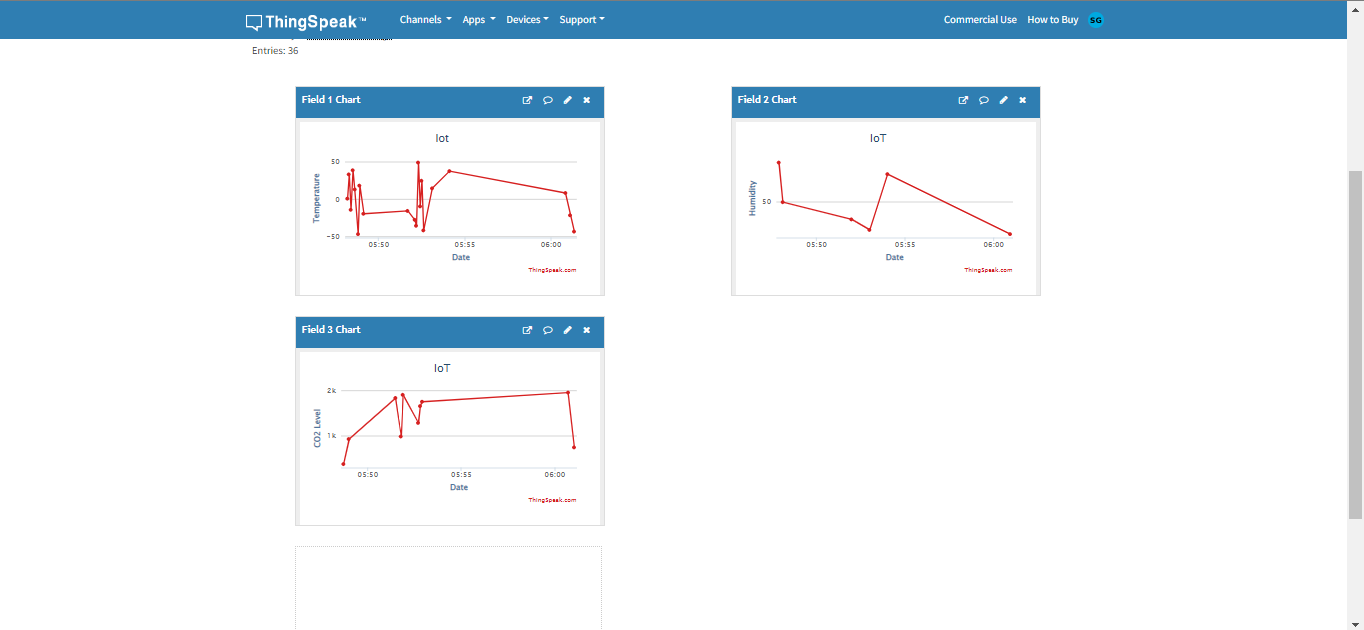
else

disp('No data found in the specified time range.');

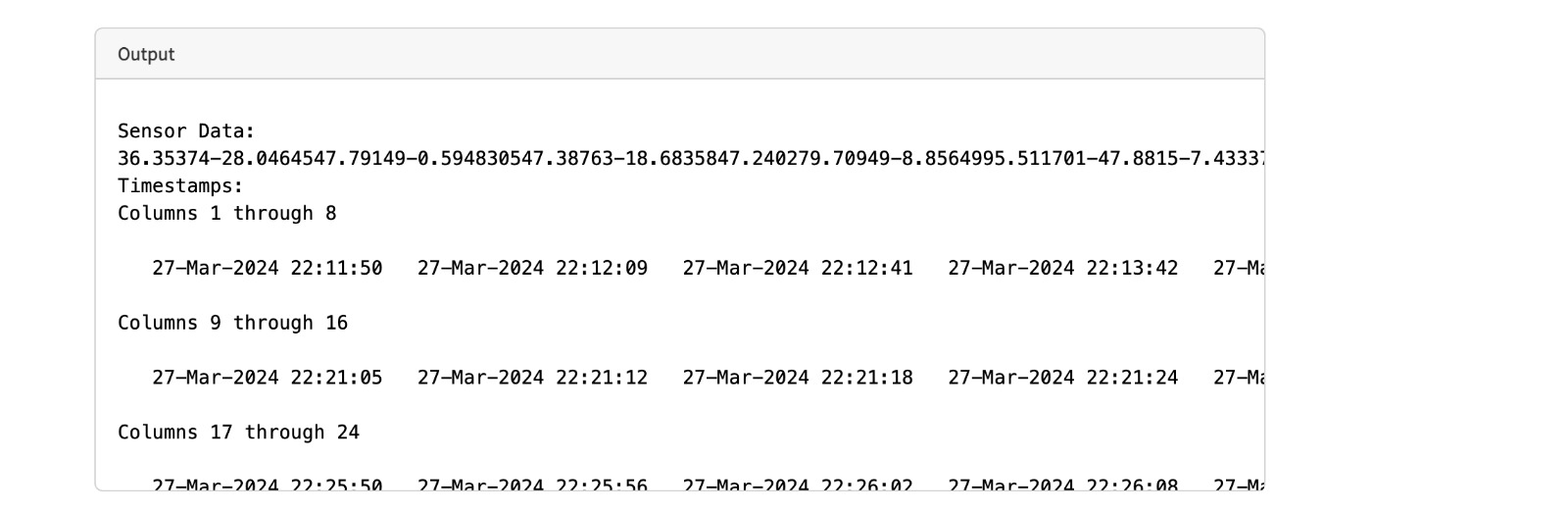
end

Thingspeak Screenshots





MATLAB SS



Wowki link : https://wokwi.com/projects/393468723481009153

Thingspeak : <https://thingspeak.com/channels/2486827/private_show>