**Project Collecting Data from Sensors connected with ESP32 and Accessing it through Docker**

**Objective**

The objective is to setup IOT system using ESP32 for collecting temperature , humidity and distance data from sensors and push it to Grafana Dashboard through MQTT and employ Docker for managing containers. The data transmission occurs to a server using a local hotspot for connectivity.

We have one microcontroller and 3 sensors

1. Temperature
2. Humidity
3. Sonar

# Requirements:

* ESP32 microcontroller
* Temperature and humidity sensor (e.g., DHT22)
* Ultrasonic sensor (e.g., HC-SR04)
* MQTT broker (e.g., Mosquitto)
* Grafana installation
* Docker installed on the Linux server
* Linux operating system
* Local hotspot for data transmission

1. **STEP : Installation**

First of all we have to install Arduino 1.8.19 . After that, we checked for board esp32 from

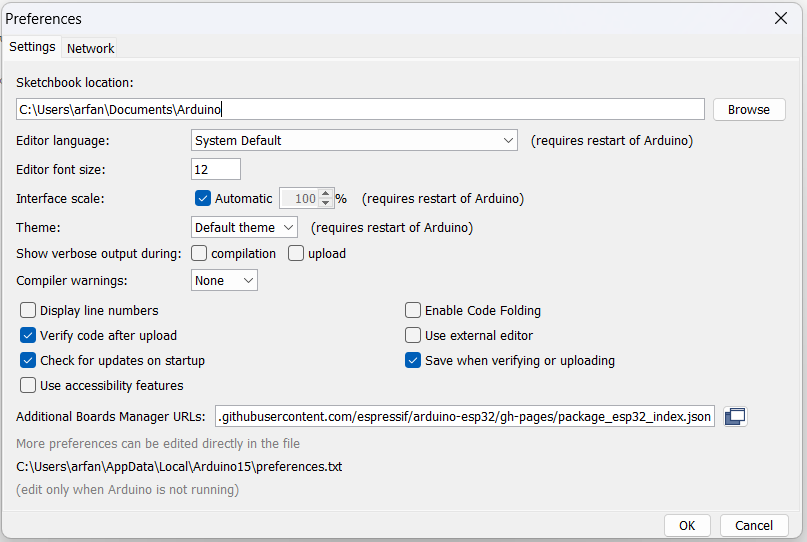
***Tools 🡪 Board Manager*** (if did not find ESP board esp32 install the following library)

**Installing Libraries**

First For installing esp32 board , Go to ***File 🡪 Preferences 🡪 Additional Board Manager URLs***

In preferences section Additional Board Manager URLs paste two URL’s separated by single comma.

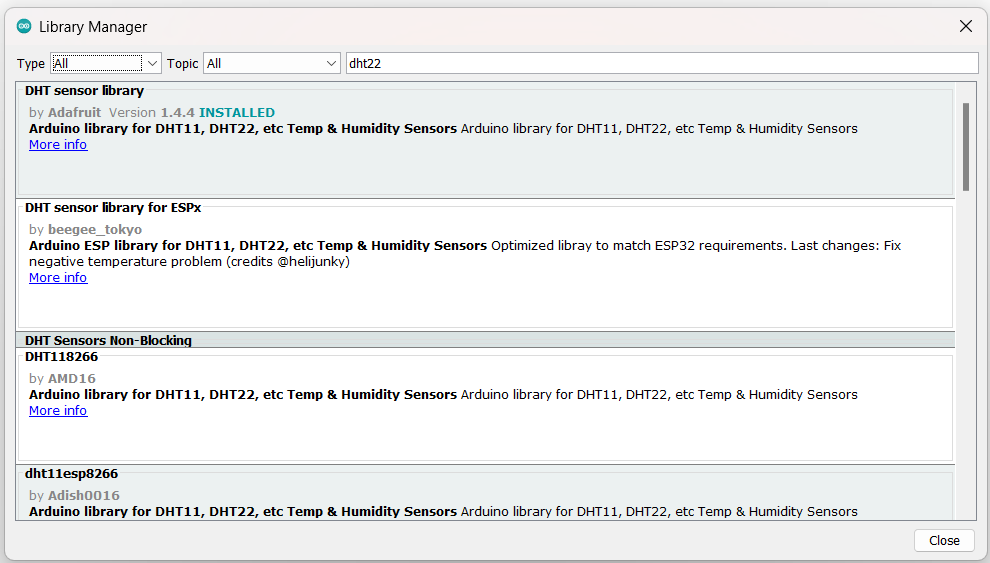
<http://arduino.esp8266.com/stable/package_esp8266com_index.json,https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json>



Now click ok. And go in board Manager (write esp32 ) and options will appear. Click install on **Esp32 by Expressif Systems** and install it by install all options .

1. **Install DHT22 Library (Sensor Library) Temperature and humidity Sensor Library**

From ***Tools -------🡪 Library Manager*** *type DHT22 ------🡪 install(all)*



Then

*Tools 🡪 Port 🡪 COM3 or COM4 (ETC)*

If port does not appear then attach microcontroller of GPU , and it will appear.

Next

Go to examples 🡺 wifi ------------🡪 wifiscan Arduino

**Next step :**

*Files 🡪 Examples 🡪 DHT Sensor Library 🡪 DHT Unified Sensor*

DHT unified sensor library has port 2 change it to port 4. (need to confirm )

We will also install Files( libraries ) lateron. After connecting sensors and Arduino with board.

**Hardware Setup:**

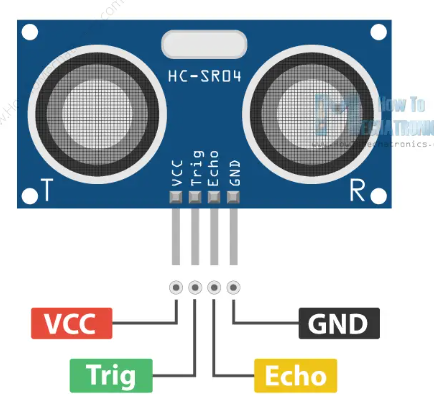
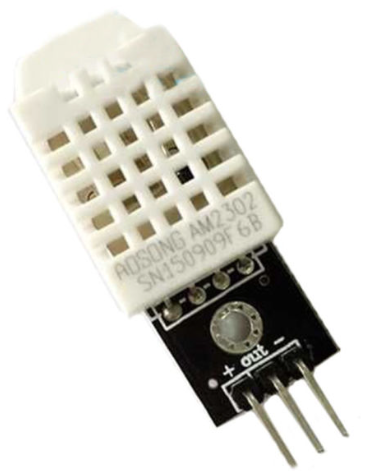
Connect the temperature and humidity sensor (DHT22) to the ESP32 following the datasheet instructions.

Connect the ultrasonic sensor (HC-SR04) to the ESP32, typically requiring two GPIO pins for trigger and echo.

Ensure proper power and ground connections for both sensors and the ESP32.

The link to the video is ---------------------------

We will connect 3 sensors (temperature , humidity ,Sonar ) with Arduino and board and Arduino with laptop.

Sonar Sensor Temperature Sensor

**Programming for ESP-32:**

# Import necessary libraries

import machine

import time

from dht import DHT22

import urequests as requests  # for HTTP requests to Grafana

# Configure DHT22 sensor

dht\_pin = machine.Pin(4)  # GPIO4

dht = DHT22(dht\_pin)

# Ultrasonic sensor configuration

trigger\_pin = machine.Pin(2)  # GPIO2

echo\_pin = machine.Pin(15)    # GPIO15

def read\_distance():

    trigger\_pin.value(0)

    time.sleep\_us(2)

    trigger\_pin.value(1)

    time.sleep\_us(10)

    trigger\_pin.value(0)

    while echo\_pin.value() == 0:

        pass

    t1 = time.ticks\_us()

    while echo\_pin.value() == 1:

        pass

    t2 = time.ticks\_us()

    return (t2 - t1) / 58

while True:

    try:

        # Read temperature and humidity

        dht.measure()

        temperature = dht.temperature()

        humidity = dht.humidity()

        # Read distance

        distance = read\_distance()

        # Send data to MQTT broker

        # Use an MQTT library suitable for MicroPython, e.g., umqtt.simple

        # Send data to Grafana via HTTP (Grafana Alerting)

        grafana\_url = "http://your\_grafana\_server:port/api/annotations"

        headers = {"Authorization": "Bearer your\_api\_token"}

        data = {

            "text": f"Temperature: {temperature}°C, Humidity: {humidity}%, Distance: {distance} cm",

            "tags": ["ESP32", "Sensor Data"]

        }

        response = requests.post(grafana\_url, json=data, headers=headers)

        response.close()

        time.sleep(300)  # Delay for 5 minutes

    except Exception as e:

        print("Error:", e)

**MQTT Broker Setup:**

* Install Mosquitto MQTT broker on Linux server.
* Configure Mosquitto to listen on a specific port (e.g., 1883).
* Create MQTT topics for temperature, humidity, and distance data

**Docker Containerization:**

* Set up four Docker containers:
* MQTT Broker container: Install and configure Mosquitto MQTT broker inside the container.
* Grafana container: Install Grafana inside the container and configure it to connect to the MQTT broker and InfluxDB database.
* InfluxDB container: Create a container with InfluxDB to store the data from MQTT.
* ESP32 Data Relay container: Run a Python or Node.js script inside this container to receive MQTT data from the ESP32 and push it into the InfluxDB container.

**Docker Compose:**

* Create a Docker Compose YAML file to define the four containers (MQTT Broker, Grafana, InfluxDB, ESP32 Data Relay).
* Specify the container configurations, dependencies, and environment variables in the YAML file.

**Container Deployment:**

* Deploy the containers on the Linux server using Docker Compose:

**docker-compose up -d**.

* Verify that all containers are running without errors:

**docker-compose ps**.

**Local Hotspot Setup:**

* Create a local hotspot on your Linux server or use an existing one to establish a Wi-Fi connection with the ESP32.
* Ensure the ESP32 is configured to connect to the hotspot.

**Testing and Monitoring:**

* Confirm that the ESP32 is publishing data to the MQTT broker via the local hotspot.
* Verify that data is being stored in the InfluxDB database.
* Access the Grafana dashboard through a web browser to visualize the sensor data.

By following these steps and using the provided Python code for the ESP32, we can set up a comprehensive IoT system that collects temperature, humidity, and distance data from sensors, pushes it to Grafana via MQTT, utilizes Docker for efficient container management, and transmits data to a Linux server via a local hotspot for remote monitoring and visualization.