Air Quality Monitoring

Phase 3: Development Part 1

Agenda

Hardware Setup

IoT Device Configuration

Data Collection Script Development

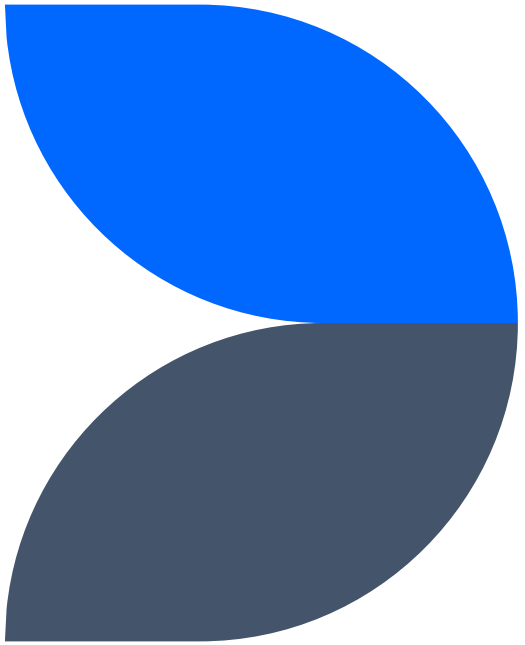
Real-Time Data Updates and Testing

Hardware Setup

ESP32 Microcontroller: The ESP32 is a versatile microcontroller with built-in Wi-Fi and Bluetooth capabilities, making it suitable for IoT applications. It will serve as the core component of your monitoring device.

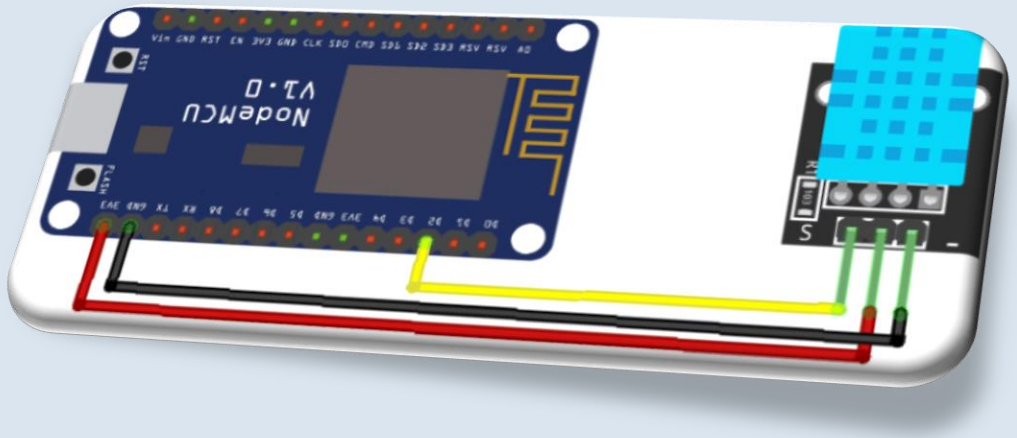
Air Quality Sensors: Connect air quality sensors such as DHT22 and gas sensors (e.g., CO, NO2, SO2, O3) to the ESP32. Ensure the sensors you choose are compatible with the ESP32's voltage and communication requirements.

IoT Device Configuration



Bill of Materials

S.N.	Components Name	Quantity
Q1	4.5	2.3
Q2	3.2	5.1
Q3	2.1	1.7
Q4	4.5	2.2



Development Environment Setup

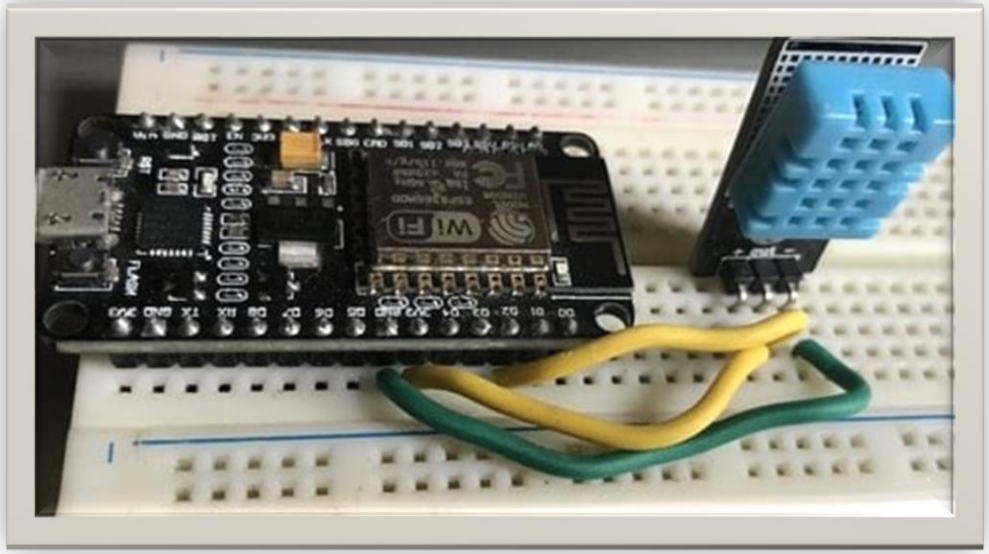
Install the necessary development tools and libraries on your computer to program the ESP32. You can use the Arduino IDE or the PlatformIO development platform for ESP32 development.

Programming

Write a program in the chosen development environment that reads data from the DHT22 sensor.

Use the appropriate libraries for the ESP32 to interact with the sensor.

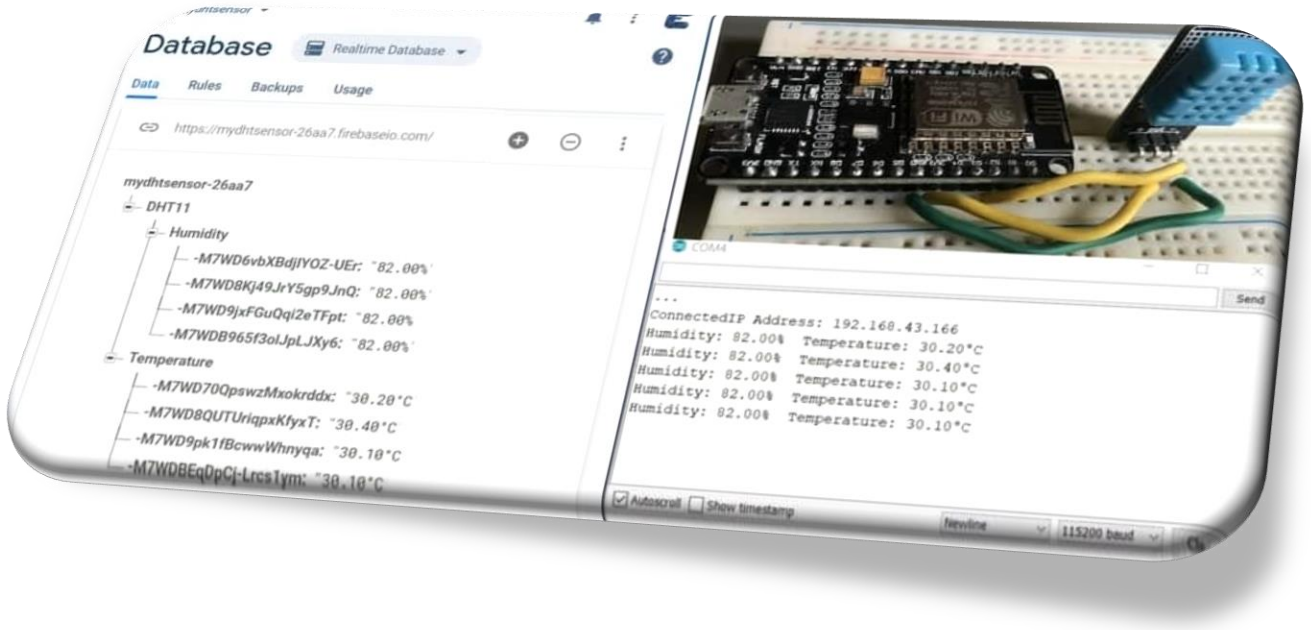
For the DHT22, you would typically use the Adafruit DHT library or similar libraries



```ino
#include <WiFi.h>
#include "DHTesp.h"
#include "ThingSpeak.h"

const int DHT_PIN = 15;
const int LED_PIN = 13;
const char* WIFI_NAME = "Wokwi-GUEST";
const char* WIFI_PASSWORD = "";
const int myChannelNumber =2304277 ;
const char* myApiKey = "6HPO49B53W7NB0EP";
const char* server = "api.thingspeak.com";

DHTesp dhtSensor;
WiFiClient client;

void setup() {
  Serial.begin(115200);
  dhtSensor.setup(DHT_PIN, DHTesp::DHT22);
  pinMode(LED_PIN, OUTPUT);
  WiFi.begin(WIFI_NAME, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED){
    delay(1000);
    Serial.println("Wifi not connected");
  }
  Serial.println("Wifi connected !");
  Serial.println("Local IP: " + String(WiFi.localIP()));
  WiFi.mode(WIFI_STA);
  ThingSpeak.begin(client);
}

void loop() {
  TempAndHumidity  data = dhtSensor.getTempAndHumidity();
  ThingSpeak.setField(1,data.temperature);
  ThingSpeak.setField(2,data.humidity);
  if (data.temperature > 35 || data.temperature < 12 || data.humidity > 70 || data.humidity < 40) {
    digitalWrite(LED_PIN, HIGH);
  }else{
    digitalWrite(LED_PIN, LOW);
  }

  int x = ThingSpeak.writeFields(myChannelNumber,myApiKey);

  Serial.println("Temp: " + String(data.temperature, 2) + "°C");
  Serial.println("Humidity: " + String(data.humidity, 1) + "%");

  if(x == 200){
    Serial.println("Data pushed successfull");
  }else{
    Serial.println("Push error" + String(x));
  }
  Serial.println("---");

  delay(100);
```

Data Collection Script Development

This data collection script is designed for an ESP32 microcontroller equipped with a DHT22 sensor. The script reads temperature and humidity data from the DHT22 sensor, formats the data, and sends it to a data-sharing platform via Wi-Fi. The ESP32 connects to a specified Wi-Fi network, collects sensor data at regular intervals, and transmits it for real-time monitoring and analysis. The script is designed to ensure accurate and reliable data collection, making it a vital component of the air quality monitoring project.



Send the real-time sensor data to Google Firebase Console using NodeMCU ESP8266 & DHT11 Sensor

Real-Time Data Updates and Testing

In this phase, the IoT device with its sensors and data collection script is tested rigorously to ensure accurate and reliable data collection. Data updates occur at predefined intervals to simulate real-time conditions, and the system is monitored for consistent performance. This phase is critical to verify the functionality and reliability of the hardware and software components in preparation for deployment and integration with the data-sharing platform

Send The Real-Time Data Updates

Thank you