ESP32 Sensor Project Guide

Connecting to Firebase and Visualizing Data on univtool

univtool Student Projects

Contents

1 Introduction

This guide will walk you through setting up your ESP32 with various sensors and connecting it to the univtool Firebase platform. By the end, you'll have a live dashboard displaying your sensor data in real-time [citation:8].

2 What You'll Need

2.1 Hardware Requirements

- ESP32 development board (ESP32 DevKitC recommended)
- USB cable for programming
- Sensors of your choice (common options):
  - **DHT22** - Temperature and humidity [citation:2][citation:4]
  - **HC-SR04** - Ultrasonic distance sensor
  - **LDR (photoresistor)** - Light intensity
  - **Soil moisture sensor** - Capacitive or resistive type
  - **MQ series** - Gas sensors (MQ2, MQ135)
  - **LM35** - Analog temperature sensor [citation:10]
- Breadboard and jumper wires
- Wi-Fi connection

2.2 Software Requirements

- Arduino IDE (download from `https://www.arduino.cc/en/software`)
- Firebase ESP Client library
- Sensor-specific libraries (DHT sensor library, etc.)
- A Google account (for Firebase access)

3 Credentials from Your Admin

Your admin will provide you with the following information. Keep them safe:

Item	Description
Firebase API Key	Your project's unique API key for authentication
Database URL	The URL of your Firebase Realtime Database (e.g., `https://your-project-default-rtdb.firebaseio.com/`)
Your Project ID	Your unique identifier (e.g., `student123`) – this is your Firebase UID
Service Account Key	JSON file for secure server-to-server communication (optional, for advanced use)
Login Credentials	Email and password to access your project dashboard on univtool.com/projects/

4 Setting Up Your Development Environment

4.1 Install Arduino IDE

1. Download and install Arduino IDE from the official website
2. Open Arduino IDE after installation

4.2 Add ESP32 Board Support

1. Go to **File** → **Preferences**
2. In the "Additional Board Manager URLs" field, add:

```
https://dl.espressif.com/dl/package_esp32_index.json
```

3. Click **OK**
4. Go to **Tools** → **Board** → **Boards Manager**
5. Search for "ESP32" and install the package by Espressif Systems

4.3 Install Required Libraries

Go to **Sketch** → **Include Library** → **Manage Libraries** and install:

- **Firebase ESP Client** by Mobitz (search "firebase") [citation:1]
- **DHT sensor library** by Adafruit (for DHT22)
- **Adafruit Unified Sensor** (required for DHT)
- **LiquidCrystal I2C** (if using LCD display)
- **ArduinoJson** by Benoit Blanchon

5 Connecting Your Sensors

5.1 DHT22 Temperature and Humidity Sensor

Connect as follows [citation:2][citation:4]:

DHT22 Pin	ESP32 Pin	Notes
VCC	3.3V	Power
DATA	GPIO4	Digital pin (can be changed in code)
GND	GND	Ground

Add a 10kOhms pull-up resistor between VCC and DATA pins.

5.2 HC-SR04 Ultrasonic Distance Sensor

HC-SR04 Pin	ESP32 Pin	Notes
VCC	5V	Power
TRIG	GPIO5	Trigger pin
ECHO	GPIO18	Echo pin

GND	GND	Ground

5.3 LDR (Light Dependent Resistor)

Use a voltage divider circuit:

- LDR in series with a 10kohms resistor between 3.3V and GND
- Connect the junction between LDR and resistor to GPIO34 (analog input)

5.4 Soil Moisture Sensor (Capacitive)

Sensor Pin	**ESP32 Pin**
VCC	3.3V
GND	GND
AOUT	GPIO35 (analog input)

6 Programming Your ESP32

6.1 Complete Code Template

Create a new sketch in Arduino IDE and paste the following code. Replace the placeholders with your actual credentials [citation:1][citation:6]:

```
#include <WiFi.h>
#include <Firebase_ESP_Client.h>
#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"

// Sensor libraries
#include <DHT.h>

// ===== REPLACE WITH YOUR CREDENTIALS =====
#define WIFI_SSID "YOUR_WIFI_SSID"
#define WIFI_PASSWORD "YOUR_WIFI_PASSWORD"
#define API_KEY "YOUR_FIREBASE_API_KEY"
#define DATABASE_URL "YOUR_FIREBASE_DATABASE_URL"
#define USER_EMAIL "your-login-email@example.com"
#define USER_PASSWORD "your-login-password"
// ==========================================

// Your unique project ID from Firebase (UID)
String projectID = "YOUR_PROJECT_ID";

// Define sensors
#define DHTPIN 4
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

#define TRIG_PIN 5
#define ECHO_PIN 18
#define LDR_PIN 34
#define SOIL_PIN 35

// Firebase objects
```

```
FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;
bool signupOK = false;

void setup() {
  Serial.begin(115200);

  // Initialize sensors
  dht.begin();
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(LDR_PIN, INPUT);
  pinMode(SOIL_PIN, INPUT);

  // Connect to WiFi
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());

  // Configure Firebase
  config.api_key = API_KEY;
  config.database_url = DATABASE_URL;

  // Sign up with email/password
  if (Firebase.signUp(&config, &auth, USER_EMAIL, USER_PASSWORD)) {
    Serial.println("Firebase signup OK");
    signupOK = true;
  } else {
    Serial.printf("Signup error: %s\n", config.signer.signupError.
        message.c_str());
  }

  config.token_status_callback = tokenStatusCallback;
  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
}

float readDistance() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  long duration = pulseIn(ECHO_PIN, HIGH);
  float distance = duration * 0.034 / 2;
  return distance;
}

```

```
void loop() {
  if (Firebase.ready() && signupOK &&
      (millis() - sendDataPrevMillis > 10000 || sendDataPrevMillis == 0)
         ) {
    sendDataPrevMillis = millis();

    // Read all sensors
    float temperature = dht.readTemperature();
    float humidity = dht.readHumidity();
    float distance = readDistance();
    int ldrValue = analogRead(LDR_PIN);
    int soilValue = analogRead(SOIL_PIN);

    // Get timestamp
    unsigned long timestamp = millis();

    // Send to Firebase - each reading as a separate entry with
       timestamp
    String basePath = "projects/" + projectID + "/data/";

    if (!isnan(temperature)) {
      String path = basePath + String(timestamp) + "/temperature";
      Firebase.RTDB.setFloat(&fbdo, path.c_str(), temperature);
    }

    if (!isnan(humidity)) {
      String path = basePath + String(timestamp) + "/humidity";
      Firebase.RTDB.setFloat(&fbdo, path.c_str(), humidity);
    }

    String pathDist = basePath + String(timestamp) + "/distance";
    Firebase.RTDB.setFloat(&fbdo, pathDist.c_str(), distance);

    String pathLDR = basePath + String(timestamp) + "/light";
    Firebase.RTDB.setInt(&fbdo, pathLDR.c_str(), ldrValue);

    String pathSoil = basePath + String(timestamp) + "/soil_moisture";
    Firebase.RTDB.setInt(&fbdo, pathSoil.c_str(), soilValue);

    // Also update latest values for quick access
    Firebase.RTDB.setFloat(&fbdo, ("projects/" + projectID + "/latest/
       temperature").c_str(), temperature);
    Firebase.RTDB.setFloat(&fbdo, ("projects/" + projectID + "/latest/
       humidity").c_str(), humidity);
    Firebase.RTDB.setFloat(&fbdo, ("projects/" + projectID + "/latest/
       distance").c_str(), distance);
    Firebase.RTDB.setInt(&fbdo, ("projects/" + projectID + "/latest/
       light").c_str(), ldrValue);
    Firebase.RTDB.setInt(&fbdo, ("projects/" + projectID + "/latest/
       soil_moisture").c_str(), soilValue);

    Serial.println("Data␣sent␣to␣Firebase");
  }
}
```

6.2 Understanding the Code

- **WiFi Connection:** The ESP32 connects to your local network [citation:1]
- **Firebase Authentication:** Uses email/password to authenticate [citation:6]
- **Data Structure:** Each reading gets a timestamp-based key, making it easy to query history
- **Live Updates:** The "latest" node stores the most recent values for quick dashboard access
- **Sensor Reading:** Each sensor is read every 10 seconds (adjustable)

7 Testing Your Setup

1. Connect your ESP32 to your computer via USB
2. Select the correct board: **Tools** → **Board** → **ESP32 Dev Module**
3. Select the correct COM port
4. Click the Upload button (right arrow)
5. Open Serial Monitor (**Tools** → **Serial Monitor**) set to 115200 baud
6. You should see connection messages and "Data sent to Firebase" every 10 seconds

8 Viewing Your Data on univtool

8.1 Logging In

1. Go to `https://univtool.com/projects/`
2. Enter the email and password provided by your admin
3. Complete the CAPTCHA verification
4. Click "Login"

8.2 Navigating Your Dashboard

After login, you'll see your project card with:

- **Live Sensor Cards:** Current temperature, humidity, etc. [citation:8]
- **Distance Map:** A visual representation with a moving dot
- **Chart:** Temperature and humidity history graph
- **Data Table:** Recent readings with timestamps
- **Control Buttons:** Stop/start updates, refresh, view historical data

8.3 Interpreting the Map

The blue dot on the 400×400 canvas moves horizontally based on your distance sensor reading. Five horizontal lines help visualize position changes.

8.4 Historical Data

Click "Historical Data" to load more past readings. The table shows the most recent entries, with the newest at the top.

9 Troubleshooting

Problem	Solution
WiFi connection fails	Check SSID and password; ensure ESP32 is within range
Firebase authentication fails	Verify API key, database URL, and login credentials [citation:1]
Sensor readings show NaN	Check sensor wiring; verify library installations
No data on dashboard	Check Serial Monitor for errors; verify project ID matches
Data not updating	Click "Start Updates" if stopped; check internet connection
Distance sensor reads 0	Check trigger/echo pins; ensure 5V power [citation:4]

10 Advanced Features

10.1 Adding More Sensors

You can extend the code to include:

- **MQ2 Gas Sensor:** For air quality monitoring [citation:2]
- **BMP280:** Barometric pressure
- **PIR Motion Sensor:** Motion detection
- **Relay Module:** Control appliances remotely

10.2 Data Logging Interval

Adjust the delay in ‘loop()‘ to change how often data is sent. Be mindful of Firebase usage limits.

10.3 Threshold Alerts

Modify the code to trigger actions when values exceed thresholds (e.g., buzzer when temperature too high) [citation:10].

11 Next Steps

1. Experiment with different sensors
2. Create multiple projects for different locations
3. Build a physical enclosure for your sensors
4. Share your dashboard with classmates
5. Extend the web dashboard with custom visualizations

12 Resources

- Firebase Documentation: `https://firebase.google.com/docs`
- ESP32 Firebase Library: `https://github.com/mobizt/Firebase-ESP-Client`
- Random Nerd Tutorials: `https://randomnerdtutorials.com` [citation:8]
- univtool Support: `mailto:projects@univtool.com`