Lab11: Firebase-1

simplified Arduino code for ESP32-S3 to send DHT11 data to Firebase, along with complete **Firebase setup instructions**:

Simple Arduino Code (ESP32-S3)

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <DHT.h>

// WiFi Credentials
const char* ssid = "";
const char* password = "";

// Firebase Configuration
const String FIREBASE_HOST = "";
const String FIREBASE_AUTH = "";
const String FIREBASE_PATH = "/sensor_data.json";

// DHT Sensor
#define DHTPIN 4       // GPIO4 (change if needed)
#define DHTTYPE DHT11  // DHT11 or DHT22

// Timing
const unsigned long SEND_INTERVAL = 10000;  // 10 seconds
const unsigned long SENSOR_DELAY = 2000;    // 2 seconds between reads

// ======= Global Objects ======= //
DHT dht(DHTPIN, DHTTYPE);
unsigned long lastSendTime = 0;
unsigned long lastReadTime = 0;

// ======= Setup ======= //
void setup() {
  Serial.begin(115200);
  Serial.println("\nESP32-S3 DHT11 Firebase Monitor");

  initDHT();
  connectWiFi();
}

// ======= Main Loop ======= //
void loop() {
  // Maintain WiFi connection
  if (WiFi.status() != WL_CONNECTED) {
    connectWiFi();
  }

  // Read sensor (with proper timing)
  if (millis() - lastReadTime >= SENSOR_DELAY) {
    float temp, hum;
    if (readDHT(&temp, &hum)) {
      // Send to Firebase (with proper timing)
      if (millis() - lastSendTime >= SEND_INTERVAL) {
        sendToFirebase(temp, hum);
        lastSendTime = millis();
      }
    }
    lastReadTime = millis();
  }
}

// ======= Sensor Functions ======= //
void initDHT() {
  dht.begin();
  Serial.println("DHT sensor initialized");
  delay(500);  // Short stabilization delay
}

bool readDHT(float* temp, float* humidity) {
  *temp = dht.readTemperature();
  *humidity = dht.readHumidity();

  if (isnan(*temp) || isnan(*humidity)) {
    Serial.println("DHT read failed! Retrying...");

    // Attempt sensor recovery
    digitalWrite(DHTPIN, LOW);  // Reset pin state
    pinMode(DHTPIN, INPUT);
    delay(100);
    initDHT();  // Reinitialize

    return false;
  }

  Serial.printf("DHT Read: %.1f°C, %.1f%%\n", *temp, *humidity);
  return true;
}
```

```
// ======= WiFi Functions ======= //
void connectWiFi() {
  Serial.print("Connecting to WiFi");
  WiFi.disconnect(true);  // Clear previous config
  WiFi.begin(ssid, password);

  int attempts = 0;
  while (WiFi.status() != WL_CONNECTED && attempts < 15) {
    delay(500);
    Serial.print(".");
    attempts++;
  }

  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("\nWiFi Connected!");
    Serial.print("IP Address: ");
    Serial.println(WiFi.localIP());
  } else {
    Serial.println("\nWiFi Connection Failed!");
  }
}

// ======= Firebase Functions ======= //
void sendToFirebase(float temp, float humidity) {
  if (WiFi.status() != WL_CONNECTED) {
    Serial.println("Cannot send - WiFi disconnected");
    return;
  }

  HTTPClient http;
  String url = "https://" + FIREBASE_HOST + FIREBASE_PATH + "?auth=" +
FIREBASE_AUTH;

  // Create JSON payload
  String jsonPayload = "{\"temperature\":" + String(temp) +
                      ",\"humidity\":" + String(humidity) +
                      ",\"timestamp\":" + String(millis()/1000) + "}";

  Serial.println("Sending to Firebase...");
  Serial.println(jsonPayload);

  http.begin(url);
  http.addHeader("Content-Type", "application/json");
  int httpCode = http.POST(jsonPayload);

  if (httpCode == HTTP_CODE_OK) {
    Serial.println("Firebase update successful");
  } else {
    Serial.printf("Firebase error: %d\n", httpCode);
    if (httpCode == -1) {
      Serial.println("Check your Firebase URL and authentication");
    }
  }

  http.end();
}
```

Complete Firebase Setup Guide

Step 1: Create Firebase Project

1. Go to Firebase Console
2. Click "Add project" → Enter project name → Continue
3. Disable Google Analytics (for simplicity) → Create project

Step 2: Set Up Realtime Database

1. In left menu, go to "Realtime Database"
2. Click "Create Database" → Start in test mode → Enable
3. Note your database URL (format: `your-project-id.firebaseio.com`)

Step 3: Get Database Secret

1. Click the gear icon → Project settings
2. Go to "Service accounts" tab
3. Under "Database secrets", click "Show" (copy this secret key)

Step 4: Database Rules Configuration

1. Go to "Realtime Database" → Rules tab
2. Replace with these temporary rules (for testing only):

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

3. Click "Publish"

Security Note: These rules allow open access. For production, implement proper authentication.

Step 5: Arduino Code Configuration

Replace these placeholders in the code:

- `YOUR_WIFI_SSID`: Your WiFi network name
- `YOUR_WIFI_PASSWORD`: Your WiFi password
- `YOUR-PROJECT-ID`: From Firebase database URL
- `YOUR-DATABASE-SECRET`: From Firebase project settings

Step 6: Required Libraries

Install these via Arduino Library Manager:

1. **DHT sensor library** by Adafruit
2. **HTTPClient** (comes with ESP32 board package)
3. **WiFi** (comes with ESP32 board package)

Step 7: Upload and Monitor

1. Connect ESP32-S3 via USB
2. Select board: "ESP32S3 Dev Module"
3. Upload the code
4. Open Serial Monitor (115200 baud) to check status

Verification:

1. In Firebase Console, go to Realtime Database
2. You should see new data appearing every 10 seconds:

```
{
  "sensor_Data": {
    "-Nxxxxxxxxxxxx": {
      "temperature": 25.5,
      "humidity": 60.2
    }
  }
}
```

Here's a detailed breakdown of the major components of your ESP32-S3 DHT11 to Firebase code:

1. **Initialization (Setup)**

```
void setup() {
  Serial.begin(115200);
  Serial.println("\nESP32-S3 DHT11 Firebase Monitor");
  initDHT();
  connectWiFi();
}
```

- **Serial Communication**: Starts serial monitor at 115200 baud for debugging
- **DHT Initialization**: Calls `initDHT()` to start the temperature/humidity sensor
- **WiFi Connection**: Calls `connectWiFi()` to establish internet connection

2. **Main Program Loop**

```
void loop() {
  // WiFi maintenance
  if (WiFi.status() != WL_CONNECTED) {
    connectWiFi();
  }

  // Sensor reading logic
  if (millis() - lastReadTime >= SENSOR_DELAY) {
    float temp, hum;
    if (readDHT(&temp, &hum)) {
      if (millis() - lastSendTime >= SEND_INTERVAL) {
        sendToFirebase(temp, hum);
        lastSendTime = millis();
      }
    }
    lastReadTime = millis();
  }
}
```

- **WiFi Monitoring**: Continuously checks and maintains WiFi connection
- **Timed Sensor Reading**:
  - Reads sensor every 2 seconds (`SENSOR_DELAY`)
  - Only sends to Firebase every 10 seconds (`SEND_INTERVAL`)
- **Timing Control**: Uses `millis()` for non-blocking delays

3. **DHT Sensor Functions**

```
void initDHT() {
  dht.begin();
  Serial.println("DHT sensor initialized");
  delay(500);
}

bool readDHT(float* temp, float* humidity) {
  *temp = dht.readTemperature();
  *humidity = dht.readHumidity();

  if (isnan(*temp) || isnan(*humidity)) {
    // Hardware reset procedure
    digitalWrite(DHTPIN, LOW);
    pinMode(DHTPIN, INPUT);
    delay(100);
    initDHT();
    return false;
  }

  Serial.printf("DHT Read: %.1f°C, %.1f%%\n", *temp, *humidity);
  return true;
}
```

- **Initialization**: Starts the DHT sensor with begin()
- **Reading Logic**:
    - Gets temperature and humidity values
    - Includes hardware reset procedure if read fails
    - Prints readings to serial monitor
- **Error Handling**: Returns false if reading fails

4. **WiFi Connection**

```
void connectWiFi() {
  Serial.print("Connecting to WiFi");
  WiFi.disconnect(true);
  WiFi.begin(ssid, password);

  int attempts = 0;
  while (WiFi.status() != WL_CONNECTED && attempts < 15) {
    delay(500);
    Serial.print(".");
    attempts++;
  }

  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("\nWiFi Connected!");
    Serial.print("IP Address: ");
    Serial.println(WiFi.localIP());
  } else {
    Serial.println("\nWiFi Connection Failed!");
  }
}
```

- **Connection Process**:
    - Disconnects any existing connection
    - Attempts to connect with credentials
    - Shows progress dots during connection
- **Timeout Handling**: Gives up after 15 attempts (7.5 seconds)
- **Status Feedback**: Prints success/failure and IP address

5. **Firebase Integration**

```
void sendToFirebase(float temp, float humidity) {
  if (WiFi.status() != WL_CONNECTED) return;

  HTTPClient http;
  String url = "https://" + FIREBASE_HOST + FIREBASE_PATH + "?auth=" + FIREBA
SE_AUTH;

  String jsonPayload = "{\"temperature\":" + String(temp) +
                      ",\"humidity\":" + String(humidity) +
                      ",\"timestamp\":" + String(millis()/1000) + "}";

  http.begin(url);
  http.addHeader("Content-Type", "application/json");

  int httpCode = http.POST(jsonPayload);

  if (httpCode == HTTP_CODE_OK) {
    Serial.println("Firebase update successful");
  } else {
    Serial.printf("Firebase error: %d\n", httpCode);
  }

  http.end();
}
```

- **Data Packaging**:
    - Creates JSON payload with sensor data
    - Includes uptime-based timestamp
- **HTTP Request**:
    - Uses HTTPS connection
    - Sets proper content-type header
- **Error Handling**:
    - Verifies WiFi before sending
    - Reports HTTP status codes
- **Resource Management**: Properly closes HTTP connection

Key Timing Mechanisms:

1. **Sensor Read Delay** (2 seconds):
    - Ensures DHT11 isn't polled too frequently (min 2s required)
    - `if (millis() - lastReadTime >= SENSOR_DELAY)`
2. **Firebase Send Interval** (10 seconds):
    - Limits data transmission to prevent flooding
    - `if (millis() - lastSendTime >= SEND_INTERVAL)`
3. **Uptime Timestamp**:
    - `millis()/1000` converts milliseconds to seconds since boot

Error Recovery Features:

- Automatic WiFi reconnection
- DHT sensor hardware reset on failure
- HTTP error code reporting
- Serial monitor feedback at every stage

Data Flow:

1. Initialize hardware → 2. Connect WiFi → 3. Read sensor → 4. Package data → 5. Send to Firebase → (Repeat)

Students Task:

Change the simulated, virtual timestamp with real timestamp according to Pakistan standard time.(NTP)