LAMPIRAN

Lampiran 1 Kode Server ESP32

```
#include <WiFi.h>
#include <WebServer.h>
#include <HTTPClient.h>
#include <WiFiClient.h>
#include <WiFiClientSecure.h>
#include <PubSubClient.h>
#include <UniversalTelegramBot.h>

const char* AP_SSID = "ESP32_IoT_Hub";
const char* AP_PASS = "iot12345";
const char* WIFI_SSID = "RedmiNote12";
const char* WIFI_PASS = "wahminta";

const char* MQTT_SERVER = "broker.hivemq.com";
const int MQTT_PORT = 1883;

const char* TOPIC_GERAK = "Kamar1/sensor/bayi/wemos1/gerak";
const char* TOPIC_SUARA = "Kamar1/sensor/bayi/wemos1/suara";
const char* TOPIC_SUHU  = "Kamar1/sensor/bayi/wemos2/suhu";
const char* TOPIC_ASAP  = "Kamar1/sensor/bayi/wemos2/asap";

#define BOT_TOKEN
"8773950442:AAGSlDQ3YI049BWe1A4wRO0TNiV4V51GwLg"
#define CHAT_ID "7456554554"

WebServer server(80);

WiFiClient espClient;

PubSubClient mqttClient(espClient);

WiFiClientSecure secureClient;

UniversalTelegramBot bot(BOT_TOKEN, secureClient);

bool gerak = false;
bool suara = false;
bool asap = false;

float suhu = 0.0;

bool lastAsap = false;
```

```
unsigned long lastDataWemos1 = 0;
unsigned long lastDataWemos2 = 0;

const unsigned long NODE_TIMEOUT = 15000;

bool wemos1Online = false;
bool wemos2Online = false;
bool smartMode = false;
bool telegramSent = false;

unsigned long smartStart = 0;

const unsigned long SMART_DELAY = 1000;

String babyStatus = "Normal";

unsigned long lastReconnect = 0;
const unsigned long reconnectInterval = 5000;

unsigned long lastTelegramCheck = 0;
const unsigned long telegramCheckInterval = 1500;

void connectWiFi();
void connectMQTT();
void publishMQTT();
void handleTelegram();
void sendMenu(String chatID);
void sendStatusSuhu(String chatID);
void handleData();
void checkNodeStatus();
void evaluateBabyStatus();

void connectWiFi()
{
  Serial.println();
  Serial.println("Menghubungkan ke WiFi Internet...");

  WiFi.begin(WIFI_SSID, WIFI_PASS);


  unsigned long startAttempt = millis();
  while (WiFi.status() != WL_CONNECTED && millis() - startAttempt <
15000)
  {
    delay(500);
    Serial.print(".");
  }
```

```
    Serial.println();

    if (WiFi.status() == WL_CONNECTED)
    {
        Serial.println("WiFi Internet Connected");
        Serial.print("IP STA : ");
        Serial.println(WiFi.localIP());
    }
    else
    {
        Serial.println("WiFi Internet GAGAL terhubung, lanjut jalan pakai AP
saja");
    }
}

void connectMQTT()
{

    int attempt = 0;

    while (!mqttClient.connected() && attempt < 3)
    {
        Serial.print("Menghubungkan MQTT...");

        String clientID = "ESP32_Server_";
        clientID += String(random(0xffff), HEX);

        if (mqttClient.connect(clientID.c_str()))
        {
            Serial.println(" Berhasil");
        }
        else
        {
            Serial.print(" Gagal rc=");
            Serial.println(mqttClient.state());

            delay(1000);
        }

        attempt++;
    }
}


void publishMQTT()
{
    if (!mqttClient.connected()) return; //
```

```
    mqttClient.publish(TOPIC_GERAK, gerak ? "1" : "0");
    mqttClient.publish(TOPIC_SUARA, suara ? "1" : "0");
    mqttClient.publish(TOPIC_SUHU,  String(suhu, 1).c_str());
    mqttClient.publish(TOPIC_ASAP,  asap ? "1" : "0");
}

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

    secureClient.setInsecure();

    WiFi.mode(WIFI_AP_STA);

    WiFi.softAP(AP_SSID, AP_PASS);

    Serial.println();
    Serial.println("=====================================");
    Serial.println(" SMART HOME BABY MONITORING SERVER");
    Serial.println("=====================================");

    Serial.print("AP IP : ");
    Serial.println(WiFi.softAPIP());

    connectWiFi();

    mqttClient.setServer(MQTT_SERVER, MQTT_PORT);
    connectMQTT();

    server.on("/data", HTTP_GET, handleData);
    server.begin();

    Serial.println("HTTP Server Ready");

    if (WiFi.status() == WL_CONNECTED)
    {
        bot.sendMessage(CHAT_ID, "✅ ESP32 Smart Baby Monitoring
Online", "");
    }
}

void handleData()
{

    String node = server.hasArg("node") ? server.arg("node") : "";

    bool gotGerakSuara = server.hasArg("gerak") || server.hasArg("suara");
```

```
bool gotSuhuAsap    = server.hasArg("suhu")  || server.hasArg("asap");

if (server.hasArg("gerak"))
{
   gerak = server.arg("gerak").toInt();
}

if (server.hasArg("suara"))
{
   suara = server.arg("suara").toInt();
}

if (server.hasArg("suhu"))
{
   suhu = server.arg("suhu").toFloat();
}

if (server.hasArg("asap"))
{
   asap = server.arg("asap").toInt();
}

if (node == "1" || gotGerakSuara)
{
   lastDataWemos1 = millis();
   wemos1Online = true;
}

if (node == "2" || gotSuhuAsap)
{
   lastDataWemos2 = millis();
   wemos2Online = true;
}

publishMQTT();
evaluateBabyStatus();

// Sensor Asap
if (asap && !lastAsap)
{
   bot.sendMessage(
      CHAT_ID,
      " 🔥 PERINGATAN!\n\nAsap terdeteksi di sekitar bayi.",
      ""
   );
}

lastAsap = asap;
```

```
    Serial.println("--------------------------------");
    Serial.print("Gerak : "); Serial.println(gerak);
    Serial.print("Suara : "); Serial.println(suara);
    Serial.print("Suhu : ");  Serial.println(suhu);
    Serial.print("Asap : ");  Serial.println(asap);
    Serial.print("Status : "); Serial.println(babyStatus);
    Serial.println("--------------------------------");

    server.send(200, "text/plain", "OK");
}


void evaluateBabyStatus()
{
    if (gerak && suara)
    {
        if (!smartMode)
        {
            smartMode = true;
            smartStart = millis();
            telegramSent = false;

            Serial.println("Smart Detection Started");
        }

        if ((millis() - smartStart) >= SMART_DELAY)
        {
            babyStatus = "Kemungkinan Bayi Menangis";

            if (!telegramSent)
            {
                bot.sendMessage(
                    CHAT_ID,
                    "😭 Kemungkinan bayi menangis.\n"
                    "Gerakan dan suara terdeteksi selama lebih dari 3 detik.\n"
                    "Silakan lakukan pengecekan.",
                    ""
                );

                telegramSent = true;
                Serial.println("Telegram Terkirim");
            }
        }
    }
    else if (gerak)
    {
        smartMode = false;
```

```
        telegramSent = false;
        babyStatus = "Bayi Bergerak";
    }
    else
    {
        smartMode = false;
        telegramSent = false;
        babyStatus = "Bayi Tidur";
    }
}

void checkNodeStatus()
{
    if (wemos1Online && millis() - lastDataWemos1 > NODE_TIMEOUT)
    {
        wemos1Online = false;
        gerak = false;
        suara = false;

        Serial.println("[WARNING] Wemos 1 (gerak/suara) tidak mengirim data,
dianggap offline");
        evaluateBabyStatus();
    }

    if (wemos2Online && millis() - lastDataWemos2 > NODE_TIMEOUT)
    {
        wemos2Online = false;

        Serial.println("[WARNING] Wemos 2 (suhu/asap) tidak mengirim data,
dianggap offline");
    }
}

void sendMenu(String chatID)
{
    String pesan =
        "👶 SMART BABY MONITORING\n\n"
        "Silakan pilih menu di bawah.";

    String keyboard =
        "[[{\"text\":\" 🌡️ STATUS SUHU\",\"callback_data\":\"SUHU\"}]]";

    bot.sendMessageWithInlineKeyboard(chatID, pesan, "", keyboard);
}

void sendStatusSuhu(String chatID)
{
    String pesan;
```

```
    pesan += " 🌡️ STATUS SUHU\n\n";
    pesan += " 🌡️ Suhu Kamar Bayi : ";
    pesan += String(suhu, 1);
    pesan += " °C";

    if (!wemos2Online)
    {
      pesan += "\n\n ⚠️ Data mungkin tidak update (node sensor suhu offline)";
    }

    bot.sendMessage(chatID, pesan, "");
}

void handleTelegram()
{

    int jumlahPesan = bot.getUpdates(bot.last_message_received + 1);

    for (int i = 0; i < jumlahPesan; i++)
    {
      String chat_id = bot.messages[i].chat_id;
      String text = bot.messages[i].text;

      if (text == "/start")
      {
        sendMenu(chat_id);
      }

      if (text == " 🌡️ STATUS SUHU")
      {
        sendStatusSuhu(chat_id);
      }

      if (bot.messages[i].type == "callback_query")
      {
        if (bot.messages[i].text == "SUHU")
        {
          sendStatusSuhu(chat_id);
        }
      }
    }
}

void loop()
{
    // Reconnect WiFi
```

```
  if (WiFi.status() != WL_CONNECTED)
  {
    if (millis() - lastReconnect >= reconnectInterval)
    {
      Serial.println("Reconnect WiFi...");
      connectWiFi();
      lastReconnect = millis();
    }
  }

  // Reconnect MQTT
  if (!mqttClient.connected())
  {
    connectMQTT();
  }

  mqttClient.loop();

  server.handleClient();

  if (WiFi.status() == WL_CONNECTED &&
    millis() - lastTelegramCheck >= telegramCheckInterval)
  {
    handleTelegram();
    lastTelegramCheck = millis();
  }

  checkNodeStatus();

  static unsigned long lastPrint = 0;

  if (millis() - lastPrint >= 5000)
  {
    Serial.println();
    Serial.println("====================================");
    Serial.print("Status Bayi : "); Serial.println(babyStatus);
    Serial.print("Gerak : "); Serial.println(gerak ? "YA" : "TIDAK");
    Serial.print("Suara : "); Serial.println(suara ? "YA" : "TIDAK");
    Serial.print("Suhu : "); Serial.print(suhu); Serial.println(" °C");
    Serial.print("Asap : "); Serial.println(asap ? "YA" : "TIDAK");
    Serial.print("Wemos1 (gerak/suara) : "); Serial.println(wemos1Online ?
"ONLINE" : "OFFLINE");
    Serial.print("Wemos2 (suhu/asap)   : "); Serial.println(wemos2Online ?
"ONLINE" : "OFFLINE");
    Serial.println("====================================");

    lastPrint = millis();
  }
```

```
}
```

Lampiran 2 Kode Wemos Client 1

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>
#include <DHT.h>

//=======================================================
// WIFI - HUBUNGKAN KE ACCESS POINT ESP32 SERVER
//=======================================================

const char* AP_SSID = "ESP32_IoT_Hub";
const char* AP_PASS = "iot12345";

// IP default softAP ESP32 adalah 192.168.4.1
// Kalau di server Serial Monitor tertulis IP lain, sesuaikan di sini
const char* SERVER_IP = "192.168.4.1";
const int   SERVER_PORT = 80;

//=======================================================
// PIN SENSOR
//=======================================================

#define DHTPIN   D4      // pin data DHT11
#define DHTTYPE  DHT11
DHT dht(DHTPIN, DHTTYPE);

#define MQ2_PIN  D5      // pin digital output MQ2 (DO)
// MQ2 modul: DO biasanya LOW saat asap/gas terdeteksi (tergantung modul,
// cek dengan Serial.println(digitalRead(MQ2_PIN)) kalau terbalik)

//=======================================================
// TIMER (non-blocking)
//=======================================================

unsigned long lastSend = 0;
const unsigned long sendInterval = 2000; // kirim data tiap 2 detik

unsigned long lastReconnect = 0;
const unsigned long reconnectInterval = 5000;

//=======================================================
// SETUP
```

```
//=======================================================

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

   pinMode(MQ2_PIN, INPUT);

   dht.begin();

   WiFi.mode(WIFI_STA);
   WiFi.begin(AP_SSID, AP_PASS);

   Serial.println();
   Serial.println("=====================================");
   Serial.println(" WEMOS NODE 1 - DHT11 + MQ2");
   Serial.println("=====================================");
   Serial.print("Menghubungkan ke ");
   Serial.println(AP_SSID);

   unsigned long startAttempt = millis();
   while (WiFi.status() != WL_CONNECTED && millis() - startAttempt <
15000)
   {
      delay(300);
      Serial.print(".");
   }

   Serial.println();

   if (WiFi.status() == WL_CONNECTED)
   {
      Serial.println("Terhubung ke server AP");
      Serial.print("IP Node : ");
      Serial.println(WiFi.localIP());
   }
   else
   {
      Serial.println("GAGAL terhubung, akan coba lagi di loop()");
   }
}

//=======================================================
// KIRIM DATA KE SERVER
//=======================================================

void sendData(float suhu, bool asap)
{
```

```
    if (WiFi.status() != WL_CONNECTED) return;

    WiFiClient client;
    HTTPClient http;

    String url = "http://";
    url += SERVER_IP;
    url += ":";
    url += SERVER_PORT;
    url += "/data?node=2&suhu=";
    url += String(suhu, 1);
    url += "&asap=";
    url += (asap ? "1" : "0");

    http.begin(client, url);
    int httpCode = http.GET();

    Serial.print("Kirim -> ");
    Serial.print(url);
    Serial.print(" | Response: ");
    Serial.println(httpCode);

    http.end();
}

//=============================================================
// LOOP
//=============================================================

void loop()
{
    // Reconnect WiFi kalau putus
    if (WiFi.status() != WL_CONNECTED)
    {
        if (millis() - lastReconnect >= reconnectInterval)
        {
            Serial.println("Reconnect ke server AP...");
            WiFi.begin(AP_SSID, AP_PASS);
            lastReconnect = millis();
        }
    }

    // Baca sensor & kirim tiap sendInterval, non-blocking
    if (millis() - lastSend >= sendInterval)
    {
        float suhu = dht.readTemperature();
        bool asap = (digitalRead(MQ2_PIN) == LOW); // sesuaikan logika
HIGH/LOW dengan modul MQ2 kamu
```

```
      if (isnan(suhu))
      {
        Serial.println("Gagal baca DHT11, lewati kiriman kali ini");
      }
      else
      {
        Serial.print("Suhu: "); Serial.print(suhu);
        Serial.print(" C | Asap: "); Serial.println(asap ? "YA" : "TIDAK");

        sendData(suhu, asap);
      }

      lastSend = millis();
    }
}
```

Lampiran 3 Kode Wemos Client 2

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>

//=======================================================
// WIFI - HUBUNGKAN KE ACCESS POINT ESP32 SERVER
//=======================================================

const char* AP_SSID = "ESP32_IoT_Hub";
const char* AP_PASS = "iot12345";

const char* SERVER_IP = "192.168.4.1"; // sesuaikan kalau IP server berbeda
const int   SERVER_PORT = 80;

//=======================================================
// PIN SENSOR
//=======================================================

#define PIR_PIN   D5     // SR501 output digital (HIGH saat ada gerakan)
#define SOUND_PIN D6     // KY037 output digital (DO), HIGH saat suara melewati ambang batas
                         // Ambang batas KY037 diatur lewat trimpot di modulnya

//=======================================================
// TIMER (non-blocking)
//=======================================================

unsigned long lastSend = 0;
```

```
const unsigned long sendInterval = 1000; // gerak & suara butuh respons lebih cepat dari suhu/asap

unsigned long lastReconnect = 0;
const unsigned long reconnectInterval = 5000;

//=======================================================
// SETUP
//=======================================================

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

  pinMode(PIR_PIN, INPUT);
  pinMode(SOUND_PIN, INPUT);

  WiFi.mode(WIFI_STA);
  WiFi.begin(AP_SSID, AP_PASS);

  Serial.println();
  Serial.println("=========================================");
  Serial.println(" WEMOS NODE 2 - KY037 + SR501");
  Serial.println("=========================================");
  Serial.print("Menghubungkan ke ");
  Serial.println(AP_SSID);

  unsigned long startAttempt = millis();
   while (WiFi.status() != WL_CONNECTED && millis() - startAttempt < 15000)
  {
    delay(300);
    Serial.print(".");
  }

  Serial.println();

  if (WiFi.status() == WL_CONNECTED)
  {
    Serial.println("Terhubung ke server AP");
    Serial.print("IP Node : ");
    Serial.println(WiFi.localIP());
  }
  else
  {
    Serial.println("GAGAL terhubung, akan coba lagi di loop()");
  }
}
```

```
//=====================================================
// KIRIM DATA KE SERVER
//=====================================================

void sendData(bool gerak, bool suara)
{
   if (WiFi.status() != WL_CONNECTED) return;

   WiFiClient client;
   HTTPClient http;

   String url = "http://";
   url += SERVER_IP;
   url += ":";
   url += SERVER_PORT;
   url += "/data?node=1&gerak=";
   url += (gerak ? "1" : "0");
   url += "&suara=";
   url += (suara ? "1" : "0");

   http.begin(client, url);
   int httpCode = http.GET();

   Serial.print("Kirim -> ");
   Serial.print(url);
   Serial.print(" | Response: ");
   Serial.println(httpCode);

   http.end();
}

//=====================================================
// LOOP
//=====================================================

void loop()
{
   // Reconnect WiFi kalau putus
   if (WiFi.status() != WL_CONNECTED)
   {
      if (millis() - lastReconnect >= reconnectInterval)
      {
         Serial.println("Reconnect ke server AP...");
         WiFi.begin(AP_SSID, AP_PASS);
         lastReconnect = millis();
      }
   }
```

```
  // Baca sensor & kirim tiap sendInterval, non-blocking
  if (millis() - lastSend >= sendInterval)
  {
    bool gerak = (digitalRead(PIR_PIN) == HIGH);
        bool suara = (digitalRead(SOUND_PIN) == HIGH); // sesuaikan
HIGH/LOW dengan hasil trimpot KY037 kamu

    Serial.print("Gerak: "); Serial.print(gerak ? "YA" : "TIDAK");
    Serial.print(" | Suara: "); Serial.println(suara ? "YA" : "TIDAK");

    sendData(gerak, suara);

    lastSend = millis();
  }
}
```

RIWAYAT HIDUP

K Arista Arya Dwijaya lahir di Desa Sudaji Kecamatan Sawan Kabupaten Buleleng Penulis menempuh pendidikan dasar hingga lulus pada tahun 2013 di SD Negeri 1 Sudaji, dilanjutkan ke jenjang sekolah menengah pertama dan diselesaikan pada tahun 2016 di SMP Negeri 3 Sawan. Jenjang pendidikan menengah atas ditempuh dan dirampungkan pada tahun 2019 di SMK Negeri 3 Singaraja. Pada tahun yang sama, penulis melanjutkan studi ke Program Studi S1 Ilmu Komputer, Jurusan Teknik Informatika, Universitas Pendidikan Ganesha, dan hingga pertengahan tahun 2026 masih tercatat aktif sebagai mahasiswa pada program studi tersebut. Sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer, penulis menyusun tugas akhir dengan judul " RANCANG BANGUN SMART HOME BABY MONITORING BERBASIS IOT ".