LAMPIRAN

LINK VIDIO PENELITIAN
https://drive.google.com/drive/folders/1OKHIyuYaWLytECF9QSTDNYiPNzFajHpW

CARA KERJA CODING

Program ESP8266 untuk terhubung ke jaringan WiFi dan menampilkan informasi jaringan.

```
#include <ESP8266WiFi.h>


const char *ssid = "HAC"; //ganti nama wifi
const char *pass = "11111111";//ganti password


WiFiClient client;


void setup() {
 Serial.begin(9600);
 delay(10);


 Serial.print(" Menghubungkan ke : ");
 Serial.println(ssid);


 WiFi.begin(ssid, pass);
 while (WiFi.status() != WL_CONNECTED)
 {
  delay(500);
  Serial.print("....");
  }
  Serial.print("\n");
  Serial.print("IP address : ");
  Serial.print(WiFi.localIP());
```

```
    Serial.print("\n");
    Serial.print("MAC : ");
    Serial.println(WiFi.macAddress());
    Serial.print("\n");
    Serial.print("IP gateway : ");
    Serial.println(WiFi.gatewayIP());
    Serial.println("");
    Serial.print("Terhubung dengan : ");
    Serial.println(ssid);
}

void loop() { }
```

Program ESP8266 sebagai penerima (receiver) sinyal Infrared (IR) dan menampilkan hasil decode ke Serial Monitor.

```
#include <Arduino.h>
#include <IRrecv.h>
#include <IRremoteESP8266.h>
#include <IRac.h>
#include <IRtext.h>
#include <IRutils.h>

// Pin IR Receiver
const uint16_t kRecvPin = D2;  // D2 pada NodeMCU = GPIO4

// Konfigurasi Serial dan buffer
const uint32_t kBaudRate = 115200;  // Ubah ke 115200 agar lebih cepat dan umum
const uint16_t kCaptureBufferSize = 1024;  // Buffer data IR

#if DECODE_AC
const uint8_t kTimeout = 50;
#else
const uint8_t kTimeout = 15;
#endif

const uint16_t kMinUnknownSize = 12;  // Ukuran minimum sinyal IR tak dikenal
#define LEGACY_TIMING_INFO false  // Tampilkan timing info lama jika ingin

// Objek untuk menerima IR
```

```
IRrecv irrecv(kRecvPin, kCaptureBufferSize, kTimeout, true);
decode_results results;

void setup() {
  // Inisialisasi Serial (tanpa SERIAL_TX_ONLY untuk menghindari error)
  Serial.begin(kBaudRate);
  while (!Serial) delay(50);  // Tunggu sampai Serial siap

  Serial.printf("\n" D_STR_IRRECVDUMP_STARTUP "\n", kRecvPin);

#if DECODE_HASH
  irrecv.setUnknownThreshold(kMinUnknownSize);
#endif

  irrecv.enableIRIn();  // Mulai menerima sinyal IR
  Serial.println("IR Receiver aktif dan siap menerima sinyal...");
}

void loop() {
  if (irrecv.decode(&results)) {
    // Tampilkan waktu penerimaan
    uint32_t now = millis();
    Serial.printf(D_STR_TIMESTAMP " : %06u.%03u\n", now / 1000, now %
1000);

    // Tampilkan peringatan jika buffer penuh
    if (results.overflow)
      Serial.printf(D_WARN_BUFFERFULL "\n", kCaptureBufferSize);

    // Versi library
```

```
    Serial.println(D_STR_LIBRARY ": v" _IRREMOTEESP8266_VERSION_ "\n");

    // Deskripsi dasar sinyal IR
    Serial.print(resultToHumanReadableBasic(&results));

    // Jika AC signal, tampilkan deskripsi tambahan
    String description = IRAcUtils::resultAcToString(&results);
    if (description.length())
      Serial.println(description);

#if LEGACY_TIMING_INFO
    Serial.println(D_STR_MESGDESC);
    Serial.println(resultToTimingInfo(&results));
    yield();
#endif

    // Selesai proses, siap menerima sinyal berikutnya
    irrecv.resume();
    yield();
  }
}
```

Sistem kontrol dan monitoring AC berbasis IoT menggunakan ESP8266, Firebase, dan komunikasi TCP ke server lokal.

```
#include <ESP8266WiFi.h>
#include <FirebaseESP8266.h>
#include <SoftwareSerial.h>

// ========== KONFIGURASI JARINGAN ==========
const char* ssid = "HAC";
const char* password = "11111111";

// ========== KONFIGURASI FIREBASE ==========
#define FIREBASE_HOST "ac-app-b5549-default-rtdb.asia-southeast1.firebasedatabase.app"
#define FIREBASE_AUTH "wq9RwYVcTNOEN1qDuZWaYwlyvUQHl8pRvlj5jZKr"

// ========== DEKLARASI OBJEK ==========
FirebaseData fbData;
FirebaseAuth fbAuth;
FirebaseConfig fbConfig;
WiFiClient client;

// ========== STRUKTUR DATA AC ==========
struct ACState {
  bool lastState;
```

```
  int lastTemp;
  String path;
  String name;
  IPAddress serverIP;
  uint16_t serverPort;
};

// Inisialisasi array AC dengan IP dan port server masing-masing
ACState acs[] = {
  {false, 0, "/devices/AC_04", "LG", IPAddress(10,33,126,78), 5000}};
const int AC_COUNT = sizeof(acs) / sizeof(acs[0]);

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

  connectWiFi();

  fbConfig.host = FIREBASE_HOST;
  fbConfig.signer.tokens.legacy_token = FIREBASE_AUTH;

  Firebase.begin(&fbConfig, &fbAuth);
  Firebase.reconnectWiFi(true);

  Serial.println("\nMencoba koneksi ke Firebase...");
  unsigned long startTime = millis();
  while (!Firebase.ready() && millis() - startTime < 15000) {
    Serial.print(".");
    delay(500);
```

```
  }

  if (!Firebase.ready()) {
    Serial.println("\nGagal terkoneksi ke Firebase!");
    Serial.println("Penyebab: " + fbData.errorReason());
    while (1) delay(1000);
  }

  Serial.println("\nBerhasil terhubung ke Firebase!");
}

void loop() {
  if (!Firebase.ready() || WiFi.status() != WL_CONNECTED) {
    Serial.println("[ERROR] Koneksi terputus, mencoba reconnect...");
    connectWiFi();
    delay(5000);
    return;
  }

  for (int i = 0; i < AC_COUNT; i++) {
    checkACState(acs[i]);
    delay(200);
  }

  delay(1000);
}

void connectWiFi() {
```

```
  Serial.print("\nMenghubungkan ke WiFi ");
  Serial.print(ssid);
  Serial.print("...");

  WiFi.begin(ssid, password);
  unsigned long startTime = millis();

  while (WiFi.status() != WL_CONNECTED && millis() - startTime < 20000) {
    delay(500);
    Serial.print(".");
  }

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

void checkACState(ACState &ac) {
  // Cek status ON/OFF
  String path = ac.path + "/isOn";
  if (Firebase.getBool(fbData, path)) {
    if (fbData.dataType() == "boolean") {
      bool currentState = fbData.boolData();
      if (currentState != ac.lastState) {
```

```
      unsigned long sendMillis = millis();
      Serial.printf("\n[%s] Status berubah: %s -> %s [at %lums]\n",
                    ac.name.c_str(),
                    ac.lastState ? "ON" : "OFF",
                    currentState ? "ON" : "OFF",
                    sendMillis);

      if (sendToServer(ac.path.substring(9) + ":" + (currentState ? "ON" : "OFF"),
sendMillis, ac.serverIP, ac.serverPort)) {
        ac.lastState = currentState;
      }
    }
  }
 } else {
  Serial.printf("\n[ERROR] Gagal baca %s: %s\n", path.c_str(),
fbData.errorReason().c_str());
 }

 // Cek temperatur
 path = ac.path + "/temperature";
 if (Firebase.getInt(fbData, path)) {
  if (fbData.dataType() == "int") {
    int currentTemp = fbData.intData();
    if (currentTemp != ac.lastTemp) {
      unsigned long sendMillis = millis();
      Serial.printf("\n[%s] Suhu berubah: %d°C -> %d°C [at %lums]\n",
                    ac.name.c_str(), ac.lastTemp, currentTemp, sendMillis);
```

```
      if (sendToServer(ac.path.substring(9) + ":TEMP:" + String(currentTemp),
sendMillis, ac.serverIP, ac.serverPort)) {
        ac.lastTemp = currentTemp;
       }
     }
   }
  } else {
   Serial.printf("\n[ERROR] Gagal baca %s: %s\n", path.c_str(),
fbData.errorReason().c_str());
  }
}

bool sendToServer(String message, unsigned long sendMillis, IPAddress serverIP,
uint16_t serverPort) {
  Serial.print("[NETWORK] Mengirim ke server: ");
  Serial.println(message);
  Serial.printf("[INFO] Waktu kirim: %lums | RSSI WiFi: %d dBm\n", sendMillis,
WiFi.RSSI());

  // Kirim ke TCP Server (opsional)
  unsigned long startMillis = millis();
  while (!client.connect(serverIP, serverPort) && (millis() - startMillis < 5000))

  if (!client.connected()) {
   Serial.println("\n[ERROR] Gagal terkoneksi ke server!");
   return false;
  }

  client.println(message);
```

```
  client.flush();

  unsigned long responseStart = millis();
  while (client.available() == 0 && (millis() - responseStart < 3000)) {
    delay(100);
  }

  if (client.available()) {
    String response = client.readStringUntil('\n');
    Serial.print("[SERVER] Respons: ");
    Serial.println(response);

    unsigned long receiveMillis = millis();
    Serial.printf("[INFO] Waktu terima: %lums setelah kirim\n", receiveMillis -
sendMillis);
  }

  client.stop();
  return true;
}
```

ESP8266 sebagai server WiFi yang mengontrol beberapa AC melalui sinyal Infrared (IR).

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <WiFiServer.h>
#include <IRremoteESP8266.h>
#include <IRsend.h>
#include <ir_Electra.h>
#include <SoftwareSerial.h>

// ======= KONFIGURASI JARINGAN =======
const char* ssid = "HAC";
const char* password = "11111111";

// ======= SERVER WIFI =======
WiFiServer server(5000);
WiFiClient client;

// ======= KONFIGURASI IR TRANSMITTER =======
const uint16_t kIrLed = 4;  // D2
IRsend irsend(kIrLed);
```

```
// ======= STRUKTUR DATA UNTUK AC =======
struct ACProfile {
  String name;
  const uint16_t* powerOnRaw;
  const uint16_t* powerOffRaw;
  const uint16_t* tempRaw[15]; // suhu 16-30 (index 0-14)
  size_t rawLength;
};

// ======= RAW DATA =======
const uint16_t ac1PowerOn[187] PROGMEM = {9054, 4438,  612, 1614,  586,
1640,  568, 560,  536, 562,  538, 562,  538, 562,  536, 1660,  588, 1638,  588, 1638,
586, 1640,  568, 1658,  570, 558,  538, 562,  538, 560,  538, 1660,  588, 540,  538,
562,  538, 562,  536, 562,  538, 562,  536, 562,  536, 1660,  588, 1638,  566, 1660,
568, 560,  536, 562,  538, 562,  538, 562,  538, 562,  538, 562,  538, 560,  540, 558,
540, 558,  544, 556,  542, 556,  544, 554,  544, 554,  574, 1628,  594, 530,  634, 466,
632, 468,  602, 498,  600, 498,  606, 494,  604, 494,  602, 498,  598, 500,  570, 530,
566, 534,  564, 534,  562, 536,  562, 538,  538, 562,  536, 562,  558, 1646,  580, 542,
558, 542,  556, 544,  556, 542,  556, 544,  556, 544,  556, 546,  554, 546,  552, 570,
528, 572,  528, 572,  526, 574,  524, 576,  524, 602,  498, 602,  498, 602,  496, 602,
498, 602,  496, 630,  470, 628,  470, 628,  470, 630,  470, 1730,  496, 654,  446, 654,
444, 654,  446, 656,  442, 680,  396, 704,  394, 706,  394, 732,  368, 732,  366, 760,
340, 1860,  366, 812,  288, 1964,  262, 942,  106}; // Isi sama seperti sebelumnya

const uint16_t ac1PowerOff[211] PROGMEM = {9022, 4470,  644, 1582,  646,
1580,  620, 508,  584, 516,  582, 516,  584, 514,  586, 1612,  620, 1606,  646, 1580,
646, 1580,  618, 1608,  620, 508,  584, 516,  584, 516,  584, 1614,  620, 508,  586,
514,  586, 514,  612, 488,  610, 488,  604, 496,  586, 1610,  620, 1606,  618, 1608,
618, 510,  584, 516,  584, 516,  586, 514,  584, 514,  586, 514,  602, 498,  586, 514,
586, 512,  588, 512,  584, 514,  584, 514,  584, 516,  586, 1612,  620, 508,  584, 516,
586, 514,  584, 516,  584, 514,  584, 516,  586, 514,  584, 514,  586, 514,  584, 516,
584, 516,  560, 538,  584, 516,  582, 518,  558, 540,  560, 540,  560, 1638,  614, 514,
560, 540,  558, 542,  558, 540,  558, 542,  558, 542,  556, 542,  556, 544,  538, 562,
538, 560,  538, 560,  540, 560,  542, 558,  542, 556,  544, 556,  570, 530,  572, 528,
570, 530,  566, 532,  566, 510,  590, 512,  592, 526,  594, 506,  566, 532,  562, 538,
562, 536,  562, 538,  560, 538,  562, 538,  560, 542,  558, 540,  560, 540,  558, 542,
558, 1666,  560, 540,  558, 1668,  558, 542,  554, 546,  554, 546,  552, 570,  528,
```

572, 526, 1674, 550, 1700, 526, 1700, 526, 1702, 524, 602, 498, 604, 496, 1728, 498, 628, 498};

const uint16_t ac1Temp17[71] PROGMEM = {9038, 4466, 594, 508, 612, 508, 612, 1646, 594, 508, 612, 508, 612, 508, 612, 508, 612, 508, 612, 1646, 592, 1646, 594, 506, 612, 1646, 592, 1646, 592, 1646, 594, 1646, 594, 1644, 594, 508, 610, 1648, 592, 508, 612, 508, 612, 508, 612, 508, 612, 508, 612, 1646, 594, 508, 612, 1646, 594, 1644, 594, 1646, 594, 1646, 594, 1646, 594, 1644, 594, 40116, 9038, 2216, 594};

const uint16_t ac1Temp18[71] PROGMEM = {9036, 4468, 590, 510, 610, 510, 610, 1648, 592, 510, 610, 510, 610, 510, 610, 510, 610, 510, 610, 1648, 592, 1650, 590, 510, 610, 1648, 590, 1648, 590, 1650, 588, 1650, 590, 1648, 590, 510, 610, 1648, 590, 510, 610, 510, 610, 510, 610, 510, 610, 510, 610, 1650, 590, 510, 610, 1648, 592, 1648, 590, 1648, 592, 1648, 590, 1648, 592, 1648, 592, 40122, 9036, 2218, 590};

// ======= RAW DATA =======

const uint16_t ac2PowerOn[67] PROGMEM = {9038, 4466, 594, 508, 612, 508, 612, 1648, 592, 510, 612, 508, 612, 508, 612, 508, 612, 508, 612, 1646, 592, 1646, 594, 508, 612, 1648, 592, 1646, 592, 1648, 592, 1648, 592, 1648, 592, 508, 612, 508, 612, 508, 612, 1646, 592, 508, 612, 508, 612, 508, 612, 1648, 592, 1646, 594, 1646, 594, 508, 612, 1646, 594, 1646, 592, 1646, 594, 1646, 594};

const uint16_t ac2PowerOff[71] PROGMEM = {9044, 4466, 592, 508, 612, 508, 612, 1648, 592, 508, 612, 508, 612, 508, 612, 510, 610, 510, 610, 1648, 592, 1646, 592, 508, 612, 1648, 592, 1646, 592, 1648, 592, 1648, 592, 1646, 592, 510, 610, 510, 612, 510, 610, 1646, 592, 510, 610, 510, 612, 508, 612, 508, 612, 1646, 592, 1646, 592, 1648, 592, 510, 610, 1646, 592, 1648, 592, 1648, 592, 1648, 592, 40124, 9038, 2216, 594};

const uint16_t ac2Temp17[] PROGMEM = {};

const uint16_t ac2Temp18[] PROGMEM = {};

const uint16_t ac2Temp19[] PROGMEM = {};

// ======= RAW DATA =======

const uint16_t ac3PowerOn[259] PROGMEM = {4058, 3988, 534, 1962, 564, 1934, 566, 1930, 568, 1932, 564, 934, 566, 1930, 564, 936, 566, 1930, 566, 932, 566, 1930, 540, 1960, 536, 960, 566, 932, 566, 932, 568, 932, 566, 930, 568, 1930, 566, 932, 566, 1930, 566, 932, 568, 1932, 566, 932, 566, 932, 568, 1930,

538, 8638, 4060, 3988, 534, 1962, 542, 1954, 566, 1932, 536, 1962, 534, 962, 538, 1958, 538, 962, 540, 1956, 536, 962, 542, 1956, 536, 1962, 536, 962, 538, 960, 538, 962, 536, 962, 538, 958, 542, 1954, 536, 964, 534, 1962, 536, 962, 538, 1960, 536, 962, 536, 962, 536, 1962, 536, 8616, 4058, 3990, 534, 1962, 536, 1962, 534, 1964, 534, 1964, 534, 964, 532, 1964, 512, 986, 510, 1988, 510, 988, 512, 1986, 512, 1986, 512, 986, 512, 1010, 512, 986, 516, 980, 544, 954, 550, 1922, 604, 920, 578, 1894, 606, 918, 580, 1894, 604, 920, 576, 922, 576, 1896, 600, 8524, 4088, 3960, 562, 1936, 562, 1936, 536, 1960, 536, 1962, 536, 962, 536, 1962, 534, 964, 534, 1964, 534, 964, 534, 1964, 534, 1964, 532, 966, 534, 986, 512, 986, 512, 986, 512, 986, 512, 1986, 510, 988, 510, 1988, 510, 988, 510, 1988, 508, 990, 508, 990, 508, 1990, 508, 8628, 906};

const uint16_t ac3PowerOff[259] PROGMEM = {4058, 3986, 536, 1960, 542, 1956, 536, 1960, 542, 1956, 566, 932, 542, 1954, 542, 956, 542, 1956, 540, 958, 538, 1960, 536, 1962, 536, 1960, 542, 956, 544, 956, 540, 958, 542, 956, 542, 1956, 542, 956, 542, 1956, 538, 962, 540, 1956, 536, 960, 542, 956, 542, 956, 542, 8610, 4060, 3986, 536, 1962, 534, 1962, 542, 1956, 538, 1960, 536, 960, 542, 1956, 536, 962, 542, 1956, 542, 956, 542, 1956, 536, 1960, 536, 1962, 536, 964, 536, 960, 542, 956, 540, 958, 538, 1960, 536, 962, 536, 1960, 536, 962, 540, 1956, 538, 960, 540, 956, 542, 958, 536, 8616, 4060, 3988, 534, 1964, 534, 1964, 536, 1964, 532, 1966, 532, 966, 534, 1962, 512, 986, 510, 1988, 510, 988, 512, 1986, 512, 1986, 512, 1986, 512, 986, 512, 986, 512, 986, 512, 988, 512, 1984, 540, 982, 546, 1928, 572, 952, 576, 1898, 602, 922, 580, 918, 578, 920, 578, 8522, 4092, 3954, 568, 1930, 566, 1932, 566, 1932, 562, 1936, 560, 938, 536, 1962, 536, 962, 536, 1962, 536, 964, 534, 1962, 534, 1964, 534, 1964, 534, 964, 534, 964, 534, 966, 534, 964, 532, 1966, 532, 966, 532, 1986, 510, 988, 512, 1986, 512, 988, 510, 988, 510, 988, 510, 8668, 4032, 4016, 506, 1992, 506, 1992, 504, 1994, 504, 1994, 502, 996, 502, 2020, 476, 1022, 476, 2022, 476, 1024, 474, 2022, 476, 2022, 476, 2022, 474, 1024, 474, 1024, 474, 1024, 476, 1022, 476, 2022, 474, 1024, 474, 2024, 474, 1024, 474, 2024, 474, 1024, 472, 1024, 474, 1024, 474};

const uint16_t ac3Temp17[] PROGMEM = {};

const uint16_t ac3Temp18[] PROGMEM = {};

const uint16_t ac3Temp19[] PROGMEM = {};

// ======= RAW DATA =======

const uint16_t ac4PowerOn[] PROGMEM = {};

const uint16_t ac4PowerOff[] PROGMEM = {};

const uint16_t ac4Temp17[] PROGMEM = {};

```
const uint16_t ac4Temp18[] PROGMEM = {};
const uint16_t ac4Temp19[] PROGMEM = {};
// ======= PROFIL AC =======
ACProfile acProfiles[] = {
  {
    "Panasonic",
    ac1PowerOn,
    ac1PowerOff,
    {},
    sizeof(ac1PowerOn) / sizeof(uint16_t)
  },
  {
    "Daikin",
    ac2PowerOn,
    ac2PowerOff,
    {},
    sizeof(ac2PowerOn) / sizeof(uint16_t)
  },
  {
    "Samsung",
    ac3PowerOn,
    ac3PowerOff,
    {},
    sizeof(ac3PowerOn) / sizeof(uint16_t)
  },
  {
    "LG",
    ac4PowerOn,
```

```
    ac4PowerOff,
    {},
    sizeof(ac4PowerOn) / sizeof(uint16_t)
  },
};
const int AC_COUNT = sizeof(acProfiles) / sizeof(acProfiles[0]);

// ======= SETUP =======
void setup() {
  Serial.begin(115200);
  irsend.begin();

  connectWiFi();
  server.begin();

  Serial.println("\nServer siap!");
  Serial.print("IP: ");
  Serial.println(WiFi.localIP());

  for (int i = 0; i < AC_COUNT; i++) {
    Serial.printf("- AC_%02d: %s\n", i + 1, acProfiles[i].name.c_str());
  }
}

// ======= LOOP =======
void loop() {
  if (!client || !client.connected()) {
    client = server.available();
```

```
    if (client) {
      Serial.println("\nClient TCP terhubung: " + client.remoteIP().toString());
    }
  }

  if (client && client.available()) {
    String command = client.readStringUntil('\n');
    command.trim();
    Serial.println("[WiFi] Perintah: " + command);
    processCommand(command, "WiFi");
  }

}

// ======= KONEKSI WIFI =======
void connectWiFi() {
  Serial.print("Menghubungkan ke WiFi ");
  Serial.print(ssid);
  Serial.println("...");
  WiFi.begin(ssid, password);
  unsigned long start = millis();
  while (WiFi.status() != WL_CONNECTED && millis() - start < 20000) {
    delay(500);
    Serial.print(".");
  }
  Serial.println(WiFi.status() == WL_CONNECTED ? "\nWiFi Terhubung!" :
"\nGagal konek WiFi.");
}
```

```
// ======= KIRIM SINYAL IR (dengan pembacaan dari PROGMEM) =======
void sendIR(const uint16_t* rawData, size_t length) {
  uint16_t* tempData = new uint16_t[length];
  for (size_t i = 0; i < length; i++) {
    tempData[i] = pgm_read_word_near(rawData + i);
  }

  for (int i = 0; i < 3; i++) {
    irsend.sendRaw(tempData, length, 38);  // 38kHz
    delay(100);
  }

  delete[] tempData;
  Serial.println("Sinyal IR dikirim");
}

// ======= PROSES PERINTAH =======
void processCommand(String command, String source) {
  unsigned long timeReceived = millis();
  long sentTimestamp = 0;

  int firstSep = command.indexOf(':');
  int secondSep = command.indexOf(':', firstSep + 1);

  String device, action;
  if (secondSep != -1) {
    device = command.substring(0, firstSep);
```

```
  action = command.substring(firstSep + 1, secondSep);
  sentTimestamp = command.substring(secondSep + 1).toInt();
} else {
  device = command.substring(0, firstSep);
  action = command.substring(firstSep + 1);
}

if (!device.startsWith("AC_")) {
  Serial.println("[ERROR] Nama perangkat salah");
  return;
}

int acIndex = device.substring(3).toInt() - 1;
if (acIndex < 0 || acIndex >= AC_COUNT) {
  Serial.println("[ERROR] Nomor AC tidak valid");
  return;
}

ACProfile& ac = acProfiles[acIndex];

if (action == "ON") {
  sendIR(ac.powerOnRaw, ac.rawLength);
} else if (action == "OFF") {
  sendIR(ac.powerOffRaw, ac.rawLength);
} else if (action.startsWith("TEMP:")) {
  int temp = action.substring(5).toInt();
  if (temp < 16 || temp > 30) {
    Serial.println("[ERROR] Suhu di luar batas");
```

```
    return;
   }
   int tempIndex = temp - 16;
   if (ac.tempRaw[tempIndex] == nullptr) {
    Serial.println("[ERROR] Data suhu tidak tersedia");
    return;
   }
   sendIR(ac.tempRaw[tempIndex], ac.rawLength);
  } else {
   Serial.println("[ERROR] Perintah tidak dikenali");
   return;
  }

  unsigned long delayMs = (sentTimestamp > 0) ? timeReceived - sentTimestamp :
0;
  int rssi = WiFi.RSSI();

  Serial.printf("[INFO] Sumber: %s\n", source.c_str());
  Serial.printf("[INFO] AC: %s | Aksi: %s\n", ac.name.c_str(), action.c_str());
  Serial.printf("[INFO] Waktu Diterima: %lu ms\n", timeReceived);
  if (sentTimestamp > 0) {
   Serial.printf("[INFO] Timestamp Kirim: %ld ms\n", sentTimestamp);
   Serial.printf("[INFO] Delay: %lu ms\n", delayMs);
  } else {
   Serial.println("[INFO] Timestamp Kirim tidak tersedia");
  }
  if (source == "WiFi") {
   Serial.printf("[INFO] WiFi RSSI: %d dBm\n", rssi);
```

```
  }

  String response = "ACK:" + device + ":" + action + ":Delay=" + String(delayMs) +
"ms";
  if (source == "WiFi" && client) {
    client.println(response);
  }
}
```