

# LAPORAN PROYEK SISTEM KONTROL LAMPU BERBASIS TELEGRAM BOT

## Kelompok 6

Jesika Natalia (22080960 )  
Kurniawan Putra Mukhlisin (22080960 )  
M.Nabil Al-Hamamy (2208096035)

---

### I. PENDAHULUAN

#### 1.1. Latar Belakang

Penerapan Internet of Things (IoT) dalam otomasi rumah (smart home) telah berkembang pesat. Salah satu implementasi dasar dalam smart home adalah sistem pengendalian perangkat listrik seperti lampu. Proyek ini bertujuan membangun sistem pengendalian dua lampu listrik 220V secara jarak jauh menggunakan platform Telegram sebagai antarmuka pengguna, dengan mikrokontroler NodeMCU ESP8266 dan modul relay sebagai pengendali fisik.

Pengendalian lampu secara otomatis maupun jarak jauh dapat memberikan kemudahan bagi pengguna, menghemat energi, dan meningkatkan keamanan, terutama saat pengguna sedang tidak berada di rumah. Teknologi ini juga relevan dalam konteks keberlanjutan energi dan efisiensi sistem rumah tangga modern.

Dalam proyek ini, dikembangkan sistem pengendalian dua buah lampu bertegangan 220V yang dapat diakses dari jarak jauh melalui aplikasi Telegram. Telegram dipilih sebagai antarmuka karena bersifat ringan, gratis, lintas platform, dan memiliki fitur *bot API* yang memungkinkan integrasi dengan sistem otomatisasi. Sistem ini dibangun menggunakan mikrokontroler NodeMCU ESP8266 sebagai otak utama, dengan modul relay sebagai aktuator untuk mengatur arus listrik ke lampu.

#### 1.2. Tujuan

- Membuat sistem kendali lampu berbasis perintah teks dari Telegram.
- Menghubungkan bot Telegram dengan NodeMCU melalui koneksi Wi-Fi dan API Telegram.
- Mengontrol beban listrik AC 220V secara aman menggunakan relay.
- Meningkatkan pemahaman tentang protokol komunikasi cloud-to-device.

#### 1.3. Manfaat

- Implementasi konsep IoT sederhana pada smart home.
- Solusi low-cost untuk kontrol perangkat jarak jauh.
- Pengenalan integrasi hardware dengan cloud-based API.

### II. LANDASAN TEORI

#### 2.1. NodeMCU ESP8266

NodeMCU adalah mikrokontroler berbasis chip ESP8266 dengan koneksi Wi-Fi terintegrasi, mendukung pengembangan berbasis Arduino IDE. NodeMCU dapat digunakan untuk menghubungkan perangkat keras ke cloud atau API seperti Telegram Bot API.

## 2.2. Modul Relay 4 Channel

Relay adalah saklar elektromagnetik yang digunakan untuk mengontrol beban listrik AC menggunakan sinyal tegangan rendah (3.3–5V). Pada proyek ini, hanya dua dari empat channel relay yang digunakan.

## 2.3. Telegram Bot API

Telegram Bot adalah akun aplikasi non-manusia yang dapat menerima dan mengirim pesan melalui API HTTP/HTTPS. Bot dapat diprogram untuk menjalankan fungsi otomatis, seperti mengontrol hardware.

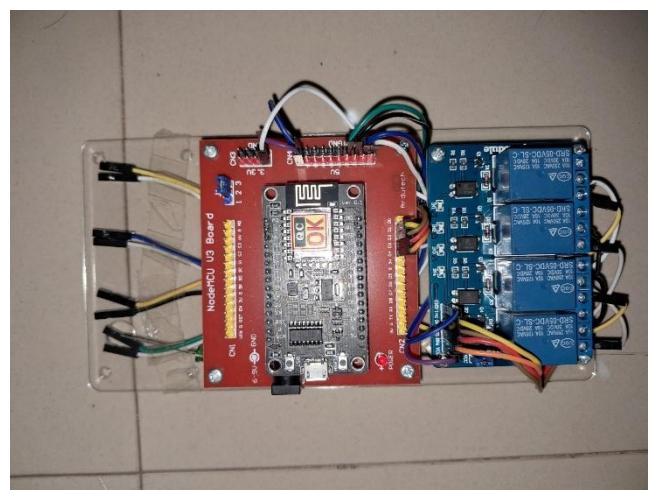
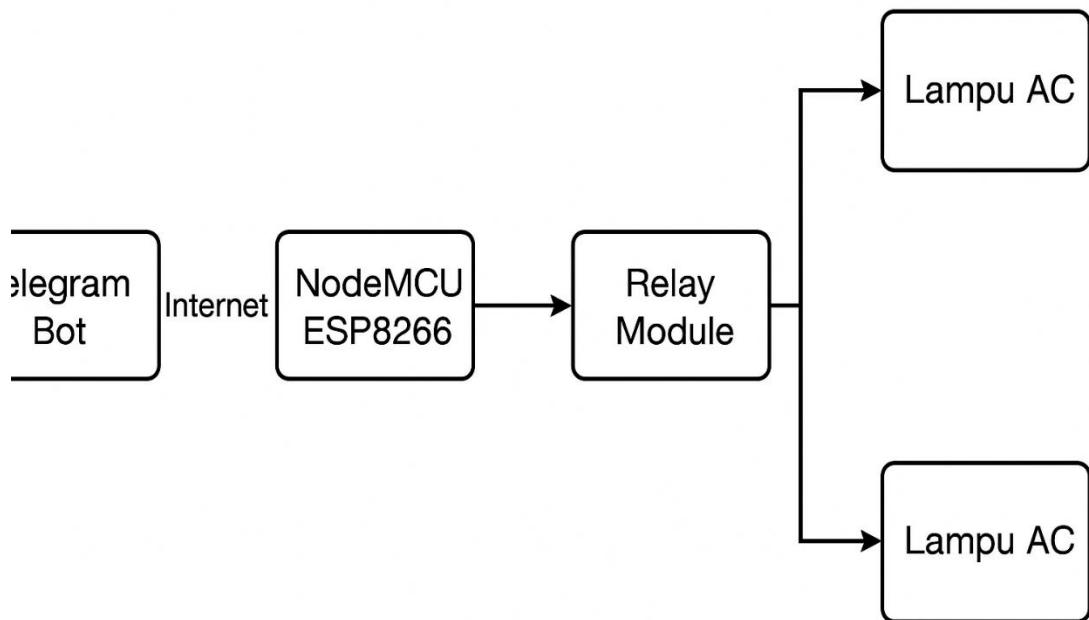
## 2.4. Topologi Cloud-to-Device

Arsitektur ini mengandalkan interaksi pengguna ke cloud server (Telegram Server), lalu mengirim data ke device (NodeMCU) melalui internet.

## III. PERANCANGAN SISTEM

### 3.1. Arsitektur Sistem

Diagram Blok:



### **3.2. Diagram Rangkaian**

<b>NodeMCU Pin</b>	<b>Relay Pin</b>	<b>Fungsi</b>
D1 (GPIO5)	IN1	Kontrol Lampu 1
D2 (GPIO4)	IN2	Kontrol Lampu 2
3V3	VCC	Catu daya relay
GND	GND	Ground

Lampu:

- Kabel fase PLN → COM1/COM2 Relay
- Kabel NO1 dan NO2 → Lampu 1 dan Lampu 2
- Kabel netral PLN → Langsung ke masing-masing lampu

### **3.3. Komponen**

<b>No</b>	<b>Komponen</b>	<b>Spesifikasi</b>
1	NodeMCU ESP8266	Wi-Fi, 3.3V logic
2	Modul Relay 4 Channel	5V, TTL
3	Lampu LED	Beban output
4	Adaptor 5V/2A	Catu daya NodeMCU
5	Kabel AC dan Jumper	Wiring
6	Telegram App	Android/iOS/Web

## **IV. IMPLEMENTASI**

### **4.1. Prosedur Bot Telegram**

- Buka @BotFather → /newbot → beri nama dan username
- Salin token API
- Token digunakan dalam sketch Arduino sebagai kunci komunikasi

### **4.2. Konfigurasi Arduino IDE**

- Install Board ESP8266
- Install Library:
  - UniversalTelegramBot.h
  - ESP8266WiFi.h
  - WiFiClientSecure.h

### 4.3. Kode Program (Sketch)

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <ArduinoJson.h> // Versi 6.x

// Ganti dengan data WiFi yang berhasil terkoneksi
const char* ssid = "OPPO A96";
const char* password = "00000000";

// Token dari BotFather (pastikan sudah benar)
const char* botToken = "7506216795:AAGq5HbIYB8_yc6dbZERRHbH-zNxp85v-aE";
const char* telegramApiBase = "https://api.telegram.org/bot";

// Pin relay
const int relay1Pin = D1;
const int relay2Pin = D2;

// Variabel untuk update Telegram
long lastUpdateId = 0;

WiFiClientSecure client;

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

    pinMode(relay1Pin, OUTPUT);
    pinMode(relay2Pin, OUTPUT);

    // Matikan relay saat boot (asumsi relay aktif LOW)
    digitalWrite(relay1Pin, HIGH);
    digitalWrite(relay2Pin, HIGH);

    // Koneksi WiFi
    WiFi.begin(ssid, password);
    Serial.print("⌚ Menghubungkan ke WiFi");
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("\n✅ Terhubung!");
    Serial.print("IP Address: ");
    Serial.println(WiFi.localIP());

    // Abaikan sertifikat SSL (tidak aman untuk produksi)
    client.setInsecure();
}

void loop() {
    if (WiFi.status() == WL_CONNECTED) {
```

```

handleTelegramUpdates();
} else {
  Serial.println("⚠ WiFi terputus, mencoba reconnect...");
  WiFi.reconnect();
}

delay(1000); // Cek pesan tiap 1 detik
}

void handleTelegramUpdates() {
HTTPClient https;

  String url = String(telegramApiBase) + botToken + "/getUpdates?offset=" +
String(lastUpdateId + 1);
  https.begin(client, url);
  int httpCode = https.GET();

  if (httpCode > 0) {
    String payload = https.getString();

    DynamicJsonDocument doc(8192);
    DeserializationError error = deserializeJson(doc, payload);

    if (error) {
      Serial.print("✖ JSON error: ");
      Serial.println(error.f_str());
      https.end();
      return;
    }

    JSONArray result = doc["result"].as<JSONArray>();

    for (JsonObject update : result) {
      lastUpdateId = update["update_id"].as<long>();
      String text = update["message"]["text"].as<String>();
      long chat_id = update["message"]["chat"]["id"].as<long>();

      Serial.print("✉ Perintah diterima: ");
      Serial.println(text);

      if (text == "/relay1_on") {
        digitalWrite(relay1Pin, LOW);
        sendMessage(chat_id, "✅ Relay 1 AKTIF");
      } else if (text == "/relay1_off") {
        digitalWrite(relay1Pin, HIGH);
        sendMessage(chat_id, "✅ Relay 1 NONAKTIF");
      } else if (text == "/relay2_on") {
        digitalWrite(relay2Pin, LOW);
        sendMessage(chat_id, "✅ Relay 2 AKTIF");
      }
    }
  }
}

```

```

        } else if (text == "/relay2_off") {
            digitalWrite(relay2Pin, HIGH);
            sendMessage(chat_id, "✅ Relay 2 NONAKTIF");
        } else {
            sendMessage(chat_id,
                "Perintah tidak dikenali.\nGunakan salah satu:\n"
                "/relay1_on\n/relay1_off\n/relay2_on\n/relay2_off");
        }
    } else {
        Serial.print("❌ Gagal akses Telegram: ");
        Serial.println(httpCode);
    }

    https.end();
}

void sendMessage(long chat_id, String text) {
    HttpClient https;
    String url = String(telegramApiBase) + botToken + "/sendMessage";

    https.begin(client, url);
    https.addHeader("Content-Type", "application/x-www-form-urlencoded");

    String postData = "chat_id=" + String(chat_id) + "&text=" + text;

    int httpCode = https.POST(postData);

    if (httpCode > 0) {
        Serial.println("✉️ Balasan terkirim");
    } else {
        Serial.print("❌ Gagal kirim pesan: ");
        Serial.println(httpCode);
    }

    https.end();
}

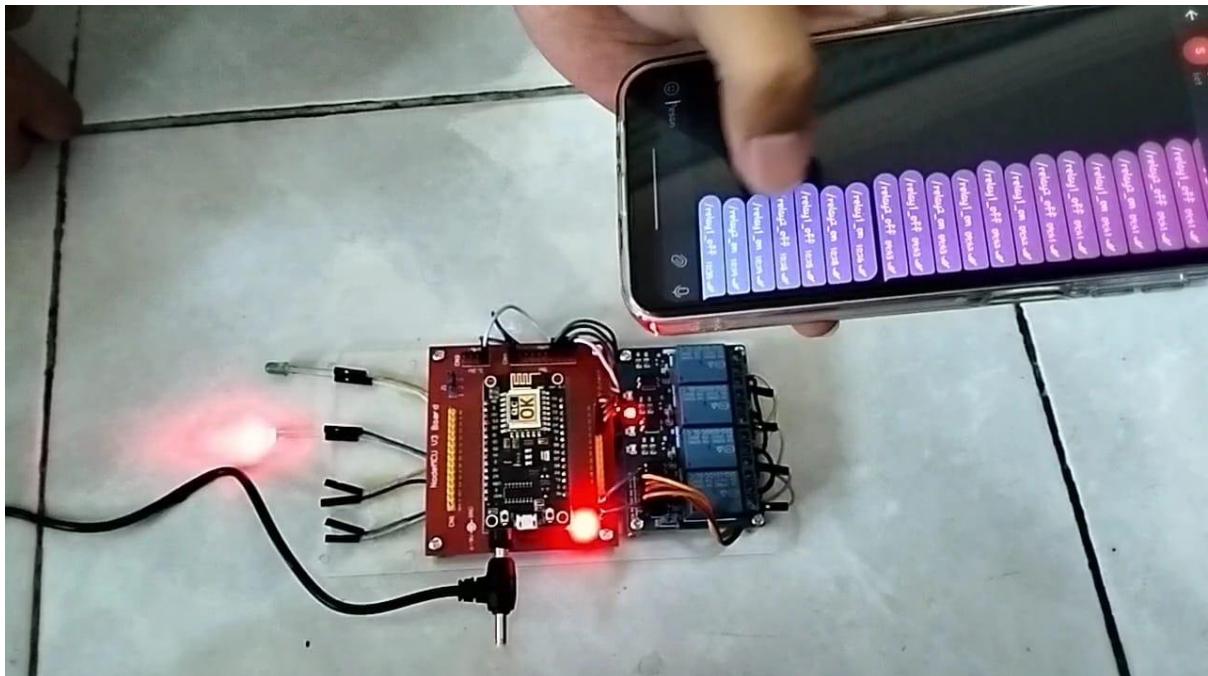
```

#### 4.4. Pengujian

/relay1\_on



/relay1\_off



/relay2\_on



/relay2\_off



<b>Uji</b>	<b>Kondisi</b>	<b>Hasil</b>
/relay1_on	Lampu 1 nyala	Berhasil
/relay1_off	Lampu 1 mati	Berhasil
/relay2_on	Lampu 2 nyala	Berhasil
/relay2_off	Lampu 2 mati	Berhasil
Tanpa Wi-Fi	Tidak terkoneksi	Tidak merespons

#### **4.5. Keamanan dan Isolasi**

- Gunakan relay dengan optocoupler
- Pastikan sambungan AC dilindungi dari sentuhan langsung
- Gunakan kabel AC standar SNI dan terminal tertutup

### **V. PENUTUP**

#### **Kesimpulan**

Sistem ini menunjukkan bahwa kontrol perangkat AC dapat dilakukan dari jarak jauh menggunakan Telegram sebagai antarmuka dan ESP8266 sebagai pengendali utama. Keamanan fisik tetap menjadi perhatian utama ketika menangani beban AC.