LAB1: Temperature Sensor with Relay Control (Telegram)

1. Overview

In this lab, you will build a tiny IoT monitoring node with an ESP32, DHT22 temperature/humidity sensor, and a relay. The ESP32 sends Telegram alerts when the temperature rises above a threshold and lets users control the relay via chat commands. Once the temperature drops below the threshold again, the relay turns off automatically**.**

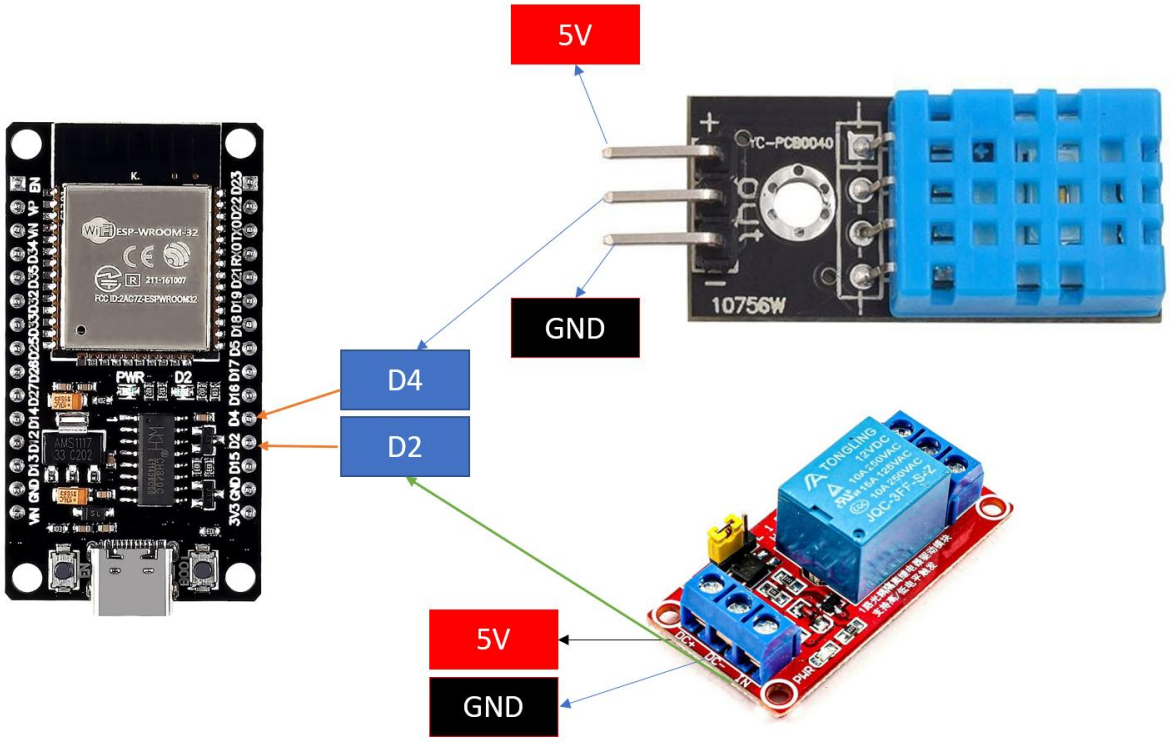
2. Learning Outcomes (CLO Alignment)

- **Design & implement** an IoT system using ESP32 + MicroPython (sensing, actuation, networking).
- **Apply programming techniques** for periodic sampling, debouncing, and simple state machines.
- **Develop a chat-based remote control** application using Telegram Bot API (HTTP requests).
- **Document & present** system design, wiring, and test evidence (screenshots/video), and reflect on reliability/ethics.
- **Evaluate** performance (sampling interval, rate limits) and **safety** (relay loads, power isolation).

3. Equipment

- ESP32 Dev Board (MicroPython firmware flashed)
- DHT22 sensor
- Relay module
- jumper wires
- USB cable + laptop with **Thonny**
- Wi-Fi access (internet)

4. Wiring



5. Tasks & Checkpoints

Task 1-Sensor Read & Print (10 pts)

- Read DHT22 every 5 seconds and print the temperature and humidity with 2 decimals.
- Evidence: serial screenshot.

Task 2-Telegram Send (15 pts)

- Implement send_message() and post a test message to your group.
- Evidence: chat screenshot.

Task 3-Bot Command (15 pts)

- Implement /status to reply with current T/H and relay state.
- Implement /on and /off to control the relay.
- Evidence: chat screenshot showing all three commands working.

Task 4-Bot Command (20 pts)

- **No messages** while T < 30 °C.
- If **T ≥ 30 °C** and relay is **OFF**, send an alert **every loop** (5 s) until /on is received.
- After /on, **stop alerts**. When **T < 30 °C**, turn relay **OFF automatically** and send a one-time "auto-OFF" notice.
- Evidence: short video (60–90s) demonstrating above behavior.

Task 5-Robustness (10 pts)

- Auto-reconnect Wi-Fi when dropped.
- Handle Telegram HTTP errors (print status; skip this cycle on failure).
- Avoid crashing on DHT OSError (skip cycle).

Task 6-Document(30 pts)

- **README.md** with wiring diagram/photo, configuration steps (token, chat id), and usage instructions.
- Include a block diagram or flowchart of your loop/state.

6. Submission & Academic Integrity

Submit a **private GitHub repo** (add instructor as collaborator). Include:

- Source code (.py files)
- README.md with setup/usage and wiring photo
- **Screenshots** of Telegram chat (/status, alerts)
- **Short demo video** (link) showing behavior under T ≥ 30 °C then cool-down