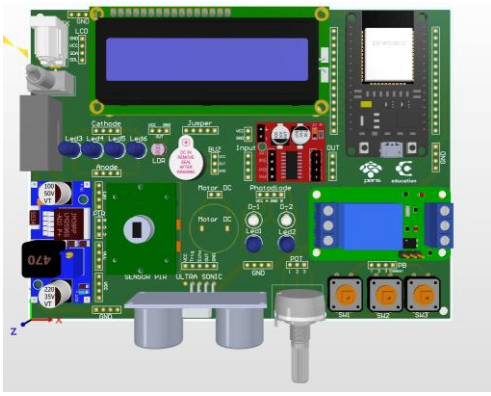
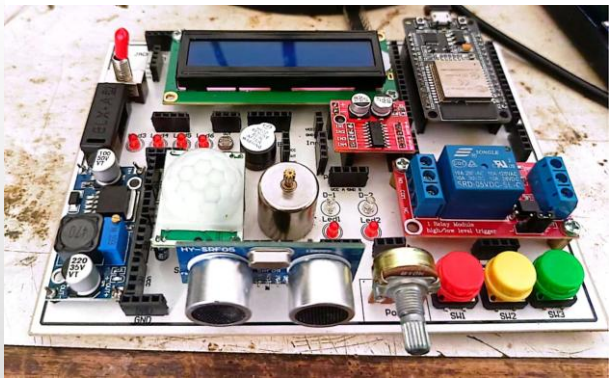
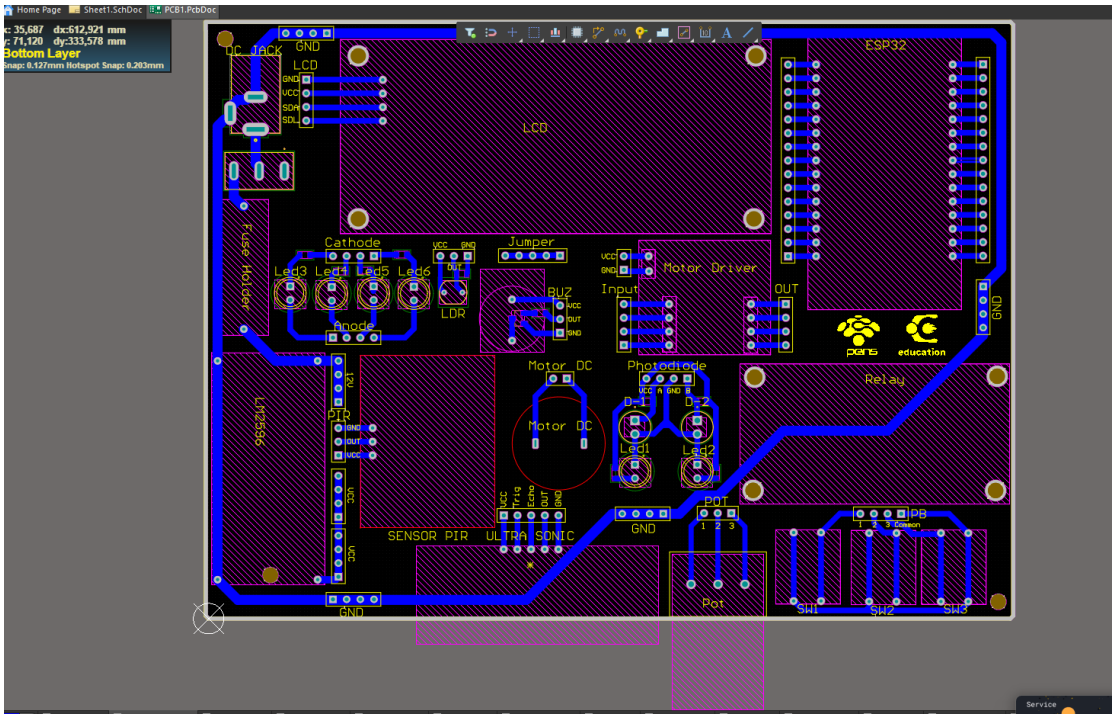
Dokumentasi Modul Pembelajaran IoT Menggunakan ESP32

1. Pendahuluan

Modul pembelajaran IoT berbasis ESP32 dirancang untuk mempermudah praktikum dan eksperimen dengan sensor serta aktuator. ESP32 dipilih karena sudah memiliki WiFi + Bluetooth sehingga dapat langsung terhubung ke MQTT broker tanpa memerlukan mikrokontroler tambahan.







Modul ini cocok digunakan dalam pembelajaran IoT, Smart Home, dan Smart Building, karena menyediakan berbagai sensor dan aktuator yang siap dipakai hanya dengan jumper.

2. Deskripsi Sistem

Sistem ini menggunakan **ESP32** sebagai mikrokontroler utama yang terhubung ke jaringan WiFi dan berkomunikasi dengan **broker MQTT**. ESP32 akan membaca data dari sensor, mengendalikan aktuator (seperti LED atau buzzer), dan menampilkan informasi pada LCD.

Data sensor (misalnya suhu dan kelembaban dari sensor DHT22, atau jarak dari sensor ultrasonik HC-SR04) akan dikirim ke broker MQTT melalui topik tertentu. Selain itu, ESP32 juga dapat menerima perintah dari broker untuk mengontrol perangkat, seperti menyalakan atau mematikan LED.

LCD yang terhubung ke ESP32 digunakan untuk menampilkan data sensor secara real-time serta status koneksi MQTT. Dengan demikian, sistem ini dapat dimanfaatkan sebagai **platform monitoring dan kontrol IoT sederhana** berbasis ESP32.

Alur kerjanya:

1. ESP32 terkoneksi ke WiFi dan broker MQTT.
2. Sensor membaca data (misalnya suhu, kelembaban, atau jarak).
3. Data dikirim ke broker MQTT dengan topik tertentu.
4. Data dapat ditampilkan di aplikasi MQTT client (misalnya **MQTTX**, **Node-RED**, atau **mosquitto_sub**).
5. Broker MQTT juga dapat mengirimkan perintah ke ESP32 (misalnya menyalakan LED).
6. LCD menampilkan informasi sensor dan status koneksi.

3. Sensor yang Digunakan

- **LDR (Light Dependent Resistor)** → sensor cahaya.

- **Ultrasonic HC-SR04** → sensor jarak/ketinggian air.

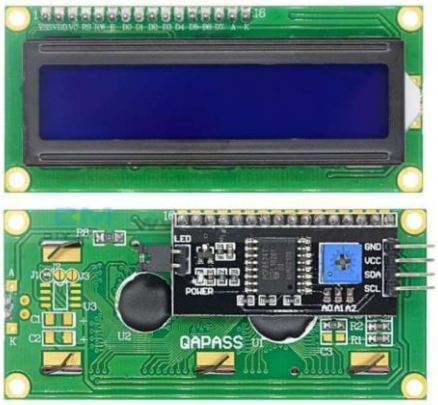
- **PIR Sensor** → mendeteksi gerakan manusia.

- **Potensiometer** → input analog (simulasi sensor).

- **Relay** → kendali peralatan listrik.

- **DC Motor + Driver MX1508** → aktuator.

- **LCD 16x2 I2C** → menampilkan data sensor.



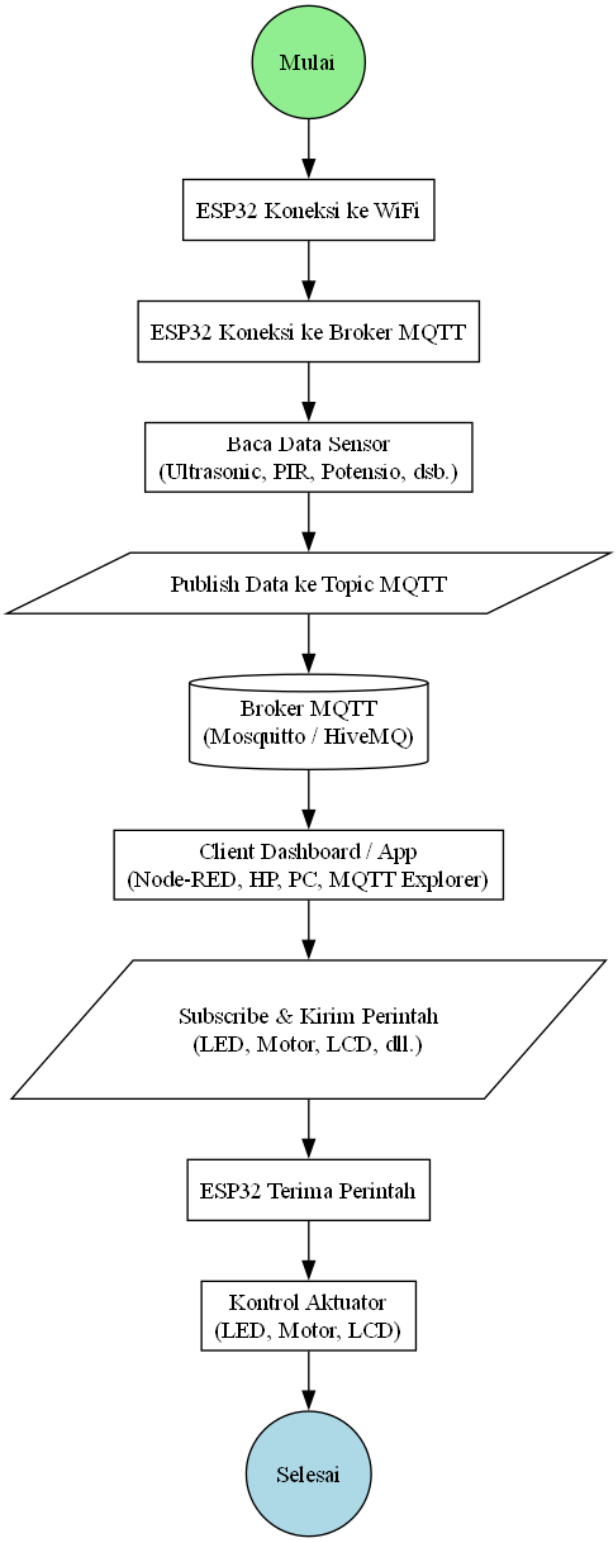
- **ESP32** → komunikasi WiFi & MQTT broker.

4. Alur Sistem

- **Inisialisasi ESP32**
  - ESP32 dinyalakan → terkoneksi ke jaringan WiFi.
  - ESP32 melakukan koneksi ke **broker MQTT** (contoh: Mosquitto lokal atau broker publik seperti HiveMQ).
- **Akuisisi Data Sensor**
  - ESP32 membaca data dari sensor yang terpasang pada modul (misalnya: Ultrasonik → jarak, Potensiometer → nilai analog, PIR → deteksi gerakan, dsb).
- **Publikasi Data ke MQTT**
  - Data sensor dikemas dalam format string/JSON.
  - ESP32 mengirimkan data ke **topic MQTT** tertentu, misalnya:
    - `iot/esp32/ultrasonic`
    - `iot/esp32/lcd`
    - `iot/esp32/led`
- **Distribusi Data**
  - Broker MQTT menerima data, lalu mendistribusikannya ke semua client yang **subscribe** ke topic tersebut.
  - Client bisa berupa:
    - Aplikasi **Node-RED Dashboard**
    - **MQTT Explorer**
    - Aplikasi Android/Website IoT
    - ESP32 lain
- **Kontrol Aktuator (Subscriber)**
  - Client (misalnya Node-RED) mengirimkan perintah ke ESP32 melalui topic lain, misalnya:
    - `iot/esp32/led/set` → ON/OFF LED
    - `iot/esp32/motor/set` → kontrol kecepatan motor

ESP32 yang sudah subscribe akan menerima pesan lalu mengeksekusinya

5. Diagram Alur Data



6. Cara Menggunakan Modul Pembelajaran IoT

1. Perlengkapan / Komponen

- Modul Pembelajaran IoT
- Kabel data microusb/type C
- Kabel jumper Male to male.

---

2. Pin mapping (rekomendasi untuk ESP32 DevKit)

Catatan: jangan gunakan pin input-only untuk output. Pin ADC (34–39) hanya input.

```
SDA (I2C LCD)  -> GPIO 21
SCL (I2C LCD)  -> GPIO 22
LDR (ADC)      -> GPIO 34
Potensiometer  -> GPIO 35
PIR            -> GPIO 13
Ultrasonic TRIG-> GPIO 5
Ultrasonic ECHO-> GPIO 18
Button         -> GPIO 14  (gunakan INPUT_PULLUP)
LED (built-in) -> GPIO 2
Relay          -> GPIO 26
Buzzer         -> GPIO 27
Motor IN1      -> GPIO 32
Motor IN2      -> GPIO 33
Motor EN (PWM) -> GPIO 25  (ledc PWM)
```

Sesuaikan kalau boardmu beda.

---

3. Wiring singkat (perangkat → pin ESP32)

- LCD I2C : SDA → 21, SCL → 22, VCC → 5V, GND → GND
- LDR : satu ujung LDR → 3.3V, ujung lain → ADC pin 34 dan juga ke resistor 10k ke GND (pembagi tegangan)
- Potensiometer : tengah → ADC 35, satu ujung → 3.3V, satunya → GND
- PIR : VCC → 5V (atau 3.3V tergantung modul), GND → GND, OUT → 13
- Ultrasonic : VCC → 5V, GND → GND, TRIG → 5, ECHO → 18 (beberapa modul HC-SR04 bekerja ok pada 5V; jika echo 5V, gunakan pembagi level atau modul yang aman)
- Push Button : satu kaki ke GPIO14, satunya ke GND, gunakan INPUT_PULLUP → baca LOW saat ditekan
- LED: LED+ → resistor 220Ω → GPIO2, LED- → GND
- Relay: IN → GPIO26, VCC → 5V, GND → GND (pastikan module relay cocok)
- Buzzer: + → GPIO27, - → GND
- Motor via L298N: IN1 → 32, IN2 → 33, ENA → 25 (PWM), motor power separate 12V/5V (sesuaikan)

- *PENTING:* beri ground bersama (ESP32 GND dan motor power GND harus sama).

---

4. Persiapan software (Arduino IDE — cara cepat)

1. Install Arduino IDE (atau gunakan PlatformIO).
2. Tambahkan ESP32 ke Board Manager:
   - File → Preferences → Additional Boards Manager URLs: `https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json`
   - Tools → Board → Boards Manager → cari `esp32` → Install.
3. Install libraries (Library Manager):
   - `PubSubClient` (untuk MQTT)
   - `ArduinoJson` (untuk JSON)
   - `LiquidCrystal_I2C` (untuk LCD I2C)
   - (opsional) `NewPing` (untuk ultrasonic, jika mau)
4. (Opsional) MQTT Broker:
   - Untuk percobaan cepat pakai broker publik `test.mosquitto.org`.
   - Untuk lokal, install Mosquitto di Linux: `sudo apt install mosquitto mosquitto-clients` (atau pakai broker di PC/server).
   - Gunakan `MQTT Explorer` atau `mosquitto_sub` untuk melihat topik.

---

5. Struktur folder proyek (rekomendasi)

```
firmware/
└── examples/
└── main_project/
    └── esp32_all_sensors/   # file utama: esp32_all_sensors.ino
docs/
README.md
```

---

6. Contoh format data (JSON)

ESP32 akan publish ke topic `iot/module1/data` payload JSON misalnya:

```
{
  "device_id":"esp32_01",
  "timestamp":"2025-09-14T21:30:00",
  "light":350,
  "distance":45,
  "pir":1,
  "potensio":512,
  "button":0
}
```

---

7. Sketch terintegrasi (siap pakai)

Salin kode di bawah, ganti `SSID`, `PASSWORD`, `mqtt_server` lalu upload ke ESP32. Kode ini:

- Konek WiFi + MQTT
- Membaca sensor sesuai pin mapping
- Publish JSON ke `iot/module1/data` setiap 2 detik
- Subscribe ke `iot/module1/cmd` untuk perintah JSON (led/relay/buzzer/motor/lcd)
- Tampilkan ringkasan di LCD

---

8. Cara pakai / uji coba

1. Edit `NAMA_WIFI`, `PASSWORD_WIFI`, dan `mqtt_server` di sketch.
2. Upload ke ESP32 via Arduino IDE. Pilih board `ESP32 Dev Module` dan COM port yang benar.
3. Buka Serial Monitor (115200) untuk cek log WiFi/MQTT.
4. Lihat topic `iot/module1/data`:
    - Gunakan `mosquitto_sub` (jika pakai Mosquitto lokal):
    - `mosquitto_sub -h test.mosquitto.org -t "iot/module1/data" -v`
    - Atau gunakan MQTT Explorer untuk subscribe topik dan lihat pesan JSON.
5. Untuk mengirim perintah (contoh nyalakan LED), publish JSON ke `iot/module1/cmd`:
    - Payload contoh:
    - `{"led":"ON"}`
    - Contoh untuk motor maju kecepatan 200:
    - `{"motor":"FWD", "motor_speed":200}`
    - Contoh menulis ke LCD:
    - `{"lcd":"Hello Class!"}`

---

9. Troubleshooting umum

- ESP32 tidak konek WiFi → cek SSID/password dan jarak sinyal.
- MQTT tidak connect → periksa `mqtt_server` reachable, firewall, dan port 1883.
- Ultrasonic baca -1 → pastikan wiring ECHO/TRIG benar; echo dari HC-SR04 adalah 5V → ESP32 input mungkin butuh divider (beberapa modul aman).
- ADC nilai terlalu besar → ESP32 ADC range 0–4095 (12-bit default); di sketch aku map ke 0–1023 untuk kemudahan.
- Motor/relay butuh power terpisah → gunakan supply terpisah dan pastikan GND bersama.

---

10. Saran pengembangan (untuk modul pembelajaran)

- Tambahkan NTP agar timestamp akurat.
- Simpan data ke database (InfluxDB/MongoDB) melalui broker atau server.
- Buat dashboard di Node-RED atau web (Flask/React).
- Tambahkan OTA update agar firmware bisa di-update lewat WiFi.
- Tambahkan proteksi level shifting untuk sensor 5V → ESP32 3.3V.

7. Kesimpulan

Modul IoT berbasis **ESP32** dengan dukungan berbagai sensor (LDR, Ultrasonic, PIR, Potensiometer, Push Button) dan aktuator (LED, Motor, LCD) memberikan sarana pembelajaran yang lengkap bagi pemula untuk memahami konsep dasar **Internet of Things (IoT)**.

Dengan menghubungkan ESP32 ke **Broker MQTT**, data sensor dapat dipublikasikan dan dipantau secara real-time melalui berbagai client (PC, smartphone, atau dashboard seperti Node-RED). Selain itu, sistem ini juga memungkinkan pengiriman perintah dari client untuk mengendalikan aktuator.

Keseluruhan percobaan menunjukkan bahwa:

1. **ESP32** mampu menjadi mikrokontroler utama yang handal untuk aplikasi IoT.
2. **MQTT** adalah protokol komunikasi ringan yang cocok untuk pertukaran data sensor secara efisien.
3. Struktur project yang modular memudahkan pemula untuk mencoba sensor/aktuator secara bertahap hanya dengan “colok-jumper”.
4. Sistem dapat dikembangkan lebih lanjut dengan menambah sensor baru, memperluas format data JSON, dan mengintegrasikan ke aplikasi mobile atau web.

Dengan modul ini, peserta dapat memahami **alur sistem IoT dari sensor → ESP32 → MQTT broker → client → aktuator**, sekaligus memperoleh pengalaman langsung dalam praktik pemrograman dan komunikasi IoT.