ALAT MEMINDAI JUMLAH ORANG DI RUANGAN

**Deskripsi**

Proyek ini bertujuan untuk membuat sebuah sistem otomatis yang dapat mendeteksi dan menghitung jumlah orang yang berada di dalam sebuah ruangan. Alat ini akan memberikan informasi jumlah orang secara real-time dan mampu mengurangi hitungan ketika seseorang keluar dari ruangan. Selain itu, sistem ini dilengkapi dengan notifikasi berupa suara dan indikator visual ketika ruangan mencapai kapasitas maksimal. Dengan menggunakan ESP32 yang terhubung ke platform MQTT, sistem ini mampu mengirim dan menerima data yang dapat diakses secara jarak jauh.

**Cara Kerja**

1. Saat seseorang melewati pintu masuk, sensor akan mendeteksi orang tersebut dan menambah hitungan jumlah orang di dalam ruangan.
2. Ketika seseorang keluar melalui pintu keluar, hitungan jumlah orang di dalam ruangan akan berkurang.
3. Jika ruangan mencapai kapasitas maksimum, akan ada pemberitahuan melalui speaker (buzzer) dan lampu LED akan menyala sebagai indikasi bahwa ruangan penuh.

**Alat dan bahan**

* Esp -32
* Sensor ( Ultrasonic ) ada 2 sensor untuk in dan out
* Speaker ( buzer )
* Kabel jumper
* Lcd + i2c
* Lampu led
* Kabel jumper
* Papan roti ( bread bord )

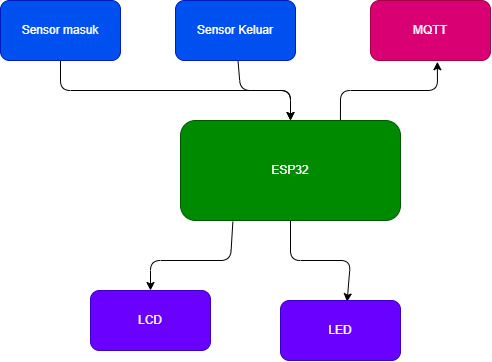
**Bahan tambahan**

* Kardus
* Lem

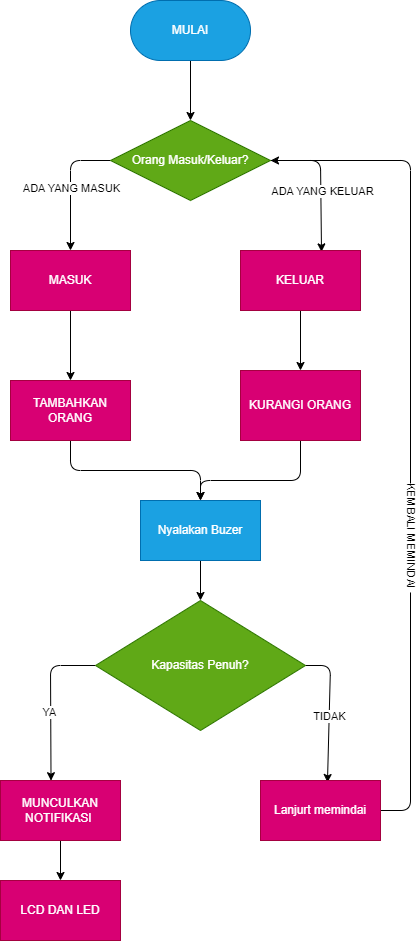
**Tambahan**

* Sistem ini menggunakan MQTT untuk mengirim dan menerima data antara ESP32 dan server.
* Jumlah orang yang berada di dalam ruangan akan ditampilkan di LCD secara real-time.
* Lampu LED akan menyala dan buzzer akan berbunyi jika ruangan sudah penuh.

**Diagram Block**

\

**Diagram alir**



**Code program sementara**

|  |
| --- |
| #include <Wire.h> |
| #include <LiquidCrystal\_I2C.h> |
| #include <WiFi.h> |
| #include <PubSubClient.h> |
|  |
| // Konfigurasi untuk LCD |
| LiquidCrystal\_I2C lcd(0x27, 16, 2); // Alamat I2C LCD bisa berbeda, sesuaikan |
|  |
| // WiFi Credentials |
| const char\* ssid = "Nama\_SSID"; // Ganti dengan nama WiFi |
| const char\* password = "Password\_WiFi"; // Ganti dengan password WiFi |
|  |
| // MQTT Broker |
| const char\* mqtt\_server = "broker.hivemq.com"; // Ganti dengan alamat broker MQTT |
|  |
| WiFiClient espClient; |
| PubSubClient client(espClient); |
|  |
| // Variabel untuk sensor |
| const int sensorMasuk = 13; // Sensor untuk pintu masuk |
| const int sensorKeluar = 12; // Sensor untuk pintu keluar |
| int jumlahOrang = 0; // Jumlah orang di dalam ruangan |
|  |
| // Pin untuk LED dan buzzer |
| const int ledPin = 14; |
| const int buzzerPin = 15; |
| const int kapasitasMaksimal = 5; // Atur kapasitas maksimal ruangan |
|  |
| void setup() { |
| // Inisialisasi LCD |
| lcd.begin(); |
| lcd.backlight(); |
|  |
| // Inisialisasi Serial |
| Serial.begin(115200); |
|  |
| // Inisialisasi sensor dan output |
| pinMode(sensorMasuk, INPUT); |
| pinMode(sensorKeluar, INPUT); |
| pinMode(ledPin, OUTPUT); |
| pinMode(buzzerPin, OUTPUT); |
|  |
| // Setup WiFi dan MQTT |
| setup\_wifi(); |
| client.setServer(mqtt\_server, 1883); |
| } |
|  |
| void setup\_wifi() { |
| delay(10); |
| Serial.println(); |
| Serial.print("Menghubungkan ke "); |
| Serial.println(ssid); |
|  |
| WiFi.begin(ssid, password); |
|  |
| while (WiFi.status() != WL\_CONNECTED) { |
| delay(500); |
| Serial.print("."); |
| } |
|  |
| Serial.println(""); |
| Serial.println("WiFi tersambung"); |
| } |
|  |
| void reconnect() { |
| // Loop sampai terhubung kembali ke MQTT |
| while (!client.connected()) { |
| Serial.print("Menghubungkan ke MQTT..."); |
| if (client.connect("ESP32\_Ruangan")) { |
| Serial.println("terhubung"); |
| } else { |
| Serial.print("gagal, rc="); |
| Serial.print(client.state()); |
| Serial.println(" coba lagi dalam 5 detik"); |
| delay(5000); |
| } |
| } |
| } |
|  |
| void loop() { |
| if (!client.connected()) { |
| reconnect(); |
| } |
| client.loop(); |
|  |
| // Deteksi orang masuk |
| if (digitalRead(sensorMasuk) == HIGH) { |
| jumlahOrang++; |
| updateStatus(); |
| delay(1000); // Debounce |
| } |
|  |
| // Deteksi orang keluar |
| if (digitalRead(sensorKeluar) == HIGH && jumlahOrang > 0) { |
| jumlahOrang--; |
| updateStatus(); |
| delay(1000); // Debounce |
| } |
|  |
| // Periksa apakah ruangan penuh |
| if (jumlahOrang >= kapasitasMaksimal) { |
| digitalWrite(ledPin, HIGH); |
| digitalWrite(buzzerPin, HIGH); |
| Serial.println("Ruangan penuh"); |
| client.publish("ruangan/status", "Ruangan penuh"); |
| } else { |
| digitalWrite(ledPin, LOW); |
| digitalWrite(buzzerPin, LOW); |
| } |
| } |
|  |
| void updateStatus() { |
| // Tampilkan jumlah orang di LCD |
| lcd.clear(); |
| lcd.setCursor(0, 0); |
| lcd.print("Orang di ruangan:"); |
| lcd.setCursor(0, 1); |
| lcd.print(jumlahOrang); |
|  |
| // Kirim update jumlah orang ke server MQTT |
| char jumlahOrangStr[10]; |
| itoa(jumlahOrang, jumlahOrangStr, 10); |
| client.publish("ruangan/jumlahOrang", jumlahOrangStr); |
|  |
| Serial.print("Jumlah orang: "); |
| Serial.println(jumlahOrang); |
| } |

**KONEKSI PIN KE ESP32**

**Sensor PIR (Masuk dan Keluar):**

* **Sensor Masuk**:
  + Pin **VCC** (Sensor PIR) ke **3.3V** (ESP32)
  + Pin **GND** (Sensor PIR) ke **GND** (ESP32)
  + Pin **OUT** (Sensor PIR) ke **GPIO 13** (ESP32)
* **Sensor Keluar**:
  + Pin **VCC** (Sensor PIR) ke **3.3V** (ESP32)
  + Pin **GND** (Sensor PIR) ke **GND** (ESP32)
  + Pin **OUT** (Sensor PIR) ke **GPIO 12** (ESP32)

**LCD I2C:**

* Pin **VCC** (LCD I2C) ke **3.3V** (ESP32)
* Pin **GND** (LCD I2C) ke **GND** (ESP32)
* Pin **SDA** (LCD I2C) ke **GPIO 21** (ESP32)
* Pin **SCL** (LCD I2C) ke **GPIO 22** (ESP32)

**LED:**

* **Anoda** (Kaki panjang, positif) dari LED ke **GPIO 14** (ESP32)
* **Katoda** (Kaki pendek, negatif) dari LED ke **GND** (ESP32)

**Buzzer:**

* **Positif** (Buzzer) ke **GPIO 15** (ESP32)
* **Negatif** (Buzzer) ke **GND** (ESP32)

**Ringkasan Pin:**

* **Sensor PIR Masuk**:
  + OUT → GPIO 13
* **Sensor PIR Keluar**:
  + OUT → GPIO 12
* **LCD I2C**:
  + SDA → GPIO 21
  + SCL → GPIO 22
* **LED**:
  + Positif → GPIO 14
* **Buzzer**:
  + Positif → GPIO 15