

**Düzce Üniversitesi**

**Mühendislik Fakültesi**

**Bilgisayar Mühendisliği Bölümü**

**BM471- Gömülü Sistem Uygulamaları**

**Final Projesi Raporu**

**İsim-Soy İsim: Enes Bilal Şeker**

**Numara: 171001020**

**Öğrenim Türü: Normal Öğretim**

**Ocak 2022**

**Özet**

Bu proje ile yaklaşık iki senedir Dünya’yı etkisi altına almış Covid-19 salgınının bulaşma etkisinin minimuma indirgenmesi amaçlanmıştır. Proje hazırlanırken virüsün bulaşma hızını en aza indiren kapalı mekan havalandırması ve sosyal mesafe önlemleri dikkate alınmıştır. Mesafe ve gaz sensörleri kullanılarak kişinin sosyal mesafesi ve girdiği ortamdaki CO2 miktarı ölçülüp, bu önlemler ihlal edildiğinde kişiye uyarı mesajı gönderilmesi amaçlanmıştır. Uyarı mesajını (SMS) telefona gönderebilmek için IFTTT (If This Than That) adlı web hizmetinden yararlanılmıştır.

1. **GİRİŞ**

Covid-19 ile mücadelenin Dünya Sağlık Örgütü (WHO) tarafından belirlenmiş birtakım kriterleri vardır [1]. Projede bu kriterlerden iki tanesini kullanıldı. Bunlar;

kişinin sosyal mesafesini güvenli bir aralıkta tutup tutmadığının tespiti,

girilen kapalı bir ortamın havalandırmasının yeterli olup olmadığının tespiti.

Dünya Sağlık Örgütü’ne göre kişinin güvende olması için etrafındaki diğer kişiler ile arasında en az bir metrelik bir mesafe bulundurması gerekiyor [2]. Bu mesafe HC-SR04 ultrasonik mesafe sensörü [3] ile ölçülmüştür. Sensörden ölçülen değer NodeMCU yardımıyla işlenmiş ve anlamlı çıktılara dönüştürülmüştür. Buradan çıkan değer kod ile ayarlanıp kişinin sosyal mesafeyi (bir metrelik sınırı) aştığı anlarda kendisine uyarı SMS’i gönderilmesi sağlanmıştır.

Yine Dünya Sağlık Örgütü’ne göre kapalı bir alanın eğer iyi bir havalandırması yoksa orada uzun süreler kalmak virüse maruz kalma oranınızı artırır. Şu haber makalesine göre [4] ortamdaki CO2 miktarı milyonda 800 parçadan (800 ppm) daha fazla ise, ortamdaki havalandırmanın yeterli olmadığı söylenebilir. Bu miktar MQ-135 [5] gaz sensörü yardımıyla ölçülmüştür. MQ sensörlerin ortamdaki sıcaklık ve neme göre ölçüm hassasiyetinin değişmesi göz önüne alınarak 0 ila 1024 arasında ayarlanan ölçüm hassasiyeti, sensör 1000’in üzerinde ölçüm yaptığı zaman kişiye uyarı SMS’i gönderecek şekilde ayarlanmıştır.

Kullanılan bütün sensörler breadboard yardımıyla NodeMCU mikrodenetleyicisine bağlanmıştır. Bütün kontrolleri ve ölçüm değerleri NodeMCU’a atılan kod sayesinde kullanılabilir hale getirilmiştir.

Uyarı SMS’i için IFTTT web hizmeti kullanılmıştır. Kısaca bu hizmetten yararlanmak için bir senaryo oluşturmanız gerekiyor ve daha sonra oluşturduğunuz senaryonuzun url’sini alıp kodunuzun içinde https isteği göndermeniz gerekiyor.

1. **MATERYAL VE YÖNTEM**
   1. **Kullanılan Malzemeler**

NodeMCU, HC-SR04, MQ-135, Breadboard, Jumper Kablo.

* 1. **Covid Önleyici Kolye**

Proje teoride kişinin dışarıya çıkarken boynuna takacağı giyilebilir bir teknoloji gibi çalışması için tasarlanmıştır. Kişi dışarıya çıkarken maskesini taktığı gibi bu kolyeyi de takacak böylece virüsten daha kapsamlı bir koruma sağlanacak.

Sensörlerden okunan değerler NodeMCU yardımıyla işlenecek ve yazdığımız koda uygun bir şekilde uygulamaya koyulacak. Belirlediğimiz koşulların sağlanması durumunda kişiye SMS ile uyarı yollayacak.

Uyarı SMS’inin başarılı bir şekilde ulaşabilmesi için hem NodeMCU’nun hem mesajın geleceği cihazın internete bağlı olması gerekmektedir. Bu projede NodeMCU cihazı mesajın geldiği telefondan açılmış WiFi hotspot noktasına bağlanmaktadır. Cihazlar bu şekilde haberleşme sağlamaktadır.

1. **SONUÇLAR VE TARTIŞMA**

Projenin sonunda her şey doğru ve başarılı bir şekilde çalışıyor. Devre, kod, bağlantılar hiçbirinde bir sıkıntı yaşanmadı. Ölçümler ve uyarı mesajları tam istenilen koşullarda, istenilen sıklıkta gönderildi. Sorunsuz çalışma bakımından kullanılmaya hazır bir cihaz ortaya çıktı.

Fakat başta tasarlandığı gibi cihazın giyilebilir özelliği entegre edilemedi. Cihaz şu an devrenin üzerinde çalışır vaziyette ve kolye olarak takılmaya uygun değil. Bunun nedenleri;

sensörlerin ve mikrodenetleyicinin çok büyük olması,

bağlantı için breadboard gereksinimi,

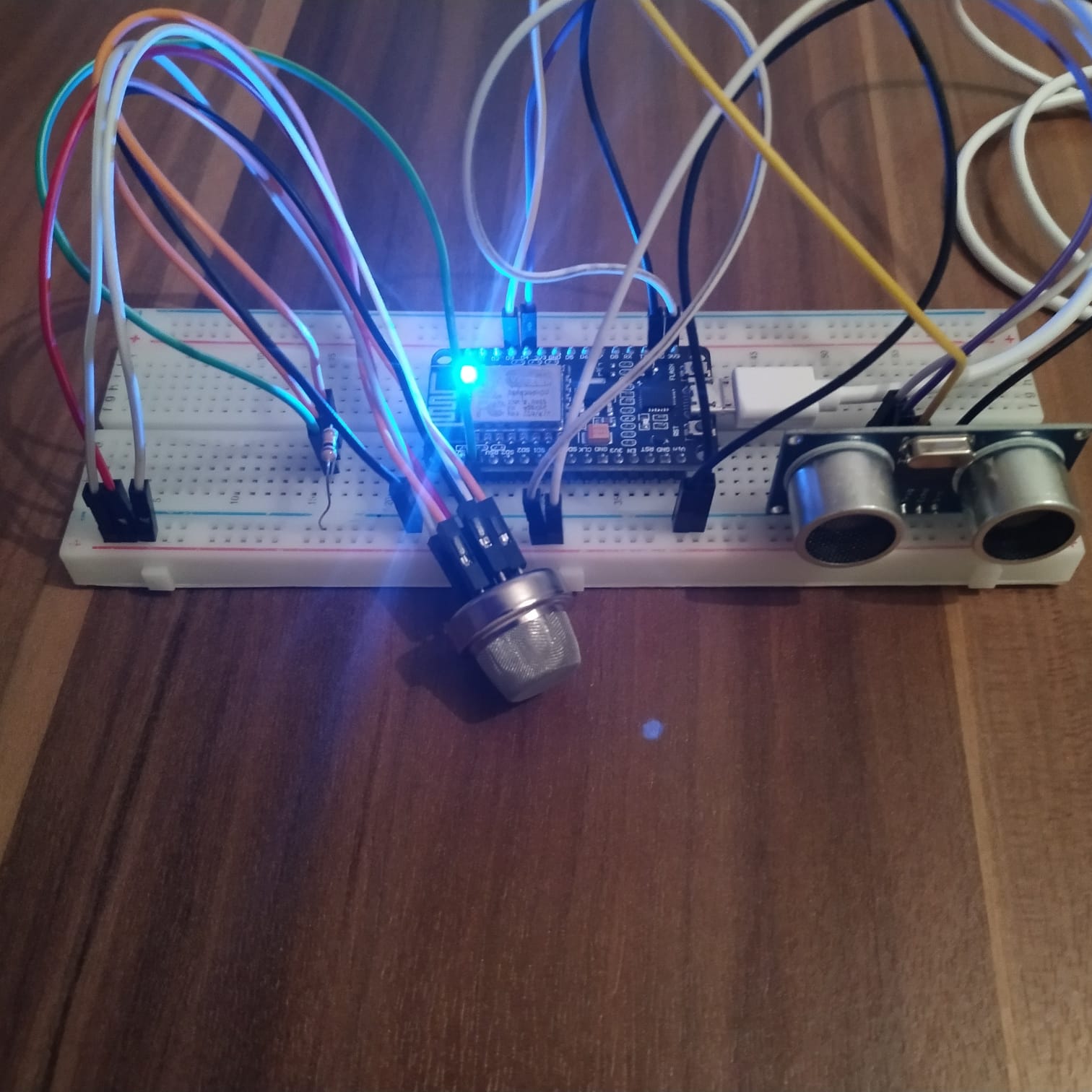
NodeMCU’nun devamlı bir güç kaynağına bağlı olmasının gerekmesi ve bunun şu anlık ancak powerbank ile sağlanıyor olması.

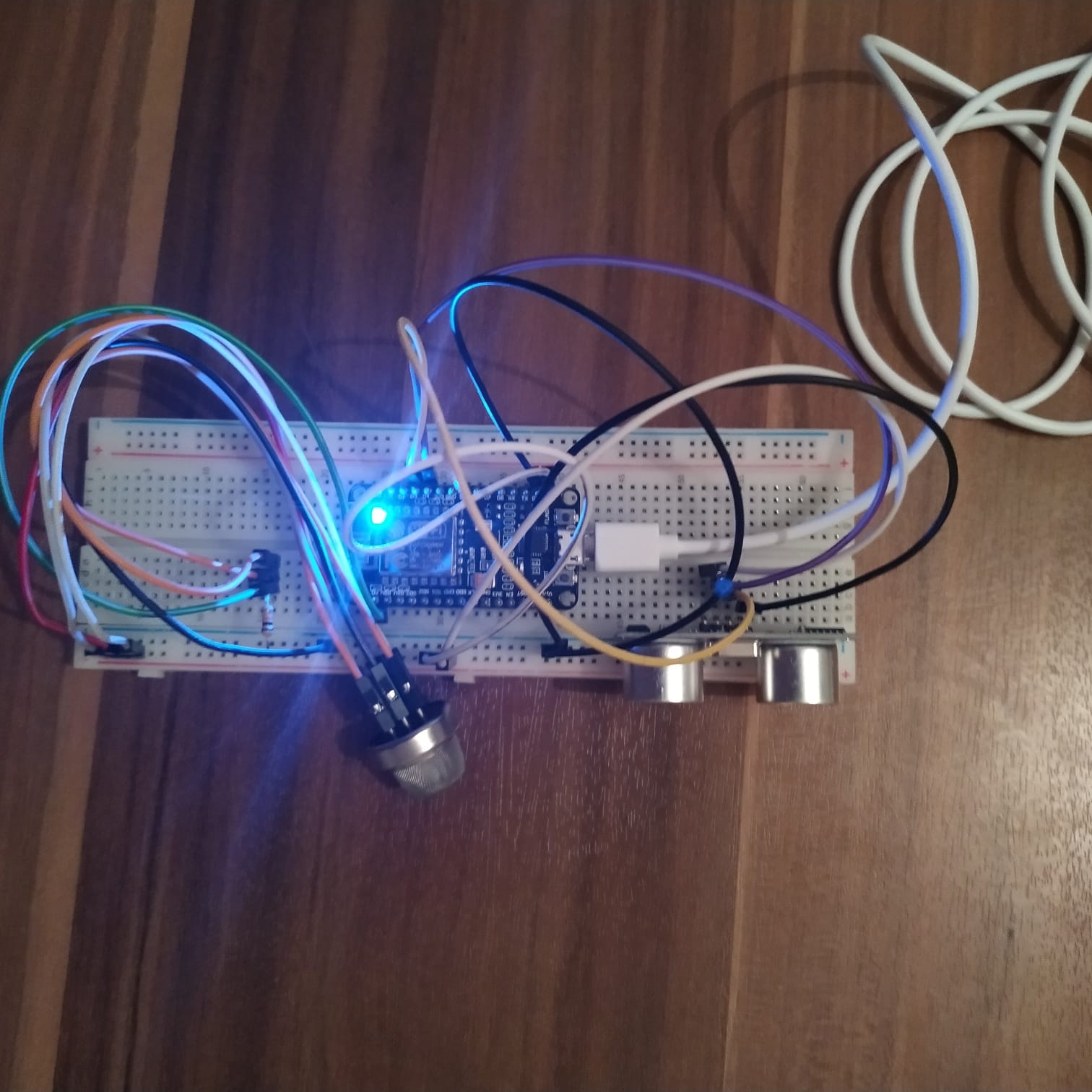
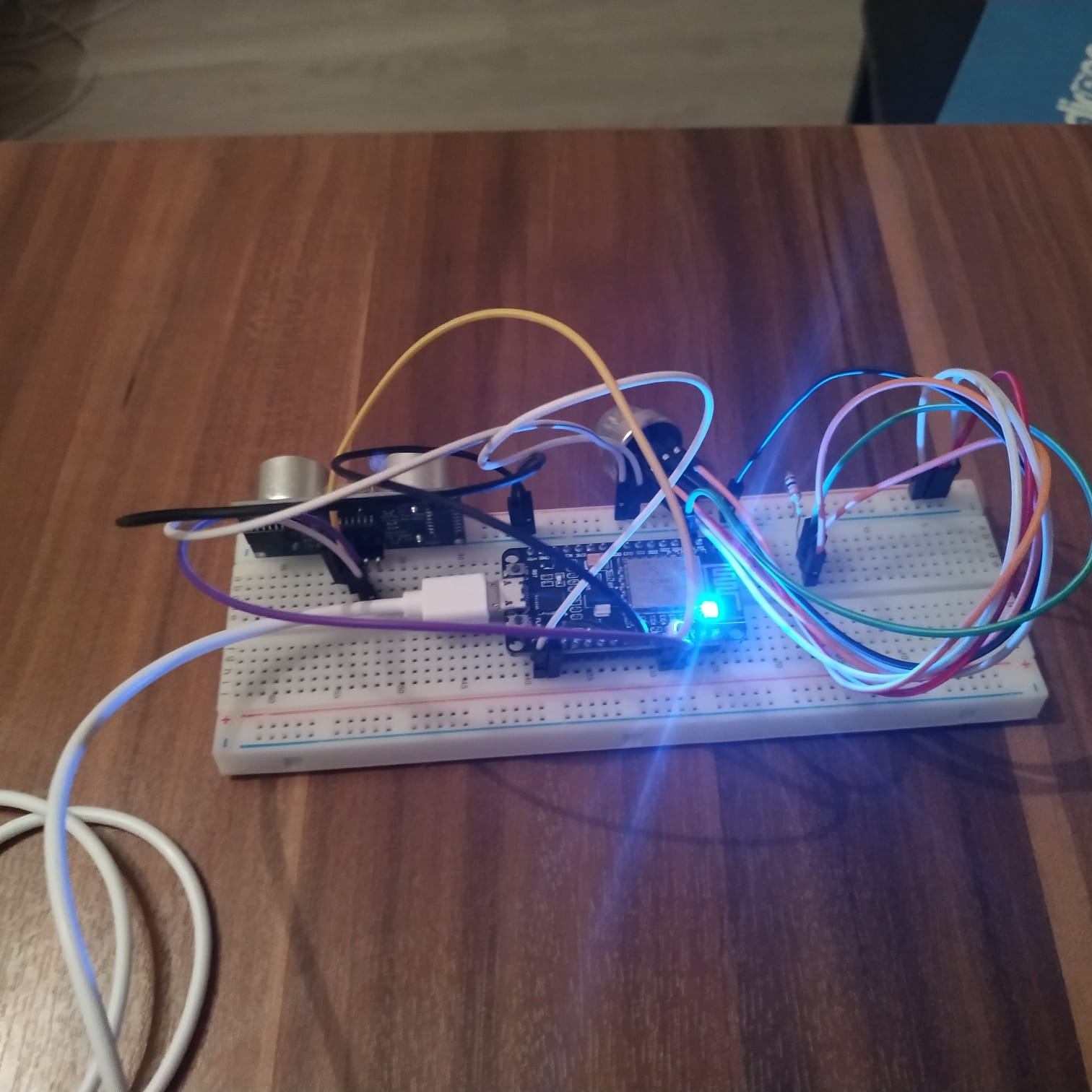
Cihazın çalışmasına engel olmayan fakat bazı durumlarda doğru çalışmasını baltalayacak bir neden de MQ-135 gaz sensörünün ortam sıcaklığına çok duyarlı olmasıdır. Ortam şartlarına göre bu sensör veri okumada bazı hatalar yapabilir.

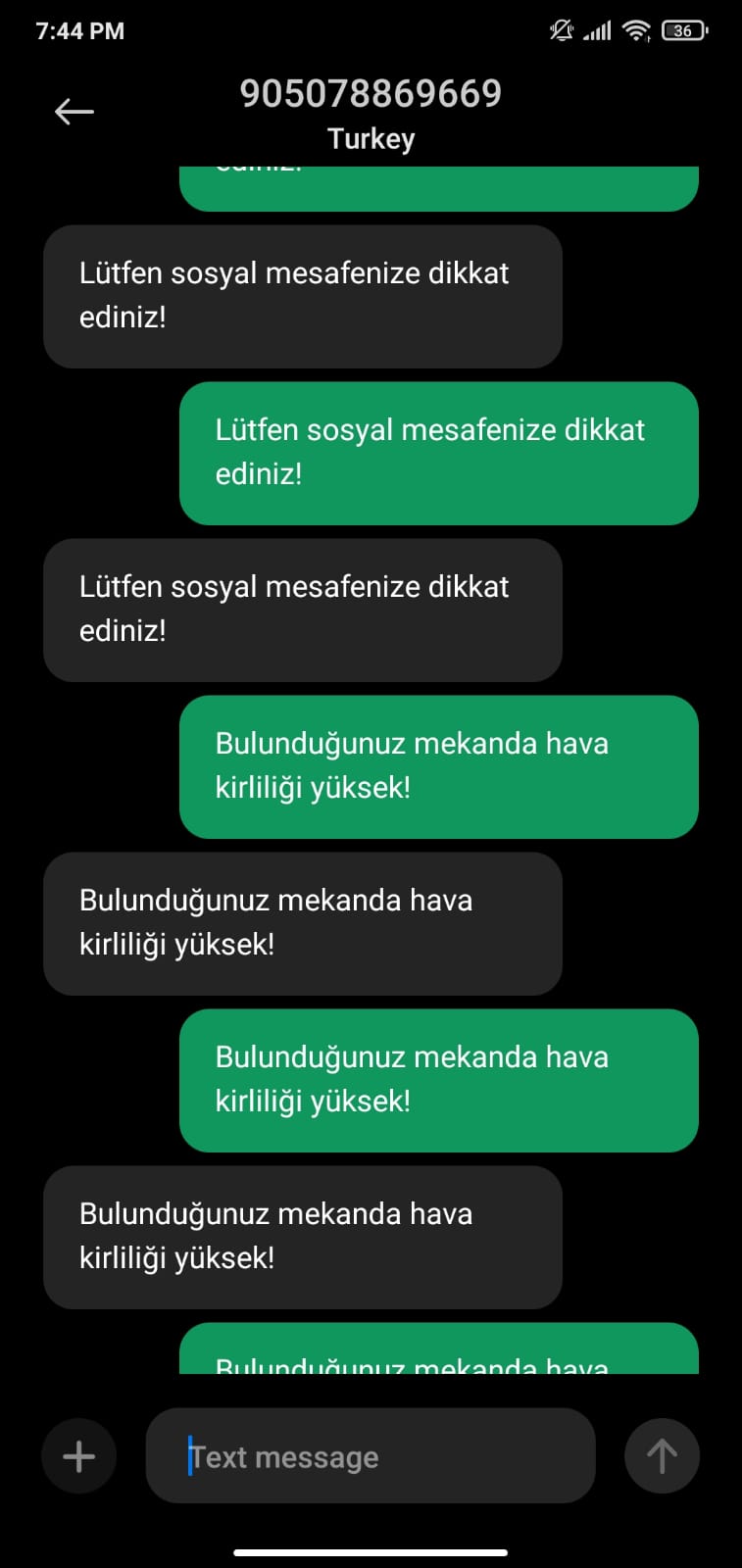
**KAYNAKÇA**

1. <https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public>
2. <https://www.who.int/westernpacific/emergencies/covid-19/information/physical-distancing>
3. <https://datasheetspdf.com/pdf/1380136/ETC/HC-SR04/1>
4. <https://www.bloomberg.com/news/articles/2021-01-11/how-to-know-if-a-building-has-good-ventilation>
5. <https://datasheetspdf.com/pdf/605077/Hanwei/MQ135/1>

**EKLER**

**FOTOĞRAFLAR**

****

****

**KOD**

#include<ESP8266WiFi.h>

#include<WiFiClientSecure.h>

int sensor = A0;

// Set the initial sensorValue to 0

int sensorValue = 0;

// defines pins numbers

const int trigPin = 2; //D4

const int echoPin = 0; //D3

//WiFi connection

const char\* ssid = "Enes";

const char\* password = "enesbilal77";

const char\* host = "maker.ifttt.com";

const int httpsPort = 443;

const char\* key = "b871167c629f5eaa3346832a7e95ad4be6b2b807";

// defines variables

long duration;

int distance;

BearSSL::WiFiClientSecure client;

void setup() {

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

Serial.begin(115200);

Serial.println();

Serial.print("connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

Serial.println("IP address: ");

Serial.println(WiFi.localIP());

client.setInsecure();

}

void loop() {

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance = duration\*0.034/2;

// Prints the distance on the Serial Monitor

Serial.print("Uzaklık: ");

Serial.println(distance);

// Read the input on analog pin 0 (named 'sensor')

// Print out the value you read

sensorValue = analogRead(sensor);

Serial.print("CO2 Miktarı: ");

Serial.println(sensorValue, DEC);

if (distance<=100){

Serial.print("connecting to ");

Serial.println(host);

Serial.print("Using key: ");

Serial.println(key);

client.setFingerprint(key);

if(!client.connect(host, httpsPort))

{

Serial.println("connection failed");

return;

}

String url = "/trigger/warning/with/key/dFytRWs5rsEqtZxA0--fCy";

Serial.print("requesting URL: ");

Serial.println(url);

client.print(String("GET ") + url + " HTTP/1.1\r\n" +

"Host: " + host + "\r\n" +

"User-Agent: BuildFailureDetectorESP8266\r\n" +

"Connection: close\r\n\r\n");

Serial.println("request sent");

}

if (sensorValue>=1000){

Serial.print("connecting to ");

Serial.println(host);

Serial.print("Using key: ");

Serial.println(key);

client.setFingerprint(key);

if(!client.connect(host, httpsPort))

{

Serial.println("connection failed");

return;

}

String url = "/trigger/warning2/with/key/dFytRWs5rsEqtZxA0--fCy";

Serial.print("requesting URL: ");

Serial.println(url);

client.print(String("GET ") + url + " HTTP/1.1\r\n" +

"Host: " + host + "\r\n" +

"User-Agent: BuildFailureDetectorESP8266\r\n" +

"Connection: close\r\n\r\n");

Serial.println("request sent");

}

delay(2000);

}