
T.C BİTLİS EREN ÜNİVERSİTESİ NESNELERİN İNTERNETİ DERSİ



MÜHENDİSLİK – MİMARLIK FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ İOT TABANLI GAZ KAÇAĞI ALGILAMA VE UYARI SİSTEMİ

Proje Adı: İoT Tabanlı Gaz Kaçağı Algılama ve Uyarı Sistemi

Dersi Veren Akademisyen: Dr. Öğrt. Üyesi İrfan ÖKTEN

Öğrenci Adı Soyadı: Zekeriya Kalkan

Öğrenci Numarası: 23080410034

1. Projenin Amacı

Günümüzde konutlarda ve endüstriyel tesislerde meydana gelen gaz sızıntıları, zehirlenme ve patlama riskleri nedeniyle ciddi can ve mal kayıplarına yol açabilmektedir. Bu projenin temel amacı, kapalı ortamlardaki yanıcı gaz seviyesini sürekli izleyerek tehlikeyi erken evrede tespit eden, Nesnelerin İnterneti (IoT) tabanlı akıllı bir güvenlik sistemi geliştirmektir.

Geliştirilen sistem, ortamdaki hava kalitesini ve çevresel parametreleri (sıcaklık ve nem) gerçek zamanlı olarak takip etmeyi hedeflemektedir. Tehlike anında sadece ortamda bulunan kişileri sesli ve görsel olarak uyarmakla kalmayıp, aynı zamanda uzak lokasyondaki kullanıcıyı mobil bildirimler aracılığıyla haberdar ederek müdahale süresini en aza indirmeyi amaçlamaktadır. Ayrıca proje, standart LED göstergeleri yerine trafik ışığı modülü kullanarak durum bildirimlerini evrensel renk kodlarıyla (Yeşil-Sarı-Kırmızı) daha anlaşılır hale getirmeyi ve farklı sensör verilerinin hibrit bir mimaride (Arduino + ESP8266) nasıl işlendiğini göstermeyi hedeflemektedir.

2. Projenin Kapsamı

Bu çalışma, donanım ve yazılım bileşenlerinin entegre edildiği hibrit bir gömülü sistem mimarisini kapsamaktadır. Proje, yerel sensör ağları üzerinden veri toplama, işleme ve bu verileri internet protokolleri aracılığıyla uzak sunuculara iletilme süreçlerini içermektedir. Proje kapsamında gerçekleştirilen temel işlevler ve sistem yetenekleri aşağıda maddeler halinde özetlenmiştir:

- **Çoklu Sensör Entegrasyonu:** Ortam güvenliği için kritik olan yanıcı gaz yoğunluğunun analog olarak ölçülmesi ve eş zamanlı olarak ortamın sıcaklık ve nem değerlerinin dijital sensörler aracılığıyla takip edilmesi.
- **Kademeli Uyarı Sistemi:** Gaz yoğunluğundaki artışın, endüstriyel standartlara atıfta bulunan "Trafik Işığı Modülü" kullanılarak üç aşamalı (Güvenli, Dikkat, Tehlike) görsel geri bildirim sistemi ile kullanıcıya aktarılması.
- **Sesli Alarm Mekanizması:** Tehlike eşiğinin aşılması durumunda piezoelektrik buzzer kullanılarak sesli ikaz verilmesi.
- **IoT Tabanlı Uzaktan Bildirim:** Sistemde bir "Ağ Geçidi" (Gateway) görevi gören Wi-Fi modülü aracılığıyla, acil durum verilerinin (Sıcaklık, Nem ve Alarm durumu) IFTTT (If This Then That) web servisine iletilmesi ve kullanıcıya anlık mobil bildirim gönderilmesi.
- **Veri Görselleştirme:** Toplanan anlık verilerin ve sistem durumunun, kullanıcı dostu bir arayüz sunan I2C protokolü tabanlı LCD ekran üzerinde görüntülenmesi.

2.1. Kullanılan Donanım ve Malzemeler

Projenin tasarım ve uygulama aşamasında, kararlı çalışma yapısı ve modülerliği nedeniyle seçilen donanım bileşenleri ve bu bileşenlerin sistemdeki görevleri aşağıdaki tabloda sunulmuştur:

Donanım Bileşeni	Sistemdeki Görevi ve Kullanım Amacı
Arduino Uno R3	Sistemin ana kontrol birimidir. Sensörlerden gelen analog ve dijital verileri işler, karar mekanizmasını çalıştırır ve çıkış birimlerini (LED, Buzzer, Ekran) yönetir.
ESP8266 Wi-Fi Modülü	Sistemin internete erişimini sağlayan ağ modülüdür. Arduino'dan aldığı alarm verilerini işleyerek HTTP protokolü üzerinden bulut sunucularına iletir.
MQ5 Serisi Gaz Sensörü	Ortamdaki yanıcı gaz (LPG, duman vb.) konsantrasyonunu algılayarak mikrodentleyiciye 0-1023 aralığında analog sinyal gönderir.
DHT11 Sensörü	Ortamin sıcaklık ve nem verilerini dijital sinyal olarak ölçer. Bu veriler hem ekranda gösterilir hem de IoT bildirimi ile kullanıcıya raporlanır.
Trafik Işığı Modülü	Tek bir modül üzerinde Yeşil, Sarı ve Kırmızı LED'leri barındırır. Gaz seviyesine göre sırasıyla "Güvenli", "Uyarı" ve "Tehlike" durumlarını görselleştirmek için kullanılır.
I2C LCD Ekran (16x2)	Sistem durumunu ("Sistem Aktif", "Gaz Kaçağı" vb.) ve ölçülen sensör değerlerini yazılı olarak kullanıcıya sunar.
Buzzer	Gaz seviyesi kritik eşiği (700 birim) aştığında aktifleşerek kullanıcıyı sesli olarak uyaran alarm bileşenidir.
Jumper Kablolar & Breadboard	Devre elemanlarının lehimsiz olarak birbirine bağlanması ve prototipin kurulması amacıyla kullanılmıştır.

3. Projenin Kısıtları

Sistemin tasarım ve gerçekleştirme aşamalarında, donanım bileşenlerinin fiziksel sınırları ve kullanılan iletişim protokollerinin gereklilikleri doğrultusunda bazı teknik kısıtlar ve optimizasyonlar uygulanmıştır:

- **Sensör Kararlılık Süresi:** Sistemde kullanılan MQ serisi gaz sensörü, elektrokimyasal yapısı gereği doğru ölçüm yapabilmek için ön ısıtma süresine ihtiyaç

duymaktadır. Bu nedenle sistem ilk açılışta sensörün kararlı hale gelmesi için kısa bir kalibrasyon süreci geçirmektedir.

- **Haberleşme Hızı Sınırı:** Arduino Uno ve ESP8266 modülleri arasındaki veri aktarımı, Arduino üzerinde donanımsal seri portun hata ayıklama (debug) için ayrılması nedeniyle, SoftwareSerial kütüphanesi kullanılarak sanal bir seri port üzerinden sağlanmıştır. Veri bütünlüğünü korumak ve paket kayıplarını önlemek adına haberleşme hızı 9600 baud rate ile sınırlandırılmıştır.
- **Ağ ve API Bağımlılığı:** Uzaktan bildirim mekanizması, aktif bir internet bağlantısına ve IFTTT sunucularının erişilebilirliğine doğrudan bağımlıdır.
- **Bildirim Sıklığı Optimizasyonu (Rate Limiting):** Tehlike anında sensörün sürekli yüksek değer okuması durumunda, sisteme ve web servisine aşırı yük binmesini engellemek amacıyla bildirimler arasına yazılımsal bir kısıtlama getirilmiştir. Sistem, bir alarm bildirimi gönderdikten sonra yeni bir bildirim göndermek için en az 15 saniye (15000 ms) beklemektedir.

4. Sistem Tasarımı ve Mimari

4.1. Genel Sistem Mimarisi

Geliştirilen proje, "Algılayıcı Düğüm" (Sensor Node) ve "Ağ Geçidi" (Gateway) olmak üzere iki temel katmanlı bir mimariye sahiptir.

1. **Algılayıcı Düğüm (Arduino Uno):** Sistemenin beyni konumundadır. Gaz ve DHT11 sensörlerinden verileri toplar, bu verileri işler ve yerel uyarı birimlerini (Trafik Işığı Modülü, Buzzer, LCD Ekran) yönetir.
2. **Ağ Geçidi (ESP8266):** Arduino'dan gelen işlenmiş alarm verilerini alır ve Wi-Fi protokolü üzerinden internete çıkarak bulut tabanlı servise (IFTTT) iletir.

4.2. Yazılım Akışı ve Karar Mekanizması

Sistem yazılımı, sensörlerden okunan verilerin belirli eşik değerlerle karşılaştırılmasına dayanan bir **Durum Makinesi (State Machine)** mantığıyla çalışmaktadır. Gaz sensöründen okunan analog değer (0-1023), 700 birimlik kritik eşik değer baz alınarak değerlendirilir.

Sistem üç ana durumda çalışmaktadır:

- **Durum 1: Güvenli (Normal) Mod:**
 - **Koşul:** Gaz değerinin güvenli sınırın (Eşik - 50) altında olması.
 - **Eylem:** Trafik ışığı modülündeki **Yeşil LED** aktif edilir, diğer uyarıcılar kapalıdır. LCD ekranda "TEMİZ" bilgisi gösterilir.
- **Durum 2: Uyarı (Warning) Modu:**
 - **Koşul:** Gaz değerinin kritik eşığa yaklaşması (Eşik - 50 ile Eşik arası).
 - **Eylem:** Trafik ışığı modülündeki **Sarı LED** aktif edilerek görsel dikkat çekilir. LCD ekranda "DİKKAT! Koku Var" uyarısı verilir.
- **Durum 3: Alarm (Danger) Modu:**
 - **Koşul:** Gaz değerinin kritik eşığı (700) aşması.
 - **Eylem:** Trafik ışığı modülündeki **Kırmızı LED** yakılır ve **Buzzer** sesli ikaz vermeye başlar.

- **IoT Tetiklemesi:** Eğer son bildirimden bu yana belirlenen süre (15 saniye) geçmişse, Arduino seri port üzerinden ALARM,Sıcaklık,Nem formatında bir veri paketi oluşturur ve ESP8266'ya gönderir.
-

5. Uygulama ve Gerçekleme

Bu bölümde, projenin donanım ve yazılım bileşenlerinin nasıl bir araya getirildiği, modüler bir yaklaşım ile ele alınmıştır. Her bir alt sistemin geliştirilme amacı, uygulama detayları ve süreç içerisinde karşılaşılan teknik zorluklara getirilen çözümler aşağıda detaylandırılmıştır.

5.1. Sensör Verilerinin Okunması ve İşlenmesi

Amaç: Ortamın güvenlik durumunu analiz edebilmek için gerekli olan gaz yoğunluğu, sıcaklık ve nem verilerinin anlık ve doğru bir şekilde toplanmasıdır.

Uygulama Detayı:

Sistemde gaz algılama işlemi için analog giriş pini (A0) kullanılarak MQ serisi sensörden 0-1023 aralığında bir voltaj değeri okunmaktadır. Eş zamanlı olarak DHT11 dijital sensörü üzerinden sıcaklık ve nem verileri çekilmektedir. Okunan bu ham veriler, seri port ekranı üzerinden saniye bazında izlenerek kalibrasyon kontrolleri yapılmıştır.

Karşılaşılan Sorun:

Gaz sensörünün ilk enerji verildiği anda kararsız değerler üretmesi ve anlık dalgalanmaların yanlış alarmlara sebep olabilme ihtimali gözlemlenmiştir.

Çözüm:

Sistem açılışına sensörün ısınması için bir bekleme süresi eklenmiş ve yazılım tarafında okunan analog verilerin ani sıçramalarını tolere edecek bir eşik değeri (700 birim) deneysel olarak belirlenmiştir.

5.2. Trafik Işığı Modülü ve Görsel Geri Bildirim

Amaç: Kullanıcının, sisteme baktığı anda sensör verilerini yorumlamaya çalışmasına gerek kalmadan, evrensel renk kodları (Yeşil-Sarı-Kırmızı) ile ortamın risk durumunu bir bakışta anlamasını sağlamaktır.

Uygulama Detayı:

Standart ayrıık LED'ler yerine, endüstriyel sistemlere atıfta bulunan entegre bir Trafik Işığı Modülü kullanılmıştır. Yazılımda oluşturulan karar yapısı ile:

- Gaz değeri güvenli aralıktaysa (Eşik - 50'den küçük) sadece **Yeşil**,

- Uyarı aralığındaysa (Eşik - 50 ile Eşik arası) sadece **Sarı**,
- Tehlike sınırındaysa (Eşik değerinden büyük) sadece **Kırmızı** ışığın yanması sağlanmıştır.

Karşılaşılan Sorun:

Durum geçişlerinde (örneğin Sarıdan Kırmızıya geçerken) önceki durumun ışığının açık kalması ve renklerin karışması sorunu yaşanmıştır.

Çözüm:

Her durum bloğu (if-else yapısı) içerisinde, aktif edilecek renk HIGH yapılırken, diğer iki rengin LOW konumuna çekilmesi zorunlu kılınarak renk karışıklığı donanımsal ve yazılımsal olarak engellenmiştir.

5.3. Arduino - ESP8266 Seri Haberleşme Protokolü

Amaç: Sensör okuma işini yapan Arduino ile internet erişimi sağlayan ESP8266 modülü arasında güvenilir bir veri köprüsü kurmaktır.

Uygulama Detayı:

Arduino üzerinde donanımsal seri portun meşgul olması nedeniyle SoftwareSerial kütüphanesi kullanılarak 2 ve 3 numaralı pinler sanal RX-TX hattı olarak tanımlanmıştır. Tehlike anında Arduino, ALARM,Sıcaklık,Nem formatında (Örnek: "ALARM,24,50") özel bir veri paketi oluşturarak bu hat üzerinden ESP8266'ya gönderir.

Karşılaşılan Sorun:

Tehlike anında sensörün sürekli yüksek değer üretmesi nedeniyle, Arduino'nun ESP8266'ya saniyede yüzlerce kez veri paketi göndermeye çalışması ve bunun sonucunda Wi-Fi modülünün kilitlenmesi.

Çözüm:

"Debounce" mantığı uygulanarak, millis() fonksiyonu ile bir zaman sayacı oluşturulmuştur. İki bildirim gönderimi arasına 15 saniyelik (15000 ms) zorunlu bir bekleme süresi eklenerek veri trafiği kontrol altına alınmıştır.

5.4. IoT Entegrasyonu ve IFTTT Bildirim Mekanizması

Amaç: Yerel ağdan bağımsız olarak, dünyanın herhangi bir yerindeki kullanıcıya gaz kaçağı durumunu ve ortam sıcaklık/nem bilgilerini iletmektir.

Uygulama Detayı:

ESP8266 modülü, seri portu sürekli dinleyerek "ALARM" ön ekiyle başlayan paketleri yakalar. Gelen paketi ayrıştırarak sıcaklık ve nem verilerini çeker. Ardından IFTTT (If This Then That) servisine ait Webhook URL'ine, API anahtarı ve sensör verilerini içeren bir HTTP GET isteği gönderir.

Test:

Yapılan denemelerde, alarm tetiklendikten sonra ortalama 3-5 saniye içerisinde mobil cihaza bildirimin ulaştığı doğrulanmıştır.

6. Test ve Sonuçlar

Geliştirilen "IoT Tabanlı Gaz Kaçağı Algılama ve Uyarı Sistemi", laboratuvar ortamında farklı senaryolar altında test edilmiştir. Test süreci, sistemin kararlılığını, sensörlerin tepki süresini ve ağ üzerinden bildirim gönderme başarısını doğrulamaya yönelik olarak kurgulanmıştır.

Aşağıdaki tabloda, sistem üzerinde gerçekleştirilen temel test senaryoları, bu senaryolara ait giriş koşulları ve sistemin ürettiği çıktılar detaylandırılmıştır:

Test No	Test Senaryosu	Girdi / Koşul	Beklenen Sonuç	Gerçekleşen Sonuç	Durum
T1	Güvenli Ortam Analizi	Gaz sensörü verisi < 650	Trafik Işığı: YEŞİL LCD: "TEMİZ" Buzzer: Pasif	Trafik Işığı: YEŞİL LCD: "TEMİZ" Buzzer: Pasif	BAŞARILI
T2	Erken Uyarı Testi	Gaz sensörü verisi 650 - 700 arası	Trafik Işığı: SARI LCD: "DİKKAT! Koku Var"	Trafik Işığı: SARI LCD: "DİKKAT! Koku Var"	BAŞARILI

Test No	Test Senaryosu	Girdi / Koşul	Beklenen Sonuç	Gerçekleşen Sonuç	Durum
			Buzzer: Pasif	Buzzer: Pasif	
T3	Alarm Durumu Testi	Gaz sensörü verisi > 700	Trafik Işığı: KIRMIZI LCD: "GAZ KACAGI !!!" Buzzer: AKTİF (1 kHz)	Trafik Işığı: KIRMIZI LCD: "GAZ KACAGI !!!" Buzzer: AKTİF	BAŞARILI
T4	IoT Bildirim İletimi	Alarm durumu tetiklendiğinde	ESP8266'nın veriyi işlemesi ve IFTTT üzerinden telefona bildirim düşmesi	Mobil cihaza, ortam sıcaklık ve nem verisini içeren bildirim ulaştı.	BAŞARILI
T5	Bildirim Sıklığı (Debounce)	Sürekli gaz kaçağı durumu (Süreklilik Testi)	Bildirimler arası en az 15 saniye bekleme süresi olması	Ardışık bildirimlerin 15 saniye arayla gönderildiği seri porttan doğrulandı.	BAŞARILI

6.1. Test Sonuçlarının Değerlendirilmesi

Yapılan testler sonucunda, MQ gaz sensörünün ortamdaki değişikliklere hızlı tepki verdiği ve Trafik Işığı Modülü'nün durum geçişlerini (Yeşil -> Sarı -> Kırmızı) hatasız bir şekilde yansıttığı gözlemlenmiştir. Özellikle IoT entegrasyonu testlerinde, Arduino ve ESP8266 arasındaki seri haberleşmenin kararlı çalıştığı ve IFTTT servisi aracılığıyla gönderilen bildirimlerin ortalama 3-5 saniye gecikme ile kullanıcıya ulaştığı tespit edilmiştir. Sistem, hedeflenen tüm güvenlik ve uyarı fonksiyonlarını başarıyla yerine getirmektedir.

EKLER

Bu bölümde, projenin yazılım mimarisi modüler bir yaklaşımla ele alınmış, donanım bileşenlerini kontrol eden kod blokları ayrı ayrı açıklanmıştır. Raporun sonunda ise sisteme yüklenen kaynak kodların tamamı sunulmuştur.

Ek A - Donanım Kontrol Kodları ve Çalışma Mantığı

Ek A.1 - Sensör Verilerinin Okunması ve Takibi

Sistemin ortamı analiz etmesi için MQ gaz sensöründen analog veri, DHT11 sensöründen ise dijital sıcaklık ve nem verisi okunmaktadır. Bu veriler değişkenlere atanarak karar mekanizmasında kullanılmak üzere hazırlanır.

C++

```
// Sensörlerden verilerin okunması
int gazDegeri = analogRead(gazSensorPin);    [cite_start]// 0-1023
arası analog değer okunur [cite: 1039]
int sicaklik = (int)dht.readTemperature();    [cite_start]// Sıcaklık
verisi (C) [cite: 1039]
int nem = (int)dht.readHumidity();            [cite_start]// Nem
verisi (%) [cite: 1040]

// Verilerin seri port üzerinden bilgisayarda gözlemlenmesi
Serial.print("Gaz:"); Serial.print(gazDegeri);
Serial.print(" Sıcaklık:"); Serial.print(sicaklik);
Serial.print(" Nem:"); [cite_start]Serial.println(nem); [cite: 1040,
1041]
```

Ek A.2 - Trafik Işığı Modülü ve Alarm Karar Yapısı

Okunan gaz verisi (gazDegeri), belirlenen kritik eşik değer (700) ile karşılaştırılır. Bu kıyaslama sonucunda Trafik Işığı Modülü üzerindeki LED'ler (Yeşil, Sarı, Kırmızı) ve Buzzer kontrol edilir.

C++

```
// Eşik Değer Kontrolü (Decision Making)
if (gazDegeri < esikDeger - 50) {
    [cite_start]// GÜVENLİ DURUM: Yeşil LED yanar, diğerleri söner
[cite: 1042]
    digitalWrite(yesilLed, HIGH);
    digitalWrite(sariLed, LOW);
    digitalWrite(kirmiziLed, LOW);
    noTone(buzzerPin);
    [cite_start]lcd.print(" (TEMİZ)"); [cite: 1043]
}
else if (gazDegeri >= esikDeger - 50 && gazDegeri < esikDeger) {
    [cite_start]// UYARI DURUMU: Sarı LED yanar [cite: 1044]
    digitalWrite(yesilLed, LOW);
    digitalWrite(sariLed, HIGH);
    digitalWrite(kirmiziLed, LOW);
    [cite_start]lcd.print("DIKKAT! Koku Var"); [cite: 1045]
}
else {
```

```

    [cite_start]// TEHLİKE DURUMU: Kırmızı LED yanar ve Buzzer öter
[cite: 1046]
    digitalWrite(yesilLed, LOW);
    digitalWrite(sariLed, LOW);
    digitalWrite(kirmiziLed, HIGH);
    tone(buzzerPin, 1000); [cite_start]// Sesli alarm aktif [cite:
1047]
    lcd.print("GAZ KACAGI !!!");
}

```

Ek A.3 - IoT Veri Gönderim Mantiğı (Zamanlayıcı Kontrolü)

Tehlike anında sürekli veri gönderip sistemi kilitlememek için millis() fonksiyonu ile bir zaman sayacı oluşturulmuştur. Her bildirim arasında 15 saniye (15000 ms) bekleme süresi uygulanır.

C++

```

// IoT Bildirim Kontrolü
unsigned long suankiZaman = millis();

[cite_start]// Eğer son bildirimden bu yana 15 saniye geçtiyse
[cite: 1048]
if (suankiZaman - sonBildirimZamani >= bildirimAraligi) {
    Serial.println("Veriler ESP'ye gönderiliyor...");

    // Veri Paketi Oluşturma: "ALARM,Sıcaklık,Nem"
    espSerial.print("ALARM,");
    espSerial.print(sicaklik);
    espSerial.print(",");
    [cite_start]espSerial.println(nem); [cite: 1049]

    sonBildirimZamani = suankiZaman; [cite_start]// Sayacı sıfırla
[cite: 1049]
}

```

Ek B - Tam Kaynak Kodları

Ek B.1 - Arduino Uno Yazılımı (Tam Hali)

Aşağıda, Arduino Uno mikrodenetleyicisine yüklenen sketch_jan11a.ino dosyasının tam içeriği yer almaktadır.

C++

```

#include <SoftwareSerial.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>

```

```

// --- PINLER ---
[cite_start]const int gazSensorPin = A0; [cite: 1034]
[cite_start]#define DHTPIN 4 [cite: 1035]
#define DHTTYPE DHT11

const int kirmiziLed = 8;
const int sariled = 9;
const int yesilled = 10;
[cite_start]const int buzzerPin = 11; [cite: 1036]

// İletişim (RX, TX)
[cite_start]SoftwareSerial espSerial(2, 3); [cite: 1036]

DHT dht(DHTPIN, DHTTYPE);
LiquidCrystal_I2C lcd(0x27, 16, 2);
int esikDeger = 700;
unsigned long sonBildirimZamani = 0;
[cite_start]const long bildirimAraligi = 15000; [cite: 1037]

void setup() {
    Serial.begin(9600);
    espSerial.begin(9600);
    dht.begin();
    lcd.init();
    lcd.backlight();

    pinMode(kirmiziLed, OUTPUT);
    pinMode(sariled, OUTPUT);
    pinMode(yesilled, OUTPUT);
    pinMode(buzzerPin, OUTPUT);

    lcd.setCursor(0, 0); lcd.print("Sistem Aktif");
    [cite_start]delay(2000); lcd.clear(); [cite: 1038]
}

void loop() {
    int gazDegeri = analogRead(gazSensorPin);
    int sicaklik = (int)dht.readTemperature();
    [cite_start]int nem = (int)dht.readHumidity(); [cite: 1039, 1040]

    // Bilgisayardan Takip
    Serial.print("Gaz:"); Serial.print(gazDegeri);
    Serial.print(" Sicaklik:"); Serial.print(sicaklik);
    Serial.print(" Nem:"); [cite_start]Serial.println(nem); [cite:
1040, 1041]

    // LCD Ekran
    lcd.setCursor(0, 0);

```

```

    [cite_start]lcd.print("Isi:"); lcd.print(sicaklik); lcd.print("C
    Nem:%"); lcd.print(nem); [cite: 1041]

    // --- DURUM KONTROLÜ ---
    if (gazDegeri < esikDeger - 50) {
        digitalWrite(yesilled, HIGH);
        digitalWrite(sariled, LOW); digitalWrite(kirmiziled, LOW);
        noTone(buzzerPin);
        [cite_start]lcd.setCursor(0, 1); lcd.print("Gaz: ");
        lcd.print(gazDegeri); lcd.print(" (TEMİZ)"); [cite: 1042, 1043]
    }
    else if (gazDegeri >= esikDeger - 50 && gazDegeri < esikDeger) {
        digitalWrite(yesilled, LOW);
        digitalWrite(sariled, HIGH); digitalWrite(kirmiziled, LOW);
        noTone(buzzerPin);
        [cite_start]lcd.setCursor(0, 1); lcd.print("DIKKAT! Koku Var");
        [cite: 1044, 1045]
    }
    else {
        // TEHLİKE DURUMU
        digitalWrite(yesilled, LOW); digitalWrite(sariled, LOW);
        digitalWrite(kirmiziled, HIGH);
        tone(buzzerPin, 1000);
        lcd.setCursor(0, 1); lcd.print("GAZ KACAGI !!! ");

        // BİLDİRİM GÖNDERME
        unsigned long suankiZaman = millis();
        if (suankiZaman - sonBildirimZamani >= bildirimAraligi) {
            Serial.println("Veriler ESP'ye gönderiliyor...");
            // Veriyi paketle: "ALARM,24,50" şeklinde gönder
            espSerial.print("ALARM,");
            espSerial.print(sicaklik);
            espSerial.print(",");
            espSerial.println(nem);

            [cite_start]sonBildirimZamani = suankiZaman; [cite: 1046,
            1047, 1048, 1049]
        }
    }
    delay(200);
}

```

Ek B.2 - ESP8266 Yazılımı (Tam Hali)

Aşağıda, Wi-Fi iletişimi sağlayan ESP8266 modülüne yüklenen sketch_jan11b.ino dosyasının tam içeriği yer almaktadır.

C++

```
#include <ESP8266WiFi.h>
```

```

#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>

const char* ssid = "Zekeriya Wifi";
const char* password = "12345678";

String IFTTT_API_KEY = "oT5aFX5hi-
Rp8Ld0Pjyx22MGdsB8b1BcpA67ALVAwTm";
[cite_start]String EVENT_NAME = "gaz_alarm"; [cite: 1024, 1025]

void setup() {
    Serial.begin(9600);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
    }
}

void loop() {
    if (Serial.available() > 0) {
        // Satırı oku: "ALARM,24,50"
        String gelenVeri = Serial.readStringUntil('\n');
        gelenVeri.trim();

        // Veriyi Virgülden Parçala
        int ilkVirgul = gelenVeri.indexOf(',');
        int ikinciVirgul = gelenVeri.indexOf(',', ilkVirgul + 1);

        if (gelenVeri.startsWith("ALARM")) {
            String sicaklik = "0";
            String nem = "0";

            // Eğer sıcaklık ve nem verisi de geldiyse onları ayıkla
            if (ilkVirgul > 0 && ikinciVirgul > 0) {
                sicaklik = gelenVeri.substring(ilkVirgul + 1,
ikinciVirgul);
                nem = gelenVeri.substring(ikinciVirgul + 1);
            }

            if(WiFi.status() == WL_CONNECTED){
                [cite_start]BildirimGonder(sicaklik, nem); [cite: 1026,
1027, 1028, 1029, 1030]
            }
        }
    }
}

void BildirimGonder(String sicaklik, String nem) {
    WiFiClient client;

```

```
HTTPClient http;

// URL'yi oluřtur: value1=Sıcaklık, value2=Nem
String url = "http://maker.ifttt.com/trigger/" + EVENT_NAME +
"/with/key/" + IFTTT_API_KEY;
url += "?value1=" + sıcaklık + "&value2=" + nem;

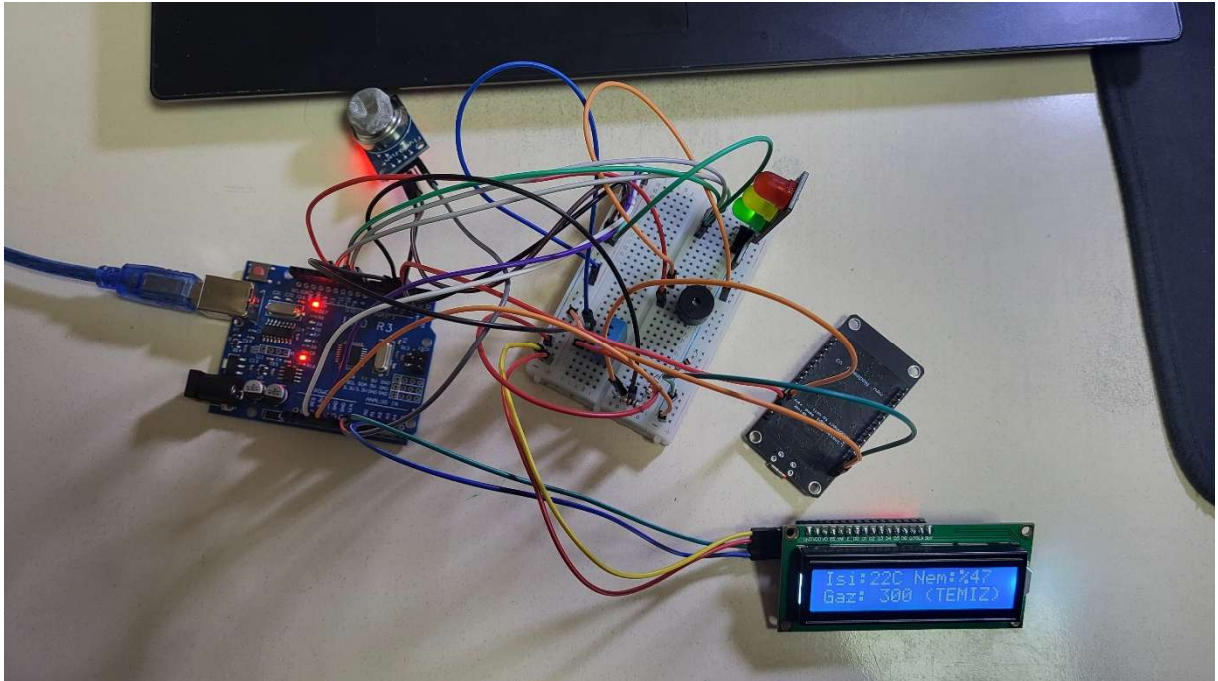
http.begin(client, url);
int httpCode = http.GET();
http.end();

[cite_start]delay(5000); [cite: 1031, 1032, 1033]
}
```

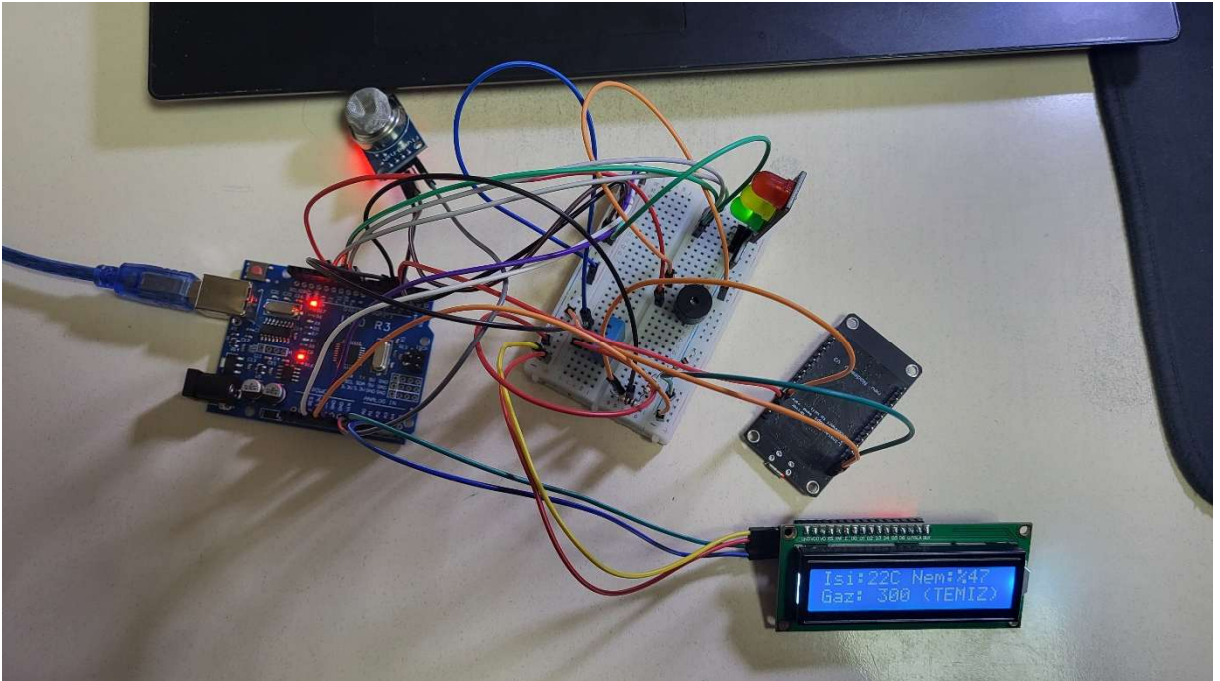
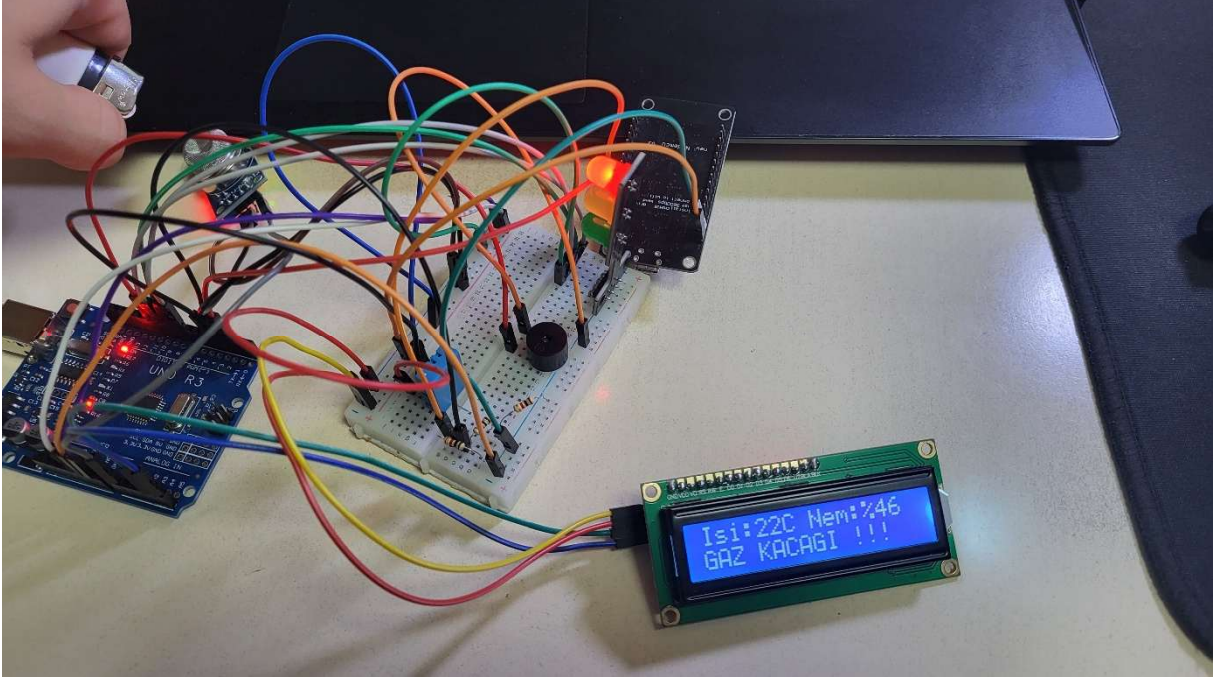
Ek C - Proje Uygulama Grselleri

Bu blmde, gerekleřtirilen projenin devre kurulumuna ve alıřma anlarına ait grseller yer almaktadır.

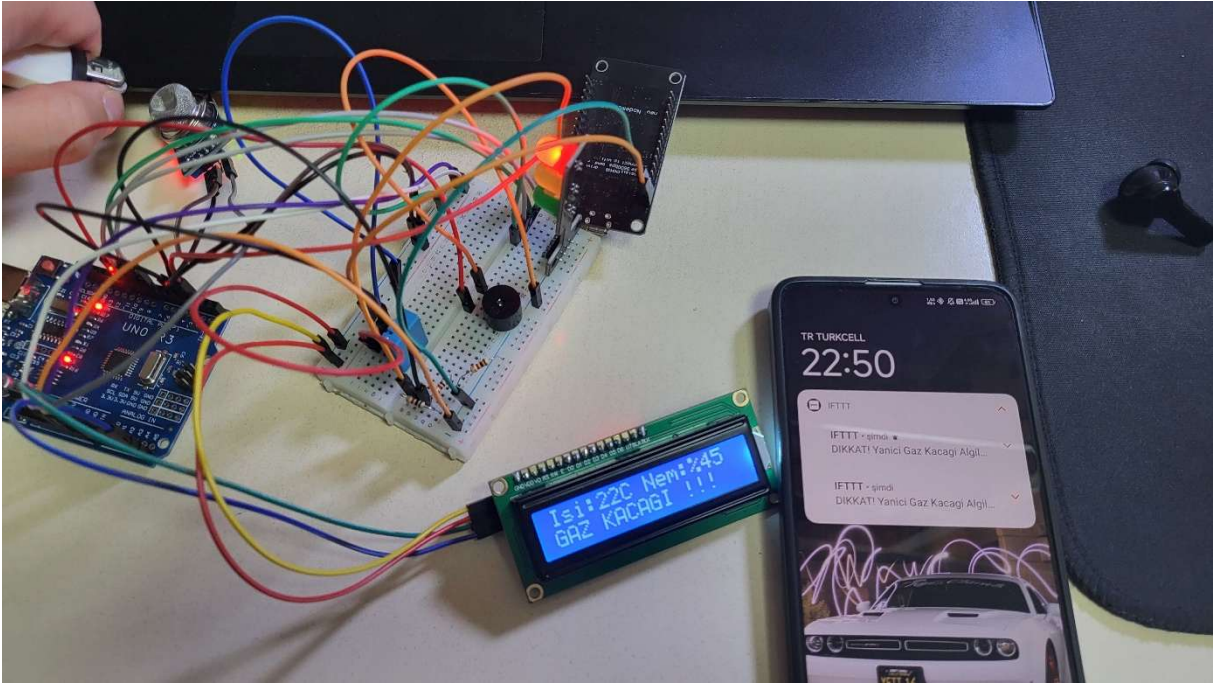
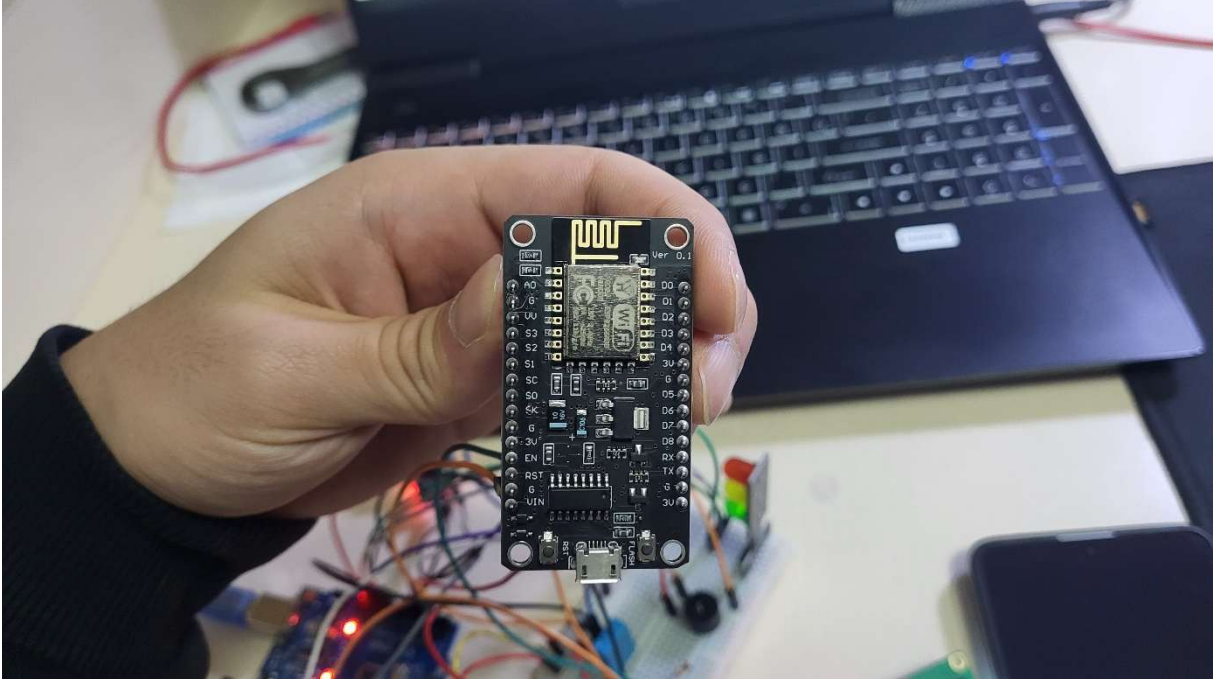
- **Grsel 1:** Sistemin genel devre kurulumu ve bileřen yerleřimi.



- **Görsel 2:** Trafik Işıđı Modölünün "Güvenli" (Yeşil) ve "Alarm" (Kırmızı) durumlarındaki çalışması.



- **Görsel 3:** ESP8266 Wi-Fi modülü ve Arduino Uno arasındaki bağlantı detayları.



Ek D - Video Anlatımı

Geliştirilen sistemin sensör tepkilerini, sesli/görsel uyarı mekanizmasını ve mobil bildirim gönderimini içeren detaylı çalışma videosuna aşağıdaki bağlantıdan ulaşabilirsiniz:

YouTube Video Linki:

<https://youtu.be/CgtrxbJrRi8?si=TTF5-SRY33pQBofc>
