



SAKARYA
ÜNİVERSİTESİ

NESNELERİN İNTERNETİ VE UYGULAMALARI

AD: Esra

SOYAD: KIZILELMA

ŞUBE: 1-A

NUMARA: B191210040

AD: Gülcan Rabia

SOYAD: Aksu

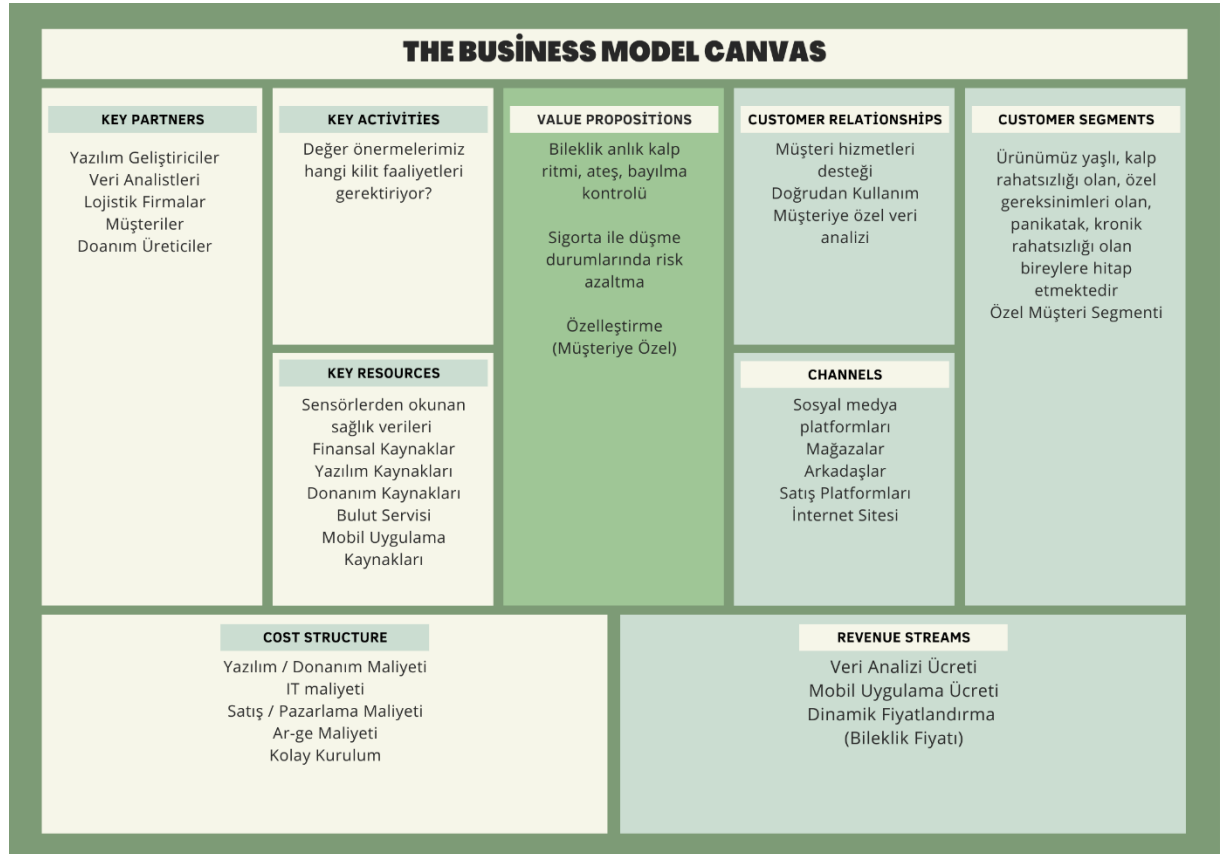
ŞUBE: 1-B

NUMARA: B201210103

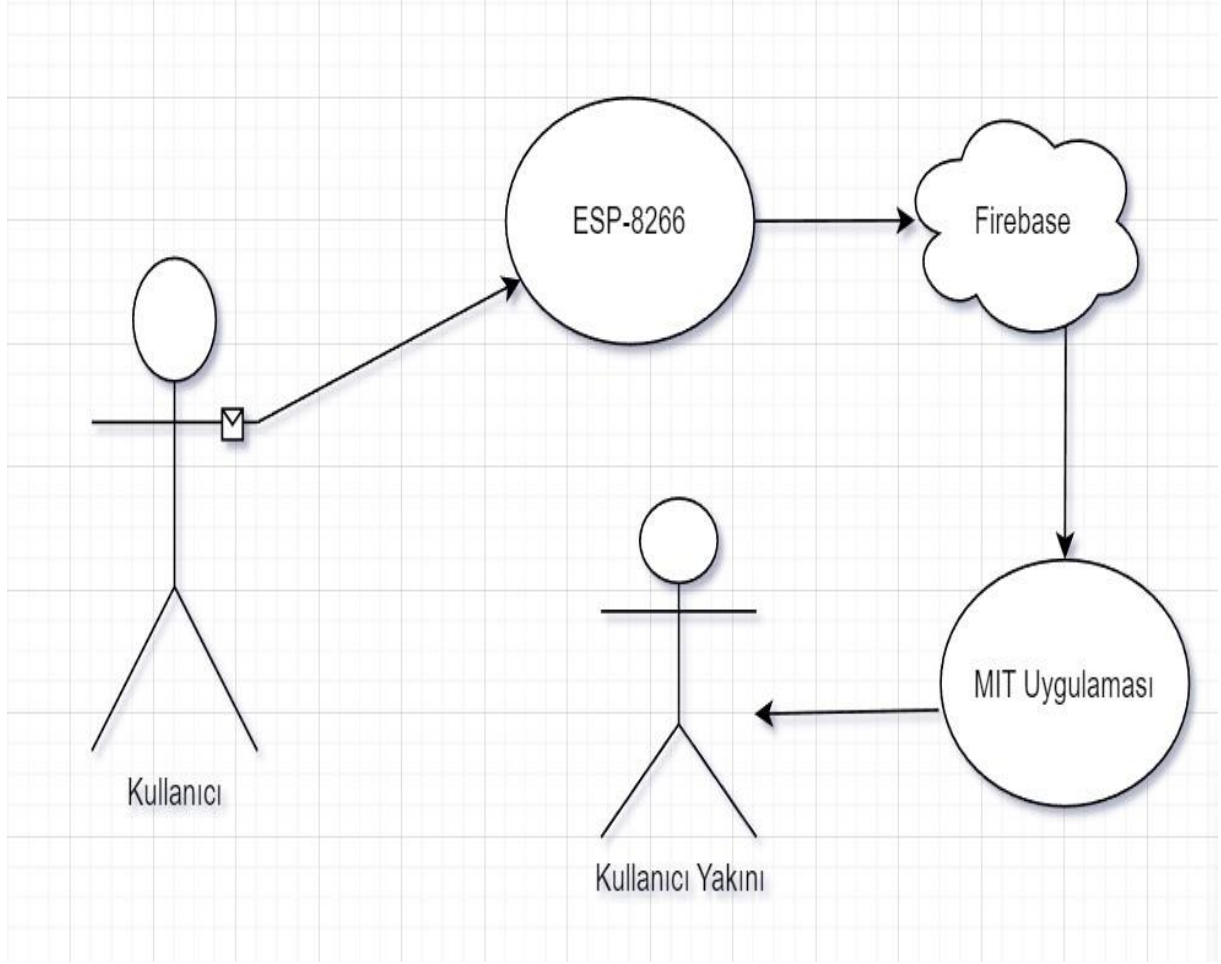
PROJE SENARYOSU

Geliştirdiğimiz projemizin adı “Çok Amaçlı Bileklik’tir.” Bu projede amacımız insanlığa yararlı bir ürün geliştirmektir. Projemizdeki hedef kitlemiz yaşlı, hasta ya da özel gereksinimli bireylerdir. Çünkü bu bireylerin günlük hayatta karşılaşılabileceği sağlık problemlerine çözüm üretmesi oldukça zordur. Biz de bu problemin önüne geçmek için yani daha kolay sonuca ulaştırmak için bu projeyi gerçekleştirdik.

Yaşlı, hasta ya da özel gereksinimli bireyler günlük yaşamlarında düşmeler, ani bayılmalar, ateşlenmeler ve kalp çarpıntısı gibi ani reaksiyonlar yaşamaktadır. Bu gibi problemlerde yalnız olmaları veya yakınlarına haber veremeyeceği durumlarda bilekliğimiz devreye girmektedir. Takılı olan bileklikteki özel sensörler sayesinde kişinin sağlık durumundaki değişiklikler kontrol edilmektedir. Kişiye özel verilen eşik değerlerin kontrolü sonucunda olması gereken sonuçtan farklı bir durum söz konusu olduğunda o etikete bağlı olan aile bireylerine kişinin durumu hakkında bilgi yollamaktayız. Bu durumlar hastanın düşmesi, kalp krizi veya kalp çarpıntısı yaşaması, ateşlenmesi ya da panikatak yaşaması gibi olaylardır.



Şekil 1: İş Modeli



Şekil 2: UML Diyagramı

GELİŞTİRİLEBİLİRLİK

Projemiz farklı tür sensör kullanıldığından büyük veri kaynağı kategorisine girmektedir. Bu noktada elde edilen verilerin analiz aşamasında günün tüm zamanlarındaki verilerin depolanmasına ihtiyacımız yoktur. Kontrol dışındaki durumlar oluşmadığı zamanlarda bu bilgilerin saklanmasına gerek yoktur. Sadece kritik anların bilgisinin veri tabanında tutulması gerekir. Bu sebeple veri depolama kısmında veri madenciliği ve yapayzeka gibi alanlardan destek almamız gerekir. Geliştirilebilirlik açısından çok amaçlı bilekliğimizdeki verilerin hastanın izniyle gerekli sağlık kuruluşlarıyla paylaşılması dahilinde hastada oluşabilecek sağlık sorunları hakkında fikir sahibi olup ön tahmin yapılabilir.

Başka bir üründen örnek vermek gerekirse ritim holter vücudumuza takıldığında tüm verileri cihaza kaydeder ve doktor daha sonra bu verileri kontrol eder. Ama bizim amacımız daha sonra değil de anlık veri akışı sağlayarak daha erken ve hızlı sonuca ulaşmaktır.

Ek olarak projemizdeki bileklik her bireyin şahsına ait olacak şekilde ayarlanmaktadır. Örneğin çocukların normal kalp ritmi ortalama 80 ile 120 arasında değişmekte iken gençlerin nabız oranı 60 ile 100 arasında değişiklik göstermektedir. Başka bir örnek vermek gerekirse her bireyin vücut sıcaklığı yaşadığı coğrafyaya ya da bağışıklık sistemine göre değişkenlik göstermektedir. Kutuplarda yaşayan insanların ortalama vücut sıcaklığı 35.9 iken çölde yaşayan insanların 36.7'dir. Bu sebeple kişiye özel bileklik tasarımı bizim için önemli bir kriter olmuştur.

TEKNOLOJİLER

Projemizde Arduino, Firebase ve MIT App Inventor uygulamalarını kullanacağız. Malzeme olarak NodeMCU, board, kablolar kullanacağız, sensör olarak; MAX30102 (nabız sensörü), GY501(ivme sensörü), CJMCU-75 (sıcaklık sensörü) kullanacağız.

MAX30102 (NABIZ SENSÖRÜ) ÖZELLİKLERİ:

- I2C haberleşme protokolüne sahiptir.
- İçerdiği kırmızı ve kızılötesi LED'lerden ışık yayarak ve bu ışığın yansımından ölçüm alarak kan emilimini ölçer.
- 3.3 besleme voltajına sahiptir.
-

GY501 (İVME SENSÖRÜ) ÖZELLİKLERİ:

- MPU 6050 Sensörü, dijital hareket işlemcisine sahip 3 eksen eksen jireskop ve 3 eksen ivme ölçen 6 DOF veya 6 eksen IMU bir modüldür. 6 eksen IMU sensörü olan MPU 6050 modülü üzerinde bulunan sensörün sıcaklık değeri çıkışı da vermektedir. Bu sayede 6 değer (ivme ve gyro), 1 değer sıcaklık olmak üzere toplam 7 değer çıkışı vermektedir.
- GY-521 MPU 6050 6DOF Gyro ve İvme Ölçer Sensör Modülü, I2C hattı üzerinden haberleşmektedir.
- MPU 6050 Modülü üzerinde I2C çıkış pinlerinin yanı sıra modülü harici sensörlere bağlamak için ekstra I2C pinleri de bulunmaktadır.
- MPU 6050 3 eksen jireskop ve ivme ölçer üzerinde çip sıcaklığını ölçmek için dahili bir sıcaklık sensörü bulunmaktadır. Bu sayede ölçüm kalibrasyonu yaparken ya da modülü kullanırken çipin dayandığı sıcaklık değeri gözlemlenebilmektedir. Sıcaklık ölçüm sensörü -40 °C ile 80 °C arasında ölçüm yapabilmektedir. Sıcaklık ölçüm doğruluğu $\pm 1^{\circ}\text{C}$ dir.
- MPU 6050 3 eksen jireskop ve ivme ölçer üzerinde dahili 5V-3.3V voltaj regülatörü bulunmaktadır. Bu sayede 5V veren Arduino gibi mikrodenetleyici kartlar ile kullanırken harici bir seviye dönüştürücüye gerek kalmadan direkt olarak mikrodenetleyici kartına bağlanabilmektedir.

- MPU6050 Modülü, kendi kendini dengeleyen robotlar, cep telefonları, navigasyon projeleri, oyun sistemleri gibi hareket uygulamalarının tespit edildiği bir çok projede kullanılabilir.

CJMCU-75 (SICAKLIK SENSÖRÜ) ÖZELLİKLERİ:

- I2C bus arabirimi, cihaz adresi 7 slave adresi 1001xxx, aynı veriyolu üzerinde sekiz cihazın dışında genişletilebilir
- 11 ADC, 0.125 ° sıcaklık çözünürlüğü sağlar
- Programlanabilir sıcaklık eşiği ve histerezis ayar noktaları
- Güç tüketimini azaltmak için kapanma modundaki akım tüketimi yalnızca 1.0µA
- Açılışta cihaz bağımsız bir sıcaklık denetleyicileri olarak kullanılabilir.
- -55 ° ~ + 125 ° sıcaklık aralığında sıcaklık doğrudan dijital sinyallere dönüştürülür ve 0.125 ° hassasiyet elde edilebilmektedir.
- MCU doğrudan dahili kayıtlarındaki I2C veri yolu verileri üzerinden okunabilir ve I2C vasıtasıyla dört veri kaydedicisi farklı işletim modlarını ayarlamak için çalışır.
- LM75A'da üç seçilebilir lojik adres pimi vardır, aynı yolda adres çakışması olmaksızın sekiz cihazın bağlanmasını mümkün kılar.

PROJE TASARIM ÖZETİ

Proje için elimizde mikrodenetleyici olarak NodeMCU vardı. Biz de NodeMCU'yu 3 sensörle nasıl kullanabileceğimizi araştırmak için sensörlerin özelliklerine baktık. 3 sensörümüz de NodeMCU da ayrı ayrı çalışıyordu fakat giriş pini olarak aynı pinleri (D1 ve D2) kullanıyorlardı. Giriş pinlerini ayırmak için her sensörün kendine ait olan adreslerini bulmamız gerekiyordu. Bu adresleri aşağıda bulunan kodla sensörleri tek tek deneyerek konsola yazdırdık.

adkontrol | Arduino 1.8.18

Dosya Düzenle Taslak Araçlar Yardım



```
#include <Wire.h>

void setup()
{
  Wire.begin();

  Serial.begin(9600);
  while (!Serial);           // Leonardo: seri monitör için bekleyin
  Serial.println("\nI2C Scanner");
}

void loop()
{
  byte error, address;
  int nDevices;

  Serial.println("Tarama...");

  nDevices = 0;
  for(address = 1; address < 127; address++)
  {
    // i2c_scanner dönüş değerini kullanır
    // olup olmadığını görmek için Wire.endTransmission
    // bir cihaz adresi kabul etti.
    Wire.beginTransmission(address);
    error = Wire.endTransmission();

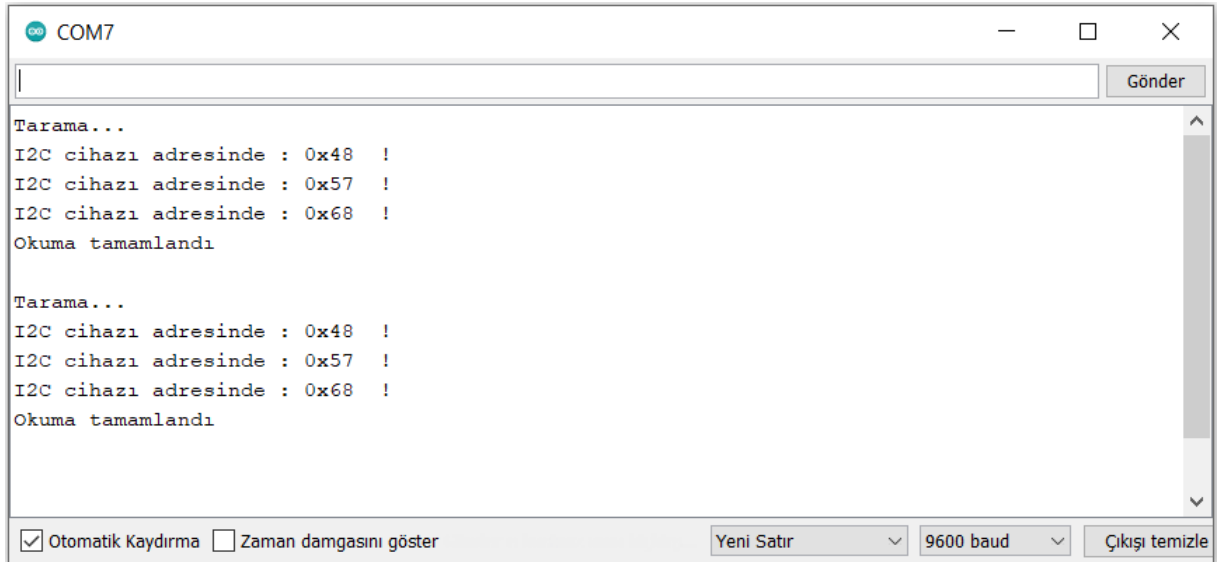
    if (error == 0)
    {
      Serial.print("I2C cihazı adresinde : 0x");
      if (address<16)
        Serial.print("0");
      Serial.print(address,HEX);
      Serial.println("  !");

      nDevices++;
    }
    else if (error==4)
    {
      Serial.print("Adreste bilinmeyen hata 0x");
      if (address<16)
        Serial.print("0");
      Serial.println(address,HEX);
    }
  }
  if (nDevices == 0)
    Serial.println("I2C cihazı bulunamadı\n");
  else
    Serial.println("Okuma tamamlandı\n");

  delay(5000);           // sonraki tarama için 5 saniye bekleyin
```

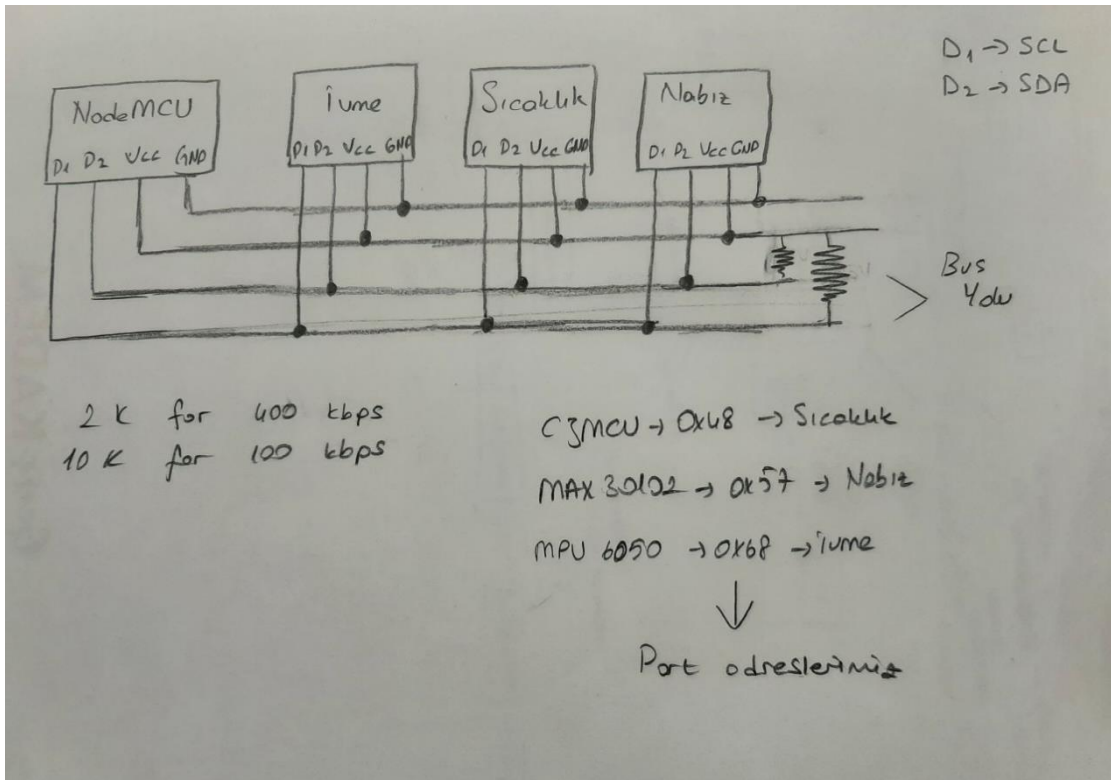
Şekil 3: Adres Belirlemek İçin Kullanılan Kodlar

Ve bu değerleri ekranda gördük.



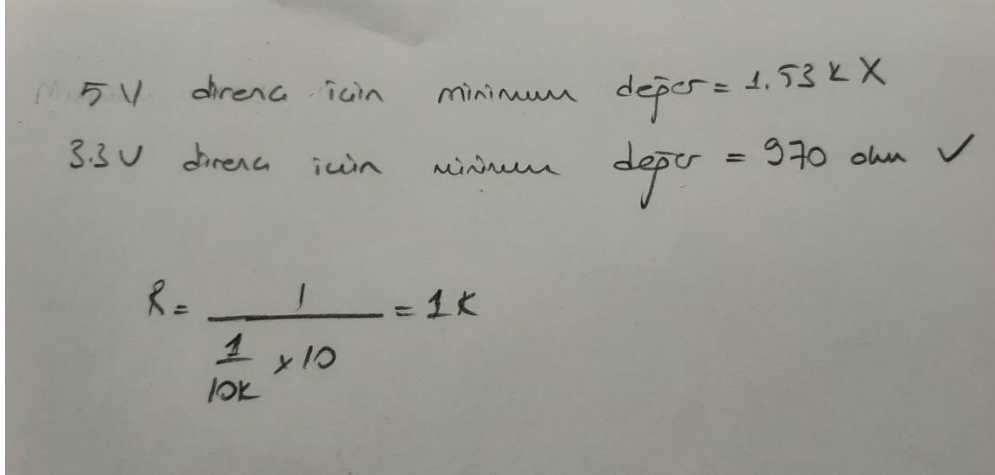
Şekil 4: Adres Değerleri

Sensörlerimiz I2C protokolü ile haberleşmekteydi. Bu sebeple 3 sensörü de aynı mikrodenetleyici üzerindeki aynı portlardan kullanabilmek için yaptığımız araştırmalar sonucu NodeMCU'nun bus yolunda düzenlemeler yapmamız gerektiğini öğrendik. D1-> SCL, D2-> SDA, G, 3V pininden çoğullama yaptık.



Şekil 5: Bus Yolu Tanımlama

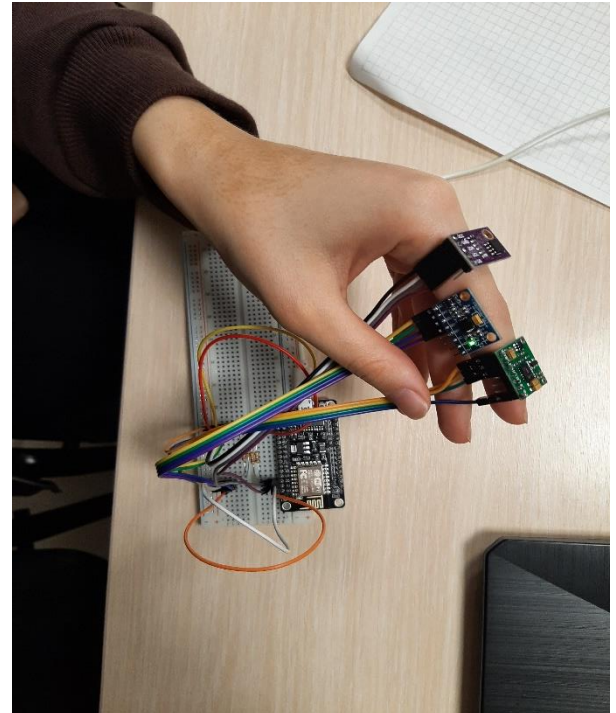
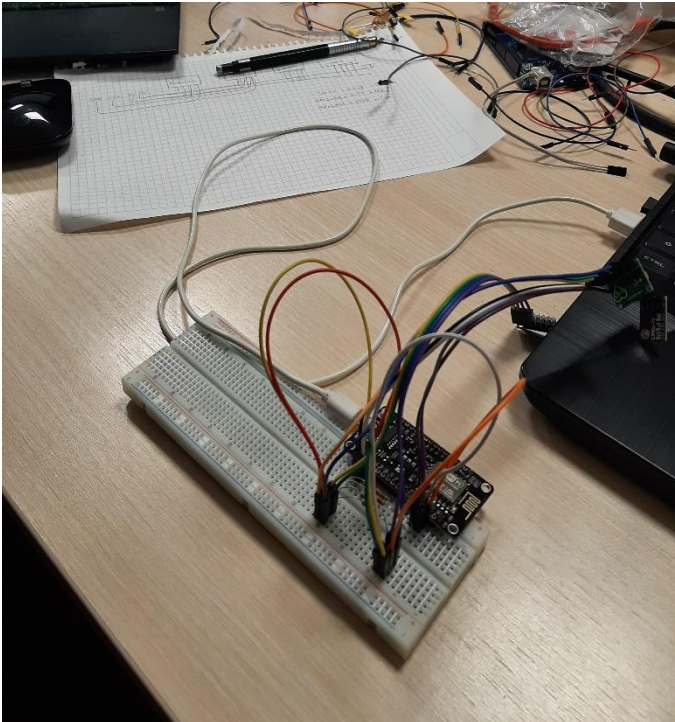
Böylece diğer sensörlerimiz de aynı pini kullanabilmiş oldu. Fakat 3 sensörü de aynı anda boarda takınca kısa devre oluştu ve sıcaklık sensörümüzden koku ve duman yükselmeye başladı. Kısa devreyi engellemek için çözüm olarak direnç kullandık. Kullandığımız dirençlerin değerini bulmak için aşağıdaki formülleri yaptık. Bu denklemden yola çıkarak 5V'luk direnç koymamız gerektiğine karar verdik.

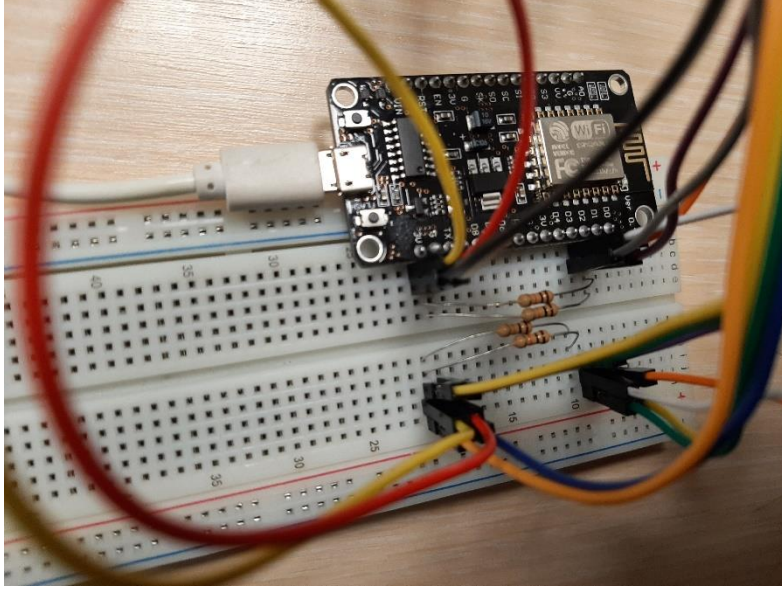

$$5V \text{ direnci için minimum değer} = 1.53 kX$$
$$3.3V \text{ direnci için minimum değer} = 970 \text{ ohm } \checkmark$$
$$R = \frac{1}{\frac{1}{10K} \times 10} = 1K$$

Şekil 6: Direnç Hesaplama

Daha sonra kodlarda 3 sensörün özelliklerini detaylandırarak kritik noktalar belirledik. Kodumuz ve firebase arasında bağlantı gerçekleştirdik. Verileri uygulama ara yüzünde görebilmek için MIT App Inventor uygulamasında ara yüz tasarladık ve uygulamamızın firebase bağlantısını yaptık.

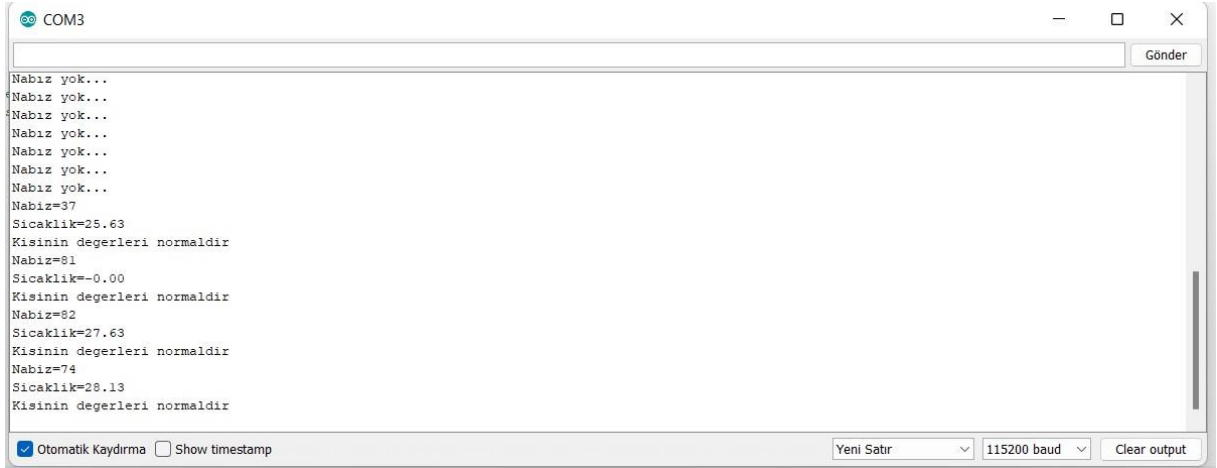
Böylece sensörden alınan bilgi kodda analiz aşamasından geçtikten sonra veriler firebase gitti ve uygulamadan bu veriyi çekmiş olduk. Devremize ait görseller aşağıda bulunmaktadır.



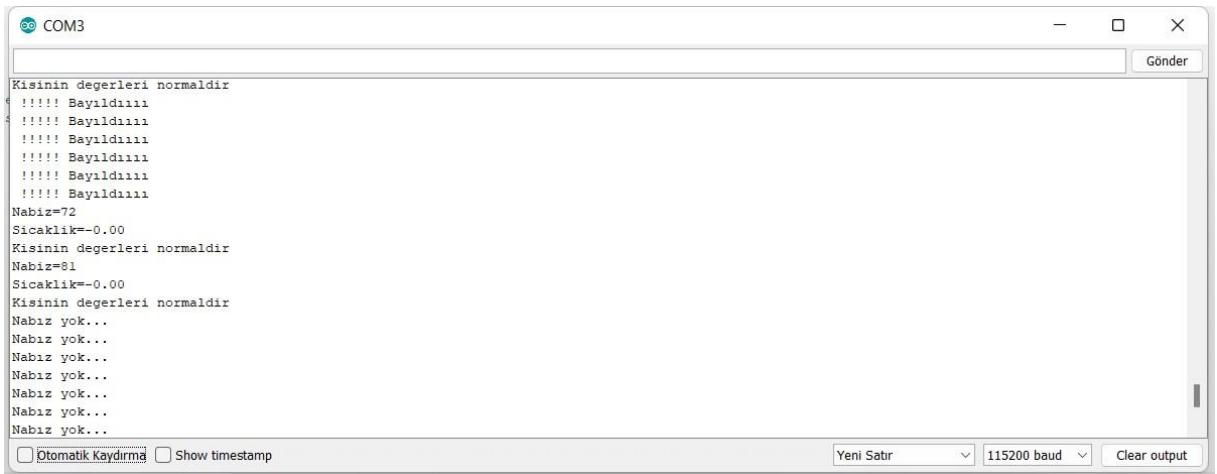


Şekil 7: Deney Fotoğrafları

Kurduğumuz döngüde kabul olarak belirli nabız altında nabız yok yazısını kontrol paneline yazdırdık. Veri tabanına gönderdiğimiz değer ile ekrana “Yakınınızın nabızı alınamıyor. Kritik bir durum olabilir.” yazısı veriyoruz.

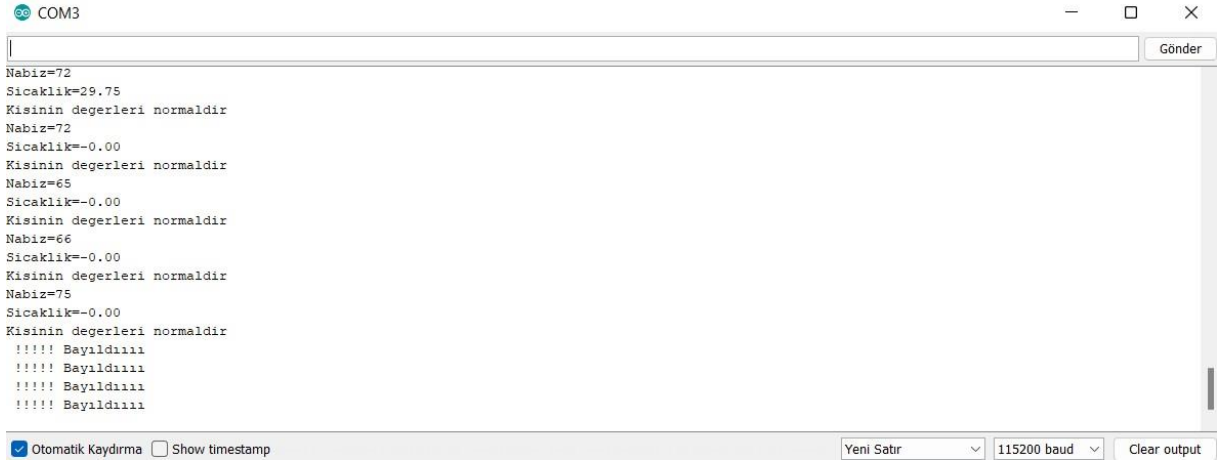


Şekil 8: Nabız Yok Yazısının COM'a Verilimi



Şekil 9: Normal Değerlerin COM'a Verilimi

Değerlerimiz sıcaklık nabız ve ivme olarak kritik sınırlar altına düşünce kullanıcının yakınına “Yakınınızın değerlerinde değişiklik mevcuttur. Bayıldı.” yazısı gösterilir. Daha sonra değerler düzelince “Kişinin değerleri normaldir.” Yazısını uygulamada gösterir.



Şekil 10: Bayıldı Değerinin COM’a Verilimi

Uygulamamızın ara yüzü de aşağıdaki gibidir. Görüntüle butonuna bastığımızda anlık kişi bilgilerini görüntülemektedir.



Şekil 11: Uygulama Ara Yüzü

UYGULAMA KODLARI

```
#include <Wire.h>

#include "MAX30105.h"

#include "heartRate.h"

#include <inttypes.h>

#include <lm75.h>

#include <Adafruit_MPU6050.h>

#include <Adafruit_Sensor.h>

#include <ESP8266WiFi.h>

#include <FirebaseArduino.h>

/* Kbalosuz ag Bilgileri */

#define WLAN_SSID "pluton"

#define WLAN_PASSWORD "123456789"

/**/Firebase Kurulum/**/

#define FIREBASE_HOST "https://akillibileklik-9260a-default-rtdb.firebaseio.com/" //url

#define FIREBASE_AUTH "AlzaSyCoyAvfa5Z59urvK8Agi7MB1pxUXF2xj8o" //firebase türetilen key

Adafruit_MPU6050 mpu;

MAX30105 particleSensor;

Templ2C_LM75 termo = Templ2C_LM75(0x48,Templ2C_LM75::nine_bits);

int temperatureAddress = 0x48; //Sicaklik

int heartrateAddress = 0x57; //Nabiz

int accelerometerAddress = 0x68; //Ivme

const byte RATE_SIZE = 4; //Daha fazla ortalama almak için bunu artırın. 4 iyidir.

byte rates[RATE_SIZE]; //Nabız dizisi

byte rateSpot = 0;

long lastBeat = 0; //Son vuruşun gerçekleştiği saat

float beatsPerMinute;

int beatAvg;

int durum = 0;
```

```

void setup()
{
  Serial.begin(115200);

  Serial.println("Start");

  Serial.print("Actual temp ");

  Serial.print(temro.getTemp());

  Serial.println(" oC");

  delay(2000);

  Wire.beginTransmission(heartrateAddress);

  ///////////////////////////////////NABIZ////////////////////////////////////

  if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) //Default olarak I2C port kullanmaktadır.
  {
    Serial.println("MAX30105 bulunamadi. Lutfen gucu kontrol ediniz.");

    while (1);
  }

  Serial.println("Isaret parmeginizi sabit bir basınca sensore yerlestin.");

  particleSensor.setup(); //Sensörü varsayılan ayarlarla yapılandırın

  particleSensor.setPulseAmplitudeRed(heartrateAddress); //Sensörün çalıştığını belirtmek için Kırmızı LED'i
düşük konuma getirin

  particleSensor.setPulseAmplitudeGreen(0); //Yeşil LED'i kapatın

  ///////////////////////////////////İVME////////////////////////////////////

  while (!Serial)

    delay(10);

  Serial.println("Adafruit MPU6050 test ediliyor!");

  // Başlatmayı deneyin!

  if (!mpu.begin()) {

    Serial.println("MPU6050 bulunamadı");

    while (1) {

      delay(10);

    }

  }

  Serial.println("MPU6050 Bulundu!");

  mpu.setAccelerometerRange(MPU6050_RANGE_8_G);

  Serial.print("İvmeölcer aralığı şu şekilde ayarlandı: ");

  switch (mpu.getAccelerometerRange()) {

```

```
case MPU6050_RANGE_2_G:
    Serial.println("+2G");
    break;
case MPU6050_RANGE_4_G:
    Serial.println("+4G");
    break;
case MPU6050_RANGE_8_G:
    Serial.println("+8G");
    break;
case MPU6050_RANGE_16_G:
    Serial.println("+16G");
    break;
}

mpu.setGyroRange(MPU6050_RANGE_500_DEG);
Serial.print("Gyro aralığı suna ayarlandı:");
switch (mpu.getGyroRange()) {
case MPU6050_RANGE_250_DEG:
    Serial.println("+ 250 deg/s");
    break;
case MPU6050_RANGE_500_DEG:
    Serial.println("+ 500 deg/s");
    break;
case MPU6050_RANGE_1000_DEG:
    Serial.println("+ 1000 deg/s");
    break;
case MPU6050_RANGE_2000_DEG:
    Serial.println("+ 2000 deg/s");
    break;
}

mpu.setFilterBandwidth(MPU6050_BAND_5_HZ);
Serial.print("Filtre bant genişliği şu şekilde ayarlandı:");
switch (mpu.getFilterBandwidth()) {
case MPU6050_BAND_260_HZ:
    Serial.println("260 Hz");
    break;
```

```

case MPU6050_BAND_184_HZ:
    Serial.println("184 Hz");
    break;
case MPU6050_BAND_94_HZ:
    Serial.println("94 Hz");
    break;
case MPU6050_BAND_44_HZ:
    Serial.println("44 Hz");
    break;
case MPU6050_BAND_21_HZ:
    Serial.println("21 Hz");
    break;
case MPU6050_BAND_10_HZ:
    Serial.println("10 Hz");
    break;
case MPU6050_BAND_5_HZ:
    Serial.println("5 Hz");
    break;
}
Serial.println("");
delay(100);
//connect to wifi.
WiFi.begin("pluton","123456789");
Serial.print("connecting");
Serial.println();
Serial.print("connected:");
Serial.println(WiFi.localIP());
Firebase.begin(FIREBASE_HOST,FIREBASE_AUTH);
Firebase.setString("durum","0");
}
void loop()
{
    delay(1000);
    /*
    * int temperatureAddress = 0x48; //Sicaklik

```



```

* int heartrateAddress = 0x57; //Nabiz

* int accelerometerAddress = 0x68; //lvme

*/

sensors_event_t a, g, temp;

mpu.getEvent(&a, &g, &temp);

long irValue = particleSensor.getIR();


if (irValue < 50000)

{ //olu durum

Serial.println("Nabız yok...");

Firebase.getString("durum");

durum = 1;

}

else if(a.acceleration.y < -5)

{

Serial.println(" !!!!! Bayıldım");

Firebase.setString("durum", "1");

durum = 2;

}

else if(thermo.getTemp() > 35)

{

Serial.println(" Ani Sıcaklık Artisi Görüldü");

durum = 3;

}

else

{

irValue = irValue / 1453;

Serial.print("Nabiz=");

Serial.println(irValue);

Serial.print("Sıcaklık=");

Serial.println(thermo.getTemp());

Serial.println("Kisinin degerleri normaldir");

durum = 4;

}

}

```

KAYNAKÇA

- <https://irojournals.com/itdw/V4/I2/05.pdf>
- <https://www.youtube.com/watch?v=QQLfzIPGjjE>
- <https://ieeexplore.ieee.org/abstract/document/8993398/authors#authors>
- <https://lastminuteengineers.com/max30102-pulse-oximeter-heart-rate-sensor-arduino-tutorial/>
- https://demirten.gitbooks.io/gomulu-linux/content/i2c_protokolunun_tanitilmasi.html
- <https://www.robotistan.com/mpu6050-6-eksen-ivme-ve-gyro-sensoru-6-dof-3-axis-accelerometer-and-gyros>
- <http://www.esp8266learning.com/esp8266-lm75-temperature-sensor-example.php>
- <https://deneyapkart.org/proje-189.html>
- <https://koddefteri.net/arduino/temel-arduino-dersleri/arduino-degiskenler-ve-veri-tipleri.html>