BOZOK ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

EV OTOMASYON SİSTEMİ

02/06/2020

YAZILIM MÜHENDİSLİĞİ PROJESİ

TEKNİK DOKÜMAN

Proje Danışmanı: Arş. Gör. Hasan ULUTAŞ

Grup 9
Esra YÜCE
Feyza YILMAZ
Halil SEÇİLMİŞ
Özlem ÖZKAYA

https://github.com/Ev-Otomasyon-Sistemi

İÇİNDEKİLER

TEKNİK DOKÜMAN

1. Mobil Uygulama

Ev otomasyon sisteminde kullanılacak sensörlerin uzaktan kontrolünü sağlamak amaçlı tasarlanmıştır. Kullanıcı evde olsun veya olmasın telefondan Ev Otomasyon Sistemi dâhilindeki bütün sistemleri kontrol edebilir.

1.1. Mobil Uygulamanın İşleyiş Mantığı

Gelişmiş özellikleri sayesinde yaşam konforunuzun üst seviyelere çıkmasına yardımcı olan Ev Otomasyon Sistemi'nde aydınlatma, kapı-perde kontrolü ve oda sıcaklığını görüntüleme komutu gibi özellikleri bulunur. Bu gibi birçok özelliğin bulunduğu sistemi kontrol etmede mobil uygulama kullanılır.

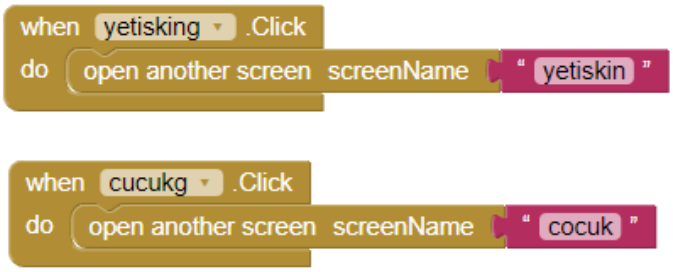
Mobil uygulama, kullanışlı ara yüzü sayesinde her yaşa hitap eden kullanıcı dostu kolay bir kullanım sağlar.

Arduino ünitesine bağlı sensörlerin kontrol bağlantısı bluetooth cihazı aracılığıyla gerçekleşir. Bağlantı sağlanan mobil uygulamadan Arduino ünitesine bağlı olan bütün sensörler kontrol edilebilir.

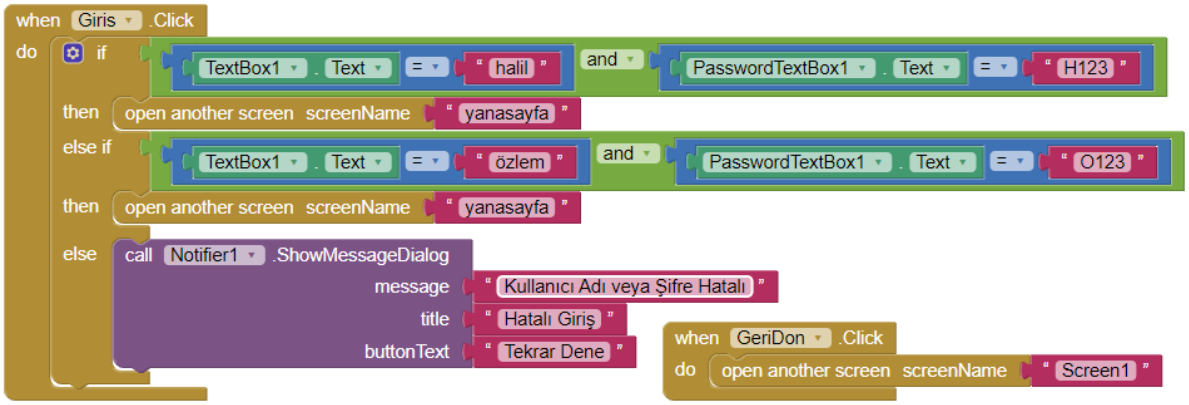
Kullanıcı hesaplarında güvenlik amaçlı bazı kısıtlamalar yer almaktadır. Bu kısıtlamalar daha çok küçük yaştaki kullanıcılar için geçerlidir. Bu sayede birçok sorunu önleyecek güvenlik sağlanır.

1.2. Mobil Uygulama Kodları ve Açıklamaları

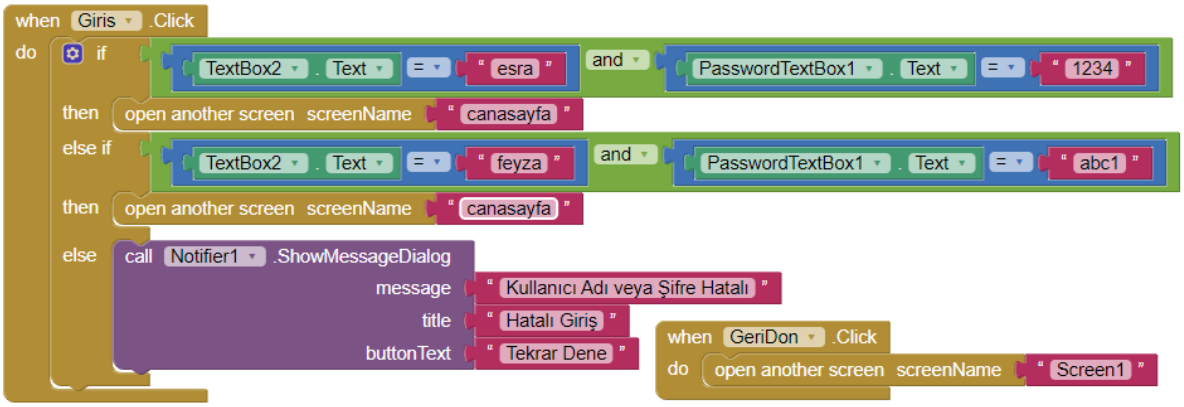
Kullanıcı Tercih Sayfası:



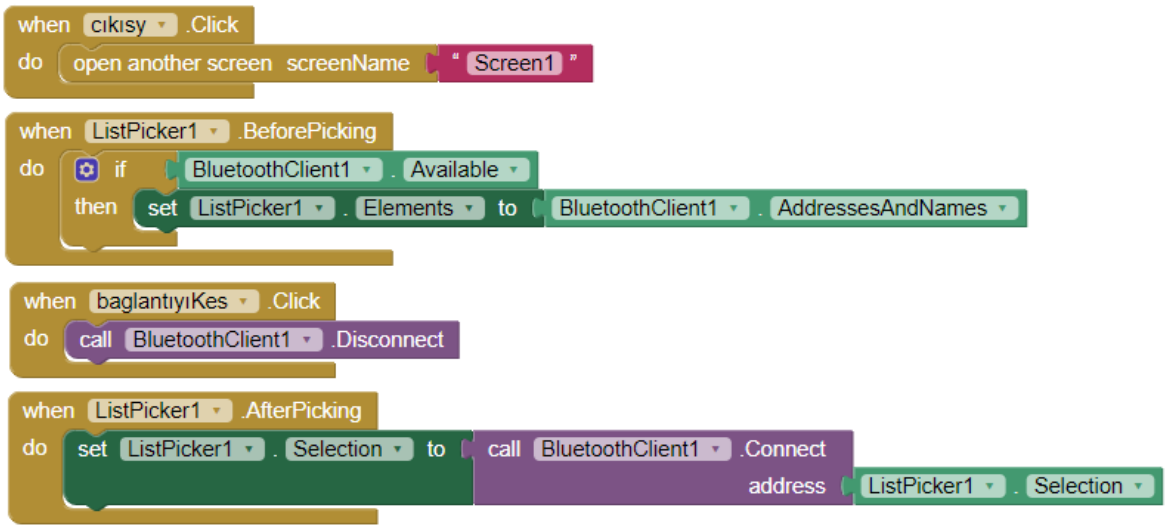
Yetişkin Giriş Ekranı

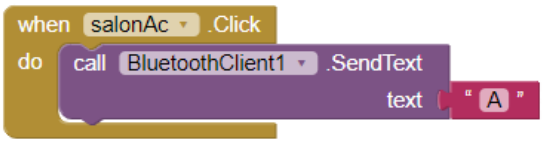


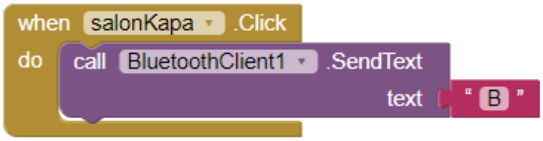
Çocuk Giriş Ekranı

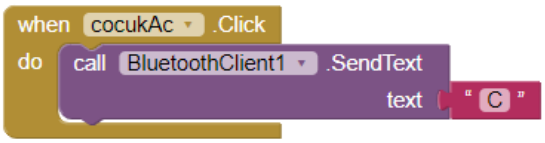


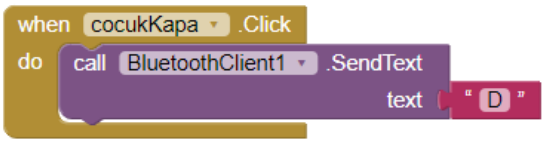
Anasayfa:

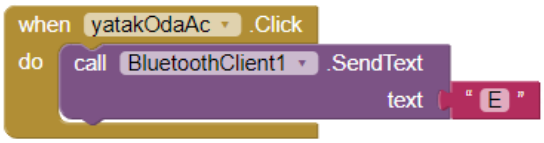


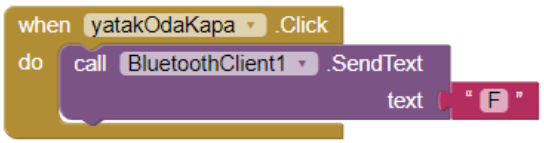

when salonAc .Click
do call BluetoothClient1 .SendText
text " A "

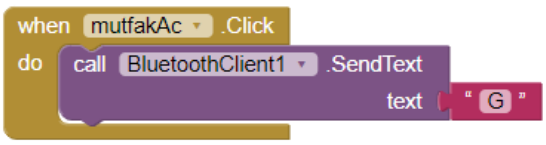

when salonKapa .Click
do call BluetoothClient1 .SendText
text " B "

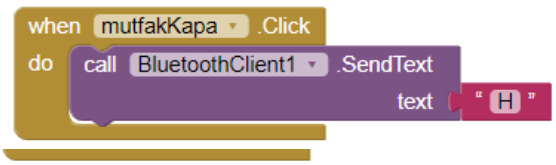

when cocukAc .Click
do call BluetoothClient1 .SendText
text " C "

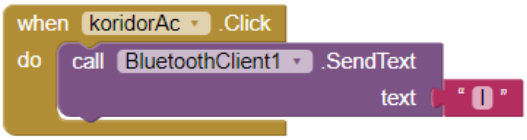

when cocukKapa .Click
do call BluetoothClient1 .SendText
text " D "

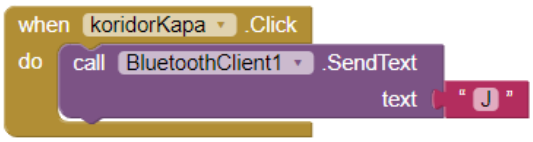

when yatakOdaAc .Click
do call BluetoothClient1 .SendText
text " E "

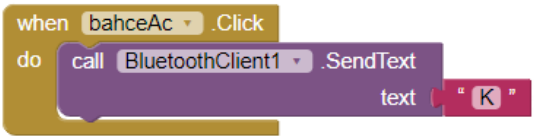

when yatakOdaKapa .Click
do call BluetoothClient1 .SendText
text " F "

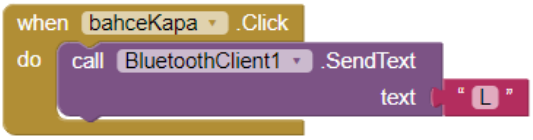

when mutfakAc .Click
do call BluetoothClient1 .SendText
text " G "

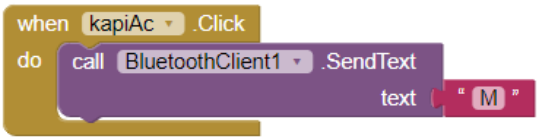

when mutfakKapa .Click
do call BluetoothClient1 .SendText
text " H "

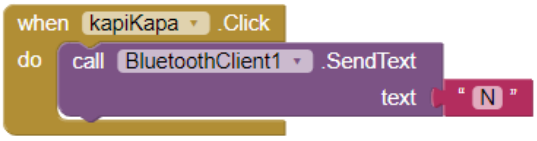

when koridorAc .Click
do call BluetoothClient1 .SendText
text " I "

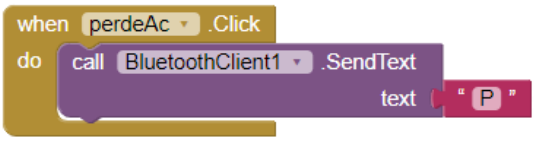

when koridorKapa .Click
do call BluetoothClient1 .SendText
text " J "

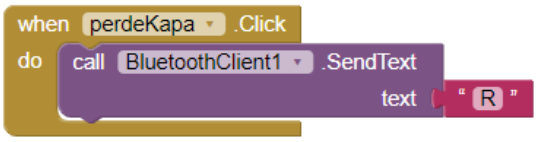

when bahceAc .Click
do call BluetoothClient1 .SendText
text " K "

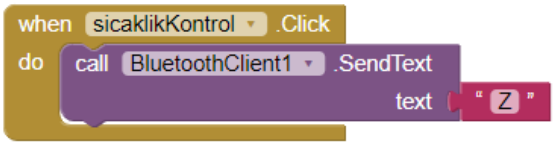

when bahceKapa .Click
do call BluetoothClient1 .SendText
text " L "

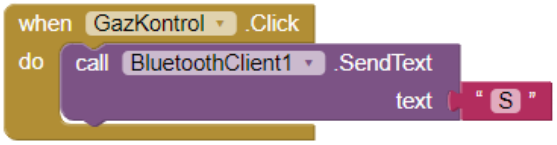

when kapiAc .Click
do call BluetoothClient1 .SendText
text " M "

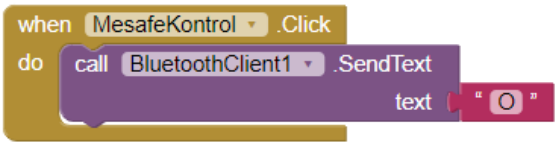

when kapiKapa .Click
do call BluetoothClient1 .SendText
text " N "

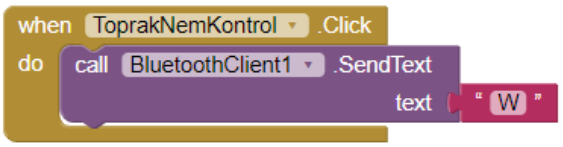

when perdeAc .Click
do call BluetoothClient1 .SendText
text " P "


when perdeKapa .Click
do call BluetoothClient1 .SendText
text " R "


when sicaklikKontrol .Click
do call BluetoothClient1 .SendText
text " Z "


when GazKontrol .Click
do call BluetoothClient1 .SendText
text " S "


when MesafeKontrol .Click
do call BluetoothClient1 .SendText
text " O "


when ToprakNemKontrol .Click
do call BluetoothClient1 .SendText
text " W "

Uygulama 5 ayrı sayfadan oluşmaktadır. Sayfa düzeni için dikey sıralamada **VerticalArrangement**(Birbirinin altında görüntülenmesi gereken bileşenlerin yerleştirileceği bir biçimlendirme öğesi.), yatay sıralamada ise **HorizontalArrangement**(Soldan sağa görüntülenmesi gereken bileşenlerin yerleştirileceği bir biçimlendirme öğesi.) kullanılmıştır. Bu öğelerin içerisine gerekli Button, TextBox ve Label bileşenleri yerleştirilmiştir.

İlk sayfada iki tane buton yer almaktadır. Butonlara tıklanarak **open another screen** komutu ile butona atanan sayfaya yönlendirme yapılır. Bu iki butonun her birine atanan farklı sayfalar dolayısıyla farklı sayfalara geçiş sağlanır. Açılan sayfalar Çocuk Girişi ve Yetişkin Girişi sayfalarıdır. Bu sayfalarda kullanıcı, kullanıcı adını ve şifresini girer. Girilen bilgiler giriş kontrolüne bağlı olan **matematik** ve **lojik** yapılar sayesinde kayıtlı kullanıcı bilgileri ile karşılaştırılır. Bilgiler doğru ise 'Giriş Yap' butonuna atanan sayfaya geçiş yapılır, bilgiler yanlış ise **'Kullanıcı adı veya şifre hatalı.'** uyarısı verip **'Tekrar dene'** seçeneği sunar. Bu hata mesajı **Notifier**(uyarı iletişim kutularını, iletileri ve geçici uyarıları görüntüler) öğesi ile sağlanır. Giriş sağlandıktan sonra son olarak Ev otomasyon sistemi maketinde bulunan sensörlerin kontrolünün sağlandığı ana sayfa açılır. Bluetooth bağlantısını sağlamak için ilk olarak **.BeforePickink** komutu ile bluetooth adresleri aranır ve **.AfterPicking** kontrolünde **.Connect address** ile çıkan ve bluetooth cihazımız ile eşleşen adres seçilir. Aynı şekilde bağlantıyı kesmek için **.Disconnect** kullanılır. Sensör kontrolünü sağlayan her bir buton bluetooth bağlantısı sayesinde **.SendText** komutu ile veri göndererek sensör kontrolünü sağlar. Kontrolü sağlanan belirli sensörlerin işlem değeri LDC ekrana yansıtılır.

2. Arduino Sistemi

Ev otomasyon sistemi özelliklerinin gerçekleştirildiği tüm sensörleri kapsayan sistemdir. Mobil uygulamadan verilen komutları işleyerek kullanıcının istediği bir şekilde geri dönüş sağlanmak amaçlanır.

2.1.Arduino Sisteminin İşleyiş Mantığı

Ev otomasyon sisteminde aşağıda tanımları verilen sensör ve modüller yer almaktadır.

Arduino Uno: çeşitli sensörlerden fiziksel bilgi almak, bu bilgiler ile çeşitli mekanizmalar tasarlamak için kullanılır.

Ayrıca motor, LED, buzzer gibi uyarıcılardan bir çıktı elde etmek için kullanılabilir.

LED: elektrik enerjisini ışığa dönüştüren yarı iletken bir devre elemanıdır.

Servo Motor: bir mekanizmanın performansını etkileyecek hataları; geri bildirim sinyalleri yardımıyla denetler. Kısa zaman aralığında hataları kontrol edecek ve bu hataları engelleyecek, çoğunlukla bir DC tür motor çeşididir.

Servo motorlar, dönüş yönünün belirli açılarda dönmesi istenilen uygulama alanlarında çok tercih edilmektedir.

Dalgıç Su Pompası: genellikle basınç ile su temini sağlar.

Çift Kanallı Motor Sürücü: iki yönde de iki motoru birbirlerinden bağımsız olarak kontrol edebileceğimiz bir motor sürücü kartıdır.

Bluetooth Modülü: kablosuz seri haberleşme uygulamaları için tasarlanmıştır.

Isı-Nem Sensörü: sıcaklık ve nem algılayıcı kalibre edilmiş dijital sinyal çıkışı veren gelişmiş bir sensördür.

Doğal Gaz ve Metan Gazı Ölçümleme Modülü: 300ppm ve 10000ppm arasında gaz kaçağı tespiti için uygun bir dizi konsantrasyonlarda metan (CNG) doğal gaz varlığını algılar.

Toprak-Nem Sensörü: toprağın içerisindeki nem miktarını veya ufak ölçekte bir sıvının seviyesini ölçmek için kullanılan bir sensördür.

Mesafe Sensörü: mesafe ölçmek amacıyla kullanılan araçtır. Sensörün bulunduğu yerden saptadığı cisim ile aradaki mesafeyi elektriksel çıkış olarak verir.

Hareket Sensörü: nesnelerin veya canlıların hareketlerini kızılötesi, ultrasonik, mikrodalga, titreşim yöntemleriyle algılanmasını ve bu algılama sonucu ilgili mekanizmaları tetiklemesini sağlayan aktif ve pasif özellikteki sensörlerdir.

Buzzer: verilen voltaja göre farklı ses sinyalleri sağlayan bir cihazdır.

Ev otomasyon sistemi maketinde, belli başlı sensörler ve modüller bir arada kullanılıp, gerçekleştirilmek istenen işlevi yerine getirecek şekilde devre sistemi tasarlanmıştır. Bunun yanında tüm sensör ve modüller Arduino uno ile bağlantılıdır.

Sistemin bütün bu bileşenleri kapsayan ve haberleşerek bir arada etkileşimini sağlayan elektronik devresi bulunmaktadır. Devre karmaşıklığı, baskı devre oluşturularak önlenmiş olup daha rahat çalışma ortamı sağlanmıştır.

Kapı sisteminde, mobil uygulama kontrolü dâhilinde kapıyı açıp kapatmak için servo motor kullanılmıştır. Servo motor ile etkileşim içerisinde olan hareket sensörü ve buzzer ile yabancı girişleri önlemek amaçlanmıştır. Hareketi algılayan hareket sensörü buzzer sistemini tetikler ve alarm uyarısı verilir.

Çiçek sulama sisteminde ise toprağın nem değerini ölçmek için toprak-nem sensörü kullanılmıştır. Toprak-nem sensörü ile etkileşimde olan dalgıç su pompası, toprağın kuruluğu algılanması durumunda çiçeklere su pompalar.

2.2.Arduino Kodları ve Açıklamaları

kod §

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);

#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT11
DHT evotomasyonu(DHTPIN, DHTTYPE);

#include <Servo.h>
Servo kapiservo;
Servo pencereservo;

const int mileri=9;
const int mgeri=10;

const int pirPin = 8;
const int gazsensor=A0;
const int trig = 3;
const int echo = 4;

const byte  latchPin = 7;
const byte dataPin  = 6;
const byte clockPin = 5;
int x = 0;
int sayac=0,btgelen,btgelenveri,gazdegeri,nemdegeri,sicaklikdegeri,mesafedegeri,skontrol;
```

#include <Wire.h> bağlantı fonksiyonlarını bulunduran kütüphane

#include <LiquidCrystal_I2C.h> :LCD ekran çalışma ve bağlantısını bulunduran kütüphane

#include "DHT.h" sıcaklık-nem sensörü bağlantısını bulunduran kütüphane

#include <Servo.h> servo motor bağlantısını bulunduran kütüphane

Burada sisteme gerekli kütüphane ekleme işlemleri yapılır.

LiquidCrystal_I2C lcd (0x27, 16, 2);

LCD ekranın adresi ve ekran boyutu hakkında bilgi verildi.

#define DHTTYPE DHT11

#define (sabit değişkenler için kullanılır) sıcaklık-nem sensörü için kullanılacak değişken tanımlandı

Servo kapiservo;

Servo pencereservo;

Servo kütüphanesindeki nesneler üretildi ve kapı ve pencere için isimlendirildi. **const**(sabit değerler için kullanılır) Burada sistemde kullanılacak değişkenlerin pinleri belirlendi. Yani arduinodaki konumları belirlendi. Çoklu led kontrolü için **Shift Register**(74HC595) kullanıldı hafızada tutma işlemleri için **.int** değişebilen yani sabit olmayan değişkenler tanımlandı.

```
int buzzerpin=13;

byte led1,led2,led3,led4,led5,led6;

byte kapi,penc;
void setup()
{
  Serial.begin(9600);
  lcd.begin(); //lcd hazırlandı
  lcd.backlight(); // arka plan ışığı ayarlandı
  kapiservo.attach(A3);  // attaches the servo on pin 9 to the servo object
  pencereservo.attach(A1 );
  pinMode(latchPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(pirPin, INPUT);
  pinMode(mileri,OUTPUT);
  pinMode(mgeri,OUTPUT);
  pinMode(buzzerpin,OUTPUT);
```

Void setup() metottu içinde genel ayarlamalar yapıldı ve bu metot sadece 1 kez çalışır. **Serial.begin()** ile seri haberleşme hızı bellirlendi. **lcd.begin()**, **lcd.backlight()** LCD ekranın genel olarak ayarladı arka ışığı. kapiservo.**attach(),** pencereservo.**attach()** servo kütüphanesinde oluşturduğumuz nesnenin Arduinodaki pini belirlendi. **pinMode()** ile ledler için tanımlanan pinleri çıkış pini olarak ayarlandı.

```
    lcd.setCursor(7,0); //yazının lcd ekrandaki konumu belirlendi
    lcd.print("EV");
    lcd.setCursor(3,1);
    lcd.print("OTOMASYONU");
    delay(3000);
    lcd.clear();


}

void loop()
{
  if(Serial.available() > 0){
      btgelenveri = Serial.read();
    }
 //Serial.println(btgelenveri);

bluetoothkontrol(btgelenveri);
gazdegeri=gaz();
nemdegeri=topraknem();
sicaklikdegeri=sicaklik();
mesafedegeri=mesafeolc();
lcdyaz(gazdegeri,nemdegeri,mesafedegeri,sicaklikdegeri,btgelenveri);
```

lcd. **setCursor()** ile LCD konumu, lcd.**print()** LCD ekrana yazılacak yazı, lcd.**clear()** LCD ekranı temizlemesi belirlendi.

void loop() metodu arduinonun yapması gereken işlemleri yazıldı. Bu metot birde fazla kez çalışabilir. İlk kullanılan **if** işlemi ile gelen veri kontrol edildi.

Serial.read() ile gelen veri aktarıldı. Sonrasında gelen değeri yukarıda tanımlanan değişkenlere atama işlemi yapıldı.

if(gazdegeri>300) koşulu ile gelen gaz değeri belirtilenden yüksek ise alarm çalması sağlandı.

if(mesafedegeri<4) koşulu ile arada bırakılan mesafe değeri belirtilenden az olduğu zaman alarm çalması sağlandı.

if(nemdegeri<400) koşulu ile nem değeri belirtilenin altında ise sulama işlemi yapılması sağlandı.

```
int bluetoothkontrol(int btgelen){


     if(btgelen==65)
    led1=1;
    if(btgelen==66)
    led1=0;

     if(btgelen==67)
    led2=1;
    if(btgelen==68)
    led2=0;

     if(btgelen==69)
    led3=1;
    if(btgelen==70)
    led3=0;

     if(btgelen==71)
    led4=1;
    if(btgelen==72)
    led4=0;

     if(btgelen==73)
    led5=1;
    if(btgelen==74)
    led5=0;
```

int bluetoothkontrol(int btgelen) burada arduino ile mobil uygulama bağlantısı kontrol ediliyor bunun doğrultusunda diğer işlemler gerçekleşiyor**.**

Mobil uygulamadaki kapat-aç butonlarına verilen harfler acii tablosundan çevrilerek if koşup yaapısı içine yazıldı. **btnglen'**nin değeri hangi sayıya eşit ise ona göre açma-kapama işlemi **y**apıldı. **(0: kapat,1: aç)**

```
    if(btgelen==77)
    kapi=0;
    if(btgelen==78)
    kapi=45;

    if(btgelen==80)
    penc=45;
    if(btgelen==82)
    penc=0;




    if(hareket()==1 && kapi==45 )
    buzzer(50);



     led(led1,led2,led3,led4,led5,led6);
     kapiservo.write(kapi);
     delay(20);
      pencereservo.write(penc);

     delay(150  );

}
```

```
int led(byte a,byte b,byte c,byte d,byte e,byte f){
  boolean dizi[] = {0, 0, f, e, d, c, b, a};  //nokta,g,f,e,d,c,b,a  = 1 acık 0 kapali
  digitalWrite(latchPin, 0);

  for (int i = 0 ; i < 8 ; i++)
  {
    x = dizi[i];                    //for dönügü boyunca dizinin i. değeri x'e atanır

    digitalWrite(dataPin, x);       //datapin'e x değeri verilir
    digitalWrite(clockPin, 1);      //saat darbesi ile yazma ve kaydırma işlemi yapılır
    digitalWrite(clockPin, 0);
  }
  digitalWrite(latchPin, 1);        //8 bitli dizi çıkışa verilir
}
```

Kapı için servo motor uygulamada butona tıklandığı zaman **delay(20)** komutu ile 0.02 saniye sonra işlevini yapar.

Pencere için servo motor uygulamada butona tıklandığı zaman **delay(150)** komutu ile 0.15 saniye sonra işlevini yapar.

Kullanılan ledlerin hepsi fonksiyonun içine parametre olarak aktarıldı. Aşağıda tekrar çağırılarak isimlendirildi.

int **led(byte,). boolean dizi[]** şeklinde dizi oluşturuldu. Çoklu led kullanılacağı için bir diziye kayıt edip işlemleri gerçekleştirildi. Sonrasında oluşturulan **for (int i = 0 ; i < 8 ; i++)** döngüsü ile ledlerin durumları hafızaya kayıt edildi.

```
void lcdyaz(int gazveri,int nemverisi,int mesafeverisi,int sicaklikverisi,int bluetoothveri){

  //Serial.println(bluetoothveri);

 if(bluetoothveri==90){

    lcd.setCursor(7,0);
    lcd.print("SIC");
    lcd.setCursor(9,1);
    lcd.print(" ");
    lcd.setCursor(7,1);
    lcd.print(sicaklikverisi);
 }
```

Lcd ekranına yazı yazmak için **lcdyaz** adında bir fonksiyon tanımlandı.

Lcdyaz fonksiyonun **gazveri, nemverisi, mesafeverisi, sicaklikverisi, bluetoothveri** olmak üzere 5 tane parametresi vardır ve gelen değerleri lcd ekranına yazdırır.

İf komutu bluetooth verisini kontrol eder. Eğer bluetooth verisi 90(sıcaklığı göster anlamına gelir) değerine eşit ise;

setCursor(7,0) komutu yazının hangi satırda hangi sutunda olacağını belirtir.

7. sutun, 0.satıra ekran imlecini ayarlar. 7.sutun, 0.satır itibariyle ekrana “SIC” yazar.

9. sutun, 1.satıra ekran imlecini ayarlar. 9.sutun, 1.satır itibariyle ekrana “ ” yazar.

7. sutun, 1.satıra ekran imlecini ayarlar. 7.sutun, 1.satır itibariyle ekrana DHT11 sensöründen aldığı **sicaklikverisi**’ni yazar.

```
if(bluetoothveri==79){
    if(mesafeverisi<100){
    lcd.setCursor(9,1);
    lcd.print(" ");
    lcd.setCursor(10,1);
    lcd.print(" ");
 }
  if(mesafeverisi<10){
    lcd.setCursor(8,1);
    lcd.print(" ");
    lcd.setCursor(9,1);
    lcd.print(" ");
 }
    lcd.setCursor(7,0);
    lcd.print("MES");
    lcd.setCursor(7,1);
    lcd.print(mesafeverisi);
}
```

İf komutu bluetooth verisini kontrol eder. Eğer bluetooth verisi 79 değerine eşit ise;

Mesafeverisine bakar.

Eğer mesafeverisi 100 değerinden küçük ise;

9. sutun, 1.satıra ekran imlecini ayarlar. 9.sutun, 1.satır itibariyle ekrana “ ” yazar.

10. sutun, 1.satıra ekran imlecini ayarlar. 10.sutun, 1.satır itibariyle ekrana “ “ yazar.

Eğer mesafeverisi 10 değerinden küçük ise;

8. sutun, 1.satıra ekran imlecini ayarlar. 8.sutun, 1.satır itibariyle ekrana “ ” yazar.

9. sutun, 1.satıra ekran imlecini ayarlar. 9.sutun, 1.satır itibariyle ekrana “ “ yazar.

Eğer hiçbiri değil ise;

7. sutun, 0.satıra ekran imlecini ayarlar. 7.sutun, 0.satır itibariyle ekrana “ MES” yazar.

7. sutun, 1.satıra ekran imlecini ayarlar. 7.sutun, 1.satır itibariyle ekrana HC-SR04 sensöründen aldığı **mesafeverisi** yazar.

```
if(bluetoothveri==83){
   if(gazveri<100){
   lcd.setCursor(9,1);
   lcd.print(" ");
   lcd.setCursor(10,1);
   lcd.print(" ");
 }
   lcd.setCursor(7,0);
   lcd.print("GAZ");
   lcd.setCursor(7,1);
   lcd.print(gazveri);

}
```

İf komutu bluetooth verisini kontrol eder. Eğer bluetooth verisi 83 değerine eşit ise;

Gazverisine bakar. Eğer **gazverisi** 100 değerinden küçük ise;

9. sutun, 1.satıra ekran imlecini ayarlar. 9.sutun, 1.satır itibariyle ekrana “ ” yazar.

10. sutun, 1.satıra ekran imlecini ayarlar. 10.sutun, 1.satır itibariyle ekrana “ “ yazar.

Eğer **gazverisi** 100 değerinden büyük ise;

7. sutun, 0.satıra ekran imlecini ayarlar. 7.sutun, 0.satır itibariyle ekrana “ GAZ” yazar.

7. sutun, 1.satıra ekran imlecini ayarlar. 7.sutun, 1.satır itibariyle ekrana MQ-2 sensöründen aldığı **gazverisi** yazar

```
if(bluetoothveri==87){
   if(nemverisi<1000){
   lcd.setCursor(10,1);
   lcd.print(" ");
   lcd.setCursor(11,1);
   lcd.print(" ");
 }
    lcd.setCursor(7,0);
   lcd.print("NEM");
   lcd.setCursor(7,1);
   lcd.print(nemverisi);
}
```

İf komutu bluetooth verisini kontrol eder. Eğer bluetooth verisi 87 değerine eşit ise;

Nemverisine bakar. Eğer **nemverisi** 1000 değerinden küçük ise;

10. sutun, 1.satıra ekran imlecini ayarlar. 10.sutun, 1.satır itibariyle ekrana “ ” yazar.

11. sutun, 1.satıra ekran imlecini ayarlar. 11.sutun, 1.satır itibariyle ekrana “ “ yazar.

Eğer **nemverisi** 1000 değerinden büyük ise;

7. sutun, 0.satıra ekran imlecini ayarlar. 7.sutun, 0.satır itibariyle ekrana “ NEM” yazar.

7. sutun, 1.satıra ekran imlecini ayarlar. 7.sutun, 1.satır itibariyle ekrana DHT11 sensöründen aldığı **nemverisi** yazar.

```
int gaz(){
  int gazdeger=analogRead(gazsensor);
  return gazdeger;
}
int mesafeolc(){
  int sure,mesafe;
  digitalWrite(trig, HIGH);
  delayMicroseconds(1000);
  digitalWrite(trig, LOW);
  sure = pulseIn(echo, HIGH);
  mesafe = (sure/2) / 29.1;
  return mesafe;
}

float sicaklik(){
  float t = evotomasyonu.readTemperature();
return t;
}
```

Gaz ın çıkış pininden bir değer okunur ve o değer return olarak ana fonksiyona döndürülür.

Gaz isminde int tipinde değer döndüren bir fonksiyon oluşturuldu.

MQ-2 sensöründen okunan analog değer **gazdeger** isimli değişkene aktarılır.

Sonuç olarak sensörden aldığı **gazdegeri**'ni return olarak ana fonksiyona döndürür ve ekrana yazdırılır.

Mesafeolc isminde int tipinde değer döndüren bir fonksiyon oluşturuldu. Sure ve mesafe isminde 2 tane değişken oluşturuldu.

İlk olarak trig pinini HIGH durumunda başlatıyoruz.

1 saniye aralıklarla ses dalgası yayılır.

Yeni ses dalgası üretebilmesi için trig pini pasif duruma getirilir.

Mikrofona herhangi bir ses dalgası çarparsa bunun için bir süre tutar. Bu süreye çarpıp geri gelmesinde iki kere iş yapmış olur. Bu süreyi 2 ye bölüp, 29.1(sesin havada yayılma hızı)'e bölünür ve mesafe adlı değişkene atanır.

Sonuç olarak geriye mesafe değişkeni return olarak ana fonksiyona döndürülür ve ekrana yazdırılır.

Sicaklik isminde **float** tipinde değer döndüren bir fonksiyon oluşturuldu.

readTemperature metodu sayesinde DHT11 sensöründen sıcaklık değeri okunur ve t değişkenine aktarılır.

Sonuç olarak sensörden aldığı sıcaklık değeri return olarak ana fonksiyona döndürülür ve ekrana yazdırılır.

```
int hareket(){
  int hdeger;
  int pirDeger = digitalRead(pirPin);

if (pirDeger == HIGH)
hdeger=1;
else
hdeger=0;

return hdeger;

}
int servokapi(int kapiderece){
kapiservo.write(kapiderece);
}
int servopencere(int penceredерece){
 pencereservo.write(penceredерece);
}
```

Hareketi algılamak için **hareket** isminde int tipinde değer döndüren bir fonksiyon oluşturuldu.

digitalRead komutuyla PIR sensöründen okuma yapılır ve **pirDegerine** aktarılır.

Eğer alınan **pirDegerinde** hareketlilik var ise **hdeger** yani hareket değeri 1 olur.

Eğer alınan **pirDegerinde** hareketlilik yok ise **hdeger** yani hareket değeri 0 olur.

Sonuç olarak sensörden aldığı hareket değeri döndürülür.

Kapının kontrolü için **servokapi** isminde int tipinde değer döndüren ve kapı derecesini alan bir fonksiyon oluşturuldu.

kapiservo.write(kapiderece); kodu ile servo motorun **kapiderecesi** açı değerine dönmesini sağlar.

Pencerenin kontrolü için **servopencere** isminde int tipinde değer döndüren ve pencere derecesini alan bir fonksiyon oluşturuldu.

pencereservo.write(kapiderece); kodu ile servo motorun **penceredерece** açı değerine dönmesini sağlar.

```
void motor(int msure){

  delay(1000);
  sayac=sayac+1;
  if(sayac>msure){
   digitalWrite(mileri,0);
  digitalWrite(mgeri,0);
  }
  else{
    digitalWrite(mileri,0);
  digitalWrite(mgeri,1);
  }
//Serial.println(sayac);

}

int topraknem(){
  int nemdeger=analogRead(A2);
   return nemdeger;
}
void buzzer(int ton){
  digitalWrite(buzzerpin,1);
  delay(ton);
   digitalWrite(buzzerpin,0);
  delay(ton);

}
```

Motor isminde, motor suresini alan ve değer döndürmeyen bir fonksiyon oluşturuldu.

1000 milisaniye bekletilir. Sayaç oluşturulur ve her seferinde sayaç 1 arttırılır.

Eğer sayaç değeri motor süresinden büyükse, **digitalWrite** fonksiyonu sayesinde motor ileri pini logıc 0 ve motor geri pini logic 0 yapılır.

Sayaç değeri motor süresinden küçükse, motor ileri pini logic 0 ve motor geri pini logıc 1 yapılır.

Toprak ve nem ölçümü yapmak için **topraknem** isimli int tipinde değer döndüren ve değer almayan bir fonksiyon oluşturuldu.

Toprak nem sensöründen okunan analog değer **nemdeger** isimli değişkene aktarılır.

Sonuç olarak sensörden aldığı **nemdegeri** döndürür.

Buzzer kontrolünü sağlamak için **buzzer** isminde değer döndürmeyen ve int tipinde değer alan bir fonksiyon oluşturuldu.

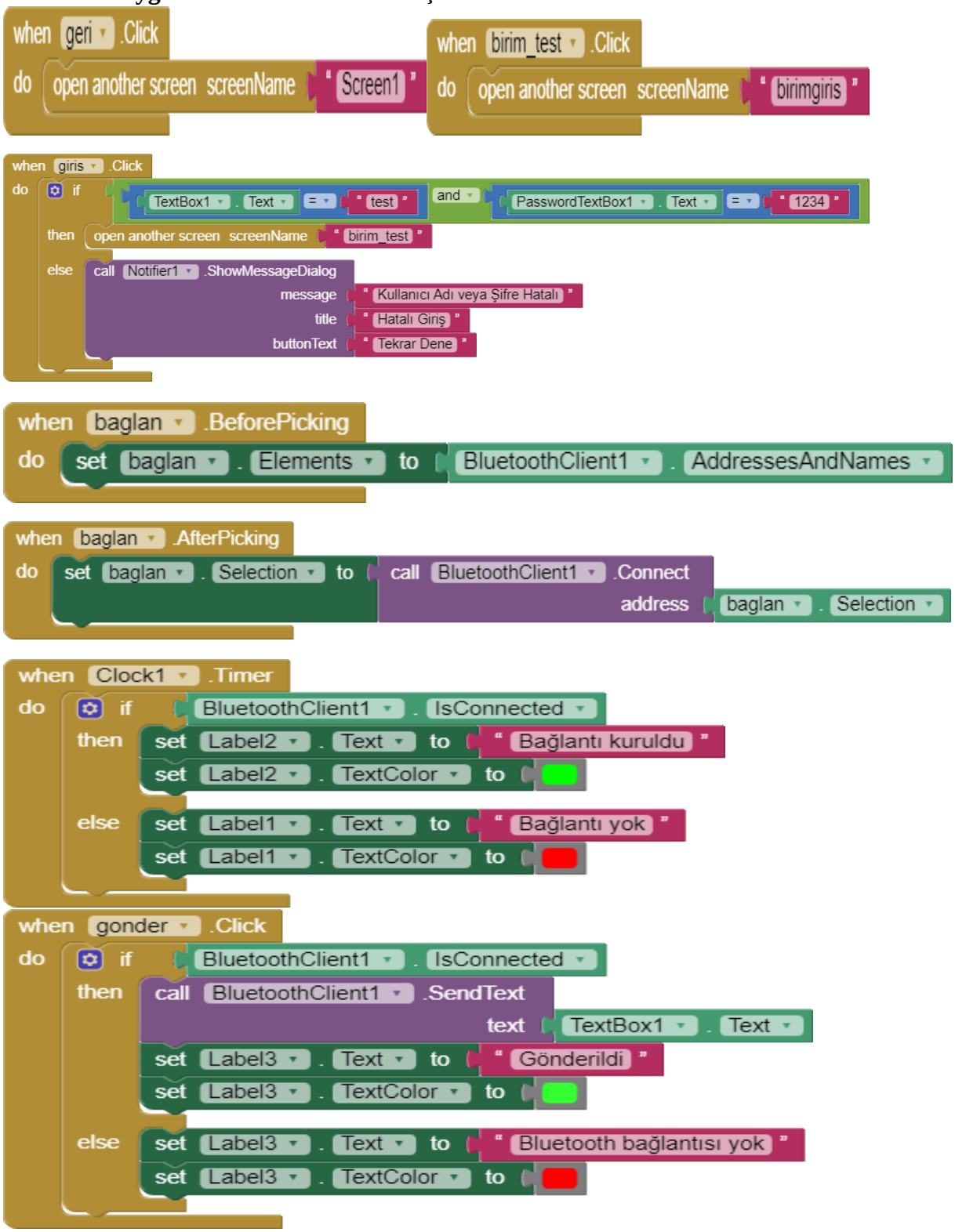
digitalWrite fonksiyonu sayesinde buzzerpin pini logıc 1 yapılır. **Delay(ton)** fonksiyonu sayesinde buzzer ses çıkartır.

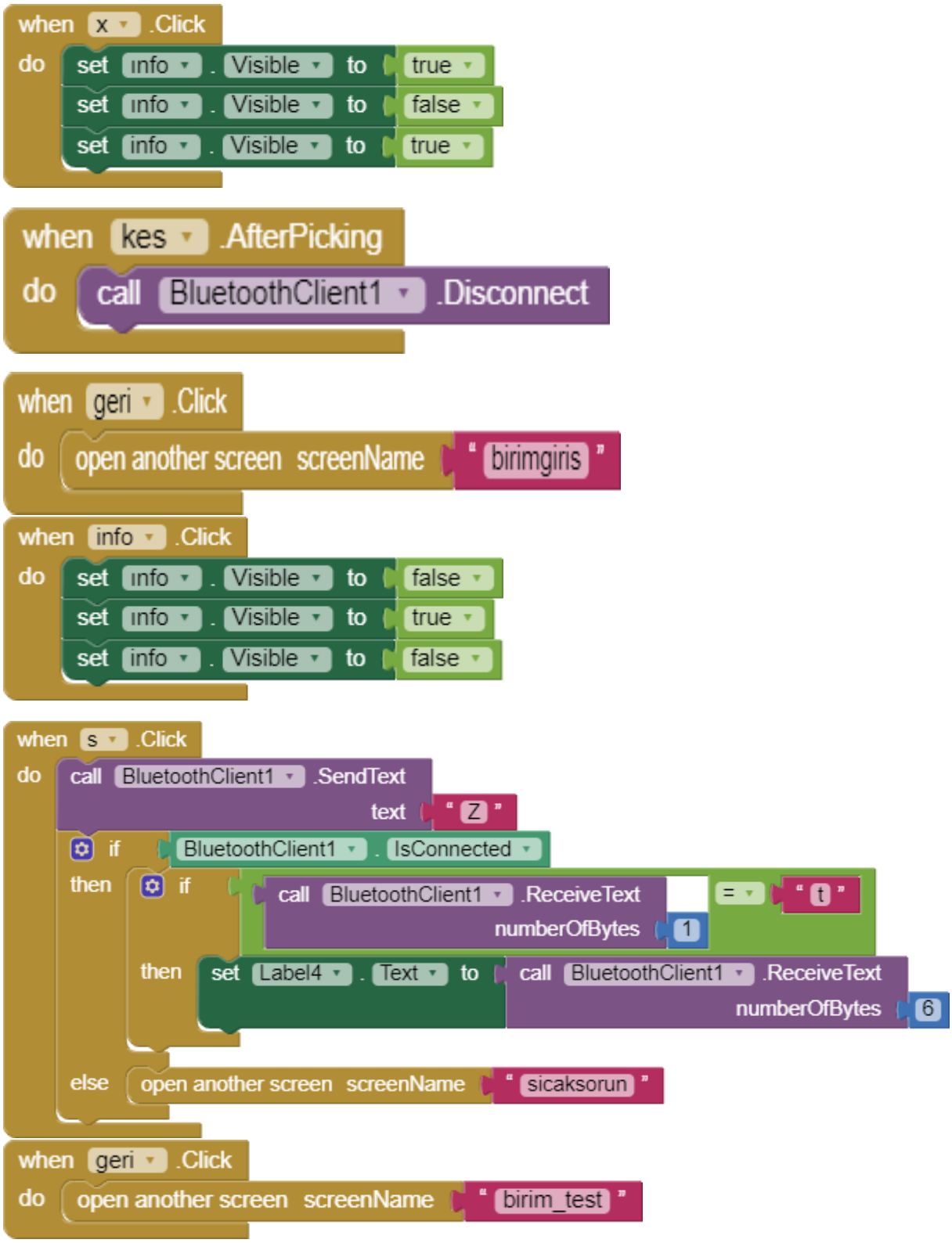
digitalWrite fonksiyonu sayesinde buzzerpin pini logıc 0 yapılır. **Delay(ton)** fonksiyonu sayesinde buzzer ses çıkartır.

3. Birim Testi

Yazılım programlamasında bir tasarım ve geliştirme yöntemidir.

3.1. Mobil Uygulama Test Kodları ve Açıklamaları




when x .Click
do set info . Visible to true
set info . Visible to false
set info . Visible to true
when kes .AfterPicking
do call BluetoothClient1 .Disconnect
when geri .Click
do open another screen screenName " birimgiris "
when info .Click
do set info . Visible to false
set info . Visible to true
set info . Visible to false
when s .Click
do call BluetoothClient1 .SendText
text " Z "
if BluetoothClient1 . IsConnected
then if call BluetoothClient1 .ReceiveText
numberOfBytes 1
= " t "
then set Label4 . Text to call BluetoothClient1 .ReceiveText
numberOfBytes 6
else open another screen screenName " sicaksorun "
when geri .Click
do open another screen screenName " birim_test "

Uygulama bu aşamada asıl uygulama hariç sadece birim testi için tasarlanmış 3 ayrı sayfadan oluşmaktadır. Sayfa düzeni için dikey sıralamada **VerticalArrangement**(Birbirinin altında görüntülenmesi gereken bileşenlerin yerleştirileceği bir biçimlendirme öğesi.), yatay sıralamada ise **HorizontalArrangement**(Soldan sağa görüntülenmesi gereken bileşenlerin yerleştirileceği bir biçimlendirme öğesi.) kullanılmıştır. Bu öğelerin içerisine gerekli Button, TextBox ve Label bileşenleri yerleştirilmiştir.

İlk sayfada 1 adet buton yer almaktadır. Butonlara tıklanarak '**open another screen'** komutu ile butona atanan sayfaya yönlendirme yapılır. Açılan sayfada kullanıcı, kullanıcı adını ve şifresini girer. Girilen bilgiler giriş kontrolüne bağlı olan **matematik** ve **lojik** yapılar sayesinde kayıtlı kullanıcı bilgileri ile karşılaştırılır. Bilgiler doğru ise giriş butonuna atanan sayfaya geçiş yapılır, bilgiler yanlış ise **'Kullanıcı adı veya şifre hatalı.'** uyarısı verip **'Tekrar dene'** seçeneği sunar. Bu hata mesajı **Notifier**(uyarı iletişim kutularını, iletileri ve geçici uyarıları görüntüler) öğesi ile sağlanır.

Testin ilk aşamasında **BluetoothClient.Disconnect**(bağlantı kesme) işlemi gerçekleştirilir ve **ListPicker** (kullanıcının aralarından seçim yapabileceği metinlerin bir listesini görüntüleyen bir buton)ile listelenir. Geri bağlanmak için **BluetoothClient.AddressesAndNames(**bağlantı) yapılır ve şifre girilmesi beklenir**.** Burada **if-then-else** komutu kullanılır şifre doğruluğuna göre yönlendirme yapılır. Label'ler yardımı ile mesaj verilir fakat butona basıldığında aktif hale gelir. Onun dışında görünmezler. Ayrıca yardımcı **info** sayesinde bağlantının nasıl olacağı hakkında bilgi verilir. Kullanıcı ev sıcaklığını görüntülemek için gerekli butona tıkladığında bağlantı olup olmadığı iç **içe if-then-else** ve **if-then** komutları ile kontrol yapılır. Eğer bağlantı var ise evin sıcaklık değerini gerekli LCD ekranda görüntüler, bağlantı yok ise **'bağlantı sorunu'** hatta mesajını verir. Bu aşamaya kadar testlerimiz bu şekilde yapılmıştır. İlerleyen zamanlarda projenin test aşamaları detaylandırılacaktır.

3.2. Arduino Test Kodları ve Açıklamaları

```
#include <sys/timeb.h>
#include "ArduinoTest2_arduino.h"

timeb t_start;
unsigned long millis() {
  timeb t_now;
  ftime(&t_now);
  return (t_now.time  - t_start.time) * 1000 + (t_now.millitm - t_start.millitm);
}

void delay(unsigned long ms) {
  unsigned long start = millis();
  while(millis() - start < ms){}
}

void initialize_ArduinoTest2_arduino() {
  ftime(&t_start);
}
```

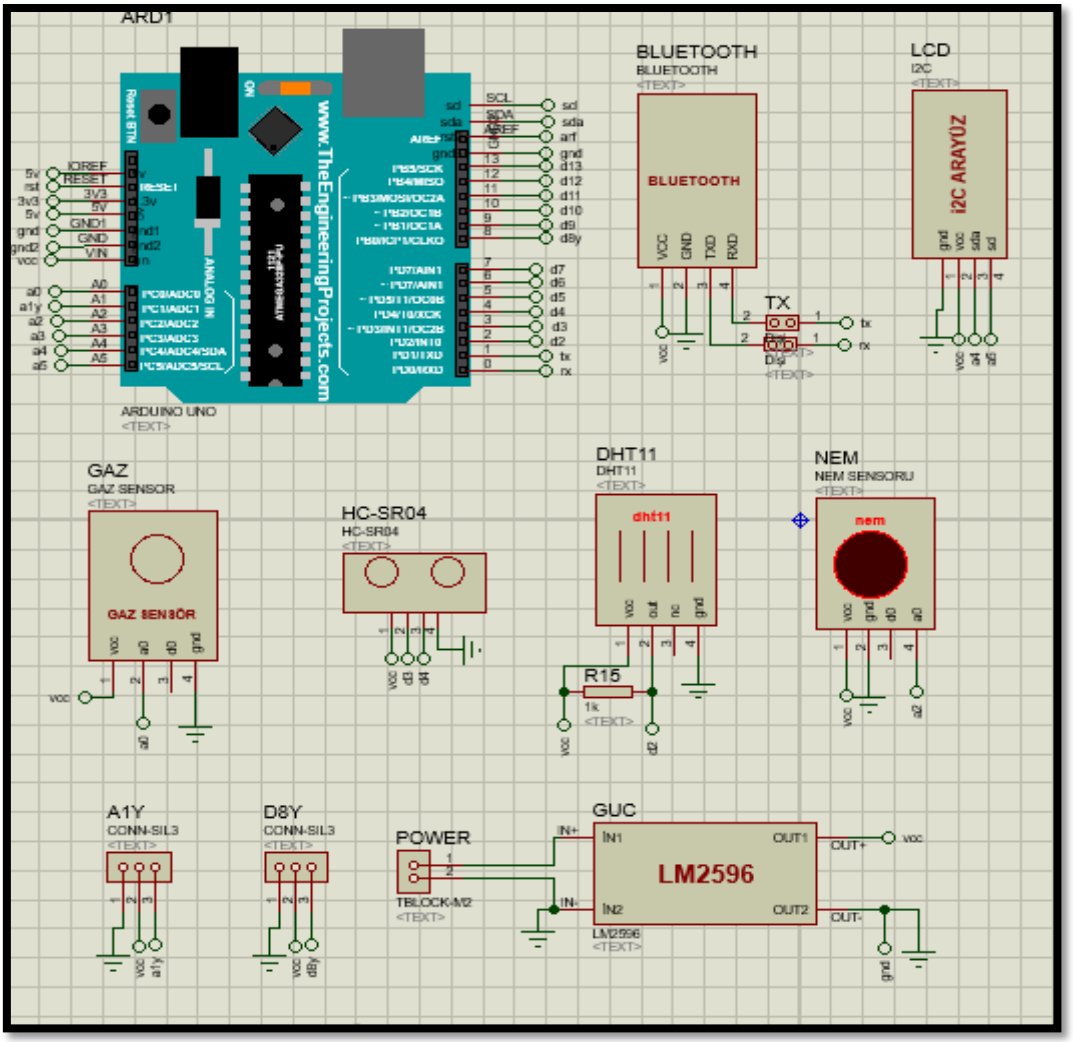
```

#pragma once

#define constrain(amt,low,high) ((amt)<(low)?(low):((amt)>(high)?(high):(amt)))
#define lowByte(w) ((unsigned char) ((w) & 0xff))
#define highByte(w) ((unsigned char) ((w) >> 8))

typedef unsigned char byte;
typedef unsigned short int word;

unsigned long millis();
void delay(unsigned long ms);
unsigned long millis();


long map(long, long, long, long, long);

void initialize_ArduinoTest2_arduino();

#include "ArduinoTest2_arduino_serial.h"
```

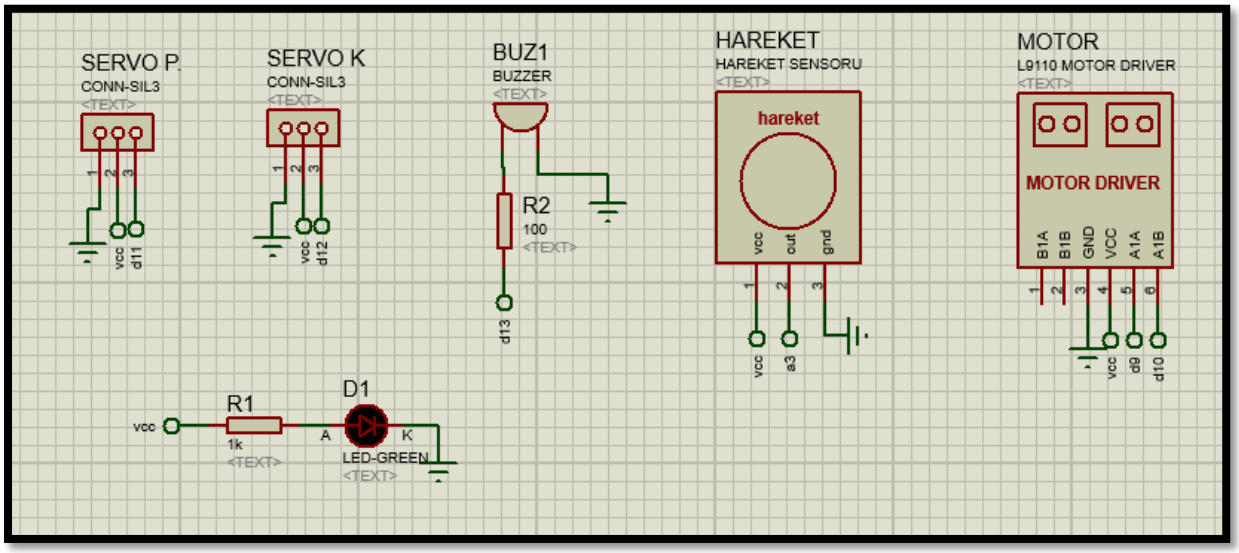
```cpp
#include <cstring>
#include <iostream>
#include <iomanip>
#include "ArduinoTest_serial.h"

void ArduinoTestSerial::begin(unsigned long speed) {
  return;   }
void ArduinoTestSerial::end() {
  return;
}
size_t ArduinoTestSerial::write( const unsigned char buf[], size_t size ) {
  using namespace std;
  ios_base::fmtflags oldFlags = cout.flags();
  streamsize oldPrec = cout.precision();
  char oldFill = cout.fill();

  cout << "Serial::write: ";
  cout << internal << setfill('0');

  for( unsigned int i = 0; i < size; i++ ){
    cout << setw(2) << hex << (unsigned int)buf[i] << " ";
  }
  cout << endl;

  cout.flags(oldFlags);
  cout.precision(oldPrec);
  cout.fill(oldFill);
  return size;
}
ArduinoTestSerial Serial;
```

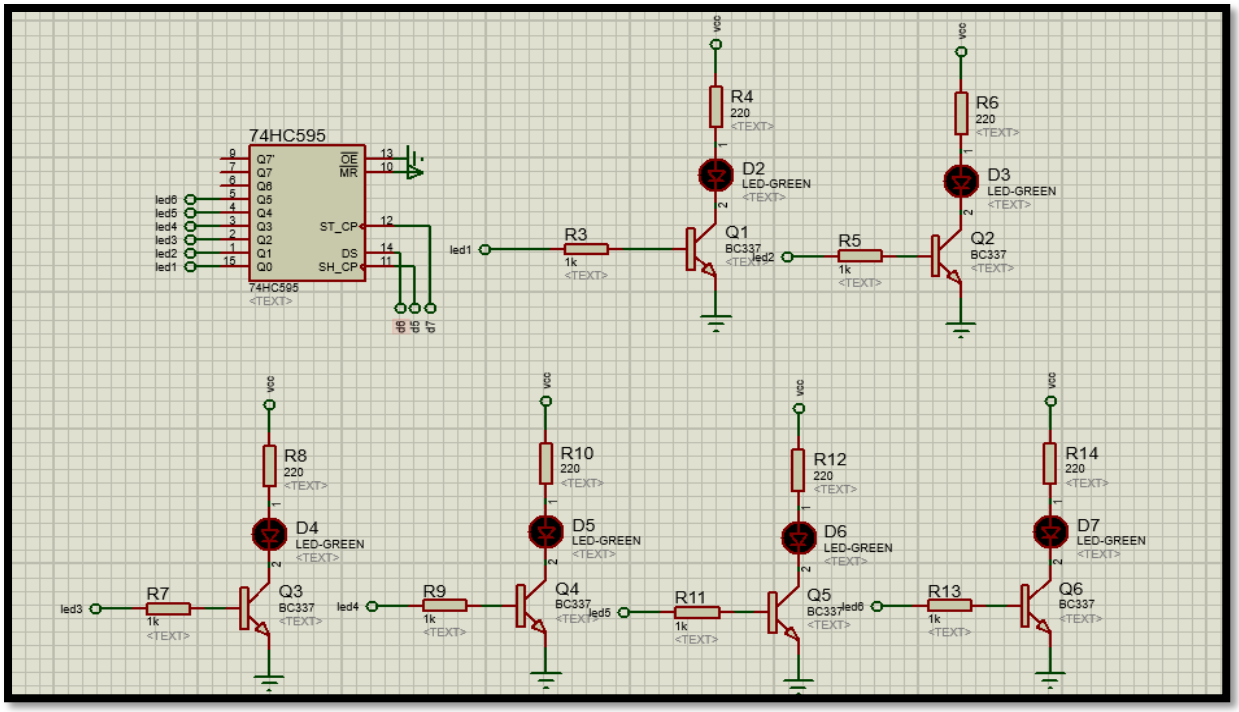
```cpp

#pragma once

#include <iostream>

class ArduinoTest {
public:
  void begin(unsigned long);
  void end();
  size_t write(const unsigned char*, size_t);
};

extern ArduinoTest Serial;
```

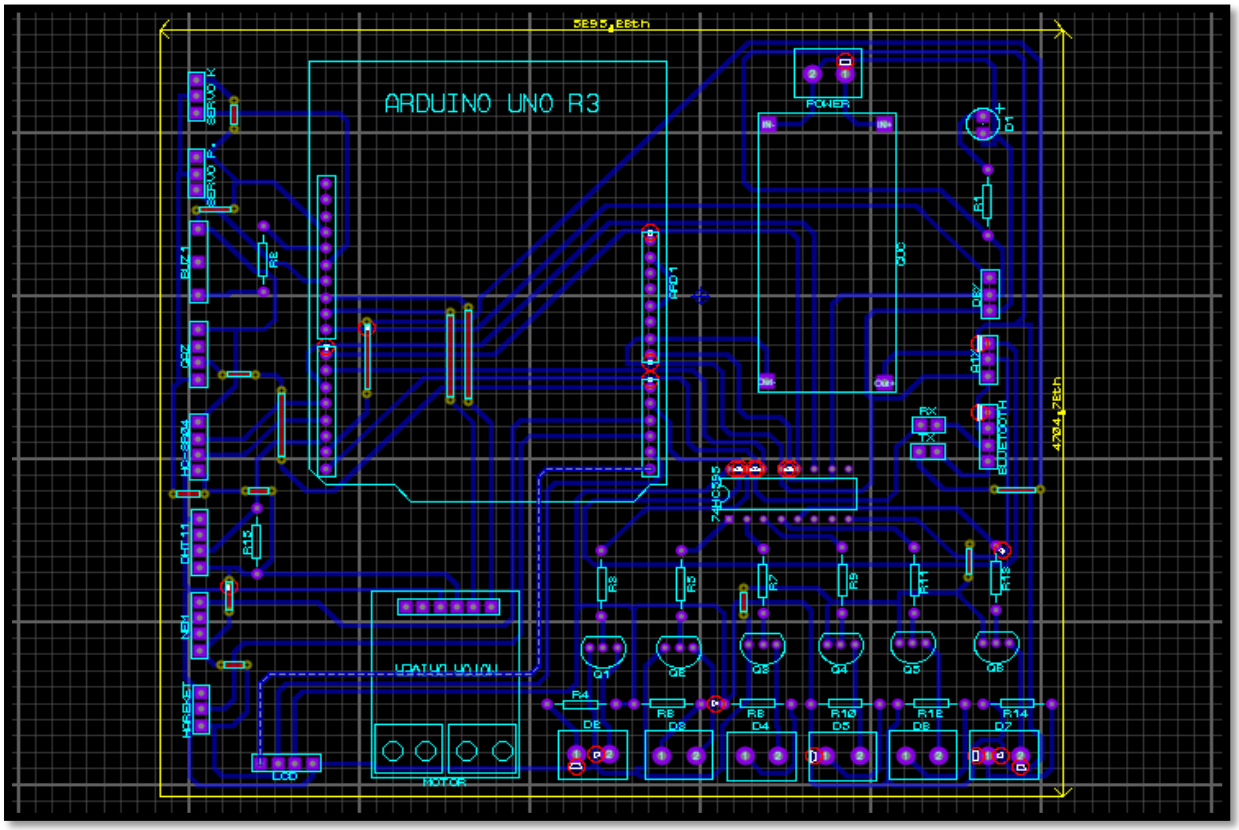
```
#include "ArduinoTest2_arduino.h"
#include "dsm2_tx.h"

using namespace std;

void millis_test() {
  unsigned long start = millis();
  cout << "millis() test start: " << start << endl;
  while( millis() - start < 10000 ) {
    cout << millis() << endl;
    sleep(1);
  }
  unsigned long end = millis();
  cout << " Sonuç - Süre: " << end - start << "ms" << endl;
}

void delay_test() {
  unsigned long start = millis();
  cout << "delay() test start: " << start << endl;
  while( millis() - start < 10000 ) {
    cout << millis() << endl;
    delay(250);
  }
  unsigned long end = millis();
  cout << "Sonuç - Süre: " << end - start << "ms" << endl;
}
```

```
  void Run_Test();
  int main(int argc, char **argv){
    initialize_mock_arduino();
    Run_Test();
  }

  void Run_Test() {
    DSM2_tx tx(6);
    tx.bind();
    for( int i = 0; i < 6; i++) {
      tx.set_channel(i, 0);
    }
    tx.send_frame();
  }
```


ARD1
ARDUINO UNO
www.TheEngineeringProjects.com
BLUETOOTH
BLUETOOTH
LCD
I2C
i2C ARAYÜZ
TX
GAZ
GAZ SENSOR
GAZ SENSÖR
HC-SR04
DHT11
dht11
NEM
NEM SENSORU
nem
R15
1k
A1Y
CONN-SIL3
D8Y
POWER
TBLOCK/M2
GUC
LM2596


SERVO P.
CONN-SIL3
SERVO K
CONN-SIL3
BUZ1
BUZZER
R2
100
HAREKET
HAREKET SENSORU
hareket
MOTOR
L9110 MOTOR DRIVER
MOTOR DRIVER
R1
1k
D1
LED-GREEN





Sistemin test aşaması için Proteus programında gerekli devre çizimi yapıldı. Bağlantı kontrolleri gerçekleşti. Devrede girdi ve çıktı kontrolü sağlandı. Gerekli devre birimlerinde doğru voltaj değerinin varlığı devrenin başarı ile sonuç verdiğini kanıtlar.

Sistem kodunda test edilmesi gereken kısımlar test adında yeni bir proje açılarak gerekli kodlar yazıldı ve ardından programı ayrı parçalara ayırarak test için izole edildi.

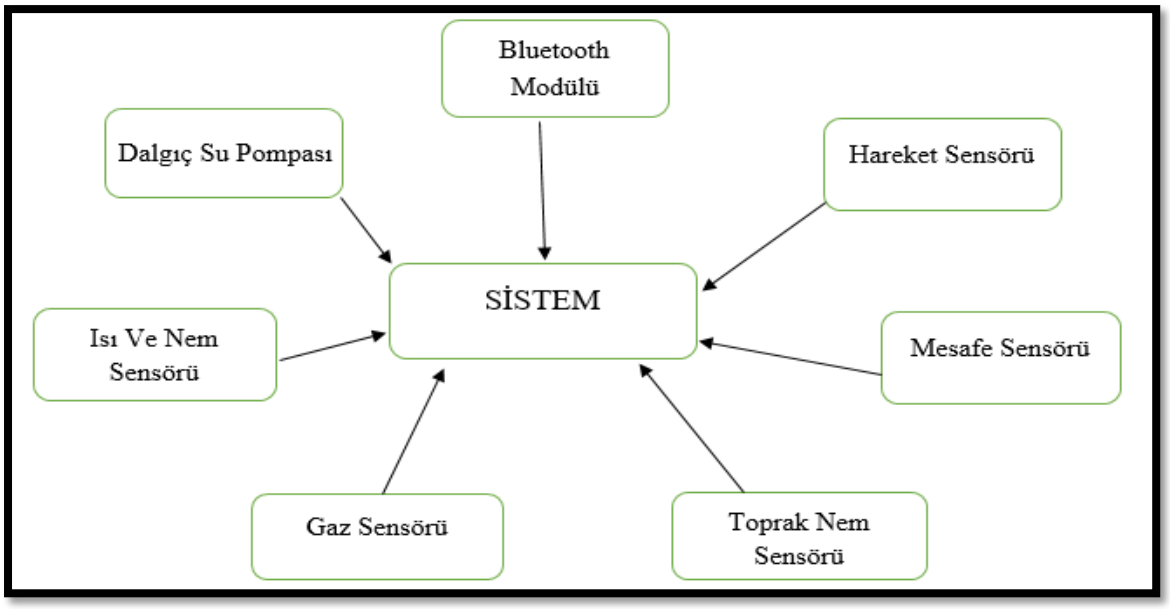
Arduino kütüphanesi tarafından sağlanan bazı destek işlevleri ArduinoTest2_arduino.cpp dosyasında mevcuttur. Kodun donanımı seri aygıtına ikili veriler yazdığında okunabilir çıktı üretmek için Arduino_Test.h yazıldı. C++ derleyicisinde derlene kodların çıktısı olan sonuç ve süre değerleri istenilen ile orantılı olması durumu testin başarılı olduğunu kanıtlar.

4. Entegrasyon Testi

Oluşturulan yazılım modüllerinin, bir araya getirerek doğruluğunu sağlamaktır. Yazılım ürünü için oluşturulan tüm modüller bir araya getirilir ve bu şekilde test edilir. Burada ki amaç: metotlar birim başına testten geçerken, modüller halinde bir araya geldiğinde bazı hatalara sebep oluyor olabilirler. Entegrasyon testleri ile ise bu tarz yazılım ürünü problemlerinin henüz canlı (prod) ortama çıkmadan veya geliştirdiğimiz yeni bir modülün de sorunsuz çalışabileceğinden hızlı bir şekilde emin olabilmemizi sağlamaktadır.

Entegrasyon birleştirme demektir. Entegrasyon testi bir yazılımın bileşenlerinin birbirine entegre edilmesi sırasında yapılabileceği gibi iki farklı yazılımın birbirine entegre edilmesi sırasında da yapılabilir. Bu yüzden entegrasyon testi, birim entegrasyon testi ve sistem entegrasyon testi olarak farklı test seviyelerinde yapılabilir. Yazılım geliştirme uzmanlarının birim test sırasında ayrı ayrı test ettikleri bileşenler birbirine entegre edildikleri zaman hataya sebep olabilirler. Entegrasyon testi, sistemin bu farklı bileşenlerinin(birimlerinin) birlikte doğru çalışıp çalışmadıklarını test etmeyi amaçlar.

Bu testler ile kullanıcının yaşaması muhtemel olası sorunların ortadan kalkması hedeflenir. Daha önceden testler sonucu sorunlar belirlenir ve çözümlenir. Bu sayede yaşanabilecek aksaklıklar en aza indirgenmiş olur. Kullanıcının tamamen kolay ve kullanışlı bir tasarım görmesini sağlamış oluruz. Ayrıca modüllerinde birbiri ile uyumu test edilir. Gerekli kod çakışmaları belirlenip çözümlenir. Bu modül ve sensörler tek tek rahat şekilde çalışır gibi bir arada sorunsuz çalışır hale getirilir. Bu testlerin uygulanıp eksiklerin giderilmesinin ardından yapılan işlemler kullanıcı karşına çıkmaya hazır hale gelmiş olur.



Gerekli sensör ve modülleri tek sistemde birleştirip sorunsuz bir şekilde çalışması için çalışmalarımız devam etmektedir. Her bir modül tek sistem üzerinde bir birinden bağımsız şekilde çalışmaktadır. Bu çalışma kullanılan sensörler ve mobil uygulama ile tetiklenip çalışmaktadır.

Sisteme gerekli sensörler teker teker aşamalı bir şekilde dahil edilip her sensörden sonra işlemin başarılı olup olmadığı test edilmiştir. Her adımda bir başka sensör eklenerek hepsinin çalışma durumu kontrol edilip onaylanmıştır.

No	Test	Fonksiyon	Prosedür	Sonuç
1	Bluetooth bağlantısı.	Mobil uygulama ile Arduino sistemi haberleşmesi.	Bağlantı adresi seçilir.	Bağlantı kuruldu. Başarılı.
2	Aydınlatma kontrolü.	Mobil uygulama kontrolünde aydınlatma açıp kapatma.	Aydınlatma butonlarına basılır.	Aydınlatma açıp kapatıldı. Başarılı.
3	Servo motor.	Kapı-perde açıp kapatma.	Kapı-perde butonlarına basılır.	Kapı-perde açıp kapatıldı. Başarılı.
4	DHT11 Isı-sıcaklık sensörü.	Isı-sıcaklık ölçümü.	Sıcaklık butonuna basılır.	Sıcaklık ölçülüp ekrana yansıtıldı. Başarılı.
5	Doğal gaz metan gazı sensörü.	Ortamdaki yabancı gaz değeri ölçümü.	Gaz değeri butonuna basılır.	Gaz değeri ölçülüp ekrana yansıtıldı. Başarılı.
6	HC-SR04 Mesafe sensörü	Cisimler arasındaki mesafe değeri ölçümü.	Mesafe değeri butonuna basılır.	Mesafe değeri ölçülüp ekrana yansıtılır. Başarılı.
7	Toprak-nem sensörü.	Toprağın nem değeri ölçümü.	Toprağın nemsiz olması durumunda sulama yapılır. Nem değeri butonuna basılır.	Nem değeri ölçülüp ekrana yansıtılır. Başarılı.
8	PIR hareket sensörü	Hareketi algılaması durumunda alarm sistemini tetikler.	Kapı zorlanarak açıldığında alarm sistemi devreye girer.	Alarm çalar. Başarılı.

5. Veri Toplama

Sensör kullanılarak yapılan uygulama veya deneysel çalışmalarda alınan ölçümlerin mobile aktarılması için veri işleme kartlarına ihtiyaç duyulmaktadır. Bu tür çalışmalarda kullanım kolaylığı açısından en çok tercih edilen platformlardan birisi de Arduino'dur. Açık kaynaklı bir geliştirme platformu olan Arduino'nun dijital ve analog girişleri sayesinde veriler okunup anlık olarak işlenebilmektedir. Ev otomasyon sistemi çalışmasında, mobil ortamda App Inventor kullanarak geliştirilen uygulama sayesinde Arduino'ya bağlanan farklı sensörler içerisinden istenilen sensörler seçilebilmektedir. Seçilen bu sensörleri kullanmak için gerekli Arduino kodu oluşturulabilmektedir. Arduino üzerindeki sensörlerden gelen veriler belirli zaman aralığında veya anlık olarak kaydedilebilmektedir. Kaydedilen veriler, istenilen periyotlarda ve özelliklerde elde edilebilmekte ve kullanılabilmektedir.

Sensörler, fiziksel veya kimyasal büyüklükleri elektriksel büyüklüklere çevirerek kullanılabilir formata dönüştürmektedirler. Sensörler kablosuz iletişim, sinyal algılama, alınan sinyali işleme ve yayma gibi yeteneklere sahiptir. Sensörler kullanılarak yapılan uygulama veya deneysel çalışmalarda alınan ölçümlerin bilgisayara aktarılması için veri toplama kartlarına ihtiyaç duyulmaktadır.

5.1. Verilerin Toplanması ve Kontrol Edilmesi

Hareket Sensörü

Projemizde hareket sensörü kullanarak herhangi bir harekette sensör durumu algılayacak. Sensör sistemi uyararak alarm devreye girecektir. Bu sensör oldukça kullanışlıdır. Artık evler bu sensör sayesinde daha güvenlidir.

Genel olarak akşam-gece saatlerinde hırsızlık olaylarının daha yoğun olduğu zamanlarda sistem tarafından veriler alınmakta olup daha çok güvenliği ön planda tutmak amaçlanmaktadır.

Işıklar

Evdeki ışıklar, kullanıcı isteği ile etkinleşir. Kullanıcı evden uzakta olsa bile ışıklara müdahale edip kapatıp açabilir.

Veriler, gündüz güneşli saatlerde ışıkların kapalı, akşam saatlerinde ışıkların açık olduğu şeklinde alınmaktadır. Aynı zamanda evden uzaktayken veya tatildeyken evin ışıklarını kontrol edip, hırsız girmemesi için akşam saatlerinde evde biri varmış hissi uyandırmak için kullanılır. Sabahleyin tekrardan ışıklar kapatılır.

Alarm

Zorlama ile eve giriş sağlandığı anda etkinleşir. Hareket sensörüyle birlikte çalışır. Genel olarak hırsızlık olaylarının yoğun olduğu zamanlarda çalışır. Eğer sistem bir hareket algılarsa buzzer devreye girer ve alarm çalar.

Mesafe Sensörü

Park işlemlerini kolaylaştırır. Burada minimum mesafe 10cm olarak ayarladığı zaman arabanın kolay bir şekilde park edilmesini sağlamaktadır.

Veriler genellikle, akşam iş dönüşünde alınmaktadır. Çünkü genel olarak akşam arabanın park edilmesi üzerine veriler akşam saatlerinde çok alınır.

Sulama

Toprağın kuruduğunu sistem anladığı zaman su vermektedir.

Veriler toprağın kuruluğuna bakıp alınmaktadır. Eğer topraktaki su miktarı az ise sistem tarafından toprağa su verilmektedir. Genel olarak tatil modu durumunda iken evdeki çiçekler sulanmaktadır.

Perde Sistemi

Bu sistem ev ortamında oldukça işe yarayan bir sistemdir. Kullanıcı, perdeyi mobil uygulama ile tek bir tuşla istediği saatlerde açar veya kapatır. Genel olarak perde sisteminden alınan veriler şu yöndedir; sabah saatlerinde perde açık, akşam saatlerinde perde kapalı durumda olduğu bilgisi alınmaktadır.

Isı-Nem Sensörü

Bu sensör sayesinde çevrenin ısı-nemi algılanır ve ona göre sisteme veri gönderir ve veri tabanında güncellenmiş olur.