Uygulamanın Tanımı ve Kullanım Alanı

Çalışmanın amacı RFID teknolojisi kullanılarak bir noktadan sadece yetkilendirilmiş kişilerin geçişinin sağlanmasıdır.

Çalışmanın 3 temel bölümü bulunmaktadır;

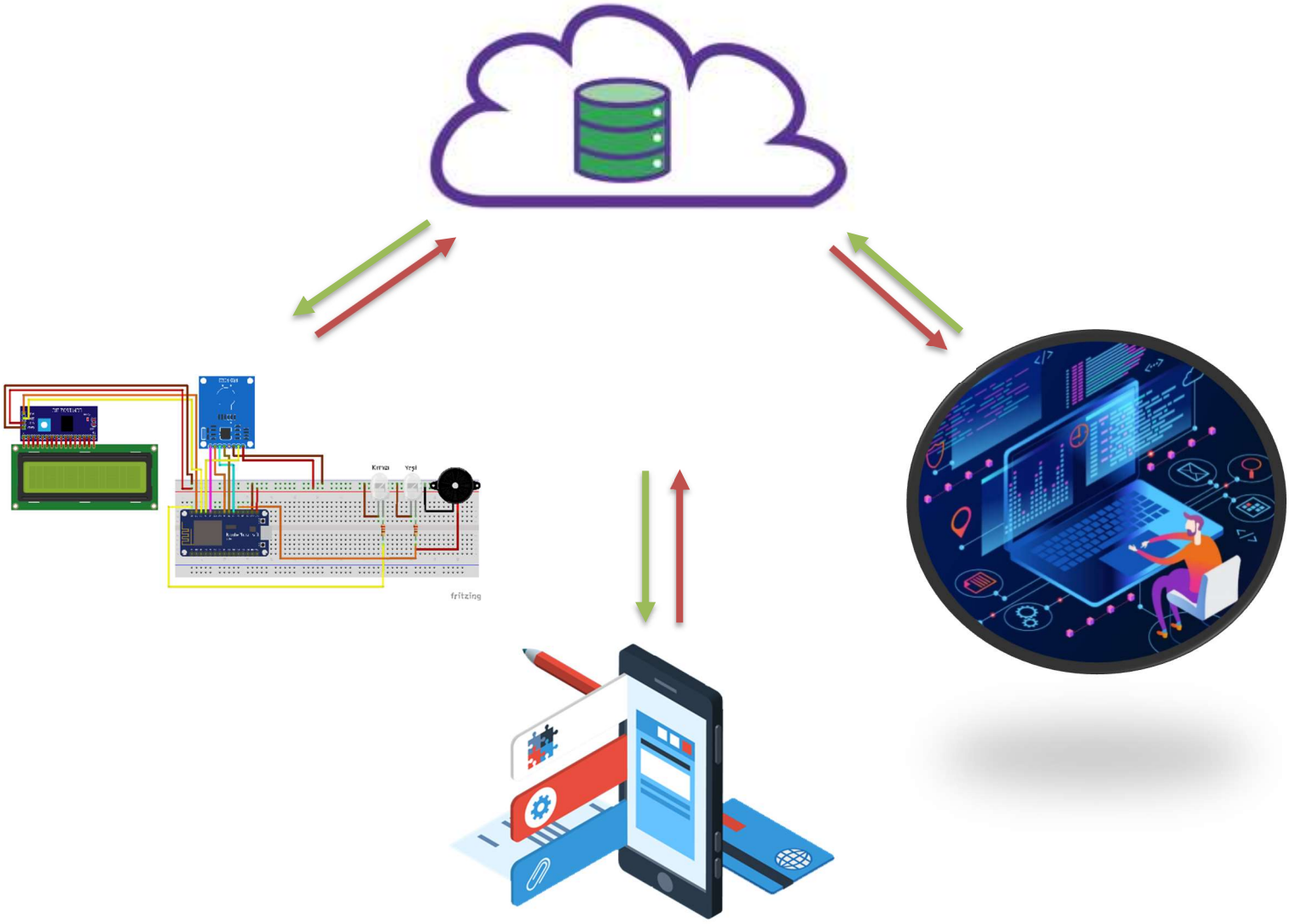
- **Fiziki Bölüm :** NodeMCU, RFID, Led ve Buzzer (Led kapı açılışını temsil etmektedir) elemanlarının olduğu bölümdür. IOT cihazın yazılımında Arduino IDE kullanılmıştır.

- **Web Uygulaması :** API kodlarının üretildiği ve NodeMCU'dan gelen Kart ID'lerine göre yetkilendirme işlemlerinin yapıldığı web arayüzüdür.
  Web arayüzü için Visual Studio ASP.Net MVC 4.0 ile birlikte gelen **Web Api** özelliği kullanılmış olup **REST** haberleşme methodu ile haberleşme sağlanmıştır. Veri Tabanı olarak da **MSSQL** kullanılmıştır.

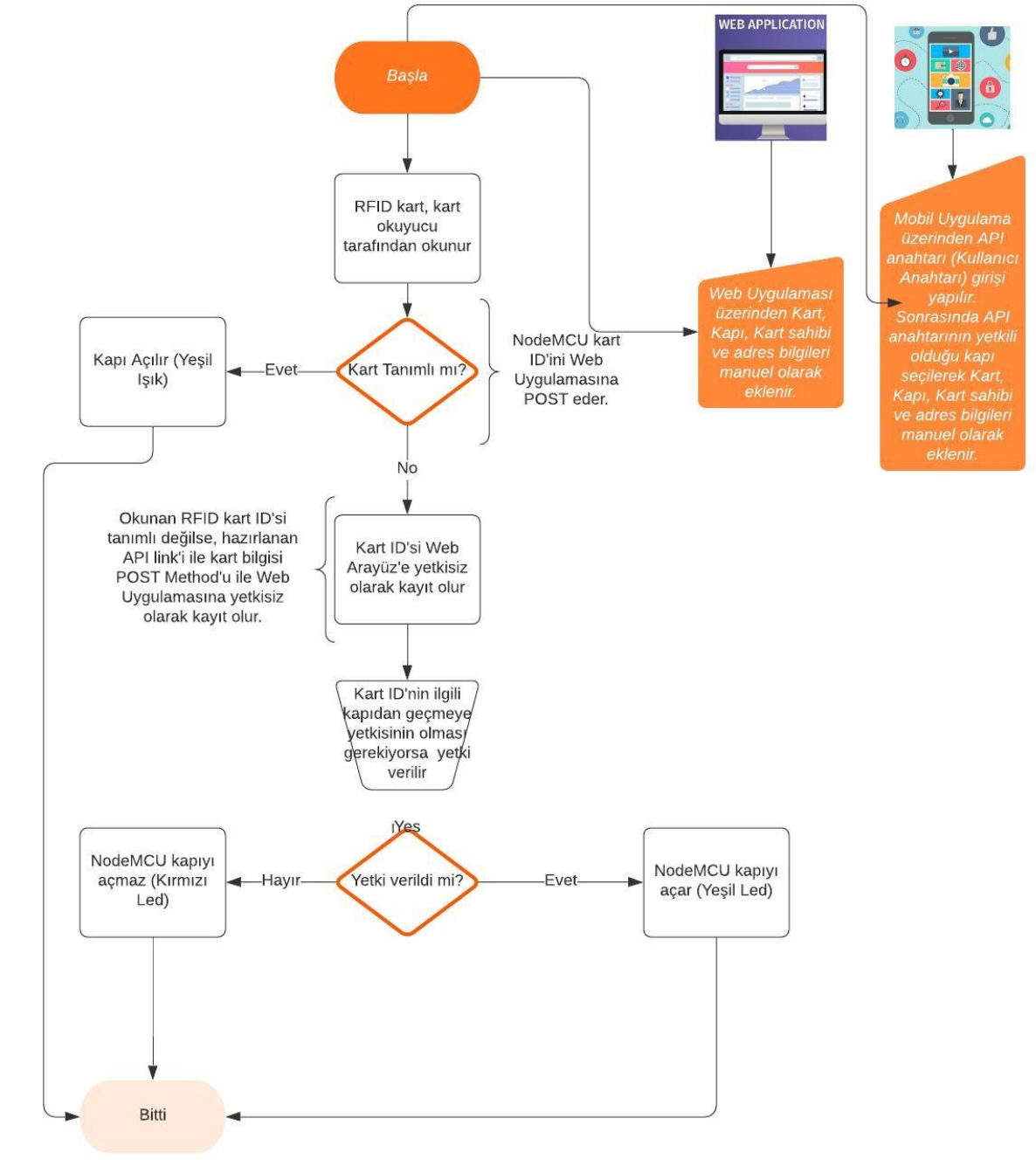
- **Mobil Uygulama :** Web uygulamasında üretilen API kodunun girilmesi sayesinde GET methodu ile yetki alınması ve yetkiye bağlı olarak Kart bilgilerinin web arayüzüne POST edilmesi işlemleri yapılmaktadır.

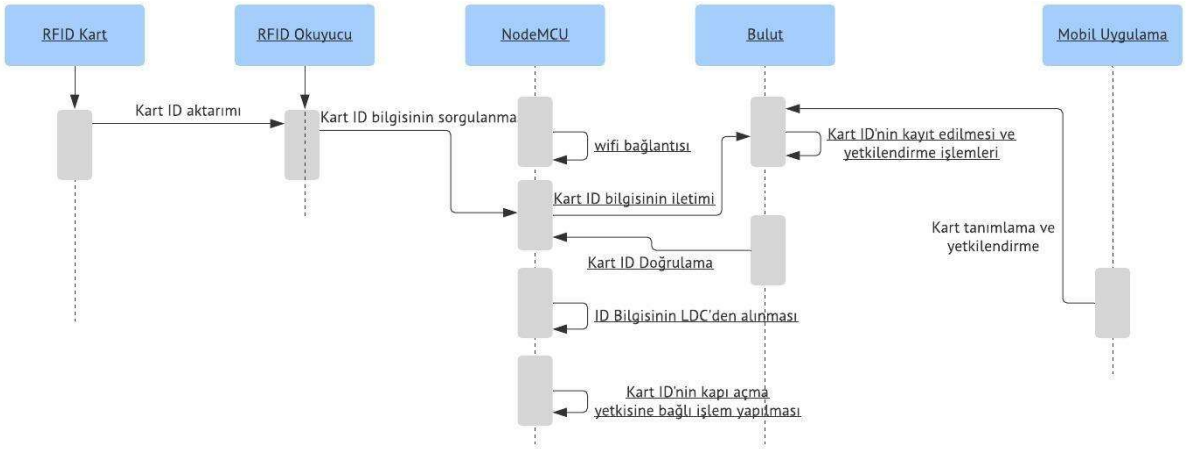
Çalışmada IOT nesnesi olarak **NodeMCU** seçilmiştir. NodeMCU'ya bağlı RFID okuyucudan gelen Kart ID bilgisi Web Uygulamasına POST edilmekte ve yetki sorgulanmaktadır. Yetki yoksa Web yada Mobil Uygulama üzerinden yetki tanımlaması yapılabilmektedir.

Sistem Mimarisi



Akış ve Dizge Diyagramı





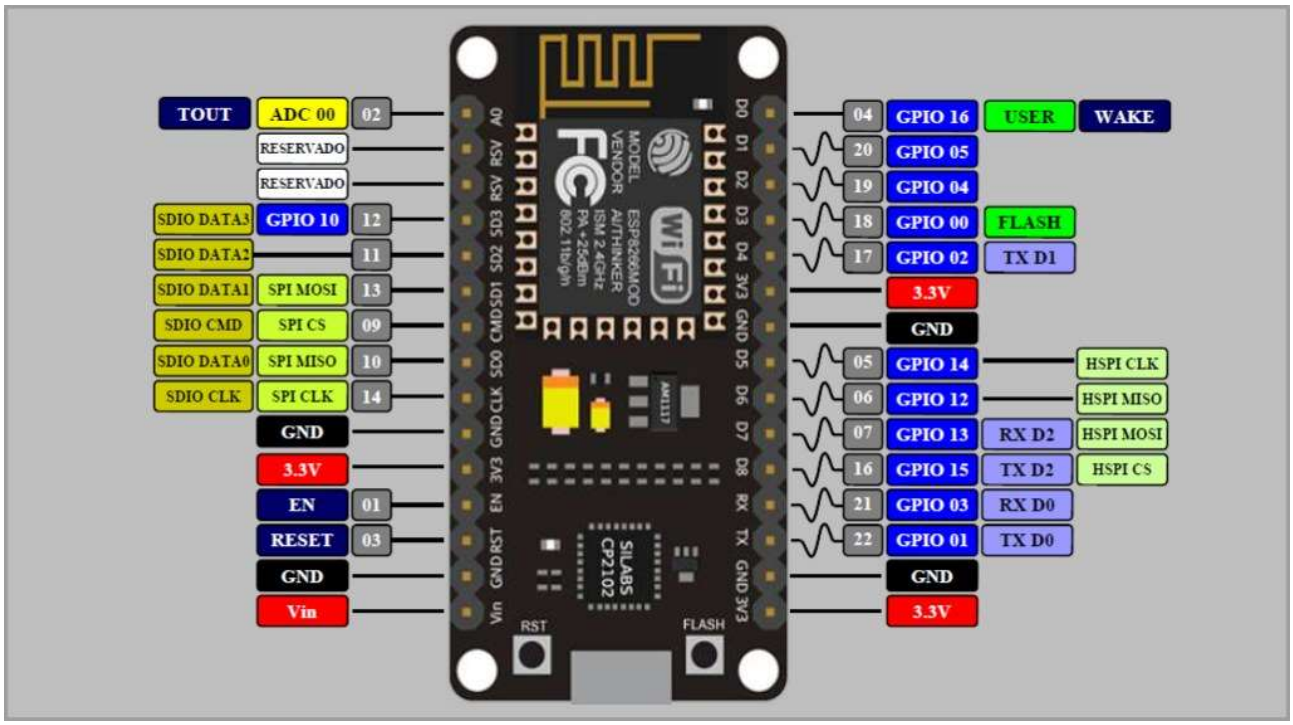
NodeMCU

NodeMCU; madeni paradan biraz daha büyük boyutta, minik bir elektronik devredir. Açık kaynaktır, ucuzdur ve yeteneklidir. İçinde ESP8266 bulunduran bir devre kartıdır.

Düşük gerilimli enerjiyle çalışır. Üzerinde çok sayıda bağlantı noktaları vardır. Bu bağlantı noktalarını kullanarak bağlayacağınız başka elektronik bileşenleri yönetebilirsiniz. Barındırdığı WiFi sayesinde kolayca IOT yani internet şeyleri olarak bilinen cihazlar yapmanıza olanak sağlar.

HTTP kütüphaneleri sayesinde web istemleri yapabilirsiniz veya web sunucusu çalıştırabilirsiniz. Bu sayede internet üzerinden bu cihazla iletişime geçebilirsiniz. Uzaktan bir şeyleri açabilir veya kapatabilirsiniz.

Ayrıca sürekli etkileşim halindedir. Programlanabilirdir, düşük maliyetli ve basittir. Ayrıca akıllı bir cihazdır ve WI-FI bağlantıya yani kablosuz bağlantıya ve kullanıma hazır gelir.



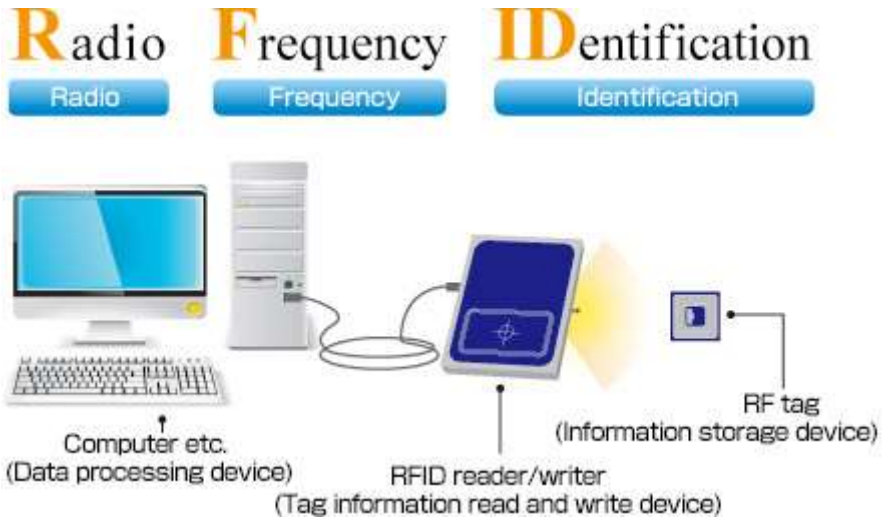
RFID

RFID nasıl çalışır?

RFID, **Otomatik Tanımlama ve Veri Yakalama** (AIDC) olarak adlandırılan bir teknoloji grubuna aittir. **AIDC yöntemleri** nesneleri otomatik olarak tanımlar, bunlar hakkında veri toplar ve bu verileri doğrudan hiçbir müdahale olmadan bilgisayar sistemlerine girer. RFID yöntemleri, bunu gerçekleştirmek için radyo dalgalarını kullanır. Basit bir düzeyde, RFID sistemleri üç bileşenden oluşur: bir RFID etiketi veya akıllı etiket, bir RFID okuyucu ve bir anten. RFID etiketleri, verileri RFID okuyucusuna (sorgulayıcı olarak da adlandırılır) iletmek için kullanılan bir entegre devre ve bir anten içerir. Okuyucu daha sonra radyo dalgalarını çok daha kullanışlı bir veri biçimine dönüştürür. Etiketlerden toplanan bilgiler daha sonra bir iletişim ara yüzü aracılığıyla veri bilgisayarında depolanabilen ve daha sonra analiz edilebilen bir ana bilgisayar sistemine aktarılır.

RFID etiketleri ve akıllı etiketler (smart labels)

RFID etiketi bir entegre devre ve bir **anten** içerir. Etiket, parçaları bir arada tutan ve çeşitli çevre koşullarından koruyan koruyucu bir malzemeden oluşur aynı zamanda. Koruyucu malzeme uygulamaya bağlıdır. Örneğin, RFID etiketleri içeren kimlik kartları tipik olarak dayanıklı plastikten yapılır ve etiket plastik katmanların arasına gömülür. RFID etiketleri çeşitli şekil ve boyutlarda gelir ve pasif veya aktif olabilir. Pasif etiketler genelde kullanılır çünkü bunlar daha küçük ve uygulanması daha az maliyetlidir. Pasif etiketler, verileri aktarabilmeleri için RFID okuyucu tarafından "güçlendirilmelidir". Pasif etiketlerden farklı olarak, aktif RFID etiketlerinde yerleşik bir **güç kaynağı** (örneğin bir batarya) vardır; böylece her zaman veri iletmelerini sağlar.

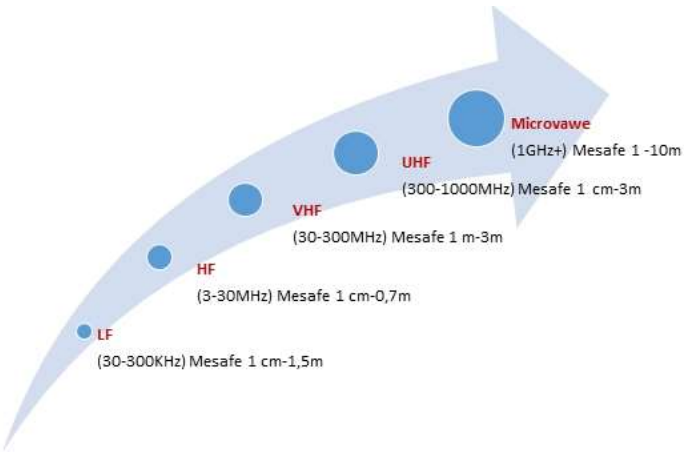


RFID Sistem bileşenleri nelerdir?

RFID sistemi, RFID etiketi ve RFID okuyucusundan oluşur. Bunların yanı sıra **RFID yazıcısı**, **RFID anteni**, sistemin kullanılacağı kontrol ve uygulama yazılımı ve sistemin kayıt edileceği alanlar da eklenebilir bu bileşenlere. RFID etiketi çip, anten ve bazen de pilden oluşur. **RFID okuyucu** ise antenlerden oluşur. RFID etiketi okuyucudan sinyal alır ve okuyucuya sinyal gönderir. Etiket, unique bir numara içerir yani benzeri yoktur. Bu, RFID okuyucu tarafından okunur. Etiketler, kullanım alanlarına göre çok farklı tasarımlarda olabilir.

	Aktif	Pasif
Güç kaynağı	Etiket içinde (Bakım gerektirir)	Radio dalgaları ile güç alır
Operasyon Sıcaklığı	Kısıtlı	1 – 3m.
Mesafe	Uzun	1 – 3m.
Bellek Kapasitesi	Büyük	1cm. – 0,7m.
Maliyet	10-100	

Aktif-Pasif RFID



RFID Etiket Frekans Aralıkları

API (Application Programming Interface)

Türkçe karşılığı "uygulama programlama arayüzü"'dür. Api üzerinde çalıştığımız uygulama ile farklı bir uygulama arasında iletişime geçmesini sağlayan yapı diyebiliriz. Başka bir değişle API, bir uygulamanın sahip olduğu işlevlerin ve yeteneklerin başka bir uygulama tarafından kullanılabilmesine olanak sağlayan bir yapıdır.

API tipik olarak Hypertext Transfer Protocol (HTTP) istek mesajları ile Extensible Markup Language(XML) ya da JavaScript Object Notation (JSON) formatında olan tepki mesajları yapısının bir tanımıdır.

REST mimarisi temelde Proxy'ye ihtiyaç duymaksızın HTTP istekleri ile basitçe gerçekleştirilebilen Get, Post, Put, Delete ve benzeri metotlarını destekler. Bu sayede de nereden ve hangi uygulamadan istekte bulunulursa bulunulsun, özel durumlar dışında Web Api'ye erişilebilecektir. Kısacası HTTP metotları kullanıldığı için bu bir tarayıcı isteği dahi olabilir.

Uygulama Programlama Arayüzü

Basitçe API (Application Programming Interface) kavramı

Yukarıdaki **Api yapısına** bakacak olursak karşı uygulamaya bir istek gönderilir ve karşılığında gönderdiğimiz uygulamaya cevap olarak bir veri alırız bu veriyi dilediğimiz gibi işleyerek geliştirmekte olduğumuz program üzerinde istediğimiz şekilde kullanabiliriz.

Api Avantajları Nelerdir ?

- API her zaman yazılımcının işini kolaylaştırmaktadır. İmkansız gibi görünen tüm proje ve içerikleri iyi bir dilde tasarlanmasını ve uygulanmasını sağlar.
- Proje içerisinde daha fonksiyonel davranma imkanı sağlanır. Çalışmaları gerçekleştirmeden önce hazırlanacak olan bir API projenin gidişatı için çok önemlidir.

- Kullanıcı odaklı ve müşterilerin beğenebileceği kolaylıkları API ile sağlayabilirsiniz.

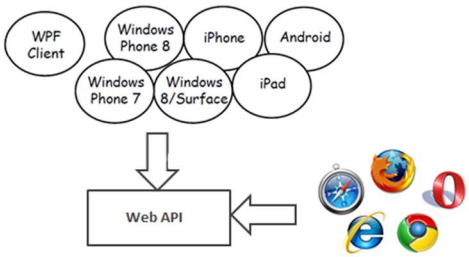
- Entegre edilerek oluşturulan projelerin başarılı olma ihtimali çok daha yüksektir. Bu duruma örnek verecek olursak bir internet sitesinin Facebook API uygulaması sayesinde kişileri tespit edip, yazıyı okuyan kişinin Facebook arkadaşlarına yazınızı önerme çalışması gibi milyonlarca farklı konsept oluşturulabilir.

Asp.Net Web Api

Asp .Net Web Api farklı türde sayısız client (browsers, mobile phones, tablets, pc, etc.) tarafından consume edilebilen HTTP protokolü üzerinden haberleşebilen servisler oluşturmak için kullanılan bir framework şeklinde tanımlayabiliriz. Asp .net MVC ile routing, controllers, action results, filter, model binders gibi ortak feature'lara sahip olduklarından bir takım benzerlikler göstermektedir.

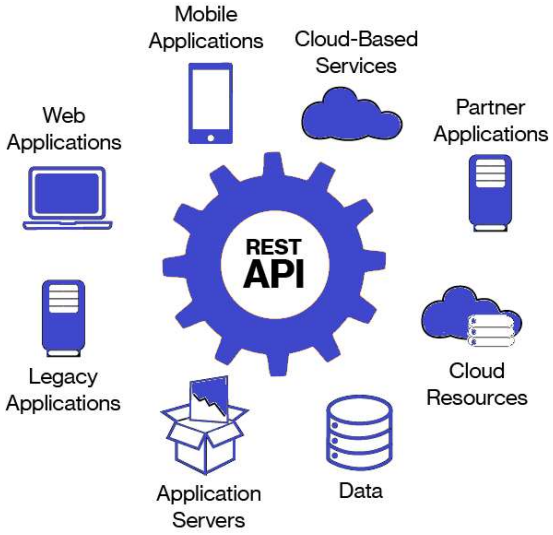
Günümüz dünyasında teknolojini gelişmesiyle birlikte firmalar artık web tabanlı uygulamalar üzerinden müşterilerine tam olarak ulaşamaz hale geldiler. İnsanlar artık günlük hayatlarının nerdeyse %50 sini akıllı telefonlar, tablet pc vs ile geçiriyorlar ve bu cihazlarda insanların hayatlarını kolaylaştıracak olan milyonlarca uygulama mevcut. Bunların yanında birde IOT ile birlikte gelecek 5 yılda dünyada 30 milyara yakın internete bağlanabilen cihazlar olacağından bahsediliyor ve buda belki milyonlarca Api geliştirmesi demek.

Firmalar veya uygulama geliştiriciler müşterilere daha kolay ve hızlı bir şekilde ulaşmada kullanmak için servislerini ve sahip oldukları verilerin bir kısmını browserlar yada internete bağlanabilen bu akıllı cihazlar tarafından consume edilebilmeleri için Api'lar geliştirmeleri gerekmektedir. Çünkü Api'lar yapısı gereği bütün programlama dilleri tarafından ortak kabul görmüş medya tiplerini (XML-JOSN..etc.) response olarak alıp gerekli parse işlemlerinden sonra kolayca kullanabilir.



Web Api sahip olduğunuz veri ve servisleri birçok farklı cihazda kullanıma sunmak için expose edebilmenizi sağlayan şahane bir framework ve dahası Web Api .Net Framework üzerinde RESTful servisler inşa etmenizi sağlayacak ideal bir open source platform. WCF Rest service'lerinin aksine Web Api HTTP protokolünün bütün özelliklerini kullanır (URIs, request/response headers, caching, versioning, çeşitli content format'ları) WCF Rest Service'lerinde yapıldığı gibi farklı cihazlar için extra config ayarları vs yapmamıza da gerek bulunmamaktadır. Request'i yapılırken dönmesi gereken response'un XML mi yoksa JSON formatında mı olacağına client'ın seçimine bırakılmıştır çünkü Web Api birden fazla medya formatında response dönebilmektedir.

Rest (REpresentational State Transfer)



REST client-server iletişimiyle ilgili bir mimari. HTTP protokol'ü ile paralel olarak gelişmiş olmasının yani sıra bugün en çok hepimizin aşina olduğu **World Wide Web** sisteminde kullanılıyor. REST mimarisini kullanan servislere genel olarak RESTful servis deniyor. Ana fikir aslında client-server arasında ki veri alışverişini SOAP, RPC gibi kompleks mimarilerle sağlamak yerine, HTTP protokolü üzerinden sağlamak. Çünkü zaten **World Wide Web** dediğimiz yapı HTTP protokolü üzerine kurulu. RESTful servisler SOAP, RPC'nin aksine basit ve hafiftirler.

Basit olmalarının yanında oldukça da esnek ve yeteneklidirler. Aslında tipik Web Servislerle yapabileceğiniz herşeyi RESTful servislerle yapabilirsiniz. Ayrıca mimari olarak nasıl olması, ne gibi özelliklere sahip olması hakkında belli yönergeler olsa da, burada SOAP gibi keskin standartları olan bir mimariden bahsetmiyoruz. Üzerine çoğu platformda (C#,JAVA vs.), bir sürü Framework yazılılmış durumda, fakat birçok platformun standart library'leri kullanılarak, kendimiz de hızlıca REstful Servisler geliştirebiliriz.

REST mimarisindeki önemli noktalardan biride her HTTP request'inde yapılması istenilen işlemin HTTP Method'larıyla (Verb) ifade edilmesi. POST, PUT, DELETE ,GET gibi. Böylece proxy ihtiyacı ortadan kalkmış oluyor ve platform bağımsız yapılar kurmak kolaylaşıyor. Şuanki modern uygulamalarda bu Method'ları harfiyen kullanmak bir zorunluluk olmasa da, standartlara uymak, işlem tutartlılığı ve güvenliği açısından önemli. Bu Method'ların üzerinden yazının ilerleyen bölümlerinde geçeceğim.

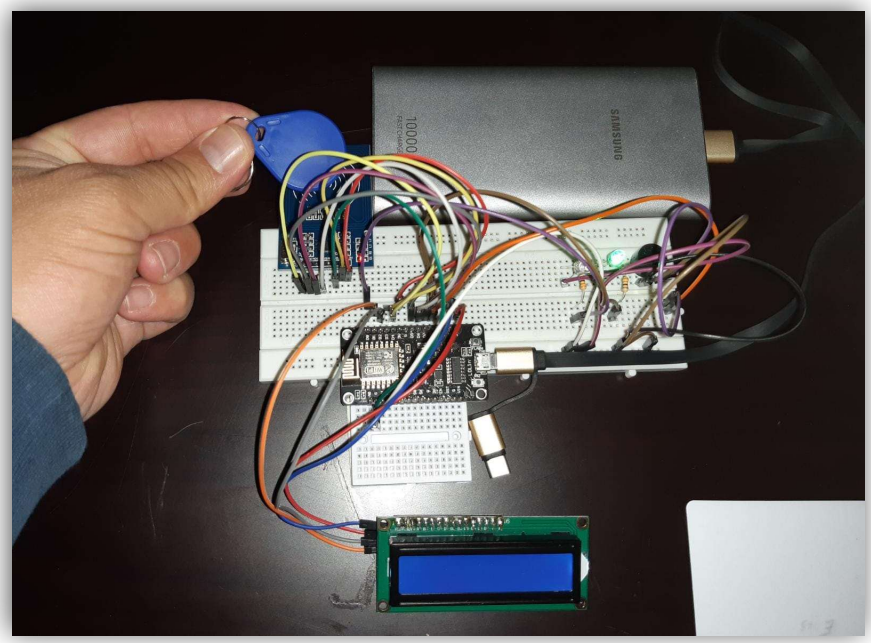
RESTful servisler'in **birçok farklı response tipi** olabilir. Bugünlerde popüler olarak JSON kullanılıyor fakat XML, CSV veya amaca bağlı olarak HMTL bile kullanılabilir. İlginç olan, önceki yıllarda bandwith düşük olmasına rağmen SOAP tabanlı servisler kullanarak kocaman kocaman verileri XML gibi verinin boyutunu daha da şişiren yöntemlerle aktarıyorduk. Şuan bandwith inanılmaz boyutlara ulaşmış durumda ve biz veri transferinde verinin boyutunu XML'e göre inanılmaz küçülten JSON kullanıyoruz, hem de JSON yıllardan beri varolmasına rağmen.

REST'in avantajları nelerdir ;

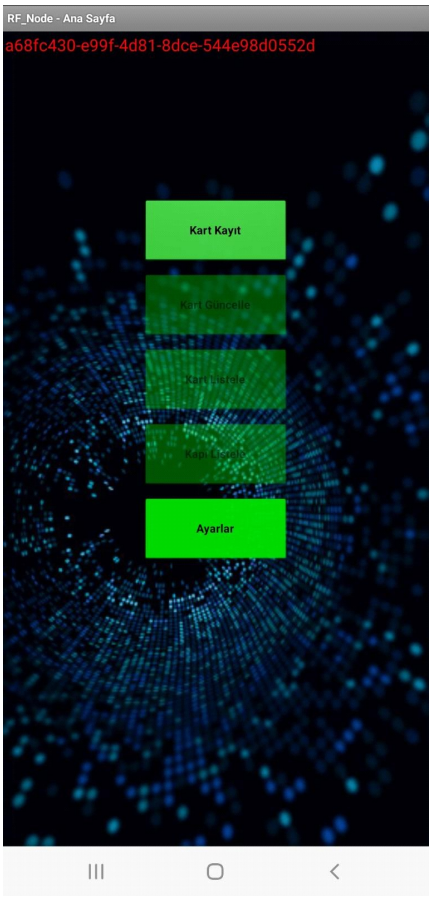
- Hafiftir, kolay extend edilebilir.
- Gelen, giden data boyutu SOAP ile karlılaştırıldığında çok ufaktır
- Tasarlaması kolaydır ve implementasyonu kolaydır, herhangi bir ekstra tool'a ihtiyacı yoktur
- HTTP üzerinden çalışır, platform bağımsızdır

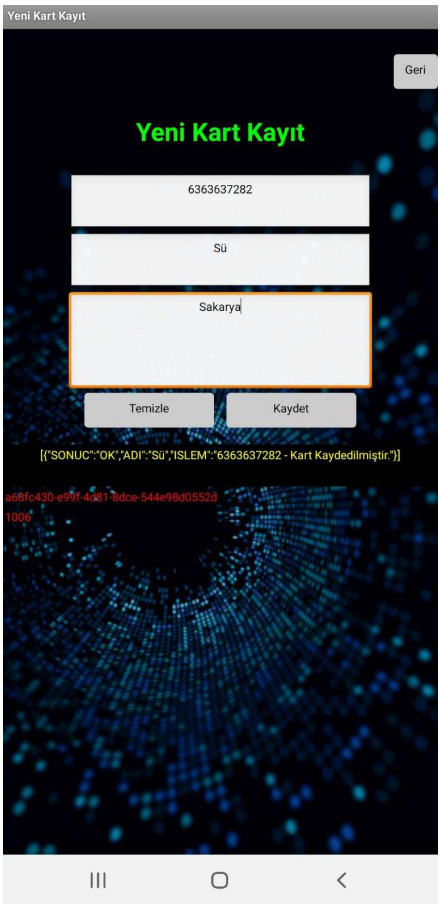
Uygulama Ekran Görüntüleri

Fiziksel Görüntü

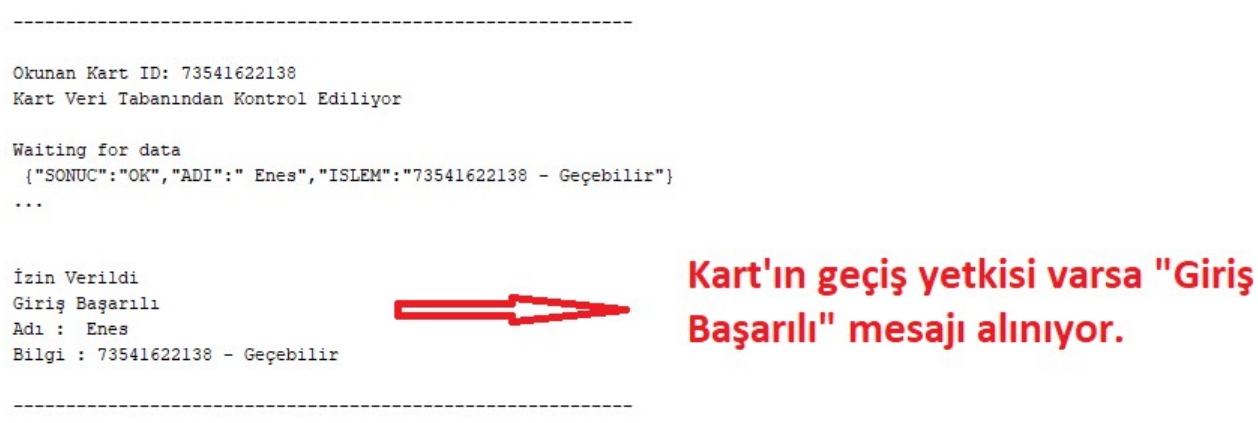


Mobil Arayüz

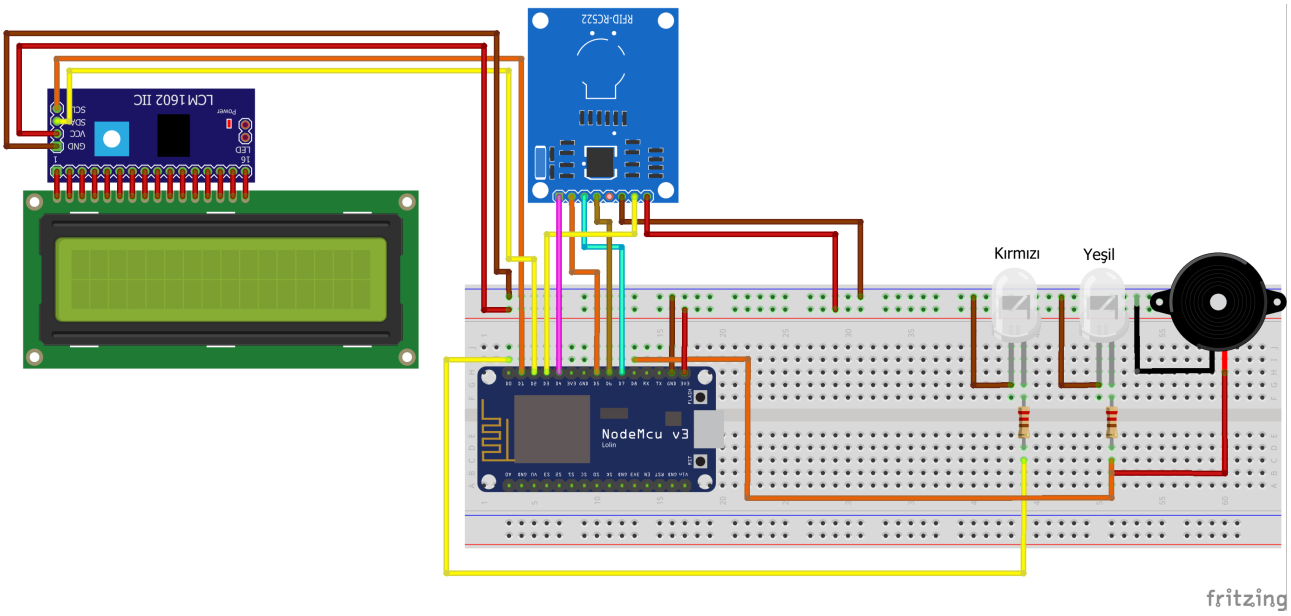




Arduino Seri Port



Fritzing Çizimi



Arduino

```
#include <SPI.h>
#include <RFID.h>
#include <ESP8266WiFi.h>
#include <ArduinoJson.h>
#include <ESP8266WebServer.h>
#include <WiFiManager.h>
#include <stdlib.h>
#include <LiquidCrystal_I2C.h>
WiFiClient client;

char servername2[]="5.180.187.194";  // remote server we will connect to
String APIKEY2 = "a216487722494bcb806fbe632545b6f9";
String kontrol = "kontrol";
String kartid = "";
String sonuc;
String isim;
String bilgi;
String result;
char bufferc[10];

#define SS_PIN D4 //SDA Piniparams
#define RST_PIN D3

int yesilLed = D8;
int kirmiziLed = D0;

RFID rfid(SS_PIN, RST_PIN);

byte ID[5] = {233,223,23,189,156};
boolean izin = true;

LiquidCrystal_I2C lcd(0x3f, 16, 2);

void(* resetFunc) (void) = 0;//declare reset function at address 0

void setup() {
lcd.begin();
lcd.backlight();
DuranYazi();


  pinMode(yesilLed, OUTPUT);
  pinMode(kirmiziLed, OUTPUT);

  Serial.begin(9600);

  WiFiManager wifiManager;
  wifiManager.setTimeout(120);
```

```
  //wifiManager.resetSettings();

if(!wifiManager.autoConnect("Enes-Wifi"))
  {
    Serial.println("failed to connect and hit timeout");
    delay(3000);
    //reset and try again, or maybe put it to deep sleep
    ESP.reset();

    delay(5000);
 }
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println("Connected");

  Serial.println("Wifi");
  Serial.println("Baglandi");

  Serial.println(WiFi.localIP());
  aktifUyari();
  delay(3000);

  SPI.begin();
  rfid.init();

   Serial.print("MAC: ");
   Serial.println(WiFi.macAddress());

}
                            void aktifUyari()
                            {
                               digitalWrite(yesilLed, HIGH);
                                 delay(150);
                               digitalWrite(yesilLed, LOW);
                                 delay(150);
                               digitalWrite(yesilLed, HIGH);
                                 delay(150);
                               digitalWrite(yesilLed, LOW);
                            //    digitalWrite(alarmPin, HIGH);
                                 delay(150);
                            //    digitalWrite(alarmPin, LOW);
                            }

void loop() {

  izin=true;
  if (rfid.isCard())
  {
    sonuc="";
    if(rfid.readCardSerial())
    {
```

```
    Serial.println("------------------------------------------------------------");
    Serial.println("   ");
    Serial.print("Okunan Kart ID: ");

    kartid=rfid.serNum[0];
    kartid=kartid+rfid.serNum[1];
    kartid=kartid+rfid.serNum[2];
    kartid=kartid+rfid.serNum[3];
    kartid=kartid+rfid.serNum[4];
    Serial.println(kartid);
    KartKontrol();
   }

   if(sonuc=="OK")
   {
    Serial.println("İzin Verildi");
    Serial.println("Giriş Başarılı");
    Serial.print("Adı : ");
    Serial.println(isim);
    Serial.print("Bilgi : ");
    Serial.println(bilgi);
     digitalWrite(yesilLed, HIGH);
      delay(100);
     digitalWrite(yesilLed, LOW);
     lcd_Giris_Basarili();
   }
   else
   {
    Serial.println("- Kart Tanımlanmadı - ");
    Serial.println(" -  Giriş Başarısız - ");
     digitalWrite(kirmiziLed, HIGH);
      delay(100);
      delay(100);
     digitalWrite(kirmiziLed, LOW);
     lcd_Giris_Basarisiz();
   }

   Serial.println("   ");
   Serial.println("------------------------------------------------------------");
   Serial.println("   ");
   rfid.halt();
  }
}
        void DuranYazi()
        {
          lcd.clear();
          lcd.setCursor(0, 0);
          lcd.print("Uydu Demirkent-C");
          lcd.setCursor(2, 1);
          lcd.print("Hos Geldiniz");
        }

        void lcd_Giris_Basarisiz()
        {
```

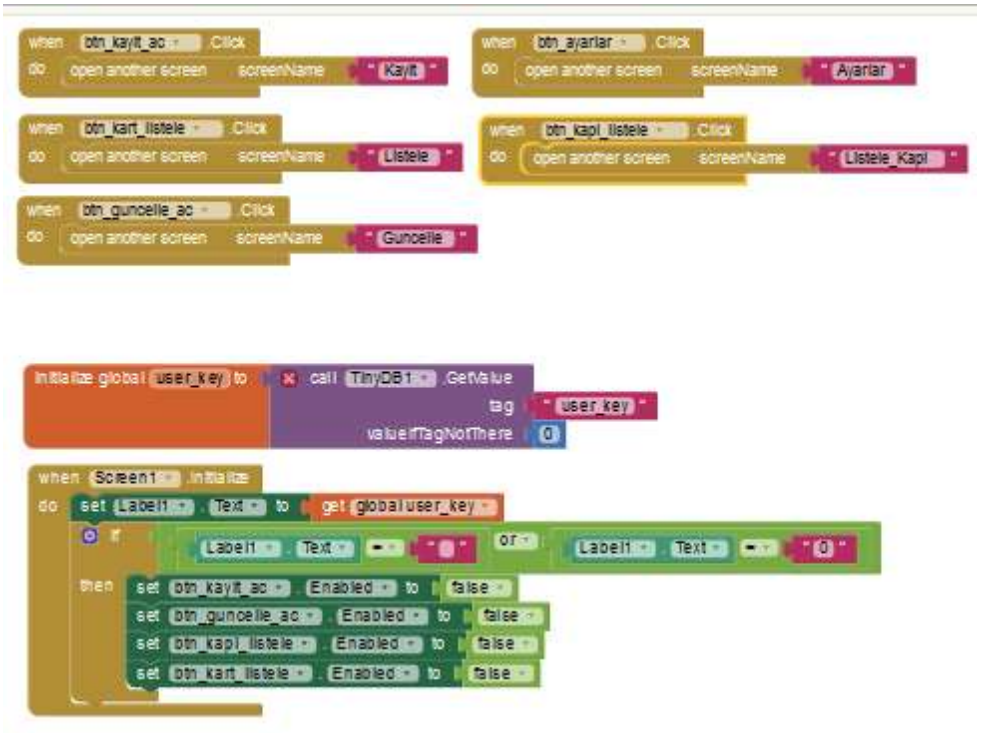
```
                lcd.clear();
                lcd.setCursor(0, 0);
                lcd.print("Kart Tanimsiz");
                lcd.setCursor(0, 1);
                lcd.print("Giris Basarisiz");
                    delay(1500);
                lcd.clear();
                 DuranYazi();
                           }
                 void lcd_Giris_Basarili()
                 {
                  lcd.clear();
                  lcd.setCursor(0, 0);
                  lcd.print("Izin verildi");
                  lcd.setCursor(0, 1);
                  lcd.print("Giris Basarili");
                       delay(1500);
                  lcd.clear();
                    DuranYazi();
                 }



void KartKontrol()
{

  Serial.println("Kart Veri Tabanından Kontrol Ediliyor");
  Serial.println();
  if (client.connect(servername2, 80)) {  //starts client connection, checks for connection
    client.println("GET
http://api.kabloaltyapi.net/EnesGecis/"+kontrol+"/"+APIKEY2+"/"+kartid+"/");
    ///client.println("GET /Export?id="+CityID2);
    client.println("Host: api.kabloaltyapi.net");
    client.println("User-Agent: ArduinoWiFi/1.1");
    client.println("Connection: close");
    client.println();
  }
  else {
    Serial.println("connection failed"); //error message if no client connect
    Serial.println();
  }

  while(client.connected() && !client.available()) delay(1); //waits for data

    Serial.println("Waiting for data");
  result="";
  while (client.connected() || client.available()) { //connected or data available
    char c = client.read(); //gets byte from ethernet buffer
      result = result+c;
      //Serial.print(c);
      if (c==']')
      client.stop(); //stop client
    }
```

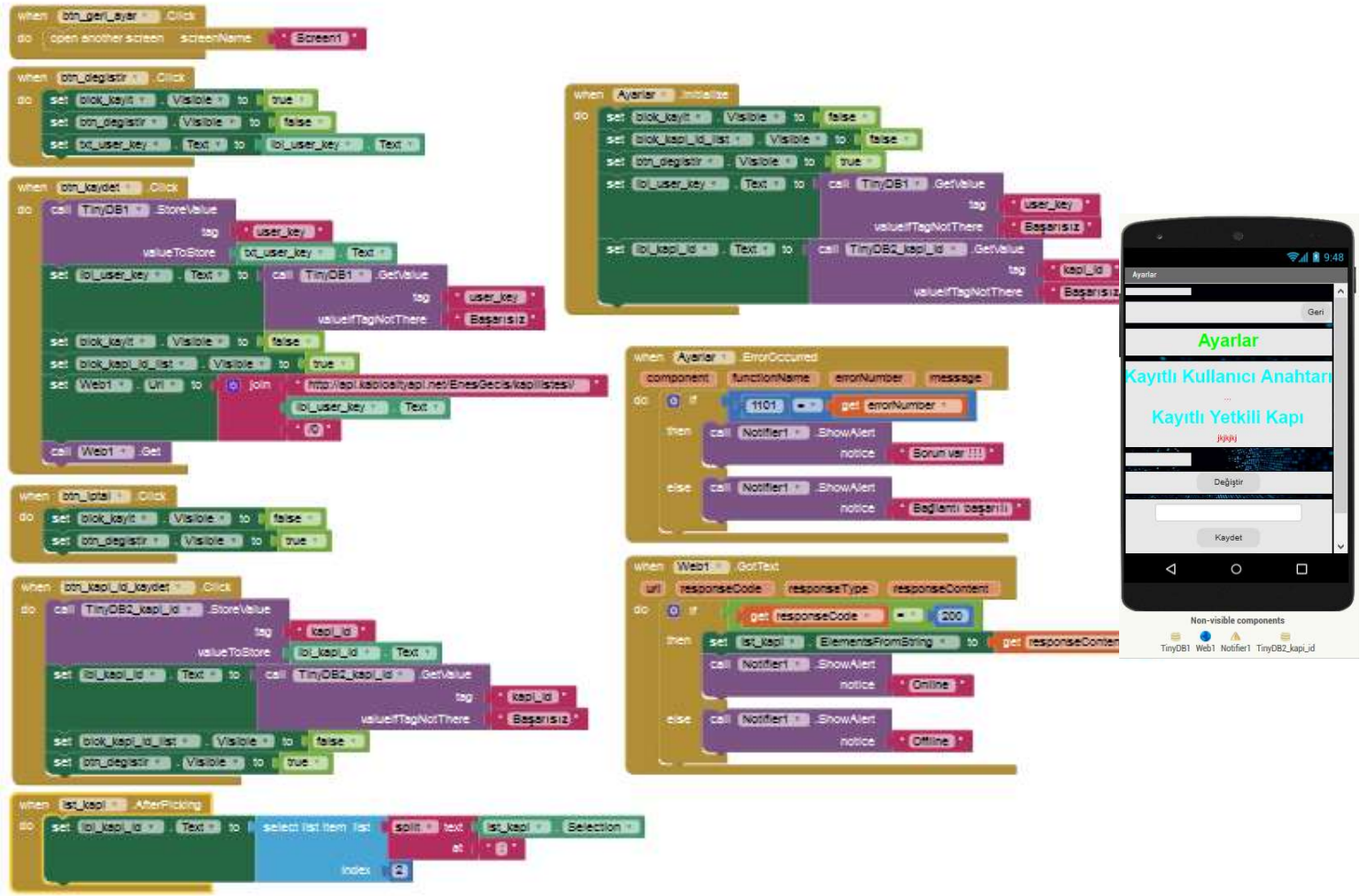
```
  result.replace('[', ' ');
  result.replace(']', ' ');
  Serial.println(result);

char jsonArray [result.length()+1];
result.toCharArray(jsonArray,sizeof(jsonArray));
jsonArray[result.length() + 1] = '\0';

StaticJsonBuffer<1024> json_buf;
JsonObject &root = json_buf.parseObject(jsonArray);
if (!root.success())
{
  Serial.println("parseObject() failed");
}

String sonuc1 = root["SONUC"];
String isim1 = root["ADI"];
String bilgi1 = root["ISLEM"];

sonuc=sonuc1;
isim=isim1;
bilgi=bilgi1;
Serial.println("...");
Serial.println("    ");
Serial.println("    ");

}
```

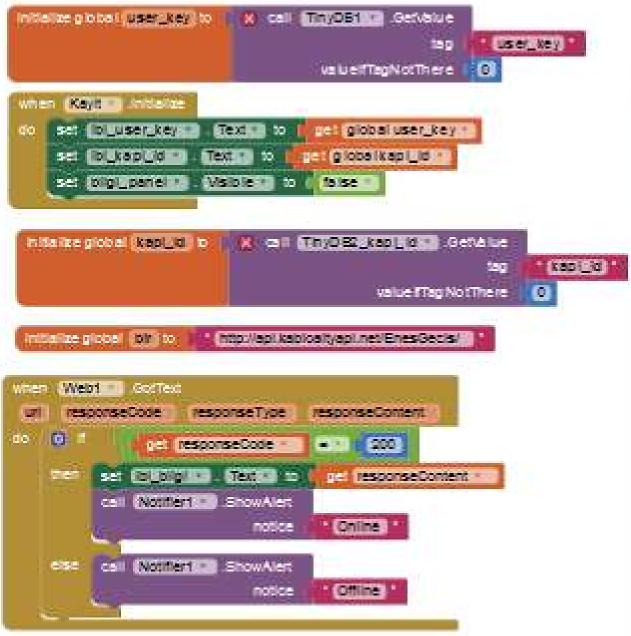
```
            config.Routes.MapHttpRoute(
                 name: "DefaultApi",
                routeTemplate:
"{controller}/{islem}/{apikey}/{id}/{isim}/{adres}/{kapiid}/{durum}",
                defaults: new { Controllers = "Gecis", isim = RouteParameter.Optional, adres
= RouteParameter.Optional, kapiid = RouteParameter.Optional, durum = RouteParameter.Optional
}
            );


        [HttpGet]
        public IHttpActionResult Kayit(string id, string apikey, string islem, string isim,
string adres, string kapiid)
        {
            DataTable dt = new DataTable();


            if (islem == "kayit")
                dt = tkod.KartKayit(apikey, id, isim, adres, kapiid);


            if (dt.Rows.Count == 0)
            {
                return NotFound();
            }
            else
                return Ok(dt);

        }


        public DataTable KartKayit(string api, string kartno, string isim, string adres,
string kapiid)
        {
            SqlConnection conn = new
SqlConnection(ConfigurationManager.ConnectionStrings[databasename].ConnectionString);

            DataTable dt = new DataTable();
            SqlCommand cmd = new SqlCommand();
            cmd.Connection = conn;
            conn.Open();
            int sonuc = 0;
            string isim1 = "";

            cmd.CommandText = "select COUNT(*) FROM API " +
                "INNER JOIN KULLANICI AS K ON K.ID=API.KULLANICI_ID " +
                "WHERE K.ID2='" + api + "' AND  API.ID='" + kapiid + "' ";

            int yetki = Convert.ToInt32(cmd.ExecuteScalar());

            if (yetki > 0)
            {
                cmd.CommandText = "select COUNT(*) FROM KARTLAR WHERE API_ID='" + kapiid +
"' AND  KART_NO='" + kartno + "' ";
                sonuc = Convert.ToInt32(cmd.ExecuteScalar());
                cmd.CommandText = "select ISIM FROM KARTLAR WHERE API_ID='" + kapiid + "'
AND KART_NO='" + kartno + "' ";
                isim1 = Convert.ToString(cmd.ExecuteScalar());

                if (sonuc == 0)
                {
```

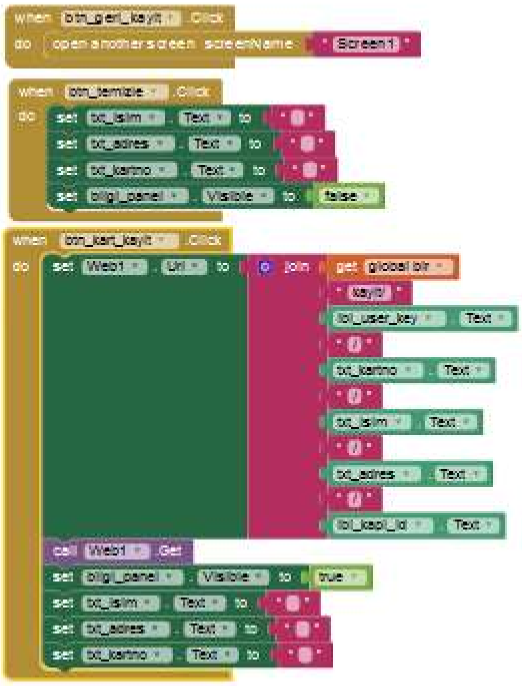
```
                    cmd.CommandText = "INSERT INTO KARTLAR (KART_NO,ISIM, ADRES, API_ID,
KAYIT_TARIHI,DURUM) VALUES('" + kartno + "','" + isim + "' ,'" + adres + "','" + kapiid + "'
,   getdate(),'true') ";
                    cmd.ExecuteNonQuery();
                }

            }
            else
                sonuc = -1;

            conn.Close();


            if (sonuc == 0)
                return KayitVar(isim, kartno + " - Kart Kaydedilmiştir.");
            else if (sonuc == -1)
                return YetkiYok("Bu işlemi yapmayaya yetkiniz yoktur.");
            else
                return KayitYok(isim1 + " - " + kartno + " - Kart zaten kayıtlı");

        }
```

Mit App Invertor






initialize global user_key to
call TinyDB1 .GetValue
tag
user_key
valueIfTagNotThere
0
when Kayit .Initialize
do
set lbl_user_key . Text to
get global user_key
set lbl_kapi_id . Text to
get global kapi_id
set bilgi_panel . Visible to
false
initialize global kapi_id to
call TinyDB2_kapi_id .GetValue
tag
kapi_id
valueIfTagNotThere
0
initialize global blr to
when Web1 .GotText
url
responseCode
responseType
responseContent
do
if
get responseCode
=
200
then
set lbl_bilgi . Text to
get responseContent
call Notifier1 .ShowAlert
notice
Online
else
call Notifier1 .ShowAlert
notice
Offline


when btn_geri_kayit .Click
do
open another screen screenName
Screen1
when btn_temizle .Click
do
set txt_isim . Text to
set txt_adres . Text to
set txt_kartno . Text to
set bilgi_panel . Visible to
false
when btn_kart_kayit .Click
do
set Web1 . Url to
join
get global blr
kayit/
lbl_user_key . Text
txt_kartno . Text
txt_isim . Text
txt_adres . Text
lbl_kapi_id . Text
call Web1 .Get
set bilgi_panel . Visible to
true
set txt_isim . Text to
set txt_adres . Text to
set txt_kartno . Text to


9:48
Yeni Kart Kayıt
Geri
Yeni Kart Kayıt
Temizle
Kaydet
Non-visible components
Web1
TinyDB1
TinyDB2_kapi_id
Notifier1

MSSQL

```
CREATE TABLE KULLANICI
  (
   ID INT IDENTITY(1,1) NOT NULL ,
   ISIM NVARCHAR(50) NULL ,
   KULLANICI_ADI NVARCHAR(50) NULL ,
   SIFRE NVARCHAR(250) NULL ,
   DURUM BIT NULL ,
   ID2 uniqueidentifier NULL DEFAULT newid() UNIQUE ,
   PRIMARY KEY ( ID ),
  );

CREATE TABLE API
  (
   ID INT IDENTITY(1,1) NOT NULL ,
   API NVARCHAR(50) NULL UNIQUE,
   ISIM NVARCHAR(50) NULL ,
   KULLANICI_ID INT NULL ,
   KAYIT_EDEN CHAR(25) ,
   KAYIT_TARIHI INT NULL ,
   DURUM BIT NULL ,
   PRIMARY KEY ( ID ),
   CONSTRAINT FK_KULLANICI FOREIGN KEY (KULLANICI_ID) REFERENCES KULLANICI(ID),
  );

CREATE TABLE KARTLAR
   (
                              ID INT IDENTITY(1,1) NOT NULL,
                              KART_NO nvarchar(50) NULL,
                              ISIM nvarchar(150) NULL,
                              ADRES nvarchar(250) NULL,
                              KAYIT_TARIHI datetime NULL,
                              DURUM bit NULL,
                              KAYIT_EDEN int NULL,
                              API_ID int NULL,
     PRIMARY KEY ( ID ),
                              CONSTRAINT FK_API FOREIGN KEY (API_ID) REFERENCES API(ID),
   );

CREATE TABLE API_LOG
   (
                              ID INT IDENTITY(1,1) NOT NULL,
                              API_ID int NULL,
                              TARIH date NULL,
                              YIL nvarchar(4) NULL,
                              AY nvarchar(2) NULL,
                              IP nvarchar(20) NULL,
                              KAYIT_TARIHI datetime NULL,
                              VERI bit NULL,
                              KART_ID int NULL,
     PRIMARY KEY ( ID ),
                              CONSTRAINT FK_API_L FOREIGN KEY (API_ID) REFERENCES API(ID),
                              CONSTRAINT FK_KART FOREIGN KEY (KART_ID) REFERENCES KARTLAR(ID),
   );
```

Busieness Canvas İş Modeli