****

**AHMET YESEVİ ÜNİVERSİTESİ**

**BİLİŞİM TEKNOLOJİLERİ VE MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**Acil Durum Uyarı Sistemi**

**GERÇEKLEŞTİRİM RAPORU**

**HAZIRLAYAN**

**142132022 Hasan ŞAHİN**

İçindekiler

[Arduino Kodları 3](#_Toc468132584)

[Raspberry Pi Kodları 4](#_Toc468132585)

[Web Api 5](#_Toc468132586)

[UygulamaController 5](#_Toc468132587)

[UygulamaDAL 7](#_Toc468132588)

[Model 7](#_Toc468132589)

[Uygulama 7](#_Toc468132590)

[Message 8](#_Toc468132591)

[Uygulama MVC Modülü 9](#_Toc468132592)

[UygulamaController 9](#_Toc468132593)

[Index 10](#_Toc468132594)

[UygulamaBLL 12](#_Toc468132595)

[Services 12](#_Toc468132596)

[Veri Okuma ve Gönderme Uygulaması 15](#_Toc468132597)

[Form1 15](#_Toc468132598)

[UygulamaBLL 16](#_Toc468132599)

[Services 17](#_Toc468132600)

[Mobile(Android,İphone,WinPhone,Universal Platform) 18](#_Toc468132601)

[Uygulama.xaml 18](#_Toc468132602)

[Uygulama.cs 19](#_Toc468132603)

[Services 20](#_Toc468132604)

# Arduino Kodları

const int analogInPin = A0

int veri = 0;

void setup() {

Serial.begin(9600);

}

void loop() {

veri = analogRead(analogInPin);

Serial.print("Okunan Değer = " );

Serial.println(veri);

delay(750);

}

Özetle A0 anolog pininden 750 ms aralıklarla okuma yapmaya yarıyor.

# Raspberry Pi Kodları

import requests

import RPi.GPIO as GPIO

import time

mseq=0

while(1):

resp = requests.get('http://dunsev.org/api/Uygulama')

if resp.status\_code != 200:

raise ApiError('GET /tasks/ {}'.format(resp.status\_code))

seq = [x['Date']for x in resp.json()]

for x in resp.json():

if x['Date']==max(seq):

mseq=x['Data']

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(18,GPIO.OUT)

print(mseq)

if int(mseq)<50:

GPIO.setup(18,GPIO.HIGH)

time.sleep(5)

else:

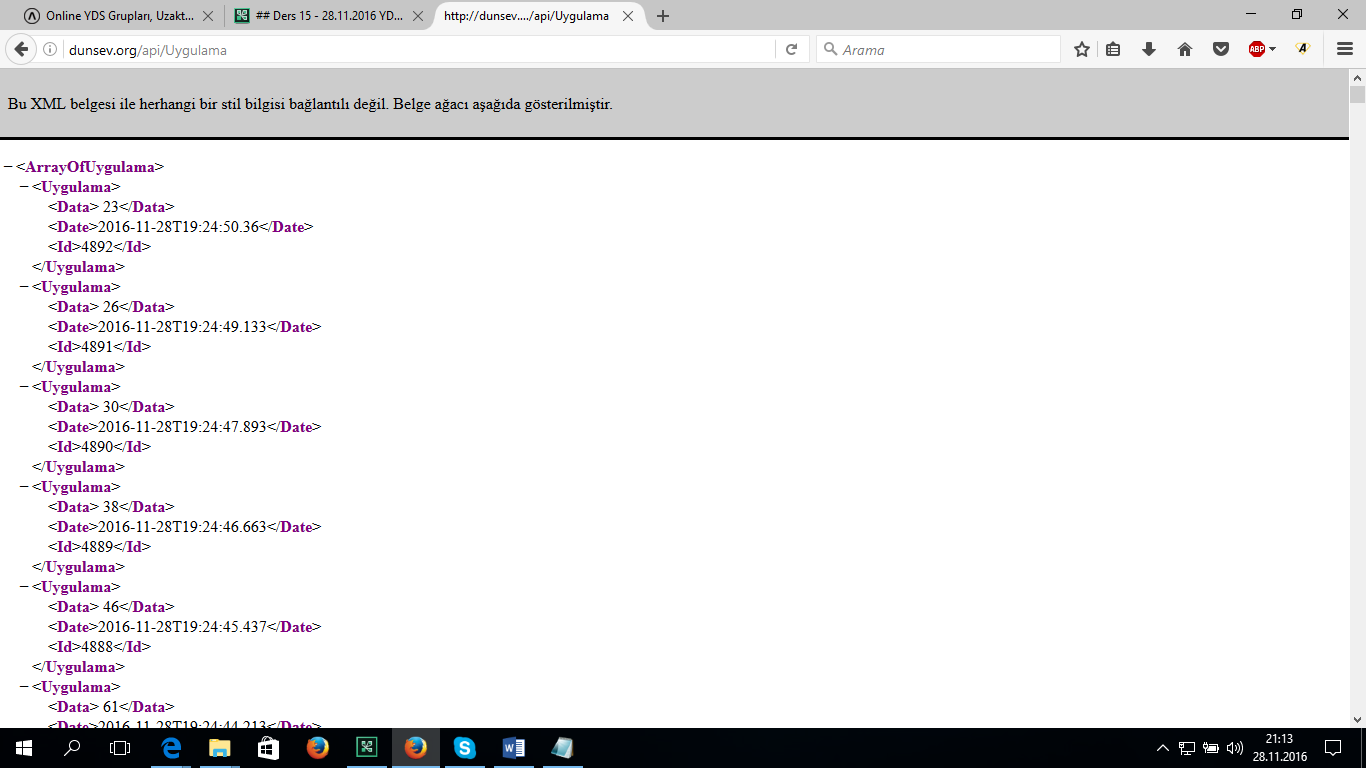
GPIO.setup(18,GPIO.LOW)

time.sleep(5)

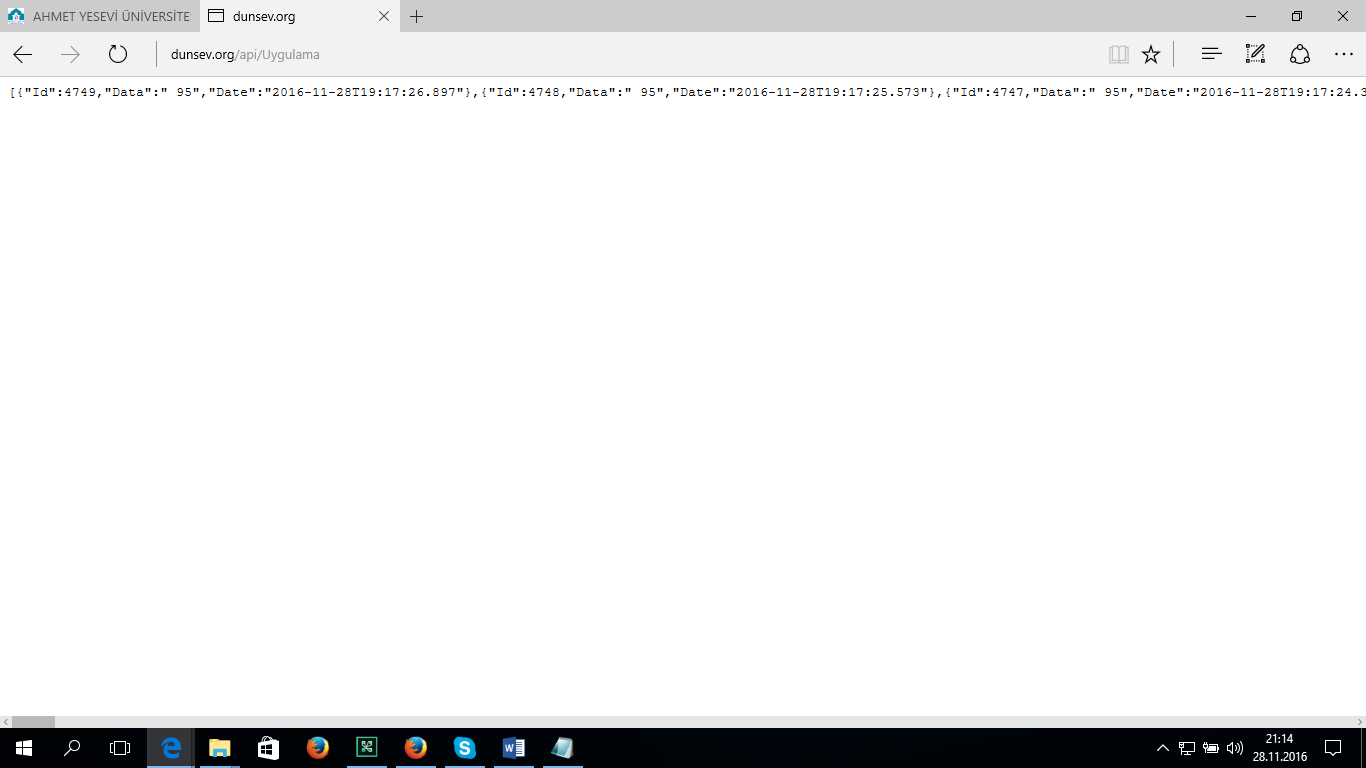
Web apiye get atıp dönen jsonunn datelerini bir diziye alıyorum. Bu dizinin maximum elemanı ile jsonun her elemanını kontrol ediyorum buradaki amaç en güncel veriyi almak. Sonra printle ekrana yazıyorum ve 50 den küçükse fanı kapatıyorum 50 den büyükse fanı çalıştırıyorum.

Not: Phyton 2.x ile çalışmıyor, Phyton 3.x ile çalışıyor.(En az 2 saatimi yedi ☺)

# Web Api



Header da istenen tipe göre xml formatında gelen veriler.



Header da istenen tipe göre json formatında gelen veriler.

## UygulamaController

namespace Dunsev.Mvc.Controllers

{

public class UygulamaController : ApiController

{

private readonly UygulamaDAL \_UygulamaDal;

public UygulamaController()

{

\_UygulamaDal = new UygulamaDAL();

}

// GET: Olay

public IEnumerable<Uygulama> Get()

{

return \_UygulamaDal.List();

}

public Message Post([FromBody]JObject value)

{

string durum = String.Empty;

try

{

\_UygulamaDal.Add(JsonConvert.DeserializeObject<Uygulama>(value.ToString()));

durum = "200";

}

catch (Exception exception)

{

durum = exception.InnerException.Message;

}

Message msg = new Message();

msg.Data = null;

msg.Information = durum;

return msg;

}

}

}

Get actionu uygulama entitysini list olarak döndürüyor.

Most actionu uygulama entitysini ekleyip hata yok ise hata olmadığını hata var ise hata kodunu message entityi ne yazıp geri döndürür. Burada newtonsoft un jsonconverter classı ile jsonu uygulama entitysine dönüştürüp add metoduna veriyor.

## UygulamaDAL

namespace Dunsev.DAL

{

public class UygulamaDAL

{

private Context db;

public UygulamaDAL()

{

db = new Context();

}

public void Add(Uygulama Uygulama)

{

db.Uygulama.Add(Uygulama);

db.SaveChanges();}

public List<Uygulama> List()

{

return db.Uygulama.OrderByDescending(o=>o.Date).ToList();

}

}

Add ile dbye ekliyoruz List ile de datihe göre sıralı listeletiyoruz.

## Model

### Uygulama

namespace Dunsev.Model

{

public class Uygulama

{

public int Id { get; set; }

public string Data { get; set; }

public DateTime Date { get; set; }

}

}

### Message

namespace Dunsev.Model

{

public class Message

{

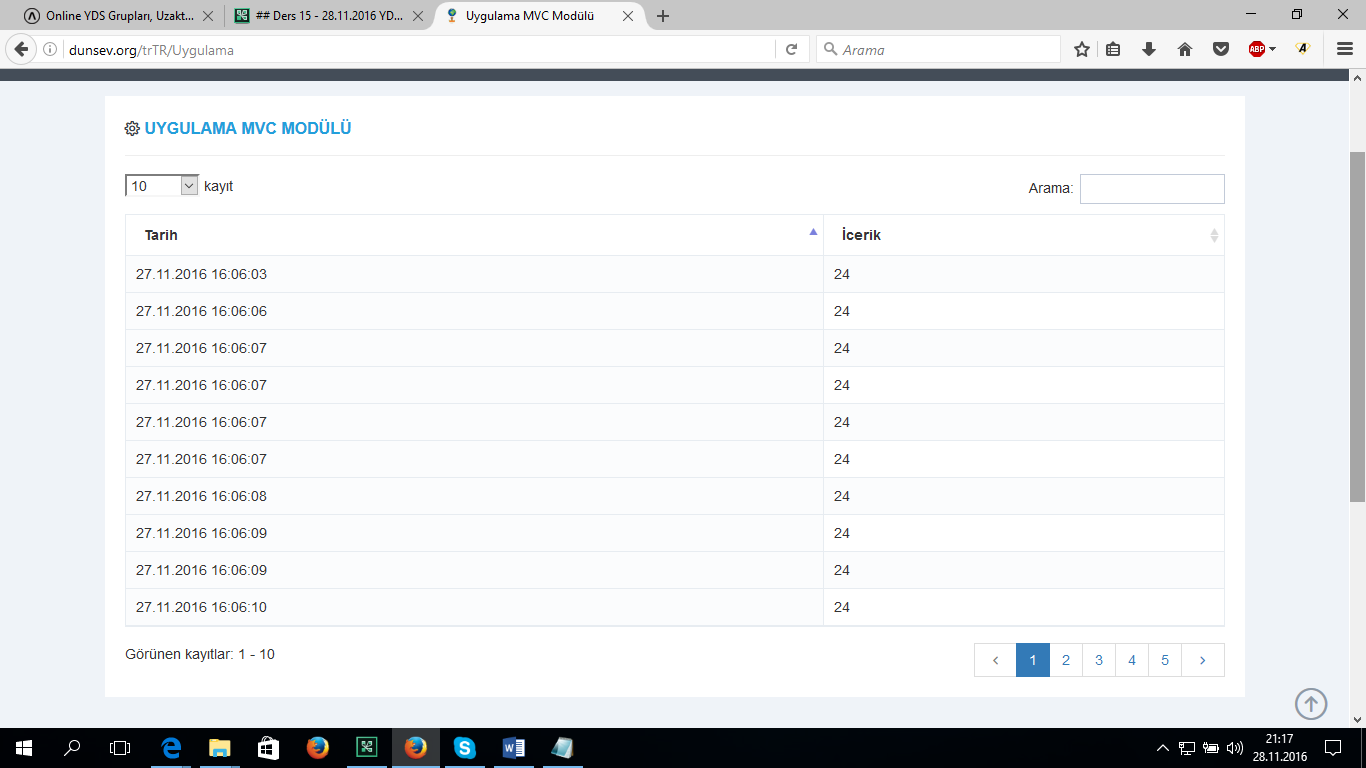
public object Data { get; set; }

public string Information { get; set; }

}

}

# Uygulama MVC Modülü



## UygulamaController

namespace Dunsev.Mvc.Areas.trTR.Controllers

{

public class UygulamaController : Controller

{

// GET: trTR/Uygulama

public ActionResult Index()

{

return View();

}

}

}

Namespacedende anlaşılacağı üzere “trTR” areasında çalıştığım için “trTR/Uygulama” veya “trTR/Uygulama/Index” yoluna gittiğimizde aşağıdaki dosyaya yönlendirme yapıyor.

## Index

@using Dunsev.BLL

@{

ViewBag.Title = "Uygulama MVC Modülü";

var UygulamaBll = new UygulamaBLL();

string[] htmlTags ={"<title>"+ViewBag.Title+"</title>",

"<meta name='description' content='DÜNSEVin resmi sitesine hoş geldiniz.' />",

"<meta name='keywords' content='DÜNSEV, ana,sayfa,serabral,palsi'>"};

ViewBag.htmlTags = htmlTags;

}

<div class="portlet light ">

<div class="portlet-title">

<div class="caption font-dark">

<i class="icon-settings font-dark"></i>

<a class="caption-subject bold uppercase" href="#">@ViewBag.Title</a>

</div>

<div class="tools"> </div>

</div>

<div class="portlet-body">

<div id="sample\_1\_wrapper" class="dataTables\_wrapper no-footer">

<table id="Olay" class="table table-striped table-bordered" cellspacing="0" width="100%">

<thead>

<tr>

<th>Tarih</th>

<th>İcerik</th>

</tr>

</thead>

<tbody>

@{

int i = 0;

string oldEven = string.Empty;

foreach (var uygulama in UygulamaBll.List())

{

i++;

if (i % 1 == 0)

{

oldEven = "old";

}

else

{

oldEven = "even";

}

<tr role="row" class="@oldEven">

<td>

@uygulama.Date

</td>

<td>

<div class="event-info fleft">

@uygulama.Data

</div>

</td>

</tr>

}

}

</tbody>

</table>

</div>

</div>

</div>

Templateyi bozmamak üzere yazdığım bu kodlarda esas olan uygulama listesinin tabloya yazılmasıdır. Burada tablonun peş peşe gelen satırlarına farklı classlar gelmesi için sayaç mod 2 ile kontrol ettim.

## UygulamaBLL

namespace Dunsev.BLL

{

public class UygulamaBLL

{

private UygulamaDAL UygulamaDal;

private Services<Uygulama> services = new Services<Uygulama>();

public UygulamaBLL()

{

UygulamaDal = new UygulamaDAL();

}

public void Add(Uygulama Uygulama)

{

services.autorequest(Uygulama);

}

public IEnumerable<Uygulama> List()

{

return UygulamaDal.List();

}

}

}

Listte authentication ile uğraşmamak için UygulamaBll den listelettim. Addde ise servicesden eklettim.

## Services

namespace Dunsev.Tools

{

public class Services<T>

{

private string access\_token;

private string servicesurl = ConfigurationSettings.AppSettings["DataSource"] ;

private void logout()

{

var url = servicesurl+"api/Account/Logout";

var pairs = new List<KeyValuePair<string, string>>

{

new KeyValuePair<string, string>("","")

};

var content = new FormUrlEncodedContent(pairs);

using (var client = new HttpClient())

{

client.DefaultRequestHeaders.Add("Authorization", "Bearer " + access\_token);

var response = client.PostAsync(url, content).Result;

}

}

private void auto()

{

string url = servicesurl + "Tkn";

var pairs = new List<KeyValuePair<string, string>>

{

new KeyValuePair<string, string>( "grant\_type", "password" ),

new KeyValuePair<string, string>( "username",ConfigurationSettings.AppSettings["UserName"] ),

new KeyValuePair<string, string> ( "password",ConfigurationSettings.AppSettings["Password"] )

};

var content = new FormUrlEncodedContent(pairs);

using (var client = new HttpClient())

{

var response = client.PostAsync(url, content).Result;

var result = response.Content.ReadAsStringAsync().Result;

JToken token = JObject.Parse(result);

access\_token = (string)token.SelectToken("access\_token");

}

}

private Message Add(T item)

{

Message result;

var url = servicesurl + "api/" + typeof(T).Name;

var content = new StringContent(JsonConvert.SerializeObject(item), Encoding.UTF8, "application/json");

using (var client = new HttpClient())

{

client.DefaultRequestHeaders.Add("Authorization", "Bearer " + access\_token);

var response = client.PostAsync(url, content).Result;

result = JsonConvert.DeserializeObject<Message>(response.Content.ReadAsStringAsync().Result);

if (response.StatusCode == HttpStatusCode.Unauthorized)

{

var x = autorequest();

}

}

return result;

}

public string autorequest(T item)

{

auto();

var x=Add(item);

logout();

return x.Information;

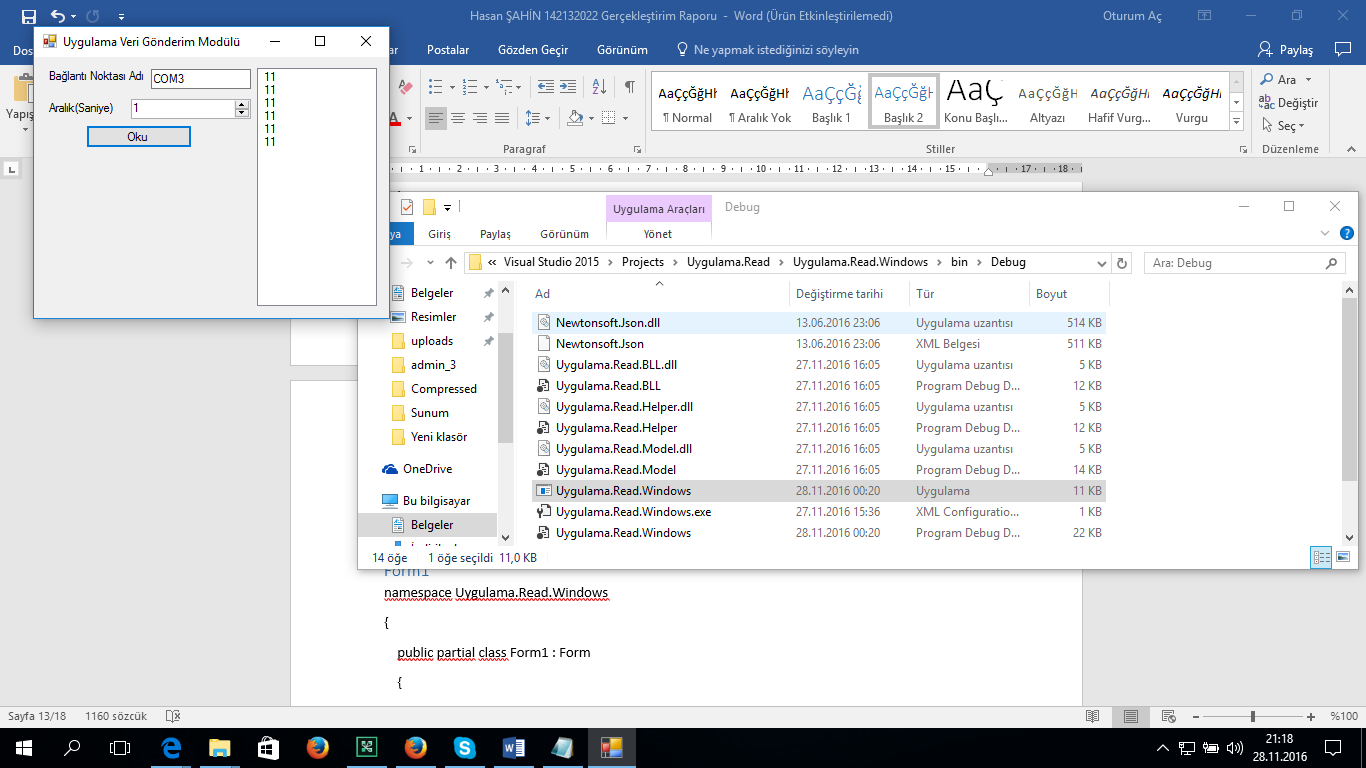
}

}

}

Auto ile token alıyor add ile itemı jsona çevirip tokenla beraber post ediyor logout ile de sessionu sanlandırıyor.

# Veri Okuma ve Gönderme Uygulaması



## Form1

namespace Uygulama.Read.Windows

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private SerialPort sp;

private readonly UygulamaBLL \_uygulamaBll=new UygulamaBLL();

private void Sp\_DataReceived(object sender, SerialDataReceivedEventArgs e)

{

string veri = sp.ReadLine();

this.BeginInvoke(new LineReceivedEvent(LineReceived), veri);

}

private delegate void LineReceivedEvent(string veri);

private void LineReceived(string veri)

{

var aralık = Convert.ToInt32(nudAralık.Text);

var parse = veri.Split('=')[1];

var data = parse.Substring(0, parse.Length - 2);

lstValue.Items.Insert(0,data);

var uygulama = new Model.Uygulama

{

Data=data,

Date=DateTime.Now

};

\_uygulamaBll.Add(uygulama);

Thread.Sleep(aralık\*1000);

}

private void btnRead\_Click(object sender, EventArgs e)

{

sp = new SerialPort(txtPort.Text, 9600);

sp.Open();

sp.DataReceived += Sp\_DataReceived;

}

}

}

Formdan port adını alıp portu açıyor gelen veriyi “=”den 2 parçaya ayıp 2. Parçasını alıyor sondaki”/n” i substring ile atıp listenin en üstüne ve web service ekleyip girilen süre kadar bekliyor.

## UygulamaBLL

namespace Uygulama.Read.BLL

{

public class UygulamaBLL

{

readonly Services \_services=new Services();

public Message Add(Model.Uygulama uygulama)

{

return \_services.Add(uygulama);

}

}

}

Adde gelen entityi servicenin add metoduna yolluyor.

## Services

namespace Uygulama.Read.Helper

{

public class Services

{

private string url = "http://dunsev.org/api/Uygulama";

public Message Add(Model.Uygulama item)

{

Message result;

var content = new StringContent(JsonConvert.SerializeObject(item), Encoding.UTF8, "application/json");

using (var client = new HttpClient())

{

var response = client.PostAsync(url, content).Result;

result = JsonConvert.DeserializeObject<Message>(response.Content.ReadAsStringAsync().Result);

}

return result;

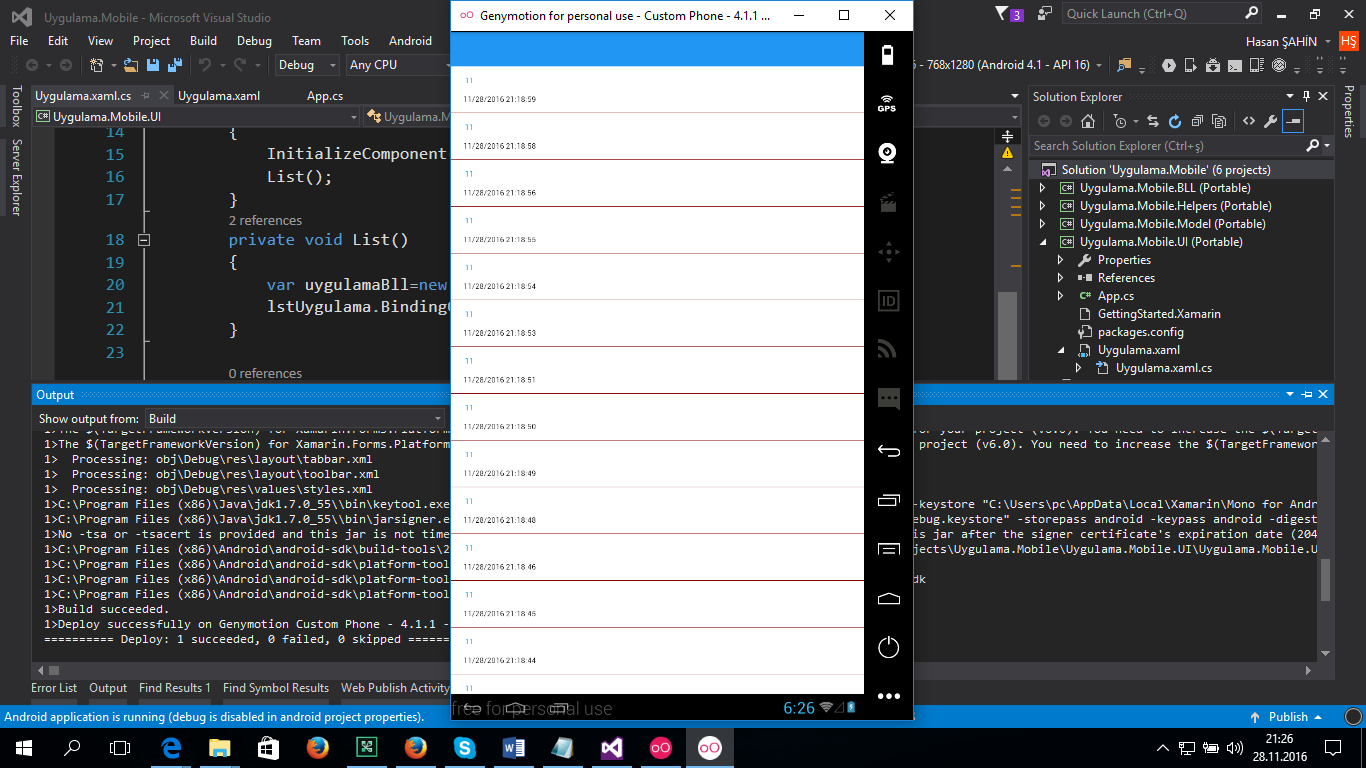
}

}

}

Gelen entityi jsona çevirip servise post ediyor. Json farmatında gelen veriyi Message tipinde entitye çevirip geri döndürüyor.

# Mobile(Android,İphone,WinPhone,Universal Platform)



## Uygulama.xaml

<?xml version="1.0" encoding="utf-8" ?>

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="Uygulama.Mobile.UI.Uygulama">

<ListView x:Name="lstUygulama" ItemsSource="{Binding .}"

VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand"

SeparatorColor="Maroon"

BackgroundColor="#ffffff"

IsPullToRefreshEnabled="True"

HasUnevenRows="True"

Refreshing="OnRefresh">

<ListView.ItemTemplate>

<DataTemplate>

<TextCell Text="{Binding Data}" TextColor="#08A8F4"

Detail="{Binding Date}" DetailColor="Black"/>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

</ContentPage>

Liste oluşturup OnRefresh eventı belirtiyorum textcell gösterilecek bilgilerin field nameleri olan “Data” ve “Date” yi belirtiyorum.

## Uygulama.cs

namespace Uygulama.Mobile.UI

{

public partial class Uygulama : ContentPage

{

public Uygulama()

{

InitializeComponent();

List();

}

private void List()

{

var uygulamaBll=new UygulamaBLL();

lstUygulama.BindingContext = uygulamaBll.List();

}

private void OnRefresh(object sender, EventArgs e)

{

List();

lstUygulama.IsRefreshing = false;

}

}

}

OnRefresh de yenileme ikonunun kaybolması için IsRefreshing i false yapıyorum.

## Services

namespace Uygulama.Mobile.Helper

{

public class Services

{

private string url = "http://dunsev.org/api/Uygulama";

public List<Model.Uygulama> List()

{

using (var client = new HttpClient())

{

var response = client.GetAsync(url).Result;

var result = JsonConvert.DeserializeObject<List<Model.Uygulama>>(response.Content.ReadAsStringAsync().Result);

return result;

}

}

}

}

Service ye get isteği göndererek dönen jsonu uygula listesine dönüştürüp geri yolluyor bu işlemin aynısı raspberry pida da yapılıyor.