T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BAKIMA MUHTAÇ HASTALARIN MOBİL CİHAZLAR ÜZERİNDEN TAKİP EDİLMESİ

YÜKSEK LİSANS TEZİ

Derya KARABAK
(Y1213.010009)

Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Ali GÜNEŞ

Temmuz, 2016

T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ MÜDÜRLÜĞÜ

Yüksek Lisans Tez Onay Belgesi

Enstitümüz Bilgisayar Mühendisliği Ana Bilim Dalı Bilgisayar Mühendisliği Tezli Yüksek Lisans Programı **Y1213.010009** numaralı öğrencisi **Derya KARABAK**'ın **"BAKIMA MUHTAÇ HASTALARIN MOBİL CİHAZLAR ÜZERİNDEN TAKİP EDİLMESİ"** adlı tez çalışması Enstitümüz Yönetim Kurulunun 30.06.2016 tarih ve 2016/18 sayılı kararıyla oluşturulan jüri tarafından oybirliği ile Tezli Yüksek Lisans tezi olarak kabul edilmiştir.

Öğretim Üyesi Adı Soyadı	**İmzası**
Tez Savunma Tarihi :22.07.2016	
1)Tez Danışmanı: Prof. Dr. Ali GÜNEŞ	.....
2) Jüri Üyesi : Yrd. Doç. Dr. Metin ZONTUL	.....
3) Jüri Üyesi : Yrd. Doç. Dr. Ferdi SÖNMEZ	.....

Not: Öğrencinin Tez savunmasında **Başarılı** olması halinde bu form **imzalanacaktır**. Aksi halde geçersizdir.

YEMİN METNİ

Yüksek Lisans tezi olarak sunduğum "Bakıma Muhtaç Hastaların Mobil Cihazlar Üzerinden Takip Edilmesi" adlı çalışmanın, tezin proje safhasından sonuçlanmasına kadarki bütün süreçlerde bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurulmaksızın yazıldığını ve yararlandığım eserlerin Bibliyografya'da gösterilenlerden oluştuğunu, bunlara atıf yapılarak yararlanılmış olduğunu belirtir ve onurumla beyan ederim. (28.06.2016)

Derya KARABAK /

ÖNSÖZ

Tez çalışmam boyunca beni yönlendiren ve yardımcı olan değerli danışmanım Sayın Prof. Dr. Ali GÜNEŞ'e ve eğitimim boyunca üzerimde emeği geçen tüm hocalarıma teşekkür eder, saygılarımı sunarım.

Temmuz 2016 **Derya KARABAK**

İÇİNDEKİLER

Sayfa

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

BAKIMA MUHTAÇ HASTALARIN MOBİL CİHAZLAR ÜZERİNDEN TAKİP EDİLMESİ

ÖZET

Bilgisayar yazılım ve donanım alanlarındaki gelişmeler sayesinde cep telefonları ve tabletler artık hayatımızın vazgeçilmez birer parçası haline gelmiştir. Gerek günlük hayatımızda gerekse bankacılık, turizm ve eğitim gibi bir çok sektörde işleri daha da kolaylaştırmak amacı ile bu cihazları kullanmaktayız. Sağlık sektörü de bu anlamda gün geçtikçe üzerinde daha çok araştırma yapılan ve gelişime açık olan sektörlerden bir tanesidir.

Özellikle yaşlı, engelli, ameliyatlı yada özel sağlık sorunlarından dolayı günlük yaşam aktivitelerini kendi başlarına tam anlamıyla gerçekleştiremeyen bakıma muhtaç hastalar için tüm dünyada olduğu gibi ülkemizde de bir çok çalışma gerçekleştirilmektedir. Evde sağlık hizmetleri, hem hastaların kendi evlerinde rahat bir ortamda sağlık kontrollerinin yapılabilmesi hem de hastanelerdeki yoğunluğun azaltılabilmesi anlamında önemli bir uygulamadır. Bu anlamda evde sağlık hizmetlerinin önemi de gün geçtikçe artmaktadır.

Bu tez çalışmasının amacı bakıma muhtaç hastaların doktorlarını beklemek zorunda kalmadan gerek kendi ihtiyaç duydukları zaman gerekse doktor isteklerine göre kendi ölçümlerini yapabilmeleri ve aynı zamanda doktorlarında hastalarını mobil olarak takip edebilmeleridir. En önemli özellik ise, bireysel olmasıdır. Örneğin tansiyon kişiden kişiye ve hastalıktan hastalığa değişen bir sağlık durumudur. Oluşturulan model ile doktor tarafından belirlenen, kişiye özel sınır değerler sisteme tanımlanır ve ölçüm bu değerler dışına çıktığı takdirde sistem tarafından doktora otomatik olarak bildirim şeklinde gönderilir. Ayrıca hastanın günlük ölçümlerini ya da geçmişe dair ölçümlerini hem hasta hem de doktor görüntüleyebilmekte ve her hangi bir durum karşısında hem hasta hem de doktor mesaj aracılığı ile iletişime geçebilmektedir.

Anahtar Kelimeler: *Sağlık Mobil Takip, Biyometrik Sensör, Evde Bakım Hizmetleri, Bakıma Muhtaç Hasta*

TRACKING OF IN NEED OF CARE PATIENT VIA MOBILE DEVICES

ABSTRACT

Mobile phones and tablets have become an indispensable part of our lives because of advances in computer hardware and software. We use these devices in order to make it easier to work in many sectors like banking, tourism and education and besides our daily life. The health sector is also one of the sectors which is open the development. And research continues to be done day by day.

Many studies are performed in our country as well as all over the world especially for elderly, handicapped, operated or in need of care patient which are unable to perform activities of daily living because of their special health problems. Home health services, it is important application for patients to make their health checks in a relaxed atmosphere in their home. Moreover it is important in order to reduce patient intensity in hospitals. For those reasons, the importance of home care services is increasing day by day.

The aim of this thesis is to be able to be monitored by doctors of patients in need of care as mobile. So, patients will be able to make their own measurements and send it to their doctors. The most important feature is that individual. For example, blood pressure values are a medical condition that changes from person to person and according to the disease. Specific limit values which are determined by doctors are defined in our model system. If the measurement is out of these values are sent to the doctor by an automated notification system. In addition, the patient's daily measurements or earlier measurements can display both the patient and the doctor. Moreover, for any purpose both patient and doctor can contact through the message.

Keywords: *Health Mobile Tracking, Biometric Sensor, Home Care Services, In Need of Care Patient*

1 GİRİŞ

Kişisel bilgisayarların ortaya çıkmasıyla birlikte hayatımıza giren bilgisayarlar gün geçtikçe fiziksel olarak küçülmektedir. Bunun sonucunda tablet ve akıllı telefonlar olarak günlük hayatımızın da vazgeçilmez birer parçası olmuştur. Dolayısı ile günümüzde gerek sosyal gerek ekonomik birçok alanda doğrudan ya da dolaylı olarak bizleri pek çok açıdan etkilemektedir. Sağlık, turizm, medya, otomotiv ve daha birçok sektörde vazgeçilmez bir hal alan bilgisayarların bu gelişimindeki temel neden donanımdaki küçülme ve ucuzlamanın yanı sıra aynı zamanda birçok alanda yazılım geliştirilmesinden kaynaklanmaktadır. Gerek masaüstü bilgisayarlar için gerekse tablet ve akıllı telefonlar için yazılan bir çok uygulama insan hayatını gittikçe kolaylaştırmaktadır.

Bilgisayar ve yazılım sektöründeki tüm bu gelişmelerden insan hayatı için en önemli şey olan sağlık sektörü de ciddi bir şekilde etkilenmiştir. Özellikle yaşlı, engelli ya da bakıma muhtaç hastaların yaşam standartlarını daha iyi bir hale getirebilmek amacıyla devletin sağladığı imkanların yanı sıra bir çok özel kuruluş, akademisyen ve arge şirketleri konunun üzerine düşmekte ve günden güne konuyla ilgili yapılan çalışmalar artmaktadır.

Bakıma muhtaç hasta tanımı; günlük yaşam ihtiyaçlarının bir bölümünü veyahut tamamını, herhangi bir destek olmaksızın, belli bir süreliğine ya da devamlı olarak karşılayamayan kişidir [1]. Bu hastalar için hem devlet hem de özel kuruluşlar tarafından evde sağlık hizmetleri verilmektedir. Gerek psikolojik gerekse tıbbi olarak hastalar bu imkanlardan faydalanabilmektedirler. Evde bakım hizmetleri için 2005 yılında yönetmelik ve genelgeler hazırlanmıştır ve bunun sonucunda özel sektör tarafından uygulamalara başlanmıştır. Ancak Sağlık Bakanlığı' na bağlı kurum ve kuruluşların hizmete başlaması ile ilgili yönerge 2010 yılında yürürlüğe konulmuştur [2].

Bakıma muhtaç hastalar için sunulan evde bakım hizmetleri yönergesinde de belirtildiği gibi görevli ekip evde sağlık hizmeti alan kişlileri belirli aralıklarla

ziyaret ederek durum değerlendirmesi sonucunda ortaya çıkan ihtiyaçlara göre yeni tanı, tedavi ve tibbi araç gereksinimlerini karşılamakla görevlidir [3].

Yönergede de belirtildiği gibi hekim belli aralıklarla hastaları ziyaret edebilmektedir. Bu proje ile yapılması hedeflenen amaç ise hastaların sürekli takibinin daha kolay bir şekilde gerçekleştirilebilmesidir. Oluşturulan model ile; doktor hastanın evine gitmeden de hastasını sistemden kontrol edebilecek ve mesajlaşma özelliği ile hastaya tavsiyelerde bulunabilecektir. Aynı zamanda hastanın herhangi bir şekilde doktoru ile iletişime geçmesi gerekirse yine mesajlaşma özelliği ile doktoruna ulaşabilecektir. Ayrıca doktorun sisteme belirttiği sağlık değerlerinin dışına çıkan bir ölçüm gerçekleştiğinde sistem tarafından doktora bildirim gönderilecektir.

Bölüm 2'de evde bakım hizmetleri ile ilgili bilgiler ve kullanılan yöntem ve uygulamalar ile ilgili bilgiler ele alınmıştır.

Bölüm 3'de projede kullanılan Arduino tabanlı sağlık platformunun özellikleri ve kullanılan sensörler ve uygulama şekilleri ile ilgili detaylı olarak bilgi verilmiştir.

Bölüm 4 mobil takip sistemi modelinin veritabanı ilişkileri, tablo yapıları ve Android mobil uygulaması ile olan iletişimi ve bilgi alışverişi detaylandırılmıştır.

Bölüm 5 ise mobil takip sistemi uygulamasının kullanıcı arayüzlerinin ve internet üzerinden haberleşme modülünün ayrıntılarını içermektedir.

Sonuç bölümünde ise oluşturulan modelin avantajları, ileriki zamanlarda nasıl daha etkin bir şekilde kullanılabileceği hakkında öneri ve görüşler sunulmuştur.

2 EVDE BAKIM HİZMETLERİ

2.1 Türkiye'de Evde Bakım Hizmetleri

Tüm Dünya'da olduğu gibi Türkiye'de de evde bakım hizmetleri son zamanlarda önemli bir hale gelmiştir. Yaşlı, engelli ya da ameliyat sonrası evde bakımı mümkün olan hastalar için verilen bu hizmetlerin bir çok açıdan önemi büyüktür. Öncelikle hastaların psikolojik açıdan evlerinde, ailelerinin yanında kendilerini hastane ortamına göre daha rahat hissetmeleri göz önüne alınabilir. Sonrasında ise hastanelerdeki hasta yoğunluğu düşünüldüğünde bu hizmet ile yoğunluk azaltılabilir duruma gelmektedir.

Türkiye'de evde sağlık hizmetleri;

- Aile hekimleri,
- A, B, ve C Grubu hastaneler ile Dal Hastaneleri bünyesinde oluşturulan evde sağlık hizmet birimleri,
- İl Sağlık Müdürlükleri bünyesinde oluşturulan Mobil Ekipler tarafından verilmektedir [2].

Yenidoğan, yatağa bağımlı ve koah gibi hastalıklar ve palyatif bakıma ihtiyacı olan hastalara evde sağlık hizmetleri verilmektedir. Bu hizmetten yararlanan hastaların kullanımına verilen bazı tıbbi aletler ise şunlardır; [2]

- Şeker ölçüm cihazı
- Mekanik ventilator,
- Ev tipi aspirator
- Pulse oksimetri
- Solunum fonksiyonu test cihazı
- Oksijen konsantratörü

- Nebülizatör

Tüm bunlar teknolojinin ilerlemesiyle birlikte ortaya çıkan gelişmelerdir. Teknolojinin gelişmesiyle daha küçük ve maliyeti düşük hale gelen cihazlar sayesinde evde bakım hizmetleri de daha kolay uygulanabilir hale gelmektedir. Sağlık ile ilgili akademik çalışmalarda spor alanında veyahut sağlığı ilgilendiren diğer bir çok alanda olduğu gibi evde bakım hizmetleri içinde kablosuz sistemlere ilgi gün geçtikçe artmaktadır. Bu sistemleri kullanan kişilerin sağlık bilgileri uzaktan, gerçek zamanlı olarak takip edilebilmekte ve uygulamanın içeriğine göre acil durumlarda müdahale edilebilmektedir.

2.2 Sağlık Alanında Kullanılan Uygulamalar ve Çalışmalar

Teknolojinin hızlı gelişimi ile birlikte akıllı telefon ve tablet kullanımı gün geçtikçe artmaktadır. Bunun sonucunda ortaya çıkan bir çok alanda farklı amaçlara hizmet eden uygulamalar insanlar tarafından sıklıkla kullanılmaktadır. Bu alanlardan bir tanesi de tabi ki sağlık alanıdır. Ülkemizde Sağlık Bakanlığı tarafından hizmete sunulan E-Nabız ve 112 Acil Yardım Butonu gibi uygulamalar bunlardan bazılarıdır. Bunlar dışında sağlık alanında farklı amaçlara yönelik daha etkili olabilecek mobil uygulamalar ile ilgili bir çok araştırma projesi de gerçekleştirilmektedir. Örneğin, 2014 yılında ev yada hastahane ortamında çıkabilecek problemlere uzaktan, hızlı ve daha doğru bir şekilde müdahale edilebilmesi amacıyla RFID sistemlerin kullanımı, hasta izlenmesi ve veri analizi ile ilgili sistem modeli sunulmuştur [4]. Bir diğer çalışma ise 2013 yılında yapılan hastaların kendi tahlillerini kendilerinin yaparak mobil cihazında bulunan uygulama sayesinde hastane sunucusuna ve hastanın doktoruna gönderilmektedir [5].

2.3 Mobil Takip Sistemi

Araştırmasını yaptığım proje ile oluşturduğum mobil uygulama sayesinde bakıma muhtaç hastalar sensörlerden elde edilen bilgileri cep telefonu ya da tablet aracılığı ile doktorlarına anlık olarak ulaştırabilecek ve ayrıca sistem açıldığı zaman doktorun girdiği sağlık değerlerinin dışına çıkan bir ölçüm gerçekleşirse hastaya atanan doktor bilgilendirilelecektir. Bunun dışında doktor istediği

hastanın geçmiş ölçüm değerlerine ve hasta bilgilerine ulaşabilecektir. Sistemin çalışma biçimi hasta ve doktor için farklıdır. Aşağıdaki gibi özetlenebilir. Hasta Tarafı; Hasta uygulamanın yüklü olduğu cihazdan uygulamaya girişini yapar ve hangi ölçümü gerçekleştirmek istiyorsa Arduino sağlık platform ile ölçümü gerçekleştirir. Ölçüm sonrasında doktora herhangi bir şekilde sormak istediği bir soru var ise mesaj gönderebilir veya doktorunun gönderdiği mesajları okuyabilir. Sisteme kayıtlı olan hastanın ölçümleri için kritik değerler atandığı doktor tarafından belirlenir. Bu yüzden, alınan ölçüm eğer ki doktorun girdiği değerler dışında ise o zaman sistem otomatik olarak doktora bildirim göndermektedir. Doktor Tarafı; Doktor uygulamanın yüklü olduğu cihazdan uygulamaya girişini yapar. Sistem sayesinde doktor, üzerine atanan hastaların tüm bilgilerini görebilir. Gelen mesajları cevaplayabilir. Hastalarının geçmişe dönük ölçümlerini gözlemleyebilir. Sistemde kullanılan 9 farklı biyometrik sensor ile hastanın hayati önem taşıyan verileri alınabilmektedir. Bu sensörler hakkında bilgiler aşağıdaki gibidir.

Nabız ve kandaki oksijen satürasyonu– Kan basıncı, solunum, vücut ısısı ve kan basıncı değerleri hayati önem taşımaktadır. Kandaki oksijen satürasyonu da son yıllarda bu değerler arasında yerini almıştır [6]. Pulse oksimetre kullanarak oksijen satürasyonunun ölçülmesi hastanelerde standart bir uygulama haline gelmiştir.

Kalp kasılıp gevşeyerek damarlara basınç yapar. Buna nabız ya da kalbin vuruş sayısı denmektedir. Nabız ölçümü, kalbin vuruş sayısının yanı sıra kalbin düzenli çalışıp çalışmadığınıda gösteren bir yöntemdir. Hazırlanan modelde nabız ve kandaki oksijen değerleri program açıldığı zaman doktor tarafından girilecek ve takip buna göre gerçekleştirilecektir.

Kan şekeri – Evde kan şekeri ölçümü ile diyabetli hastaların hem hastanede yatma süreleri kısalacak ve yaşam kaliteleri arttırılmış olacak hem de maliyeti azaltmış olacaktır. Hastanın evde ölçüm yapması oluşabilecek ataklar ve komplikasyonların tanısı, tedavisi ve önlenmesi sürecinde de önemli rol oynar. Kan şekeri için genel olarak belirlenen değerler Çizelge 2.1'de verilmiştir [7]. Ancak hazırlanan modelde kan şekerinin olması gereken değer aralığı program açıldığı zaman doktor tarafından girilecek ve takip buna göre gerçekleştirilecektir.

Çizelge 2.1: Kan Şekeri Değerleri

	Açlık kan şekeri	2 saatlik tokluk kan şekeri
Normal	100 mg veya altında	140 mg altında
Bozulmuş glikoz toleransı (diyabet adayı)	100 - 125 mg arasında	141 – 200 mg arasında
Diyabetik	125 mg üzerinde	200 mg üzerinde

ECG – Ekektrokardiyografi, kalbin kasılma ve gevşeme esnasında ortaya çıkan elektriksel aktiviteyi ortaya çıkaran işaretlerdir. Kısaca EKG adı verilir. İşaretin gösterilmesini sağlayan cihaz adı da elektrokardiyograftır. EKG testi ile kalp kasının kasılma, ritm ve iletim bozuklukları, kasta kalınlaşma ve kalp boşluklarında genişleme gibi bir çok veri elde edilebilir [8].

EKG testi ile yeni kalp ameliyatı olmuş bir hasta ya da kalp rahatsızlığı olan bir hasta sürekli ve gerçek zamanlı olarak takip edilebilir ve alınan veriler doktora ulaştırılabilir. Hazırlanan modelde değer aralığı program açıldığı zaman doktor tarafından girilecek ve takip buna göre gerçekleştirilecektir.

Kan basıncı – Kan basıncı kontrolü bir çok hastalığın, özellikle de yüksek tansiyon (hipertansiyon), kalp ve damar hastalıklarının tanı ve tedavisi için çok önemli bir yer tutar. 2005 yılında ülkemizde yapılan PatenT araştırmasına göre yaklaşık 15 milyon yüksek tansiyon hastası vardır [9]. Yüksek tansiyon, tedavi edilebilir ancak ömürboyu tedavisi sürdürürlmesi gereken bir hastalıktır [10]. Bu nedenle, evde hastanın kendisi tarafından yapılan ölçümler ile hem hasta hem de doktor açısından hastalığın kontrolü daha kolay bir hale gelecektir. Kan basıncı hastanın cinyetine, yaşına, boyuna ve hastalıklarına bağlı olarak kişiden kişiye değişiklik gösterebilir. Bu yüzden, hazırlanan modelde alt sınır ve üst sınır program açıldığı zaman doktor tarafından girilecek ve takip buna göre gerçekleştirilecektir.

EMG – Nörolojik bir yöntem olan elektromiyogram ile vücuttaki kas ve sinirlerin elektrik aktivitesinin ölçümü gerçekleştirilir. Bir çok kas ve sinir hastalığının tanı ve tedavisi sürecinde önemli bir rol oynar.

Vücut sıcaklığı – Vücut sıcaklığı bir çok hastalığın belirlenmesi ve takibi açısından hayati bir öneme sahiptir. Diğer sensörlerden alınan bilgilerle birlikte

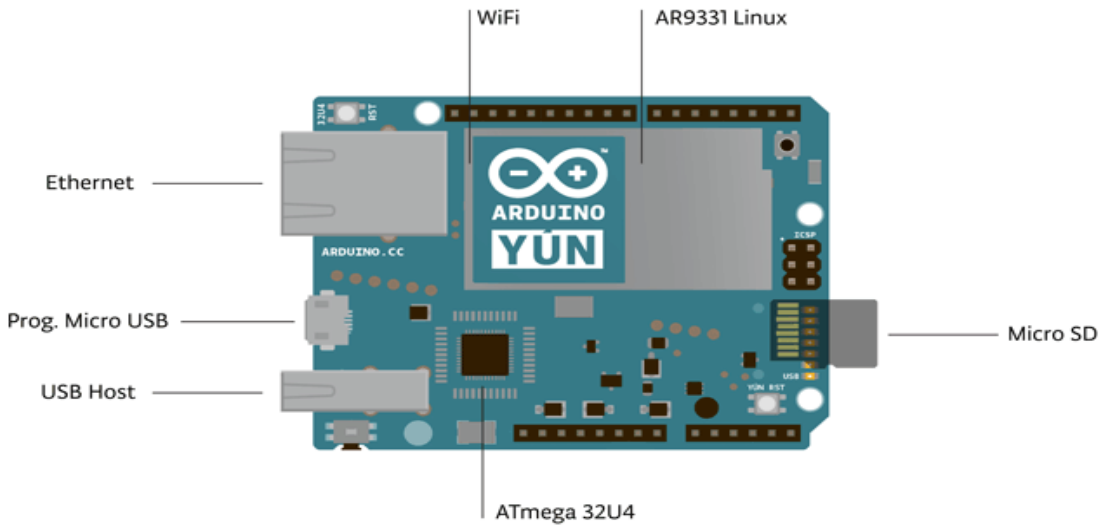
de değerlendirildiğinde bir çok farklı hastalığın tehşis ve tedavisinde önemli rol oynamaktadır. Ayrıca bunun dışında tek başına ateş olarak baş gösterdiği zaman bile vücut ısısı belli bir sıcaklığın üzerine çıktığı ya da aşağısına indiği takdirde ciddi sağlık sorunlarına neden olabilmektedir.

Galvanic deri tepkisi – Cildteki nem seviyesinin değişmesi ile birlikte cildin elektriksel direncinin değişimine bağlı olarak ölçülen deri iletkenliğine galvanik deri tepkisi denmektedir. Bu durum ter bezlerinin aktifliğine bağlıdır. Stres, heyecan ya da bunlar gibi ani duygu durum değişikliklerinde değişim gösterebileceği gibi kalp, böbrek, enfeksiyon ve tansiyon gibi bir çok hastalığın belirtilerinden bir tanesi de olabilmektedir. Bu nedenle sistemden gelen galvanik deri tepkisi ölçüm değerinin, doktor tarafından diğer veriler ile birlikte değerlendirilmesi önem arz etmektedir. Sonuç olarak hasta takibi ve uzaktan hasta kontrolü için önemli ölçümlerden bir tanesidir.

3 ARDUINO SENSÖR PLATFORMU

3.1 Arduino Yun Platformu

Arduino açık kaynak kodlu fiziksel bir programlama platformudur. Farklı uygulamalar için bir çok farklı çeşidi bulunmak ile birlikte bu tez çalışması için Arduino Yun kullanılmıştır. (Şekil3.1) Arduino Yun ATmega32u4 ve AtherosAR9331 mikroişlemcilerine sahiptir. Atheros işlemci Linux u desteklemektedir. Aynı zamanda Arduino Yun kullanılmasındaki ana sebep ise cihaz üzerinde Ethernet girişi ve Wifi bulunmasıdır. Bu şekilde proje kapsamında oluşturulan web servis ile iletişim daha kolay bir şekilde sağlanmaktadır. Proje Arduino IDE ile yazılmış ve karta yüklemesi gerçekleştirilmiştir.



Şekil 3.1: Arduino Yun

3.2 Sistemde Kullanılan Sensörler

Arduino Sensör Platformu, tıbbi uygulamalarda olduğu gibi çeşitli biyometrik uygulamaların geliştirilmesine imkan sağlamaktadır. Bu platform sayesinde,

oluşturulacak mobil uygulamada izlemesini gerçekleştirebileceğimiz sensörler ve özellikleri aşağıda sıralanmıştır. Farklı 9 adet sensör bulunmaktadır. (Şekil 3.2)

- Hava akımı (Airflow)
- Glikozölçer (Glucometer)
- Nabız ve kandaki oksijen (SPO2)
- Kan basıncı (Blood Pressure)
- Vücut ısısı (Temperature)
- Elektromyografi (EMG)
- Elektrokardiyogram (ECG)
- Galvanik vücut tepkisi (Galvanik Skin Response)
- Hasta Pozisyonu (Accelerometer)

Şekil 3.2: Arduino Sağlık Platformu

3.2.1 Kandaki oksijen ve nabız sensörü (SpO2)

Pulse oksimetre fonksiyonel hemoglobinin oksijen doygunluğunu göstermeye yarayan bir yöntemdir. Doygunluk kanda çözünen hemoglobin ve deoksihemoglobin tespitine dayanan oksijen miktarını ölçmek için kullanılır. Sensör, iki farklı dalga boyu HbO2 ve Hb spektrumlarınının gerçek farkını ölçmek için kullanılır. (Şekil 3.3)

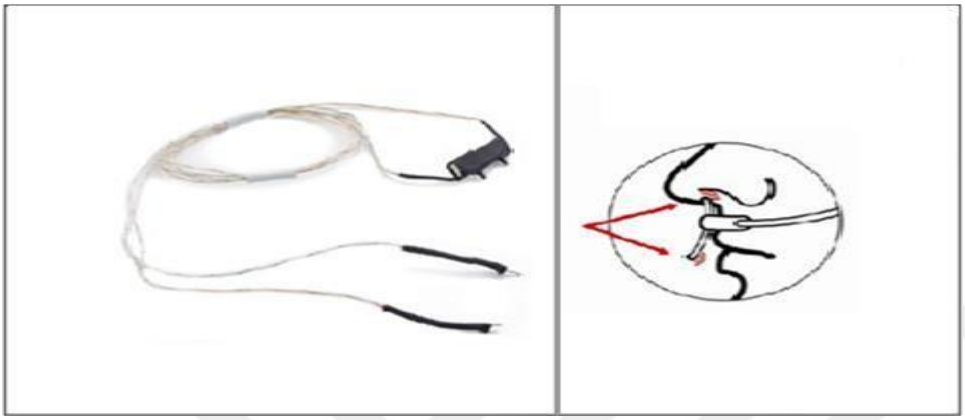
Şekil 3.3: Kandaki Oksijen ve Nabız Sensörü

Aşağıdaki kod kullanılarak, Arduino platformu ile haberleşme ve bilgi aktarımı sağlanabilir [11].

```
#include <eHealth.h>
 int cont = 0;
void setup() { Serial.begin(115200);
 eHealth.initPulsioximeter();
  PCintPort::attachInterrupt(6,
ReadPulsioximeterData,RISING);} void loop() {
  Serial.print("PRbpm : ");
  Serial.print(EHealth.getBPM());
  Serial.print(" %SPo2 : ");
  Serial.print(eHealth.getOxygenSaturation());
  Serial.print(" \n ");
  delay(500);}
void readPulsioximeter(){
  cont ++;
  if (cont == 50) {
    eHealth. readPulsioximeter();
    cont= 0; }}
```

3.2.2 Hava akımı sensörü (Nefes)

Hava akımı sensörü, solunum yardımına ihtiyacı olan hastaların hava akımı hızının ölçümü için kullanılan bir alettir. (Şekil 3.4) Cihaz, kulak arkasında uygun esnek bir iplik ve buruna yerleştirilen 2 çatal uç (sivri uç) ile hava akımını ölçer. Esnek bir yapıya sahip olduğundan dolayı kullanımı da gayet kolaydır.



Şekil 3.4: Hava Akış Sensörü

Aşağıdaki kod kullanılarak, Arduino platformu ile haberleşme ve bilgi aktarımı sağlanabilir [11].

```
#include <eHealth.h>

void setup() {

Serial.begin(11520);

void loop() {

int air = eHealth.getAirFlow();
eHealth.airFlowWave(air);
```

3.2.3 Glikozölçer sensor

Glikozölçer sensor, kandaki glikoz yoğunluğunu yaklaşık olarak hesaplamak için kullanılan bir sensördür.(Şekil 3.5) Sivri bir cisim ile deri delinerek alınan bir damla kan tek kullanımlık ölçüm çubuğu ile cihaza yerleştirilir ve kandaki glikoz seviyesi hesaplanır. Sonrasında ölçüm mg/dl ya da mmol/l olarak cihaz ekranında görülür. Cihaz ayrıca tarih ayarlarına ve kendi hafıza belleğine sahiptir.

Şekil 3.5: Glikozölçer Sensör

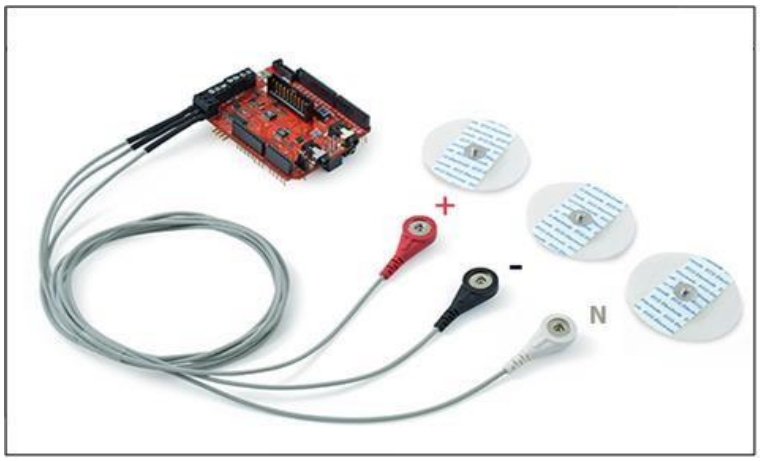
Aşağıdaki kod kullanılarak, Arduino platformu ile haberleşme ve bilgi aktarımı sağlanabilir [11].

```
#include <eHealth.h>

void setup() {

eHealth.readGlucometer();

Serial.begin(115200); delay(100);   }

void loop() {

uint8_t numberOfData
=eHealth.getGlucometerLength();

Serial.println(numberOfData, DEC); delay(100);

for (int i = 0; i<numberOfData; i++) {
Serial.print(F("Measure number   "));

Serial.println(i+1);,

 Serial.print(F("Date-    > "));

Serial.print(eHealth.glucoseDataVector[i].day));

Serial.print(F("of "));

Serial.print(eHealth.numberToMonth(eHealth.glucos
eDataVector[i].month));
```

```
Serial.print(F("of "));
Serial.print(2000 +
eHealth.glucoseDataVector[i].year);
Serial.print(F(" at "));
if (eHealth.glucoseDataVector[i].hour < 10) {
Serial.print(0); }
Serial.print(eHealth.glucoseDataVector[i].hour);
Serial.print(F(" :"));
if (eHealth.glucoseDataVector[i].minutes < 10) {
Serial.print(0);}
if (eHealth.glucoseDataVector[i].meridian ==0xBB)
Serial.print(F(" pm"));
else if
(eHealth.glucoseDataVector[i].meridian==0xAA)
Serial.print(F(" am"));
Serial.print(F(" Glucose value : "));
Serial.print(eHealth.glucoseDataVector[i].glucose);
Serial.println(F(" mg/ dl "));}
delay(20000);
```

3.2.4 Elektrokardiyogram (ECG) sensörü

Elektrokardiyogram, kalbin atışı sırasında üretilen elektrik sinyallerini ölçüp kalpteki ritim bozukluğu, damar hastalıkları ya da kalp ile ilgili diğer sorunların tespiti için kullanılan bir yöntemdir. Ölçüm sonucu alınan grafiğe Elektrokardiyogram (EKG), kullanılan cihaza ise Elektrokardiyograf denir. Kardiyak patoloji veya damar tıkanıklığından bayılma ve çarpıntıya kadar bir çok kalp aktivitesi tespit edilebildiğinden dolayı modern tıpta en çok kullanılan yöntemlerden bir tanesidir.



Şekil 3.6: Elektrokardiyogram Sensörü

Aşağıdaki kod kullanılarak, Arduino platformu ile haberleşme ve bilgi aktarımı sağlanabilir [11].

```
#include
<eHealth.h> void
setup() {
Serial.begin(1152
00); } void loop()
{

float ECG =
eHealth.getECG();
Serial.print("ECG value : ");
Serial.print(ECG, 2);

delay(10); }
```

3.2.5 Vücut ısı sensörü

Vücut ısısı, kan basıncı ve nabız gibi hayati önem taşıyan bir bulgudur. Beyin tarafından kontrol edilen vücut ısısı bir çok hastalığın başlangıcının yahut devam ediyor olduğunun göstergesidir. Bazı hastalıkların vücut ısısının değişmesi ile ilgili karakteristik özellikleri vardır ve aynı şekilde bazılarının gidişatı vücut ısısı ile izlenerek doktor tarafından gerekli tedavi uygulanabilir. Şekil 3.7' de kullanılan sensörün görseli mevcuttur.

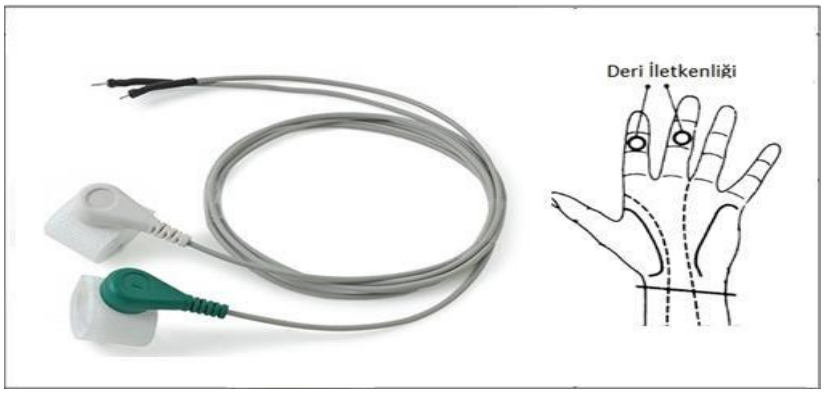
Şekil 3.7: Vücut Isı Sensörü

Aşağıdaki kod kullanılarak, Arduino platformu ile haberleşme ve bilgi aktarımı sağlanabilir [11].

```
#include <eHealth.h>

void setup() {

Serial.begin(11520);}

void loop() {

float temperature = eHealth.getTemperature();
Serial.print("temperature (ºC): ");
Serial.print(temperature, 2);

Serial.println(" "); delay(1000); }
```

3.2.6 Galvanik deri tepki sensörü

Galvanik deri tepkisi sensörü, vücudumuzun soğutma sistemi olan terlemenin ölçümü için kullanılan bir cihazdır. (Şekil 3.8) Vücut ısısının belli bir derece arasında kalması terleme ile sağlanır. Ancak bazı durumlarda aşırı terlemeler meydana gelebilir. Bu durum genetik olabileceği gibi aynı zamanda bir hastalık belirtiside olabilmektedir.



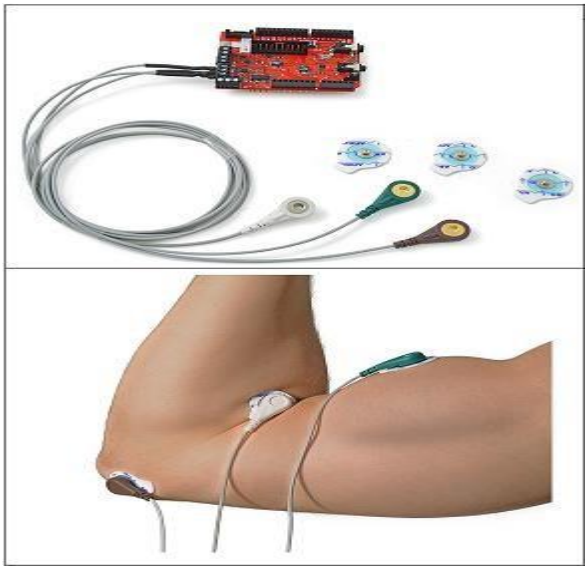
Şekil 3.8: Galvanik Tepki Sensörü

Aşağıdaki kod kullanılarak, Arduino platformu ile haberleşme ve bilgi aktarımı sağlanabilir [11].

```
#include
<eHealth.h> void
setup() {
Serial.begin(11520
0); } void loop() {
float conductance =
eHealth.getSkinConductance();
float resistance = eHealth.getSkinResistance();
float conductanceVol =
eHealth.getSkinConductanceVoltage();
Serial.println("Conductance : %f \n", conductance);
Serial.println("Resistance : %f \n", resistance);
Serial.println("Conductance Voltage : %f \n",
conductanceVol);
delay(1000); }
```

3.2.7 Elektromiyogram sensörü

Elektromiyogram, vücudumuzda bulunan kasların ve sinirlerin elektrik potansiyelinin ölçümünün yapıldığı nörolojik bir yöntemdir. Motor nöron hastalıkları ve kas hastalıkları gibi hastalıkların tanı ve tadavisi sürecinde kullanılır. Ölçüm cihazının adı elektromiyogram, grafiğine ise elektromiyografi denmektedir. (Şekil 3.9)



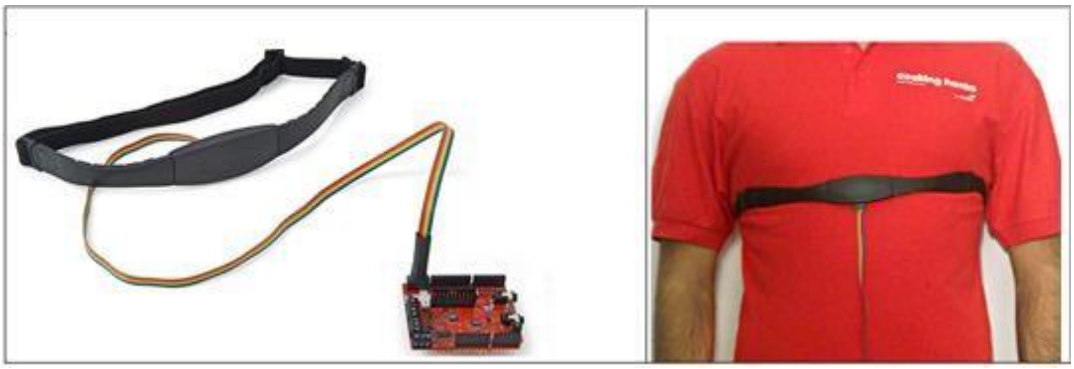
Şekil 3.9: Elektromiyogram Sensörü

Aşağıdaki kod kullanılarak, Arduino platformu ile haberleşme ve bilgi aktarımı sağlanabilir [11] .

```
#include
<eHealth.h> void
setup() {
Serial.begin(1152
00); } void loop()
{int EMG
=eHealth.getEMG
();
  Serial.print("EMG
  value: ");
  Serial.print(EMG);
  delay(100);   }
```

3.2.8 Vücut pozisyonu sensörü (ivmeölçer)

Sol ve sağ, sırtüstü, oturma ya da ayakta ve yüzüstü olmak üzere beş farklı pozisyon bu sensör ile tespit edilebilir. (Şekil 3.10) Sensör sayesinde yaşlı, engelli ya da yatalak ve bakıma muhtaç hastaların durumu ile ilgili bilgi edinme, müdahale ve kontrol sağlanabilir.



Şekil 3.10: Vücut Pozisyonu Sensörü

Aşağıdaki kod kullanılarak, Arduino platformu ile haberleşme ve bilgi aktarımı sağlanabilir [11].

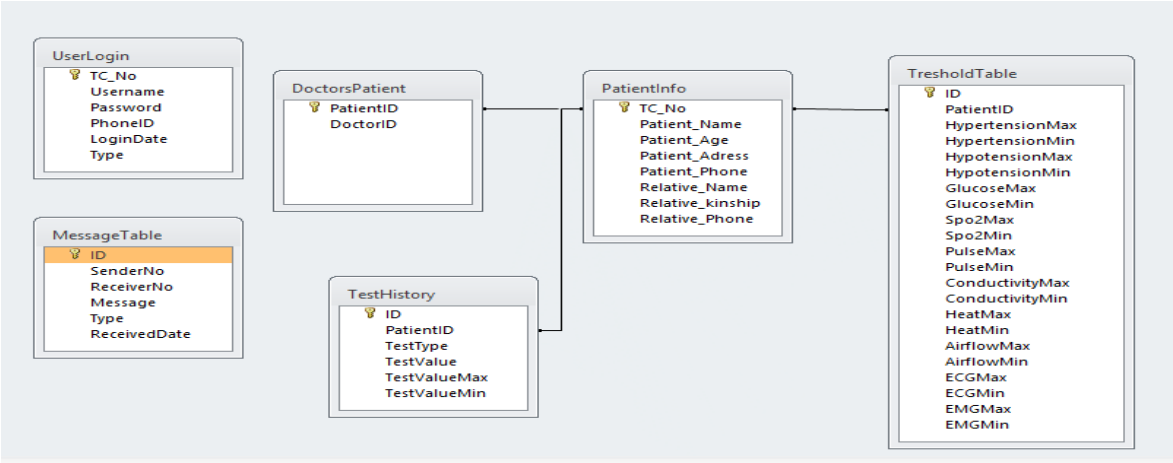
```
#include
<eHealth.h> void
setup()
{Serial.begin(115
200);
 eHealth.initPositionSensor();}
 void loop()
{Serial.print("Current position :
");
 uint8_t position =
 eHealth.getBodyPosition();
 eHealth.printPosition(position);
```

4 MOBİL TAKİP SİSTEMİ VERİTABANI TASARIMI

4.1 Veritabanı Tablolarının Tasarımı

Sistemde kullanılan EhealthDB isimli merkezi veritabanı hem kullanımı kolay hemde düşük maliyetli olmasından dolayı Microsoft Access programı ile oluşturulmuştur. Veritabanı tabloları hastaların ölçüm değerlerini, doktorlara gönderilen ölçümleri, doktor tarafından hasta için belirlenen sınır değerlerini, doktor ve hastaların bilgilerini, hasta yakınlarının bilgilerini ve hem hasta tarafından hem de doktor tarafından atılan mesaj bilgilerini tutmaktadır. Ayrıca hastanın, doktorun girdiği sınır değerleri dışında bir ölçüm sonucu gelmesi durumunda sistem tarafından doktora gönderilen bildirimlerde veritabanında tutulmaktadır. Veritabanında 6 farklı tablo bulunmaktadır. Tablo isimleri aşağıdaki gibidir. Tablo ilişkilendirmeleri Şekil 4.1'deki gibidir.

- UserLogin
- MessageTable
- DoctorsPatient
- TestHistory
- PatientInfo
- TresholdTable



Şekil 4.1: EhealthDB Veritabanı Tablo İlişkilendirmeleri

UserLogin tablosu ile uygulamaya giriş yapan kullanıcıların doktor yada hasta olup olmadığı veritabanından kontrol edilir. Eğer sistemde kayıt varsa ve giriş yapan doktorsa doktorun açılış ekranı, hasta ise hastanın açılış ekranı açılır. Tablonun diğer tablolar ile ilişkisi bulunmamakta ve aşağıdaki alanlardan oluşmaktadır.

- TC_No
- Username
- Password
- PhoneID
- LoginDate
- Type

MessageTable tablosu uygulamadaki hastaların ve doktorların birbirlerine gönderdikleri mesajları tutan tablodur. Bunun yanısıra hastanın ölçümü sonucunda eğer doktorun belirlediği kritik değerler dışına çıkan bir ölçüm var ise doktora gidecek olan bildirimler de bu tabloda tutulmaktadır. Aşağıdaki alanlardan oluşmaktadır. Bu alanlar içerisinde bulunan SenderNo ve ReceiverNo mesajı gönderen kişinin (hasta veya doktor) Tc numarasıdır. Tablonun diğer tablolar ile ilişkisi bulunmamaktadır.

- ID
- SenderNo

- ReceiverNo
- Message
- Type
- ReceivedDate

DoctorsPatient tablosu doktor ve hastaların ID lerini tutan tablodur. Hangi hastanın hangi doktora atandığı bilgilerini içerir. PatientID alanı ile PatientInfo tablosundaki TC_No alanı ile ilişkilidir. Ve yine PatientID alanı ile TextHistory tablosundaki ID ile ilişkilidir.

- PatientID
- DoctorID

Test History tablosu hastanın önceki ölçüm bilgilerini tutan tablodur. Tablo hastanın daha önceden gerçekleştirmiş olduğu ölçüm değerlerini, doktorun o ölçüm için belirlemiş olduğu kritik değer aralıklarını ve ölçüm tarih ve saatini tutmaktadır. Tablo alan isimleri aşağıdaki gibidir.

- ID
- PatientID
- TestType
- TestValue
- TestValueMax
- TestValueMin
- TestDate

PatientInfo tablosu hastanın bilgilerini tutan tablodur. TC_No, isim, yaş, adres, telefon, hasta yakını ismi, yakınlık derecesi ve yakının numarasını tutar. Alan isimleri aşağıdaki gibidir.

- TC_No
- Patient_Name
- Patient_Age

- Patient_Adress
- Patient_Phone
- Relative_Name
- Relative_kindship
- Relative_phone

TresholdTable tablosu ile hastanın doktor tarafından belirlenen kritik değerleri tutulur. Alan isimleri aşağıdaki gibidir.

- ID
- PatientID
- HypertensionMax
- HypertensionMin
- HypotensionMax
- HypotensionMin
- GlucoseMax
- GlucoseMin
- Spo2Max
- Spo2Min
- PulseMax
- PulseMin
- ConductivityMax
- ConductivityMin
- HeatMax
- HeatMin
- AirflowMax
- AirflowMin
- ECGMax

- ECGMin
- EMGMax
- EMGMin

4.2 Veritabanı ve Android Uygulama Arasındaki Veri İşlemleri

Sistemde Microsoft Access ile oluşturulan EhealthDB isimli merkezi veritabanı ile Android uygulama arasındaki veri alışverişi .Net Framework 4.5 de C# programlama dili ile yazılan Web Servis aracılığı ile gerçekleştirilmektedir. Web servis içerisine aktarılan EhealthDB ile Get ve Post metodları, Android tarafındanda çağırılarak istemci/sunucu mimarisine dayanan bir iletişim gerçekleşir. Web servis metodları içerisinde her fonksiyona ait (Android uygulamanın içerisinde bulunan her aktivitede gerçekleşecek işlem için) ayrı ayrı Sql kodları bulunmaktadır. Bu şekilde ekleme, güncelleme, kaydetme ve ölçme işlemleri gerçekleşir. Web Seviste kullanılan metodlar ve oluşturulma amaçları aşağıdaki gibidir.

- UserLogin
- LoggedUser
- DoctorPatients
- MessageToDoctor
- DoctorNotifications
- MessageToPatient
- SendMessageToDoctor
- SendMessageToPatient
- PatientInformation
- PatientTestHistoryByDoctor
- PatientTestHistoryByPatient
- TresholdValues
- SaveTresholdValues

UserLogin ile merkezi veritabanında kayıtlı olan kullanıcıların bilgileri kontrol edilerek eğer kullanıcı veritabanında kayıtlı ise son giriş tarihleri ile birlikte hangi telefon ile giriş yapılmışsa telefonun id si kaydı gerçekleştirilir. Eğer yirmidört saat içerisinde kullanıcı giriş yaptıysa LoggedUser metodu çalışır.

```
[HttpPost]
    public UserLoginResponse UserLogin([FromBody]UserLoginRequest
UserLogin)
    {
      connection.Open();
      OleDbCommand cmd = new OleDbCommand(@"Update UserLogin
                                  Set PhoneID = "
                                  Where PhoneID = @PhoneID", connection);
      cmd.Parameters.AddWithValue("@PhoneID", UserLogin.PhoneID);
      cmd.ExecuteNonQuery();
      cmd = new OleDbCommand("Select * From UserLogin", connection);
      OleDbDataReader reader = cmd.ExecuteReader();
      while (reader.Read())
      {
        if (reader["Username"].ToString() == UserLogin.Username &&
reader["Password"].ToString() == UserLogin.Password)
          {
            cmd = new OleDbCommand("Update UserLogin Set PhoneID =
@PhoneID, LoginDate = @LoginDate Where Username = @Username",
connection);
            cmd.Parameters.AddWithValue("@PhoneID",
UserLogin.PhoneID);
            cmd.Parameters.AddWithValue("@LoginDate",
DateTime.Now.AddDays(1).ToString("dd.MM.yyyy HH:mm:ss"));
```

```
                cmd.Parameters.AddWithValue("@Username",
UserLogin.Username);
                cmd.ExecuteNonQuery();
                string type = reader["Type"].ToString();
                connection.Close();
                return new UserLoginResponse
                {
                   PhoneID = UserLogin.PhoneID,
                   Type = type
                };
             }
          }
          connection.Close();
          return new UserLoginResponse
             {
                PhoneID = "false"
             };
       }
```

LoggedUser ile kullanıcı daha önceden o telefon ile giriş yapmış ise kaydedilen telefon id si ve önceki giriş saati kontrol edilir. Kullanıcı eğer yirmi dört saat içerisinde daha önce uygulamayı kullanmış ise tekrardan kullanıcı adı ve şifresi istenmeyerek kullanıcının direk olarak uygulamayı açabilmesi sağlanır.

```
[HttpPost]
      public LoggedUserResponse
LoggedUser([FromBody]LoggedUserRequest LoggedUser)
      {
         connection.Open();
```

```
        OleDbCommand cmd = new OleDbCommand("Select * From
UserLogin", connection);
        OleDbDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {
            DateTime LoginDate = Convert.ToDateTime(reader["LoginDate"]);
            DateTime CurrentTime = DateTime.Now;
            if (reader["Username"].ToString() == LoggedUser.Username &&
reader["PhoneID"].ToString() == LoggedUser.PhoneID && LoginDate >
CurrentTime)
            {
                string phoneid = reader["PhoneID"].ToString();
                string type = reader["Type"].ToString();
                connection.Close();
                return new LoggedUserResponse
                {
                    PhoneID = phoneid,
                    Type = type
                };
            }
        }
        connection.Close();
        return new LoggedUserResponse
        {
            PhoneID = "false"
        };
    }
```

DoctorPatients ile hangi hastanın hangi doktora atandığı kontrol edilir. Bu şekilde doktor uygulamayı açtığı zaman üzerine atanan hastaları hastalarım bölümünden görebilecektir.

```
[HttpPost]
    public MyPatientsResponse
DoctorPatients([FromBody]MyPatientsRequest MyPatientsRequest)
    {
        connection.Open();
        OleDbCommand cmd = new OleDbCommand("Select d.PatientID,
p.Patient_Name From DoctorsPatient d, PatientInfo p, UserLogin u where
d.PatientID = p.TC_No and d.DoctorID = u.TC_No and u.PhoneID =
@PhoneID", connection);
        cmd.Parameters.AddWithValue("@PhoneID",
MyPatientsRequest.PhoneID);
        OleDbDataReader reader = cmd.ExecuteReader();
        MyPatientsResponse Response = new MyPatientsResponse();
        Response.Response = new List<MyPatientsListItem>();
        while (reader.Read())
        {
            Response.Response.Add(new MyPatientsListItem
            {
                PatientTcNo = reader["PatientID"].ToString(),
                PatientName = reader["Patient_Name"].ToString()
            });
        }
        connection.Close();
        return Response;
    }
```

MessageToDoctor ile doktora önceden gelmiş olan mesajların kaydı tutulur.

```
[HttpPost]
    public MessageToDoctorResponse
MessageToDoctor([FromBody]MessageToDoctorRequest MessageToDoctor)
    {
      connection.Open();
      OleDbCommand cmd = new OleDbCommand(@"Select
p.Patient_Name, m.Message, m.SenderNo, m.ReceivedDate
                                From MessageTable m, PatientInfo p,
UserLogin u
                                where m.SenderNo = p.TC_No
                                 and m.Type = 'message'
                                 and m.ReceiverNo = u.TC_No
                                 and u.PhoneID = @PhoneID
                                order by m.ReceivedDate desc", connection);
      cmd.Parameters.AddWithValue("@PhoneID",
MessageToDoctor.PhoneID);
      OleDbDataReader reader = cmd.ExecuteReader();
      MessageToDoctorResponse Response = new
MessageToDoctorResponse();
      Response.Response = new List<MessageToDoctorListItem>();
      while (reader.Read())
      {
        Response.Response.Add(new MessageToDoctorListItem
        {
          PatientTcNo = reader["SenderNo"].ToString(),
```

```
                PatientName = reader["Patient_Name"].ToString(),
                Message = reader["Message"].ToString(),
                ReceivedDate =
Convert.ToDateTime(reader["ReceivedDate"]).ToString("dd.MM.yyyy
HH:mm")
            });
        }
        connection.Close();
        return Response;
    }
```

DoctorNotification ile hastanın doktor tarafından belirlenen kritik değerleri dışına çıkan herhangi bir ölçüm var ise kontrol edilir. Bu sayede doktorun bildirimler bölümüne mesaj düşer.

```
[HttpPost]
    public DoctorNotificationResponse
DoctorNotification([FromBody]DoctorNotificationRequest
DoctorNotification)
    {
        connection.Open();
        OleDbCommand cmd = new OleDbCommand(@"Select
p.Patient_Name, m.Message, m.SenderNo, m.ReceivedDate
                                From MessageTable m, PatientInfo p,
UserLogin u
                                where m.SenderNo = p.TC_No
                                  and m.Type = 'notification'
                                  and m.ReceiverNo = u.TC_No
                                  and u.PhoneID = @PhoneID
                                order by m.ReceivedDate desc", connection);
```

```
            cmd.Parameters.AddWithValue("@PhoneID",
        DoctorNotification.PhoneID);
            OleDbDataReader reader = cmd.ExecuteReader();
            DoctorNotificationResponse Response = new
        DoctorNotificationResponse();
            Response.Response = new List<DoctorNotificationListItem>();
            while (reader.Read())
            {
               Response.Response.Add(new DoctorNotificationListItem
               {
                  PatientTcNo = reader["SenderNo"].ToString(),
                  PatientName = reader["Patient_Name"].ToString(),
                  Message = reader["Message"].ToString(),
                  ReceivedDate =
        Convert.ToDateTime(reader["ReceivedDate"]).ToString("dd.MM.yyyy
        HH:mm")
               });
            }
            connection.Close();
            return Response;
        }
```

MessageToPatient ile hastaya önceden gelmiş olan mesajların kaydı tutulur.

```
        [HttpPost]
        public MessageToPatientResponse
        MessageToPatient([FromBody]MessageToPatientRequest MessageToDoctor)
        {
            connection.Open();
```

```
        OleDbCommand cmd = new OleDbCommand(@"Select m.Message,
m.SenderNo, m.ReceivedDate
                                        From MessageTable m, UserLogin u
                                        where m.Type = 'message'
                                          and m.ReceiverNo = u.TC_No
                                          and u.PhoneID = @PhoneID
                                        order by m.ReceivedDate desc", connection);
        cmd.Parameters.AddWithValue("@PhoneID",
MessageToDoctor.PhoneID);
        OleDbDataReader reader = cmd.ExecuteReader();
        MessageToPatientResponse Response = new
MessageToPatientResponse();
        Response.Response = new List<MessageToPatientListItem>();
        while (reader.Read())
        {
           Response.Response.Add(new MessageToPatientListItem
           {
              DoctorTcNo = reader["SenderNo"].ToString(),
              Message = reader["Message"].ToString(),
              ReceivedDate =
Convert.ToDateTime(reader["ReceivedDate"]).ToString("dd.MM.yyyy
HH:mm")
           });
        }
        connection.Close();
        return Response;
     }
```

SendMessageToDoctor ile doktora gönderilen mesajların merkezi veritabanına eklenmesi sağlanır.

```
[HttpPost]
    public SendMessageToDoctorResponse
SendMessageToDoctor([FromBody]SendMessageToDoctorRequest
MessageData)
    {
      connection.Open();
      try
      {
        OleDbCommand cmd = new OleDbCommand(@"Select TC_No
                          From UserLogin
                          where PhoneID = @PhoneID", connection);
        cmd.Parameters.AddWithValue("@PhoneID",
MessageData.PhoneID);
        OleDbDataReader reader = cmd.ExecuteReader();
        reader.Read();
        string SenderID = reader["TC_No"].ToString();
        cmd = new OleDbCommand(@"Select DoctorID
                    From DoctorsPatient
                    where PatientID = @SenderID", connection);
        cmd.Parameters.AddWithValue("@SenderID", SenderID);
        reader = cmd.ExecuteReader();
        reader.Read();
        string ReceiverID = reader["DoctorID"].ToString();
        cmd = new OleDbCommand(@"Insert into MessageTable
(SenderNo,ReceiverNo,Message,Type,ReceivedDate)
```

```
Values(@SenderID,@ReceiverID,@Message,'message',@ReceivedDate)",
connection);
            cmd.Parameters.AddWithValue("@SenderID", SenderID);
            cmd.Parameters.AddWithValue("@ReceiverID", ReceiverID);
            cmd.Parameters.AddWithValue("@Message",
MessageData.Message);
            cmd.Parameters.AddWithValue("@ReceivedDate",
DateTime.Now.ToString("dd.MM.yyyy HH:mm"));
            cmd.ExecuteNonQuery();
            return new SendMessageToDoctorResponse
            {
                Response = "true"
            };
        }
        catch (Exception e)
        {
            return new SendMessageToDoctorResponse
            {
                Response = "false"
            };
        }
    }
```

SendMessageToPatient ile hastaya gönderilen mesajların merkezi veritabanına eklenmesi sağlanır.

```
[HttpPost]
    public SendMessageToPatientResponse
SendMessageToPatient([FromBody]SendMessageToPatientRequest
MessageData)
```

```
    {
        connection.Open();
        try
        {
            OleDbCommand cmd = new OleDbCommand(@"Select TC_No
                                    From UserLogin
                                    where PhoneID = @PhoneID", connection);
            cmd.Parameters.AddWithValue("@PhoneID",
MessageData.PhoneID);
            OleDbDataReader reader = cmd.ExecuteReader();
            reader.Read();
            string SenderID = reader["TC_No"].ToString();
            string ReceiverID = MessageData.PatientID;
            cmd = new OleDbCommand(@"Insert into MessageTable
 (SenderNo,ReceiverNo,Message,Type,ReceivedDate)
Values(@SenderID,@ReceiverID,@Message,'message',@ReceivedDate)",
connection);
            cmd.Parameters.AddWithValue("@SenderID", SenderID);
            cmd.Parameters.AddWithValue("@ReceiverID", ReceiverID);
            cmd.Parameters.AddWithValue("@Message",
MessageData.Message);
            cmd.Parameters.AddWithValue("@ReceivedDate",
DateTime.Now.ToString("dd.MM.yyyy HH:mm"));
            cmd.ExecuteNonQuery();
            return new SendMessageToPatientResponse
            {
               Response = "true"
```

```
            };
        }
        catch (Exception e)
        {
            return new SendMessageToPatientResponse
            {
                Response = "false"
            };
        }
        finally
        {
            connection.Close();
        }
    }
```

PatientInformation ile hastaların merkezi veritabanında bulunan PatientInfo isimli tablosundaki bilgileri seçilir.

```
[HttpPost]
    public PatientInformationResponse
PatientInformation([FromBody]PatientInformationRequest PatientInformation)
    {
        PatientInformationResponse Response = new
PatientInformationResponse();
        connection.Open();
        OleDbCommand cmd = new OleDbCommand(@"Select *
                                From PatientInfo
                                where TC_No = @PatientID", connection);
```

```
            cmd.Parameters.AddWithValue("@PatientID",
        PatientInformation.PatientID);
            OleDbDataReader reader = cmd.ExecuteReader();
            reader.Read();
            Response.TC_No= reader["TC_No"].ToString();
            Response.Patient_Name = reader["Patient_Name"].ToString();
            Response.Patient_Age = reader["Patient_Age"].ToString();
            Response.Patient_Adress = reader["Patient_Adress"].ToString();
            Response.Patient_Phone = reader["Patient_Phone"].ToString();
            Response.Relative_Name = reader["Relative_Name"].ToString();
            Response.Relative_kinship = reader["Relative_kinship"].ToString();
            Response.Relative_Phone = reader["Relative_Phone"].ToString();
            connection.Close();
            return Response;
        }
```

PatientTestHistoryByDoctor ile hastanın önceki ölçüm değerlerinin merkezi veritabanındaki TestHistory tablosundan seçimi ve uygulamanın doktor tarafından gösterimi sağlanır.

```
    [HttpPost]
        public PatientTestHistoryByDoctorResponse
    PatientTestHistoryByDoctor([FromBody]PatientTestHistoryByDoctorRequest
    PatientTestHistoryByDoctor)
        {
            PatientTestHistoryByDoctorResponse Response = new
    PatientTestHistoryByDoctorResponse();
            Response.Response = new
    List<PatientTestHistoryByDoctorListItem>();
            connection.Open();
```

```
            OleDbCommand cmd = new OleDbCommand(@"Select *
                                                From TestHistory t, UserLogin u
                                                where t.PatientID = u.TC_No
                                                 and t.PatientID = @PatientID", connection);
            cmd.Parameters.AddWithValue("@PatientID",
        PatientTestHistoryByDoctor.PatientID);
            OleDbDataReader reader = cmd.ExecuteReader();
            while (reader.Read())
            {
               Response.Response.Add(new PatientTestHistoryByDoctorListItem
                {
                    Patient_Name = reader["Username"].ToString(),
                    TestType = reader["TestType"].ToString(),
                    TestValue = reader["TestValue"].ToString(),
                    TestValueMax = reader["TestValueMax"].ToString(),
                    TestValueMin = reader["TestValueMin"].ToString(),
                    TestDate =
        Convert.ToDateTime(reader["TestDate"]).ToString("dd.MM.yyyy HH:mm")
                });
            }
            connection.Close();
            return Response;
        }
```

PatientTestHistoryByPatient ile hastanın önceki ölçüm değerlerinin merkezi veritabanındaki TestHistory tablosundan seçimi ve uygulamanın hasta tarafından gösterimi sağlanır.

```
[HttpPost]
    public PatientTestHistoryByPatientResponse
PatientTestHistoryByPatient([FromBody]PatientTestHistoryByPatientRequest
PatientTestHistoryByPatient)
    {
        PatientTestHistoryByPatientResponse Response = new
PatientTestHistoryByPatientResponse();
        Response.Response = new
List<PatientTestHistoryByPatientListItem>();
        connection.Open();
        OleDbCommand cmd = new OleDbCommand(@"Select *
                                From TestHistory t, UserLogin u
                                where t.PatientID = u.TC_No
                                 and u.PhoneID = @PhoneID
                                Order by TestDate Desc", connection);
        cmd.Parameters.AddWithValue("@PhoneID",
PatientTestHistoryByPatient.PhoneID);
        OleDbDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {
          Response.Response.Add(new  PatientTestHistoryByPatientListItem
          {
            Patient_Name = reader["Username"].ToString(),
            TestType = reader["TestType"].ToString(),
            TestValue = reader["TestValue"].ToString(),
            TestValueMax = reader["TestValueMax"].ToString(),
            TestValueMin = reader["TestValueMin"].ToString(),
```

```
                TestDate =
Convert.ToDateTime(reader["TestDate"]).ToString("dd.MM.yyyy HH:mm")
            });
        }
        connection.Close();
        return Response;
    }
```

TresholdValues ile hastanın doktor tarafından daha önceden belirlenmiş kritik değerlerinin seçimi sağlanır.

```
[HttpPost]
    public TresholdValuesResponse
TresholdValues([FromBody]TresholdValuesRequest TresholdValues)
    {
        connection.Open();
        string query = @"Select * From TresholdTable
                        where PatientID = @PatientID";
        query = query.Replace("@PatientID","'" + TresholdValues.PatientID +
"'");
        OleDbDataAdapter adapter = new OleDbDataAdapter(query,
connection);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        connection.Close();
        return new TresholdValuesResponse
        {
            Response = DatatableToJsonSerialize(dt)
        };
```

SaveTresholdValues ile hastanın kritik değerlerinde doktor tarafından herhangi bir değişiklik yapıldığı zaman merkezi veritabanındaki TreholdTable tablosundaki değerler yerine yenilerinin kaydı sağlanır.

```
[HttpPost]
    public SaveTresholdValuesResponse
SaveTresholdValues([FromBody]SaveTresholdValuesRequest
TresholdValues)
    {
      try
      {
        string query = @"Update TresholdTable
                          Set HypertensionMax = '@HypertensionMax',
                          HypertensionMin = '@HypertensionMin',
                          HypotensionMax = '@HypotensionMax',
                          HypotensionMin = '@HypotensionMin',
                          GlucoseMax = '@GlucoseMax',
                          GlucoseMin = '@GlucoseMin',
                          Spo2Max = '@Spo2Max',
                          Spo2Min = '@Spo2Min',
                          PulseMax = '@PulseMax',
                          PulseMin = '@PulseMin',
                          ConductivityMax = '@ConductivityMax',
                          ConductivityMin = '@ConductivityMin',
                          HeatMax = '@HeatMax',
                          HeatMin = '@HeatMin',
                          AirflowMax = '@AirflowMax',
```

```
                    AirflowMin = '@AirflowMin',
                    ECGMax = '@ECGMax',
                    ECGMin = '@ECGMin',
                    EMGMax = '@EMGMax',
                    EMGMin = '@EMGMin'
                    Where PatientID = '@PatientID'";
            query = query.Replace("@PatientID", TresholdValues.PatientID);
            query = query.Replace("@HypertensionMax",
TresholdValues.HypertensionMax);
            query = query.Replace("@HypertensionMin",
TresholdValues.HypertensionMin);
            query = query.Replace("@HypotensionMax",
TresholdValues.HypotensionMax);
            query = query.Replace("@HypotensionMin",
TresholdValues.HypotensionMin);
            query = query.Replace("@GlucoseMax",
TresholdValues.GlucoseMax);
            query = query.Replace("@GlucoseMin",
TresholdValues.GlucoseMin);
            query = query.Replace("@Spo2Max", TresholdValues.Spo2Max);
            query = query.Replace("@Spo2Min", TresholdValues.Spo2Min);
            query = query.Replace("@PulseMax", TresholdValues.PulseMax);
            query = query.Replace("@PulseMin", TresholdValues.PulseMin);
            query = query.Replace("@ConductivityMax",
TresholdValues.ConductivityMax);
            query = query.Replace("@ConductivityMin",
TresholdValues.ConductivityMin);
            query = query.Replace("@HeatMax", TresholdValues.HeatMax);
```

```
            query = query.Replace("@HeatMin", TresholdValues.HeatMin);
            query = query.Replace("@AirflowMax",
TresholdValues.AirflowMax);
            query = query.Replace("@AirflowMin",
TresholdValues.AirflowMin);
            query = query.Replace("@ECGMax", TresholdValues.ECGMax);
            query = query.Replace("@ECGMin", TresholdValues.ECGMin);
            query = query.Replace("@EMGMax", TresholdValues.EMGMax);
            query = query.Replace("@EMGMin", TresholdValues.EMGMin);
            connection.Open();
            OleDbCommand cmd = new OleDbCommand(query, connection);
            cmd.ExecuteNonQuery();
            connection.Close();
        }
        catch (Exception e)
        {
            connection.Close();
            return new SaveTresholdValuesResponse
            {
                Response = "false"
            };
        }
        return new SaveTresholdValuesResponse
        {
            Response = "true"
        };
    }
```

4.3 Android Uygulaması Veritabanı İşlemleri

Android'in kendi veritabanı olan SQLite ile yazılmıştır. Sadece hasta yada doktorun kullanıcı adı, şifre ve telefon id sini tutmaktadır. Bu şekilde yeni giriş yapan ya da daha önceden giriş yapmış olan kullanıcıların kayıtları tutulur. Kullanıcının ilk girişi esnasında istenen kullanıcı adı ve şifre telefon id si ile birlikte veritabanına kaydedilir. Bunun sonucunda kullanıcı yirmi dört saat içerisinde uygulamaya tekrar giriş yapması halinde bu sefer kullanıcıdan kullanıcı adı ve şifresi istenmemekte ve direk olarak uygulamaya giriş yapabilmektedir. Bütün bu işlemler Android de yazılan aşağıda kodları verilmiş olan DataBase class ı içerisinde yapılmaktatır. Sonrasında SigninPage class ı içerisinde çağırılarak kontrol edilip Bölüm 4.2'de belirtilen UserLogin ve LoggedUser web servis fonksiyonları çağırılarak kontrolü sağlanır.

```
public class DataBase extends SQLiteOpenHelper
{
   private boolean signin;
   private String username;
   private final static String DBAdi="EHealth";
   private final static int surum=1;
   private final static String tablo="USers";
   private final static String columnName="Username";
   private ArrayList<ArrayList<String>>select=new ArrayList<>();


   @Override
   public void onCreate(SQLiteDatabase db) {
      db.execSQL("create table USers (ID integer primary key autoincrement,
Username String)");
   }
   @Override
   public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
      db.execSQL("drop table if exist USers");
```

```
    onCreate(db);
}
public DataBase(Context context)
{
    super(context, DBAdi, null, surum); }
public String select()
{
    SQLiteDatabase db=this.getReadableDatabase();
    Cursor cursor=null;
    String query="select * from USers";
    cursor=db.rawQuery(query, null);
    if(cursor.moveToFirst())
    {
        return cursor.getString(1);
    }
    cursor.close();
    db.close();
    return "false";}
public void insert(String username)
{SQLiteDatabase db=this.getWritableDatabase();
    String query="insert into USers (Username) values('"+username+"') ";
    db.execSQL(query);
    Log.d("inserted", String.valueOf(select()));
    db.close();
}  public void delete(String gelenId)
{
    SQLiteDatabase db=this.getWritableDatabase();
    String query="delete from USers where Username = '"+gelenId+"' ";
    db.execSQL(query); }
public boolean login(String user)
{
    if (select().length()>0)
    {
        delete(select());
```

```
        Log.d("empty","Empty"+select());
        insert(user);
        return true; }
    else
    {return false;
    }
public void logout()
{
    delete(select());}
public boolean isSignin()
{if (select().length()>0)
    {
        Log.d("user", ""+String.valueOf(select()));
        return true;}
    else
    {
        Log.d("boş", ""+String.valueOf(select()));
        return false;
    }}
public String getUsername() {
    if(select().length()>0)
    {Log.d("username",""+username);
        username=select();}
    return username;
```

5 MOBİL TAKİP SİSTEMİ MODELİNİN GELİŞTİRİLMESİ

5.1 Uygulamanın Arayüz Tasarımı

Mobil takip sistemi uygulaması Android Studio ile geliştirilmiştir. Uygulamanın veritabanı ve Android ile bağlantısı Bölüm 4'de açıklandığı gibi Web servis aracılığı ile internet üzerinden sağlanmaktadır. Ancak sisteme kaydedilen verilerin kontrolü Android uygulaması tarafından gerçekleştirilip veritabanına yansımaktadır. Uygulama içerisinde kullanılan ekranlar aşağıda açıklamaları ile birlikte verilmiştir.

Sisteme giriş için hem doktor hemde hasta tarafından kullanılan ekran aşağıdaki gibidir. (Şekil 5.1) Bu ekranda kullanıcı adı ve şifre ile veritabanına iletişim sağlanarak sisteme giriş yapılmaktadır.

Şekil 5.1: Sisteme Giriş Ekranı

Sistem giriş yapan kullanıcıya göre iki bölümden oluşmaktadır. Bunlardan bir tanesi hasta içinken diğeri doktor için oluşturulmuştur.

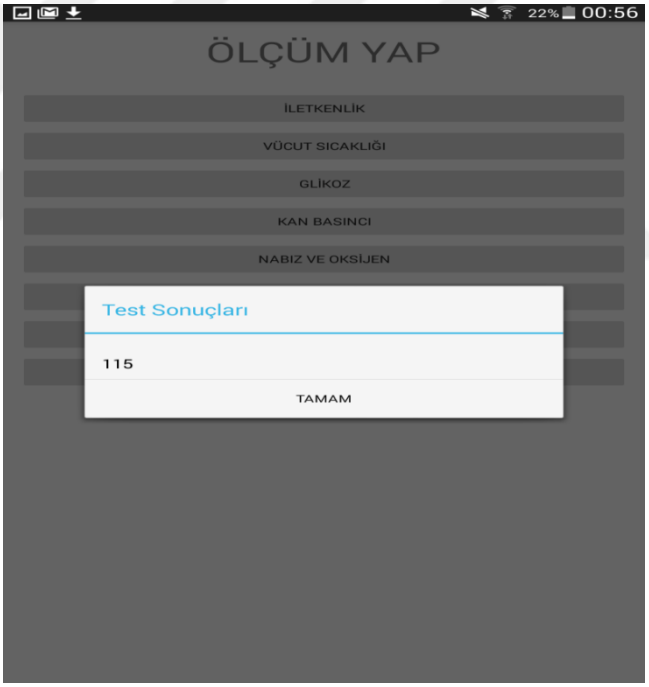
5.1.1 Hasta İşlemleri Modülü

Hastanın ölçümlerini yapabildiği, doktordan gelen mesajlarını okuyabildiği, geçmiş ölçümlerini görebildiği ve doktoruna mesaj gönderebildiği ana ekran bulunmaktadır. (Şekil 5.2)

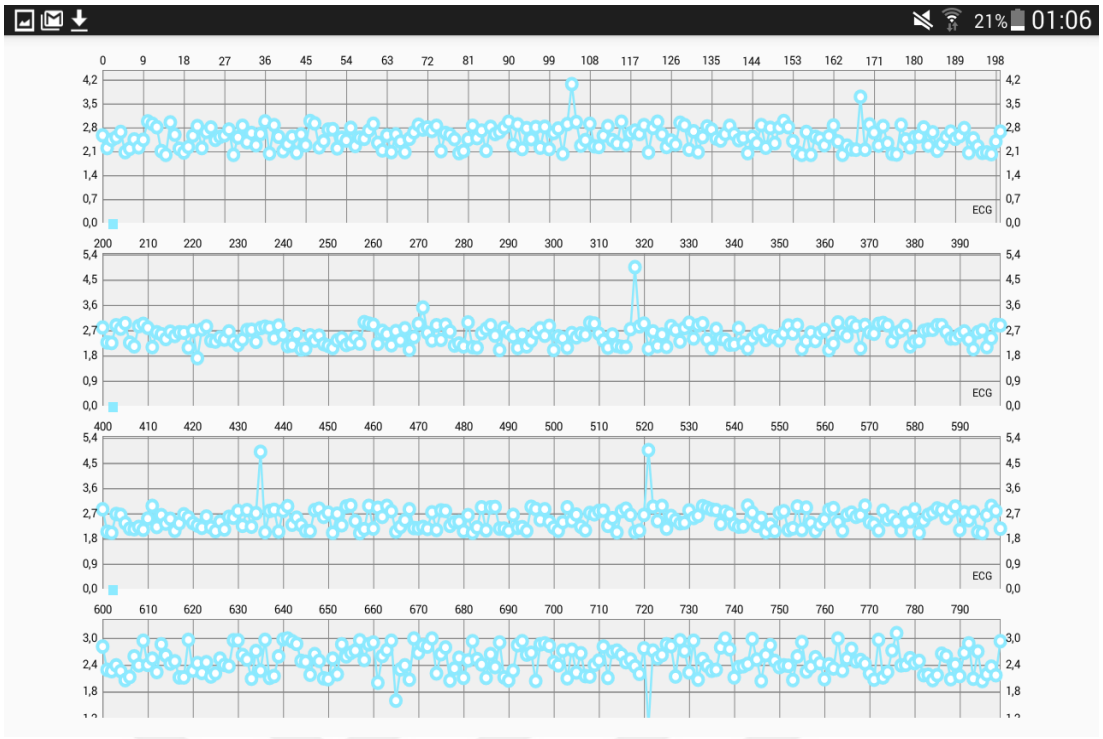
Şekil 5.2: Hasta Açılış Ekranı

Hasta eğer ölçüm yapmak istiyorsa Ölçüm Yap butonuna tıklar ve Şekil 5.3'de ki ekran açılır. Hangi ölçümü yapmak istiyorsa cihazı takar ve ölçüm yapmak istediği butona basar. İletkenlik, vücut sıcaklığı, glikoz, kan basıncı (yüksek ve düşük tansiyon), nabız ve oksijen ve nefes ölçümleri tek bir değer vermekteyken EKG ve EMG değerleri chart olarak gelmektedir. Tek bir değer olarak örneğin glikoz değeri alınmak isteniyorsa butona tıklanır ve karşımıza Şekil 5.4' de ki test onucu ekranı gelmektedir. Ancak EKG ve EMG değerleri sırasıyla Şekil 5.5 ve Şekil 5.6' da ki gibi gelmektedir.

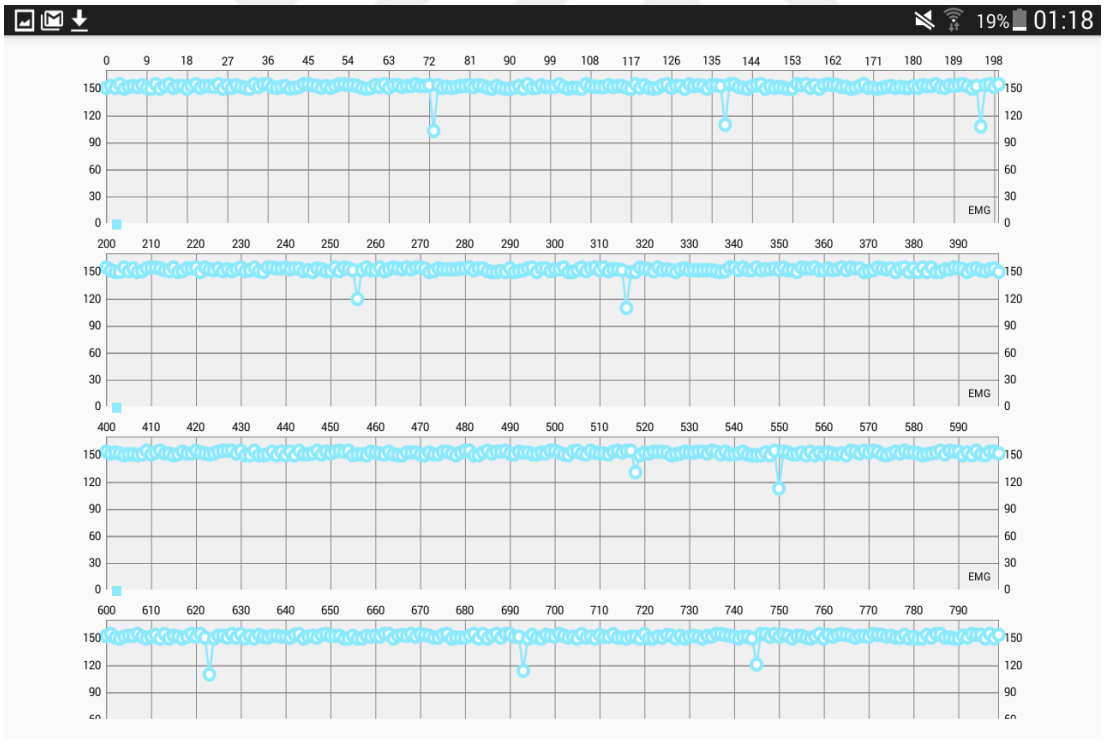
Şekil 5.3 : Ölçüm Yap Ekranı



Şekil 5.4: Glikoz Ölçümü Test Sonucu

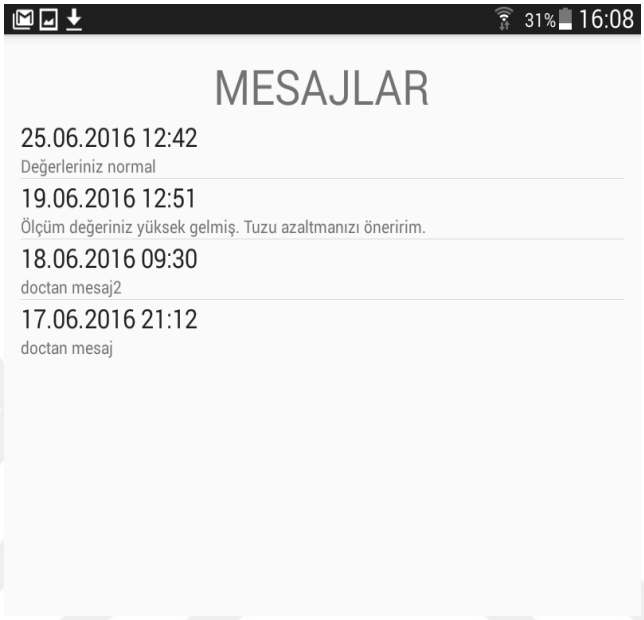


Şekil 5.5: EKG Test Sonucu

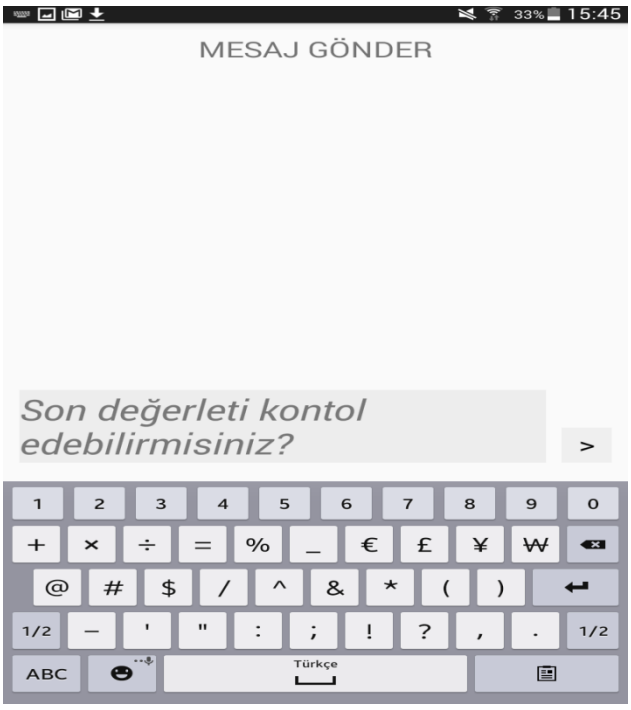


Şekil 5.6: EMG Test Sonucu

Mesajlarım bölümü doktor tarafından hastaya gelen mesajları hastanın görebilmesi amacıyla oluşturulmuştur. (Şekil 5.7) Benzer şekilde doktoruma mesaj bölümü de hastanın danışmak ya da paylaşmak istediği herhangi bir şey bulunması durumunda doktoruna mesaj gönderebilmesi amacıyla oluşturulmuştur. (Şekil 5.8)



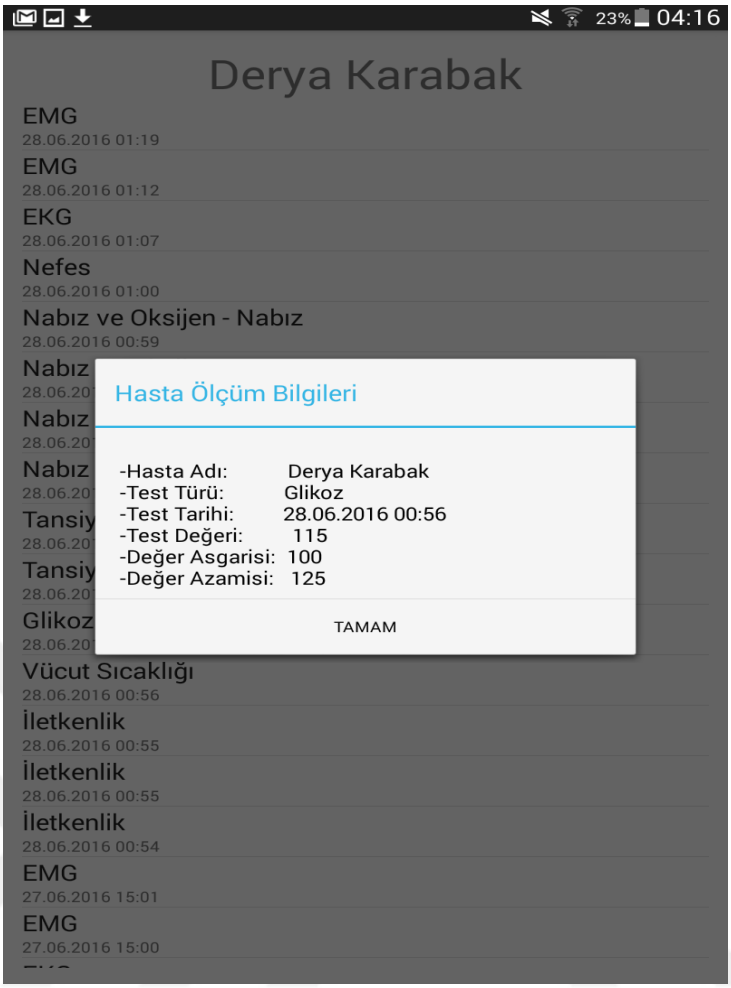
Şekil 5.7: Doktordan Hastaya Gelen Mesajlar Ekranı



Şekil 5.8: Doktora Mesaj Ekranı

Hasta tarafında bulunan bir diğer ekran ise Ölçümlerim ekranıdır. Bu ekran ile hasta daha önceden gerçekleştirmiş olduğu ölçümlerini en son yapmış olduğu ölçümden eski tarihlerdekilere doğru sıralanmış biçimde görebilmektedir. (Şekil 5.9) Herhangi bir ölçümün üzerine tıkladığı takdirde de o ölçüm ile ilgili hasta adı, test türü, test tarihi, test değeri ve doktor tarafından belirlenmiş olan kritik asgari ve azami değerlerin bilgileri görebilmektedir. (Şekil 5.10)

Şekil 5.9: Hastanın Geçmiş Ölçümleri Ekranı



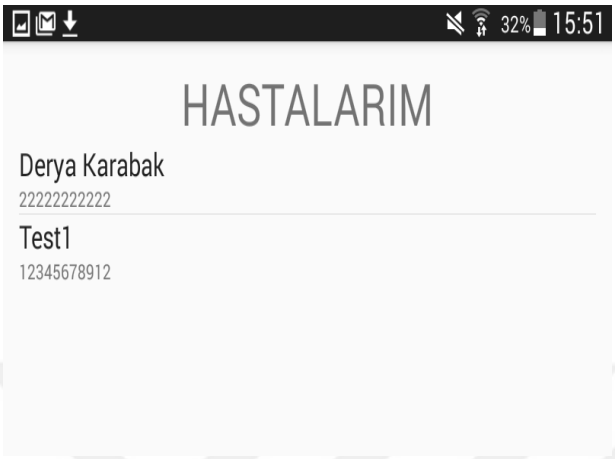
Şekil 5.10: Hasta Ölçüm Bilgileri Ekranı

5.1.2 Doktor İşlemleri Modülü

Doktorun, üzerine atanan hastaları, bu hastalar ile ilgili gelen bildirimleri görebileceği ve hastalardan gelen mesajlarını okuyabileceği ana ekran bulunmaktadır. (Şekil 5.11)

Şekil 5.11: Doktor Ana Ekranı

Hastalarım bölümünde doktor ilk olarak üzerine atanmış olan hastaları görmektedir. (Şekil 5.12) Sonrasında ise hangi hasta ile ilgili işlem yapmak istiyorsa o hastayı seçer ve karşımıza Şekil 5.13' de ki ekran gelir.



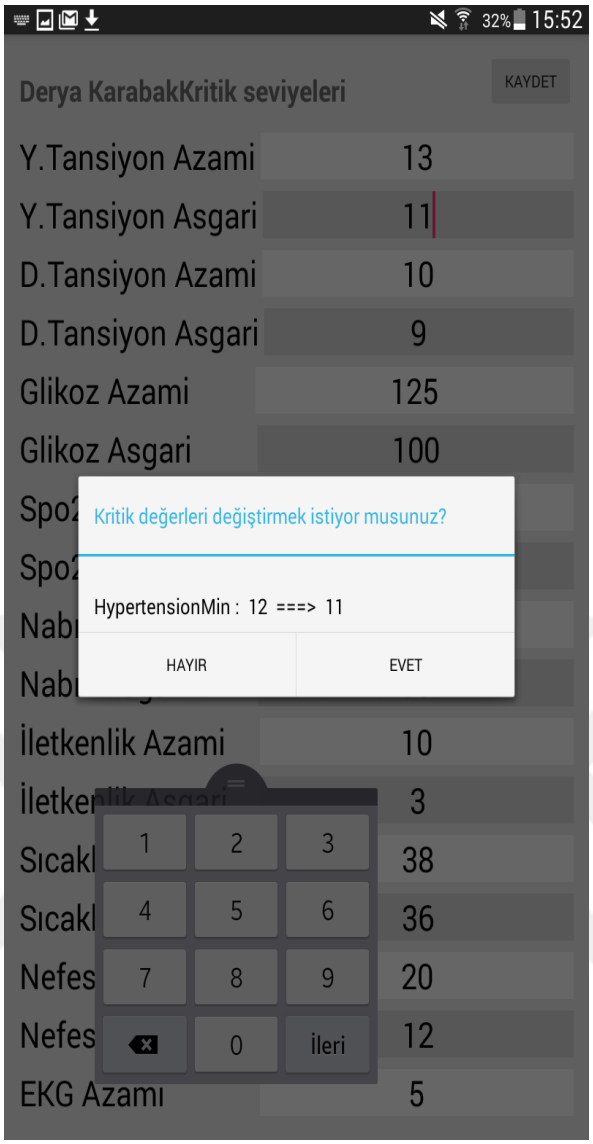
Şekil 5.12: Hastalarım Ekranı

Şekil 5.13: Hasta Bilgileri Ekranı

Doktor hastanın geçmiş değerleri görmek için ölçümler butonuna tıklar ve hastanın geçmiş tüm ölçümlerini görebilir. Burada sistem, hasta tarafında verilen Şekil 5.9 ve Şekil 5.10'da ki bilgilere yönlendirir. Sistemde gerçekleştirilen tüm ölçümler için genel olarak normal sayılan değerler mevcuttur. Ancak daha öncede belirtildiği gibi bu değerler kişiden kişiye, hastalıktan hastalığa değişkenlik gösterebilmektedir.Bu nedenle her hasta için doktor tarafından kişiye özel kritik değer sisteme girilmekte ve ölçüm sonuçları bu değerlere göre normal, düşük ve

yüksek olarak belirlenmektedir. Kritik değerler doktor tarafından Şekil 5.14' de ki gibi istenildiğinde güncellenip kaydedilebilmektedir. Herhangi bir yanlışlık olmaması açısından değişiklik yapıldığı takdirde Şekil 5.15' de ki gibi bir uyarı verilmektedir. Evet denilirse kayıt işlemi gerçekleşmekte, hayır denilirse bir önceki ekrana geri dönülmektedir.

Şekil 5.14: Kritik Değerler Ekranı

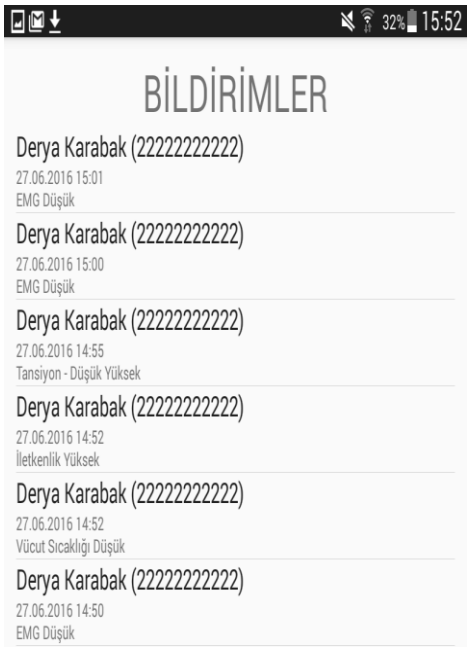


Şekil 5.15: Kritik Değerlerin Değişim Ekranı

Hasta bilgileri ekranında bulunan Hasta Detay bölümü hastanın adı, soyadı, yaşı, adresi ve hasta yakınının adı soyadı, yakınlık derecesini ve telefonunu gösteren ekrandır. (Şekil 5.16) Yanı sıra doktor herhangi bir durum ile ilgili hastaya mesaj göndermek istediği takdirde hastaya mesaj bölümüne girerek hastasına mesaj gönderebilmektedir.

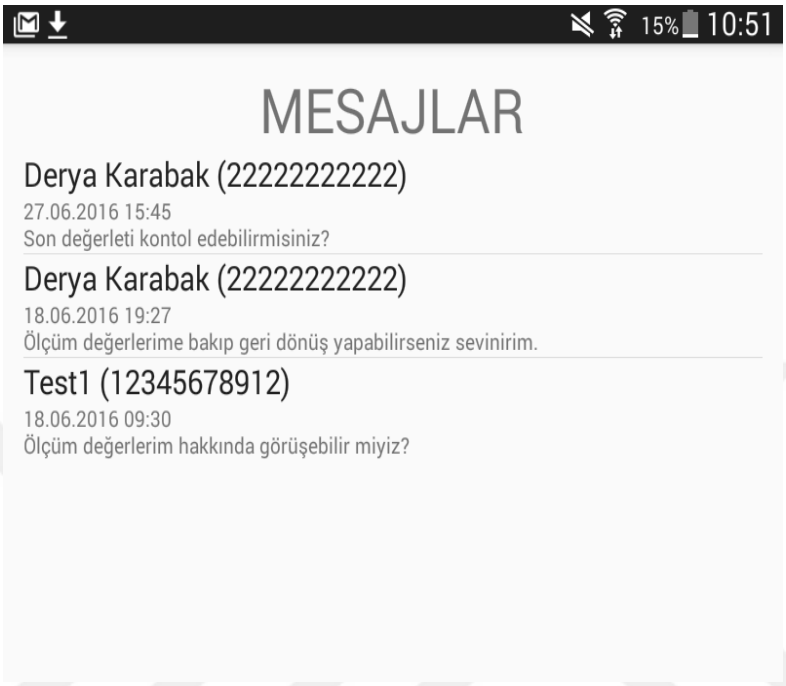
Şekil 5.16: Hasta Detay Ekranı

Doktor tarafının diğer bir bölümü ise bildirimlerdir. (Şekil 5.17) Bu bölümün oluşturulma amacı ise hastanın yaptığı herhangi bir ölçüm eğer belirtilen kritik değerlerin dışında ise doktora bildirim olarak gitmesi ve normal mesajlar için karışması önleyerek daha hızlı bir şekilde doktora ulaştırmaktır. Doktor gelen bildirimin üzerine tıkladığı zaman ise hastanın bilgilerinin bulunduğu Şekil 5.13'de ki ekrana sistem tarafından yönlendirilir.



Şekil 5.17: Bildirimler Ekranı

Son olarak doktor ana ekranında bulunan mesajlar bölümüü ise hasta tarafından doktora gönderilen mesajların doktor tarafından görülebildiği bölümdür. (Şekil 5.18)



Şekil 5.18: Doktora Gelen Mesajlar Ekranı

5.2 Mobil Takip Sisteminin Haberleşme Modülü

Sistemde kullanılan Arduino Yun Bölüm 3.1'de açıklanıldığı gibi içerisinde Linux işletim sistemi ve Wifi bulunduran bir cihazdır. Bu nedenle haberleşmeyi sağlamak için extra olarak başka bir cihaz kullanılmamıştır. Cihazın internet arayüzü vardır ve default olarak IP adresi alınır. Bilgisayara takıldığı zaman cihazın Web arayüzünden wifi ayarlaması yapılır. Haberleşme modülü amacı Arduino'dan alınan bilgilerin uygulamaya aktarımının sağlanmasıdır. İnternet üzerinden haberleşme için Arduino için de Web Servis kullanılmıştır. Wifi iletişiminin sağlanabilmesi Arduino IDE de aşağıdaki kodlar ile sağlanmıştır.

```
#include "eHealth.h"

#include "PinChangeInt.h"

#include <Bridge.h>

#include <YunServer.h>
```

```
#include <YunClient.h>
#include <HttpClient.h>
YunServer server;
int cont = 0;
int airprev = 2;
void setup() {
  Bridge.begin();
  Serial.begin(9600);
  while(!Serial); }
void loop()
{
  YunClient client = server.accept();
  String Data = GetString();
  Data.replace("\"","");
  Serial.println(Data); }
String GetString()
{
  String url = "http://beybalik.com.tr/asptest/api/values/GetQueue";
  HttpClient client;
  String data = "";
  client.get(url);
  while (client.available()) {
    char c = client.read();
    data = data + c; }
  if(data.length() != 0 && data != "null")
  { return data; }
```

```
  else
  {Serial.println("Data is null");}
  return "null";}
void SendJsonString(String JsonString)
{ YunClient client;
  if (client.connect("www.beybalik.com.tr", 80)) {
    Serial.println("Connected..");
    Process p;
    Serial.print("\n\nSending data... ");
    p.begin("curl");
    p.addParameter("-k");
    p.addParameter("--request");
    p.addParameter("POST");
    p.addParameter("--data");
    p.addParameter(JsonString);
    p.addParameter("--header");
    p.addParameter("content-type: application/json");

p.addParameter("http://www.beybalik.com.tr/asptest/api/values/ArduinoPost/");
    p.run();
    Serial.println("Sent");}}
```

Bunun yanı sıra Arduino ve Android uygulamanın haberleşebilmesi için iki tane web servis fonksiyonu bulunmaktadır. Bunlardan bir tanesi yukarıdaki String GetString method içerisinde kullanılan GetQueue metoduyken diğeri ise void SendJsonString metodu içerisinde kullanılan ArduinoPost metodudur.

GetQueue metodu ile gerçekleştirilen işlem: Arduino ile hangi ölçümün yapılacağı Android uygulama ile belirlenir ve gönderilir. Gönderilen veri

Orders.txt adında bir dosyada tutulur ve Arduino bu dosyadan bilgiyi aldıktan sonra gerekli ölçümü yapar.

```
[HttpGet]
    public string GetQueue()
    {StreamReader sr = new
StreamReader(HostingEnvironment.MapPath(@"~/Orders.txt"));
      string ReturnText = sr.ReadLine();
      sr.Close();
      File.WriteAllLines(HostingEnvironment.MapPath(@"~/Orders.txt"),
File.ReadAllLines(HostingEnvironment.MapPath(@"~/Orders.txt")).Skip(1))
;
      return ReturnText; }
```

ArduinoPost metodu ile gerçekleştirilen işlem: GetQueue metodundan alınan bilgiler sonrasında, ölçüm sonuçları OrdersResponse.txt adında bir dosyada tutulur ve Android uygulamaya gönderilir.

```
[HttpPost]
    public void ArduinoPost([FromBody]ArduinoPostRequest Request)
    {StreamWriter sw =
File.AppendText(HostingEnvironment.MapPath(@"~/OrderResponse.txt"));
      try
      {  foreach (var item in Request.Data) {
          sw.WriteLine(item); }
      }
      catch (Exception e)
      {sw.Close();}
      sw.Close();}
```

6 SONUÇ VE ÖNERİLER

Hayatımızı gün geçtikçe daha da kolaylaştıran teknoloji sayesinde artık bir çok bilgiye ve veriye kolay ulaşabilir hale gelmemiz sayesinde insan hayatı için en önemli şey olan sağlık alanındaki gelişmeler de gün geçtikçe artmaktadır. Bu tez çalışması ile hedeflenen amaç ise gerek Türkiye'de gerekse dünyada son yıllarda uygulaması gittikçe artan evde sağlık hizmetlerini daha kolay ve herkes için daha ulaşılabilir hale getirmektir.

Mobil takip sistemi ile sensörlerden alınan veriler web servis aracılığı ile Android uygulama ile bağlantı kurup verilerin depolanmasını, gerektiği zaman hastanın doktora ölçüm bilgilerini iletebilmesini ve aynı zamanda ölçüm değerleri doktor tarafından belirtilen değerlerin dışına çıktığı takdirde doktora bildirim gönderilmesini sağlamaktadır.

Oluşturulan sistem sayesinde evde sağlık hizmeti sağlayan uzmanların hastaların sağlık durumlarını uzaktan izleyebilmeleri ve dolayısıyla bu sayede daha çok hastaya ulaşabilmesi, hastaların daha kolay ve pratik bir şekilde kendi ölçümlerini yapmaları ve hastahaneye bağımlı kalmadan ev ortamında daha rahat koşullarda tedavi görmeleri hedeflenmiştir.

Bu tezde oluşturulan sistem, ileride video kamera veyahut fotograf gibi bir çok açıdan geliştirilip iyileştirilebilecek bir model tasarısı özelliğindedir. Halihazırda bulunan sistemler ile de entegre edilip kullanılabilecek ve hem hastaların hem de sağlık görevlilerinin işini kolaylaştıracak başka uygulamalara örnek teşkil edebileceği düşünülerek hazırlanmıştır.

KAYNAKLAR

[1] Tufan, İ, Aile ve sosyal politikalar bakanlığı yaşlı ve özürlü hizmet genel müdürlüğü'nün hazırladığı bakım hizmetleri stratejisi ve eylem planı takdir, eleştiri, öneri http://www.itgevakif.com/pdfs/elestiri_sicher.pdf

[2] T.C Sağlık Bakanlığı Strateji Geliştirme Başkanlığı, Koç, O 2011, Hastane rolleri, personel planlaması, evde sağlık hizmetleri, 25 Eylül, https://sgb.saglik.gov.tr/content/images/haberler/kizilcahamam_sunumlar/3_dr_orhan_koc_hastane_rolleri_evde_bakim_hizmetleri.pptx

[3] Türkiye Halk Sağlığı Kurumu, Sağlık bakanlığınca sunulan evde bakım hizmetlerinin uygulama usul ve esasları hakkında yönerge, İncelendiği Tarih 22 Şubat 2016, http://ailehekimligi.gov.tr/genel-mevzuat/yoenergeler/603-salk-bakanlnca-sunulan-evde-salk-hizmetlerinin-uygulama-usul-ve-esaslar-hakkında-yoenerge-.html

[4] Aktaş, F., Çeken, C., Erdemli, Y.E. (2014). *Biyomedikal Uygulamaları için Nesnelerin İnterneti Tabanlı Veri Toplama ve Analiz Sistemi.* Tıp Teknolojileri Ulusal Bildirisinde Sunulan Bildiri. Kapadokya, Türkiye, 25-27 Eylül.

[5] Cura, T., (2013). A low Cost Mobile Patient Monitoring System Proposal For Healthcare. Alphanumeric Journal, 1(1), 013-026.

[6] Türk Hipertansiyon ve Böbrek Hastalıkları Kan Basıncı Ölçüm Grubu, *Evde kan basıncı takibi hakkında genel bilgiler,* incelendiği tarih 23 Şubat 2016, http://www.turkhipertansiyon.org/pdf/dogruKanBasinci/3-5.pdf

[7] Kumbasar, D., *Kalp sağlığı,* Türk Geriatri Derneği, incelendiği tarih 23 Şubat 2016, http://www.turkgeriatri.org/pdfler/kalpsagligi.pdf

[8] Hatemi, H., *Şeker hastalığı tedavisi,* Türk Diabet ve Obezite Vakfı, incelendiği tarih 23 Şubat 2016, http://diabetvakfi.org/inf.php?partid=5&catid=5&pid=33

[9] Milli Eğitim Bakanlığı 2011, Acil Sağlık Hizmetleri, Ekg (Elektrokardiyografi), Ankara, Turkey

[10] Simon SB, Clark RA., *Using pulse oximetry: a review of pulse oximetry use in acute care medical wards.* Clinical Effectiveness in Nursing, 2002;6;106-110.

[11] Libelium. Guidance for e-Health Sensor Platform. Documentation/Tutorial.

EKLER

EK A: Uygulamanın C# Kodları

```
using DenemeAppV4.Models;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Web.Hosting;
using System.Web.Http;
using System.Data.OleDb;
using System.Data;
using System;
using System.Text;
using System.Threading;
namespace DenemeAppV4.Controllers
{
    public class ValuesController : ApiController
    {
        OleDbConnection connection = new
OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=|DataDirectory|\\EhealthDB.accdb;User ID=admin");
```

```
public List<Dictionary<string, object>>
DatatableToJsonSerialize(DataTable dt)
{List<Dictionary<string, object>> rows = new List<Dictionary<string,
object>>();
Dictionary<string, object> row;
foreach (DataRow dr in dt.Rows)
{row = new Dictionary<string, object>();
foreach (DataColumn col in dt.Columns)
{row.Add(col.ColumnName, dr[col]); }
rows.Add(row);
}return rows; }
public void InsertDoctorNotification(string PhoneID, string Message,
string PatientID)
{
OleDbCommand cmd = new OleDbCommand(@"Select DoctorID
From DoctorsPatient where PatientID = @PatientID",
connection);
cmd.Parameters.AddWithValue("@PatientID", PatientID);
OleDbDataReader reader = cmd.ExecuteReader();
reader.Read();
string DoctorID = reader["DoctorID"].ToString();
cmd = new OleDbCommand(@"Insert into MessageTable
(SenderNo, ReceiverNo, Message, Type, ReceivedDate)
Values(@SenderID, @ReceiverID, @Message,
'notification', @ReceivedDate)", connection);
cmd.Parameters.AddWithValue("@SenderID", PatientID);
cmd.Parameters.AddWithValue("@ReceiverID", DoctorID);
```

```
            cmd.Parameters.AddWithValue("@Message", Message);
            cmd.Parameters.AddWithValue("@ReceivedDate",
DateTime.Now.ToString("dd.MM.yyyy HH:mm"));
            cmd.ExecuteNonQuery();}
        public void InsertHistory(string PhoneID, string Testname, string
Testvalue, int counter)
        {string TestnameMax = "";
            string TestnameMin = "";
            OleDbCommand cmd = new OleDbCommand(@"Select TC_No
                    From UserLogin where PhoneID = @PhoneID", connection);
            cmd.Parameters.AddWithValue("@PhoneID", PhoneID);
            OleDbDataReader reader = cmd.ExecuteReader();
            reader.Read();
            string PatientID = reader["TC_No"].ToString();
            switch (Testname)
            {case "Pulseoximeter":
                  if (counter == 0)
                  {TestnameMax = "Spo2Max";
                     TestnameMin = "Spo2Min";
                     Testname = "Nabız ve Oksijen - Spo2";}
                  else if (counter == 1)
                  {TestnameMax = "PulseMax";
                     TestnameMin = "PulseMin";
                     Testname = "Nabız ve Oksijen - Nabız";}
                  break;
               case "BloodPressure":
                  if (counter == 0)
```

```
                {TestnameMax = "HypertensionMax";
                   TestnameMin = "HypertensionMin";
                   Testname = "Tansiyon - Yüksek";}
                else if (counter == 1)
                {TestnameMax = "HypotensionMax";
                   TestnameMin = "HypotensionMin";
                   Testname = "Tansiyon - Düşük";}
                break;
              default:
                TestnameMax = Testname + "Max";
                TestnameMin = Testname + "Min";
                break; }
            if (Testname == "Glucose")
            {Testname = "Glikoz";}
            if (Testname == "Heat")
            {Testname = "Vücut Sıcaklığı";}
            if (Testname == "Conductivity")
            {Testname = "İletkenlik";}
            string query = @"Select @TestValueMax, @TestValueMin
               From TresholdTable where PatientID = @PatientID";
            query =
query.Replace("@TestValueMax",TestnameMax).Replace("@TestValueMi
n",TestnameMin);
            cmd = new OleDbCommand(query, connection);
            cmd.Parameters.AddWithValue("@PatientID", PatientID);
            reader = cmd.ExecuteReader();
            reader.Read();
```

```
            string TestValueMax = reader[TestnameMax].ToString();
            string TestValueMin = reader[TestnameMin].ToString();
            if (Convert.ToDouble(Testvalue) >
Convert.ToDouble(TestValueMax))
            {InsertDoctorNotification(PhoneID, Testname + " Yüksek",
PatientID); }
            else if (Convert.ToDouble(Testvalue) <
Convert.ToDouble(TestValueMin))
            {InsertDoctorNotification(PhoneID, Testname + " Düşük",
PatientID); }
            cmd = new OleDbCommand(@"Insert into TestHistory
              (PatientID, TestType, TestValue, TestValueMax, TestValueMin,
TestDate)  Values(@PatientID, @TestType, @TestValue, @TestValueMax,
@TestValueMin, @TestDate)", connection);
            cmd.Parameters.AddWithValue("@PatientID", PatientID);
            cmd.Parameters.AddWithValue("@TestType", Testname);
            cmd.Parameters.AddWithValue("@TestValue", Testvalue);
            cmd.Parameters.AddWithValue("@TestValueMax",
TestValueMax);
            cmd.Parameters.AddWithValue("@TestValueMin", TestValueMin);
            cmd.Parameters.AddWithValue("@TestDate",
DateTime.Now.ToString("dd.MM.yyyy HH:mm:ss"));
            cmd.ExecuteNonQuery();
        }
        public void InsertHistory(string PhoneID, string Testname,
IEnumerable<string> lines)
        {
            string TestnameMax = Testname + "Max";
```

```
string TestnameMin = Testname + "Min";
string Testvalue = "";
OleDbCommand cmd = new OleDbCommand(@"Select TC_No
    From UserLogin where PhoneID = @PhoneID", connection);
cmd.Parameters.AddWithValue("@PhoneID", PhoneID);
OleDbDataReader reader = cmd.ExecuteReader();
reader.Read();
string PatientID = reader["TC_No"].ToString();
string query = @"Select @TestValueMax, @TestValueMin
                    From TresholdTable
                    where PatientID = @PatientID";
query = query.Replace("@TestValueMax",
TestnameMax).Replace("@TestValueMin", TestnameMin);
cmd = new OleDbCommand(query, connection);
cmd.Parameters.AddWithValue("@PatientID", PatientID);
reader = cmd.ExecuteReader();
reader.Read();
string TestValueMax = reader[TestnameMax].ToString();
string TestValueMin = reader[TestnameMin].ToString();
if (Testname == "Airflow")
{Testname = "Nefes";}
if (Testname == "ECG")
{Testname = "EKG";}
foreach (var item in lines)
{if (Convert.ToDouble(item) > Convert.ToDouble(TestValueMax))
   {Testvalue = Testname + " Yüksek";
```

```
                InsertDoctorNotification(PhoneID, Testvalue, PatientID);
                break; }
            else if (Convert.ToDouble(item) <
Convert.ToDouble(TestValueMin))
            {Testvalue = Testname + " Düşük";
                InsertDoctorNotification(PhoneID, Testvalue, PatientID);
                break; }
            else
            {Testvalue = Testname + " Normal";}}
        cmd = new OleDbCommand(@"Insert into TestHistory
       (PatientID, TestType, TestValue, TestValueMax, TestValueMin,
TestDate)
                        Values(@PatientID, @TestType, @TestValue,
@TestValueMax, @TestValueMin, @TestDate)", connection);
        cmd.Parameters.AddWithValue("@PatientID", PatientID);
        cmd.Parameters.AddWithValue("@TestType", Testname);
        cmd.Parameters.AddWithValue("@TestValue", Testvalue);
        cmd.Parameters.AddWithValue("@TestValueMax",
TestValueMax);
        cmd.Parameters.AddWithValue("@TestValueMin", TestValueMin);
        cmd.Parameters.AddWithValue("@TestDate",
DateTime.Now.ToString("dd.MM.yyyy HH:mm:ss"));
        cmd.ExecuteNonQuery();}
    [HttpGet]
    public string GetQueue()
    {
        StreamReader sr = new
StreamReader(HostingEnvironment.MapPath(@"~/Orders.txt"));
```

```
        string ReturnText = sr.ReadLine();
        sr.Close();
        File.WriteAllLines(HostingEnvironment.MapPath(@"~/Orders.txt"),
File.ReadAllLines(HostingEnvironment.MapPath(@"~/Orders.txt")).Skip(1
));
        return ReturnText; }
    [HttpPost]
    public PostQueueResponse PostQueue([FromBody]PostQueueRequest
PostQueueData)
    {StreamWriter sw =
File.AppendText(HostingEnvironment.MapPath(@"~/Orders.txt"));
        sw.WriteLine(PostQueueData.Text);
        sw.Close();
File.Create(HostingEnvironment.MapPath(@"~/OrderResponse.txt")).Close
();
        if (PostQueueData.Type == "Single")
        {Thread.Sleep(5000); }
        else if (PostQueueData.Type == "Chart")
        {Thread.Sleep(45000); }
        PostQueueResponse Response = new PostQueueResponse();
        Response.Response = new List<PostQueueListItem>();
        var lines =
File.ReadLines(HostingEnvironment.MapPath(@"~/OrderResponse.txt"));
        connection.Open();
        if (PostQueueData.Type == "Chart")
        {InsertHistory(PostQueueData.PhoneID, PostQueueData.Text,
lines); }
        int counter = 0;
```

```
            foreach (var item in lines)
            {
                if (PostQueueData.Type == "Single")
                {InsertHistory(PostQueueData.PhoneID, PostQueueData.Text,
item, counter); }
                counter++;
                Response.Response.Add(new PostQueueListItem
                {ListItem =
item});}File.Create(HostingEnvironment.MapPath(@"~/OrderResponse.txt
")).Close();
            connection.Close();
            return Response; }
        [HttpPost]
        public void ArduinoPost([FromBody]ArduinoPostRequest Request)
        {StreamWriter sw =
File.AppendText(HostingEnvironment.MapPath(@"~/OrderResponse.txt"));
            try
            { foreach (var item in Request.Data)
                {sw.WriteLine(item); }
            }
            catch (Exception e)
            {sw.Close();}
            sw.Close();
        }
        [HttpPost]
        public UserLoginResponse UserLogin([FromBody]UserLoginRequest
UserLogin)
```

```
{
    connection.Open();
    OleDbCommand cmd = new OleDbCommand(@"Update UserLogin
        Set PhoneID = " Where PhoneID = @PhoneID", connection);
    cmd.Parameters.AddWithValue("@PhoneID", UserLogin.PhoneID);
    cmd.ExecuteNonQuery();
    cmd = new OleDbCommand("Select * From UserLogin",
connection);
    OleDbDataReader reader = cmd.ExecuteReader();
    while (reader.Read())
    {
      if (reader["Username"].ToString() == UserLogin.Username &&
reader["Password"].ToString() == UserLogin.Password)
      {cmd = new OleDbCommand("Update UserLogin Set PhoneID =
@PhoneID, LoginDate = @LoginDate Where Username = @Username",
connection);
        cmd.Parameters.AddWithValue("@PhoneID",
UserLogin.PhoneID);
        cmd.Parameters.AddWithValue("@LoginDate",
DateTime.Now.AddDays(1).ToString("dd.MM.yyyy HH:mm:ss"));
        cmd.Parameters.AddWithValue("@Username",
UserLogin.Username);
        cmd.ExecuteNonQuery();
        string type = reader["Type"].ToString();
        connection.Close();
        return new UserLoginResponse
        {
          PhoneID = UserLogin.PhoneID,
```

```
                Type = type
            };}
        connection.Close();
        return new UserLoginResponse
            {PhoneID = "false"};}
    [HttpPost]
    public LoggedUserResponse
LoggedUser([FromBody]LoggedUserRequest LoggedUser)
    {connection.Open();
        OleDbCommand cmd = new OleDbCommand("Select * From
UserLogin", connection);
        OleDbDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {
            DateTime LoginDate =
Convert.ToDateTime(reader["LoginDate"]);
            DateTime CurrentTime = DateTime.Now;
            if (reader["Username"].ToString() == LoggedUser.Username &&
reader["PhoneID"].ToString() == LoggedUser.PhoneID && LoginDate >
CurrentTime)
            {string phoneid = reader["PhoneID"].ToString();
                string type = reader["Type"].ToString();
                connection.Close();
                return new LoggedUserResponse
                {PhoneID = phoneid,
                    Type = type
                };
```

```
            connection.Close();
            return new LoggedUserResponse
            {PhoneID = "false"};}
        [HttpPost]
        public MyPatientsResponse
DoctorPatients([FromBody]MyPatientsRequest MyPatientsRequest)
        {connection.Open();
            OleDbCommand cmd = new OleDbCommand("Select d.PatientID,
p.Patient_Name From DoctorsPatient d, PatientInfo p, UserLogin u where
d.PatientID = p.TC_No and d.DoctorID = u.TC_No and u.PhoneID =
@PhoneID", connection);
            cmd.Parameters.AddWithValue("@PhoneID",
MyPatientsRequest.PhoneID);
            OleDbDataReader reader = cmd.ExecuteReader();
            MyPatientsResponse Response = new MyPatientsResponse();
            Response.Response = new List<MyPatientsListItem>();
            while (reader.Read())
            {Response.Response.Add(new MyPatientsListItem
               {PatientTcNo = reader["PatientID"].ToString(),
                  PatientName = reader["Patient_Name"].ToString()});}
            connection.Close();
            return Response; }
        [HttpPost]
        public MessageToDoctorResponse
MessageToDoctor([FromBody]MessageToDoctorRequest
MessageToDoctor)
        {connection.Open();
```

```
        OleDbCommand cmd = new OleDbCommand(@"Select
p.Patient_Name, m.Message, m.SenderNo, m.ReceivedDate
                                From MessageTable m, PatientInfo p,
UserLogin u
                                  where m.SenderNo = p.TC_No
                                   and m.Type = 'message'
                                   and m.ReceiverNo = u.TC_No
                                   and u.PhoneID = @PhoneID
                                  order by m.ReceivedDate desc",
connection);
        cmd.Parameters.AddWithValue("@PhoneID",
MessageToDoctor.PhoneID);
        OleDbDataReader reader = cmd.ExecuteReader();
        MessageToDoctorResponse Response = new
MessageToDoctorResponse();
        Response.Response = new List<MessageToDoctorListItem>();
        while (reader.Read())
        {Response.Response.Add(new MessageToDoctorListItem
           {PatientTcNo = reader["SenderNo"].ToString(),
              PatientName = reader["Patient_Name"].ToString(),
              Message = reader["Message"].ToString(),
              ReceivedDate =
Convert.ToDateTime(reader["ReceivedDate"]).ToString("dd.MM.yyyy
HH:mm")
           });
        connection.Close();
        return Response; }
     [HttpPost]
```

```
public DoctorNotificationResponse
DoctorNotification([FromBody]DoctorNotificationRequest
DoctorNotification)
{connection.Open();
   OleDbCommand cmd = new OleDbCommand(@"Select
p.Patient_Name, m.Message, m.SenderNo, m.ReceivedDate
                                From MessageTable m, PatientInfo p,
UserLogin u
                                 where m.SenderNo = p.TC_No
                                  and m.Type = 'notification'
                                  and m.ReceiverNo = u.TC_No
                                  and u.PhoneID = @PhoneID
                                 order by m.ReceivedDate desc",
connection);
   cmd.Parameters.AddWithValue("@PhoneID",
DoctorNotification.PhoneID);
   OleDbDataReader reader = cmd.ExecuteReader();
   DoctorNotificationResponse Response = new
DoctorNotificationResponse();
   Response.Response = new List<DoctorNotificationListItem>();
   while (reader.Read())
   {
      Response.Response.Add(new DoctorNotificationListItem
      {
         PatientTcNo = reader["SenderNo"].ToString(),
         PatientName = reader["Patient_Name"].ToString(),
         Message = reader["Message"].ToString(),
```

```
                ReceivedDate =
Convert.ToDateTime(reader["ReceivedDate"]).ToString("dd.MM.yyyy
HH:mm")
            });
        connection.Close();
        return Response;
    }
    [HttpPost]
    public MessageToPatientResponse
MessageToPatient([FromBody]MessageToPatientRequest
MessageToDoctor)
    {connection.Open();
        OleDbCommand cmd = new OleDbCommand(@"Select
m.Message, m.SenderNo, m.ReceivedDate From MessageTable m,
UserLogin u
                                    where m.Type = 'message'
                                     and m.ReceiverNo = u.TC_No
                                     and u.PhoneID = @PhoneID
                                    order by m.ReceivedDate desc",
connection);
        cmd.Parameters.AddWithValue("@PhoneID",
MessageToDoctor.PhoneID);
        OleDbDataReader reader = cmd.ExecuteReader();
        MessageToPatientResponse Response = new
MessageToPatientResponse();
        Response.Response = new List<MessageToPatientListItem>();
        while (reader.Read())
        {Response.Response.Add(new MessageToPatientListItem
```

```
            {DoctorTcNo = reader["SenderNo"].ToString(),
               Message = reader["Message"].ToString(),
               ReceivedDate =
Convert.ToDateTime(reader["ReceivedDate"]).ToString("dd.MM.yyyy
HH:mm")
            });
         connection.Close();
         return Response;
      }
      [HttpPost]
      public SendMessageToDoctorResponse
SendMessageToDoctor([FromBody]SendMessageToDoctorRequest
MessageData)
      {connection.Open();
         try
         {OleDbCommand cmd = new OleDbCommand(@"Select TC_No
                                   From UserLogin
                                   where PhoneID = @PhoneID", connection);
            cmd.Parameters.AddWithValue("@PhoneID",
MessageData.PhoneID);
            OleDbDataReader reader = cmd.ExecuteReader();
            reader.Read();
            string SenderID = reader["TC_No"].ToString();
            cmd = new OleDbCommand(@"Select DoctorID
                        From DoctorsPatient
                        where PatientID = @SenderID", connection);
            cmd.Parameters.AddWithValue("@SenderID", SenderID);
```

```
            reader = cmd.ExecuteReader();
            reader.Read();
            string ReceiverID = reader["DoctorID"].ToString();
            cmd = new OleDbCommand(@"Insert into MessageTable
(SenderNo,ReceiverNo,Message,Type,ReceivedDate)
Values(@SenderID,@ReceiverID,@Message,'message',@ReceivedDate)",
connection);
            cmd.Parameters.AddWithValue("@SenderID", SenderID);
            cmd.Parameters.AddWithValue("@ReceiverID", ReceiverID);
            cmd.Parameters.AddWithValue("@Message",
MessageData.Message);
            cmd.Parameters.AddWithValue("@ReceivedDate",
DateTime.Now.ToString("dd.MM.yyyy HH:mm"));
            cmd.ExecuteNonQuery();
            return new SendMessageToDoctorResponse
            {
               Response = "true"
            };
          }
          catch (Exception e)
          {return new SendMessageToDoctorResponse
            {Response = "false"
            };}
      [HttpPost]
      public SendMessageToPatientResponse
SendMessageToPatient([FromBody]SendMessageToPatientRequest
MessageData)
```

```
    {
        connection.Open();
        try
        {OleDbCommand cmd = new OleDbCommand(@"Select TC_No
                From UserLogin where PhoneID = @PhoneID",
connection);
            cmd.Parameters.AddWithValue("@PhoneID",
MessageData.PhoneID);
            OleDbDataReader reader = cmd.ExecuteReader();
            reader.Read();
            string SenderID = reader["TC_No"].ToString();
            string ReceiverID = MessageData.PatientID;
            cmd = new OleDbCommand(@"Insert into MessageTable

(SenderNo,ReceiverNo,Message,Type,ReceivedDate)
Values(@SenderID,@ReceiverID,@Message,'message',@ReceivedDate)",
connection);
            cmd.Parameters.AddWithValue("@SenderID", SenderID);
            cmd.Parameters.AddWithValue("@ReceiverID", ReceiverID);
            cmd.Parameters.AddWithValue("@Message",
MessageData.Message);
            cmd.Parameters.AddWithValue("@ReceivedDate",
DateTime.Now.ToString("dd.MM.yyyy HH:mm"));
            cmd.ExecuteNonQuery();
            return new SendMessageToPatientResponse
            {Response = "true"};
        catch (Exception e)
        {return new SendMessageToPatientResponse
```

```
            {Response = "false"};}
            finally
            {connection.Close();}}
        [HttpPost]
        public PatientInformationResponse
PatientInformation([FromBody]PatientInformationRequest
PatientInformation)
        {
            PatientInformationResponse Response = new
PatientInformationResponse();
            connection.Open();
            OleDbCommand cmd = new OleDbCommand(@"Select *
              From PatientInfo where TC_No = @PatientID", connection);
            cmd.Parameters.AddWithValue("@PatientID",
PatientInformation.PatientID);
            OleDbDataReader reader = cmd.ExecuteReader();
            reader.Read();
            Response.TC_No= reader["TC_No"].ToString();
            Response.Patient_Name = reader["Patient_Name"].ToString();
            Response.Patient_Age = reader["Patient_Age"].ToString();
            Response.Patient_Adress = reader["Patient_Adress"].ToString();
            Response.Patient_Phone = reader["Patient_Phone"].ToString();
            Response.Relative_Name = reader["Relative_Name"].ToString();
            Response.Relative_kinship = reader["Relative_kinship"].ToString();
            Response.Relative_Phone = reader["Relative_Phone"].ToString();
            connection.Close();
            return Response;
```

```
    }
    [HttpPost]
    public PatientTestHistoryByDoctorResponse
PatientTestHistoryByDoctor([FromBody]PatientTestHistoryByDoctorRequ
est PatientTestHistoryByDoctor)
    {
        PatientTestHistoryByDoctorResponse Response = new
PatientTestHistoryByDoctorResponse();
        Response.Response = new
List<PatientTestHistoryByDoctorListItem>();
        connection.Open();
        OleDbCommand cmd = new OleDbCommand(@"Select *
                                From TestHistory t, UserLogin u
                                where t.PatientID = u.TC_No
                                 and t.PatientID = @PatientID
                                 Order by TestDate Desc", connection);
        cmd.Parameters.AddWithValue("@PatientID",
PatientTestHistoryByDoctor.PatientID);
        OleDbDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {Response.Response.Add(new PatientTestHistoryByDoctorListItem
          {
            Patient_Name = reader["Username"].ToString(),
            TestType = reader["TestType"].ToString(),
            TestValue = reader["TestValue"].ToString(),
            TestValueMax = reader["TestValueMax"].ToString(),
            TestValueMin = reader["TestValueMin"].ToString(),
```

```
                TestDate =
Convert.ToDateTime(reader["TestDate"]).ToString("dd.MM.yyyy
HH:mm")
            });
        connection.Close();
        return Response;
    }
    [HttpPost]
    public PatientTestHistoryByPatientResponse
PatientTestHistoryByPatient([FromBody]PatientTestHistoryByPatientRequ
est PatientTestHistoryByPatient)
    {
        PatientTestHistoryByPatientResponse Response = new
PatientTestHistoryByPatientResponse();
        Response.Response = new
List<PatientTestHistoryByPatientListItem>();
        connection.Open();
        OleDbCommand cmd = new OleDbCommand(@"Select *
                                From TestHistory t, UserLogin u
                                where t.PatientID = u.TC_No
                                  and u.PhoneID = @PhoneID
                                Order by TestDate Desc", connection);
        cmd.Parameters.AddWithValue("@PhoneID",
PatientTestHistoryByPatient.PhoneID);
        OleDbDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {Response.Response.Add(new  PatientTestHistoryByPatientListItem
          {
```

```
                Patient_Name = reader["Username"].ToString(),
                TestType = reader["TestType"].ToString(),
                TestValue = reader["TestValue"].ToString(),
                TestValueMax = reader["TestValueMax"].ToString(),
                TestValueMin = reader["TestValueMin"].ToString(),
                TestDate =
Convert.ToDateTime(reader["TestDate"]).ToString("dd.MM.yyyy
HH:mm")
            });
        }
        connection.Close();
        return Response; }
    [HttpPost]
    public SafeExitResponse SafeExit(SafeExitRequest SafeExit)
    {
        connection.Open();
        OleDbCommand cmd = new OleDbCommand(@"Update UserLogin
            Set PhoneID = " Where PhoneID = @PhoneID", connection);
        cmd.Parameters.AddWithValue("@PhoneID", SafeExit.PhoneID);
        cmd.ExecuteNonQuery();
        connection.Close();
        return new SafeExitResponse
        {
            Response = "true"};}
    [HttpPost]
    public TresholdValuesResponse
TresholdValues([FromBody]TresholdValuesRequest TresholdValues)
```

```
        {
            connection.Open();
            string query = @"Select * From TresholdTable
                        where PatientID = @PatientID";
        query = query.Replace("@PatientID","'" + TresholdValues.PatientID + "'");
            OleDbDataAdapter adapter = new OleDbDataAdapter(query, connection);
            DataTable dt = new DataTable();
            adapter.Fill(dt);
            connection.Close();
            return new TresholdValuesResponse
            {Response = DatatableToJsonSerialize(dt) };}
        [HttpPost]
        public SaveTresholdValuesResponse SaveTresholdValues([FromBody]SaveTresholdValuesRequest TresholdValues)
        {
            try
            {
                string query = @"Update TresholdTable
                                Set HypertensionMax = '@HypertensionMax',
                                HypertensionMin = '@HypertensionMin',
                                HypotensionMax = '@HypotensionMax',
                                HypotensionMin = '@HypotensionMin',
                                GlucoseMax = '@GlucoseMax',
```

```
                    GlucoseMin = '@GlucoseMin',
                    Spo2Max = '@Spo2Max',
                    Spo2Min = '@Spo2Min',
                    PulseMax = '@PulseMax',
                    PulseMin = '@PulseMin',
                    ConductivityMax = '@ConductivityMax',
                    ConductivityMin = '@ConductivityMin',
                    HeatMax = '@HeatMax',
                    HeatMin = '@HeatMin',
                    AirflowMax = '@AirflowMax',
                    AirflowMin = '@AirflowMin',
                    ECGMax = '@ECGMax',
                    ECGMin = '@ECGMin',
                    EMGMax = '@EMGMax',
                    EMGMin = '@EMGMin'
                    Where PatientID = '@PatientID'";
            query = query.Replace("@PatientID", TresholdValues.PatientID);
            query = query.Replace("@HypertensionMax",
TresholdValues.HypertensionMax);
            query = query.Replace("@HypertensionMin",
TresholdValues.HypertensionMin);
            query = query.Replace("@HypotensionMax",
TresholdValues.HypotensionMax);
            query = query.Replace("@HypotensionMin",
TresholdValues.HypotensionMin);
            query = query.Replace("@GlucoseMax",
TresholdValues.GlucoseMax);
```

```
            query = query.Replace("@GlucoseMin",
TresholdValues.GlucoseMin);
            query = query.Replace("@Spo2Max", TresholdValues.Spo2Max);
            query = query.Replace("@Spo2Min", TresholdValues.Spo2Min);
            query = query.Replace("@PulseMax",
TresholdValues.PulseMax);
            query = query.Replace("@PulseMin", TresholdValues.PulseMin);
            query = query.Replace("@ConductivityMax",
TresholdValues.ConductivityMax);
            query = query.Replace("@ConductivityMin",
TresholdValues.ConductivityMin);
            query = query.Replace("@HeatMax", TresholdValues.HeatMax);
            query = query.Replace("@HeatMin", TresholdValues.HeatMin);
            query = query.Replace("@AirflowMax",
TresholdValues.AirflowMax);
            query = query.Replace("@AirflowMin",
TresholdValues.AirflowMin);
            query = query.Replace("@ECGMax", TresholdValues.ECGMax);
            query = query.Replace("@ECGMin", TresholdValues.ECGMin);
            query = query.Replace("@EMGMax",
TresholdValues.EMGMax);
            query = query.Replace("@EMGMin", TresholdValues.EMGMin);
            connection.Open();
            OleDbCommand cmd = new OleDbCommand(query,
connection);
            cmd.ExecuteNonQuery();
            connection.Close();
        }
```

```
            catch (Exception e)
            {connection.Close();
               return new SaveTresholdValuesResponse
               {Response = "false"};}
            return new SaveTresholdValuesResponse
            {Response = "true"};
         }
```

EK B: Uygulamanın Android Kodları

```
public class AlertHelper {
   private Context context;
   private String connectionMessage="İnternet Bağlantınız
Bulunmamaktadır";
   String connectionTitle="Uyarı";
   AlertHelper(Context context)
   {this.context=context; }
   public void getErrorMessage(String errorMessage, String title)
   {new Warning(errorMessage,title).execute();}
   public boolean isConnect()
   {ConnectivityManager check =
(ConnectivityManager)this.context.getSystemService(Context.CONNECTI
VITY_SERVICE);
      NetworkInfo[] info = check.getAllNetworkInfo();
      int connection=0;
      for (int i = 0; i<info.length; i++){
   if (info[i].getState() == NetworkInfo.State.CONNECTED){
connection++;}}
      if(connection>0)
      {return true; }
      else
      {new Warning(connectionMessage, connectionTitle).execute();}
      return false;
```

```
    }
    class Warning extends AsyncTask<String,Void,String>
    {String message;
        String title;
        public Warning (String message, String title)
        {this.message=message;
            this.title=title; }
        @Override
        protected String doInBackground(String... params) {
            return null;
        @Override
        protected void onPostExecute(String result) {
            AlertDialog.Builder dialog;
            dialog = new AlertDialog.Builder(context);
            dialog.setCancelable(true);
            dialog.setTitle(title);
            dialog.setMessage(message);
            dialog.setNeutralButton("TAMAM", new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which)
{dialog.dismiss();}});
            dialog.show();}}}

public class ChartPage extends AppCompatActivity {
    ArrayList<String> dataArray;
    String testType="";
    LinearLayout layout;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_chart_page);
        getPageInfo();fillChart();}
    private void getPageInfo()
```

```
{Intent intent=getIntent();
   Bundle info = intent.getExtras();
   dataArray=info.getStringArrayList("dataArray");
   testType=info.getString("testType");
}
private void fillChart()
{layout=(LinearLayout)findViewById(R.id.chartLayout);
   int element=0;
   for(int k=1;k<=4;k++){
      LineChart lineChart = new LineChart(this);
      lineChart.setLayoutParams(new
LineChart.LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT,
170)); ArrayList<Entry> entries = new ArrayList<>();
      ArrayList<String> chartNate=new ArrayList<>();
      for (int i=0;i<(dataArray.size()/4);i++)
      {
 chartNate.add(String.valueOf(((k-1)*dataArray.size()/4)+i));
         try
 {entries.add(new Entry(Float.valueOf(dataArray.get(((k-
1)*dataArray.size()/4)+i)),i));
         }catch (Exception e)
         {Log.d("EntryCaption", "" + String.valueOf(e)) }}
      LineDataSet dataset = new LineDataSet(entries, "");
      LineData data = new LineData(chartNate, dataset);
      lineChart.setData(data);
      lineChart.setDescription(testType);
      layout.addView(lineChart); }}}

public class CriticalValuesPage extends AppCompatActivity {
   String patientID;
   Context context=this;
   String title;
   ArrayList<String> criticalValues=new ArrayList<>();
   ArrayList<String> criticalNames=new ArrayList<>();
```

```
    ArrayList<EditText> criticalEdits=new ArrayList<>();
    ArrayList<String> tempValues=new ArrayList<>();
    Button saveButton;
    TextView titleView;
    JSONObject postObject;
    int changes;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_critical_values_page);
        getPageInfo();
        teachTools();
        getValues();
        toolsActions();}
    private void getPageInfo()
    {Intent intent=getIntent();
        Bundle info = intent.getExtras();
        patientID=info.getString("identityNo");
        title=info.getString("PageTitle");}
    private void getValues()
    {String url="http://beybalik.com.tr/asptest/api/values/TresholdValues";
        String warnMessage="Bir Hata Oluştu. Bu yüzden verileri
görüntüleyemiyoruz";
        postObject = new JSONObject();
        try {
            postObject.put("PatientID", patientID);
        } catch (JSONException e) {
            e.printStackTrace();
        }JsonPostRequest(url,warnMessage); }
    private void sendValues()
{String
url="http://beybalik.com.tr/asptest/api/values/SaveTresholdValues";
```

```
String warnMessage="Bir Hata Oluştu. Bu yüzden kaydetme işleminizi
gerçekleştiremiyoruz";
      postObject = new JSONObject();
      try {
         postObject.put("PatientID", patientID);
         for (int
i=0;i<criticalValues.size();i++){postObject.put(criticalNames.get(i),temp
Values.get(i));
            Log.d("gidenObje",tempValues.get(i)); }
      } catch (JSONException e) {
         e.printStackTrace();}
      JsonPostRequest(url,warnMessage);
   }private void teachTools()
   {criticalValues=new ArrayList<>();
      criticalNames=new ArrayList<>();
      criticalEdits=new ArrayList<>();
      tempValues=new ArrayList<>();
      saveButton=(Button)findViewById(R.id.saveButton);
      titleView=(TextView)findViewById(R.id.criticalDescView);
      titleView.setText(title);
      criticalNames.add("HypertensionMax");
      criticalNames.add("HypertensionMin");
      criticalNames.add("HypotensionMax");
      criticalNames.add("HypotensionMin");
      criticalNames.add("GlucoseMax");
      criticalNames.add("GlucoseMin");
      criticalNames.add("Spo2Max");
      criticalNames.add("Spo2Min");
      criticalNames.add("PulseMax");
      criticalNames.add("PulseMin");
      criticalNames.add("ConductivityMax");
      criticalNames.add("ConductivityMin");
      criticalNames.add("HeatMax");
      criticalNames.add("HeatMin");
```

```
        criticalNames.add("AirflowMax");
        criticalNames.add("AirflowMin");
        criticalNames.add("ECGMax");
        criticalNames.add("ECGMin");
        criticalNames.add("EMGMax");
criticalNames.add("EMGMin");criticalEdits.add((EditText)findViewById(
R.id.HypertensionMaxEdit));
criticalEdits.add((EditText)findViewById(R.id.HypertensionMinEdit));
criticalEdits.add((EditText)findViewById(R.id.HypotensionMaxEdit));
criticalEdits.add((EditText)findViewById(R.id.HypotensionMinEdit));
criticalEdits.add((EditText)findViewById(R.id.GlucoseMaxEdit));
criticalEdits.add((EditText)findViewById(R.id.GlucoseMinEdit));
criticalEdits.add((EditText)findViewById(R.id.Spo2MaxEdit));
criticalEdits.add((EditText)findViewById(R.id.Spo2MinEdit));
criticalEdits.add((EditText)findViewById(R.id.PulseMaxEdit));
        criticalEdits.add((EditText)findViewById(R.id.PulseMinEdit));
criticalEdits.add((EditText)findViewById(R.id.ConductivityMaxEdit));
criticalEdits.add((EditText)findViewById(R.id.ConductivityMinEdit));
criticalEdits.add((EditText)findViewById(R.id.HeatMaxEdit));
criticalEdits.add((EditText)findViewById(R.id.HeatMinEdit));
criticalEdits.add((EditText)findViewById(R.id.AirflowMaxEdit));
criticalEdits.add((EditText)findViewById(R.id.AirflowMinEdit));
criticalEdits.add((EditText)findViewById(R.id.ECGMaxEdit));
criticalEdits.add((EditText)findViewById(R.id.ECGMinEdit));
criticalEdits.add((EditText)findViewById(R.id.EMGMaxEdit));
criticalEdits.add((EditText)findViewById(R.id.EMGMinEdit));
    }
    private void toolsActions()
    {saveButton.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View v) {
             callConfirm(1); }});
    private void JsonPostRequest (final String url, final String
warnMessage)
```

```
{final ProgressDialog pDialog = new ProgressDialog(this);
    pDialog.setMessage("Loading...");
    pDialog.show();
    JsonObjectRequest jsonObjReq = new
JsonObjectRequest(Request.Method.POST,
            url, postObject,
            new Response.Listener<JSONObject>() {
              @Override
              public void onResponse(JSONObject response) {if(new
AlertHelper(context).isConnect())
                  {Log.d("JsonPostOnResponse", "" +
String.valueOf(response));
                    JsonParse parse=new JsonParse();
                    JSONArray array=new JSONArray();
                    array=parse.getJsonArray(response);
                    if
(url.equals("http://beybalik.com.tr/asptest/api/values/TresholdValues")){
              listele(array);
                  }else{
                      String result=new
JsonParse(response).getString("Response");
                      Log.d("result Response",result);
     if(result.equals("true"))
                      {tempValues=new ArrayList<>();
          for (int i=0;i<criticalValues.size();i++)
tempValues.add(criticalValues.get(i)); }
              pDialog.hide();
                  new AlertHelper(context).getErrorMessage("Kaydetme
işleminiz tamamlanmıştır", title); }
                      else
     {pDialog.hide();
                        new
AlertHelper(context).getErrorMessage("Kaytetme işlemi Başarısız. Lütfen
```

```
verileriniz kontrol ediniz.", title); }}}
                    pDialog.hide();}
                }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                new
AlertHelper(context).getErrorMessage(warnMessage,"Uyarı!");
                Log.d("JsonResponNotification"," "+ String.valueOf(error));
                pDialog.hide();}});
        Volley.newRequestQueue(context).add(jsonObjReq); }
    private void listele(JSONArray liste)
    {JSONObject object=new JsonParse().getObjectFromArray(liste, 0);
        if(object!=null&&object.length()>0){
            JsonParse parse=new JsonParse(object);
            for (int i=0; i<criticalNames.size(); i++)
        {criticalValues.add(parse.getString(criticalNames.get(i)));
criticalEdits.get(i).setText(criticalValues.get(i)); }
    private void callConfirm(int button)
    {String message="";
        changes=0;
        for (int i=0;i<criticalNames.size();i++)
        {
        tempValues.add(String.valueOf(criticalEdits.get(i).getText()));
            if(!tempValues.get(i).equals(criticalValues.get(i)))
            {changes++;
  message+="\n"+criticalNames.get(i)+" :  "+criticalValues.get(i)+"  ===>
"+tempValues.get(i); }}
        if (changes>0)
        {new Confirm(message,button).execute();}
        else
        {if(button==0)
            {startActivity(new Intent(context,DoctorScreen.class));}}}
    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event) { if
```

```
(keyCode == KeyEvent.KEYCODE_BACK && event.getRepeatCount()
== 0) {
          callConfirm(0);
          return false; }
       return super.onKeyDown(keyCode, event); }
   class Confirm extends AsyncTask<String,Void,String>
   {boolean onay=false;
      int buttton=1;
      String message="";
      String title="Kritik değerleri değiştirmek istiyor musunuz?";
      public Confirm(String message,int button)
      {this.buttton=button;
         this.message=message; }
      public Confirm(String title, String message)
      {this.message=message;
         this.title=title; }
      private void callPopup()
      {AlertDialog.Builder dialog;
         dialog = new AlertDialog.Builder(context);
         dialog.setCancelable(true);
         dialog.setTitle(title);
         dialog.setMessage(message);
         dialog.setNegativeButton("HAYIR", new
DialogInterface.OnClickListener() {
            @Override public void onClick(DialogInterface dialog, int
which) {
               if (buttton == 0) {
                  if (changes<3) {
      startActivity(new Intent(context, DoctorScreen.class));
                  }else
                  {if(!onay)
                     {dialog.dismiss()   title = "Veriler kaydedilmeyecek.
Sayfada kalmak istiyor musunuz?";
                        onay = true;
```

```
                        changes = 0;
                        callPopup();
                    }else{
      startActivity(new Intent(context, DoctorScreen.class));}}}}});
          dialog.setPositiveButton("EVET", new
DialogInterface.OnClickListener() {
              @Override
    public void onClick(DialogInterface dialog, int which) {
                if (!onay)
                {sendValues();}}});
          dialog.show();}
      @Override
      protected String doInBackground(String... params) {
          return null; }
      @Override
      protected void onPostExecute(String result)
      {callPopup();}}}

public class CustomAdapter extends BaseAdapter
{private LayoutInflater layoutInflater;
   private ArrayList<ViewStructure> viewStructures;
   CustomAdapter(AppCompatActivity
activity,ArrayList<ViewStructure> structure)
{layoutInflater=(LayoutInflater)activity.getSystemService(Context.LAYO
UT_INFLATER_SERVICE);
      viewStructures=structure; }
   @Override
   public int getCount() {
      return viewStructures.size();}
   @Override
   public ViewStructure getItem(int position) {
      return viewStructures.get(position); }
   @Override
   public long getItemId(int position) {
```

```
            return position; }
        @Override
        public View getView(int position, View convertView, ViewGroup
    parent)
        {View rowView;
            rowView=layoutInflater.inflate(R.layout.viewrow,null);
            TextView
    titleView=(TextView)rowView.findViewById(R.id.customTitle);
            TextView
    descriptionView=(TextView)rowView.findViewById(R.id.customDescrip
    tion);
            ViewStructure structure=viewStructures.get(position);
            titleView.setText(structure.getTitle());
            descriptionView.setText(structure.getDescription());
            return rowView; }}
public class DoctorScreen extends AppCompatActivity {
   String phoneId;
   boolean returnResult=true;
   Button myPatientButton;
   Button notificationButton;
   Button messageButton;
   Button logoutButton;
   Context context;
   @Override
   protected void onCreate(Bundle savedInstanceState)
   {super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_doctor_screen);
      teachTools();
      toolActions();}
   public void teachTools()
   {phoneId= Settings.Secure.getString(this.getContentResolver(),
Settings.Secure.ANDROID_ID);
      myPatientButton=(Button)findViewById(R.id.myPatientButton);
```

```
        notificationButton=(Button)findViewById(R.id.notificationButton);
        messageButton=(Button)findViewById(R.id.messageButton);
        logoutButton=(Button)findViewById(R.id.logoutButton);
        context=this;
    }public void toolActions()
    { myPatientButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String url="http://beybalik.com.tr/asptest/api/values/DoctorPatients";
                openNotification("HASTALARIM", url); }});
        notificationButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String
url="http://beybalik.com.tr/asptest/api/values/DoctorNotification";
                openNotification("BİLDİRİMLER",url); }});
        messageButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
              String url="http://beybalik.com.tr/asptest/api/values/MessageToDoctor";
                openNotification("MESAJLAR",url); }});
        logoutButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if(JsonPostRequest())
                {logout(true); }}});}
    public void logout(boolean isConfirm)
    {if (isConfirm)
        {DataBase dataBase=new DataBase(context);
            dataBase.logout();
            Intent intent=new Intent(context,SigninPage.class);
            startActivity(intent); }}
    private boolean JsonPostRequest()
    {
```

```
        String url="http://beybalik.com.tr/asptest/api/values/SafeExit";
        final ProgressDialog pDialog = new ProgressDialog(this);
        pDialog.setMessage("Loading...");
        pDialog.show();
        JSONObject object = new JSONObject();
        try {
           object.put("PhoneID", phoneId);
        } catch (JSONException e) {
           e.printStackTrace();}
        JsonObjectRequest jsonObjReq = new
JsonObjectRequest(Request.Method.POST,
              url, object,
              new Response.Listener<JSONObject>() {
                 @Override
                 public void onResponse(JSONObject response) {
                    if(new AlertHelper(context).isConnect())
                    {Log.d("JsonPostOnResponse", "" + String.valueOf(response));
                       returnResult=true; }
                    pDialog.hide();}}, new Response.ErrorListener() {
           @Override
           public void onErrorResponse(VolleyError error) {
              new Confirm("İnternet bağlantınızda bir sorun var.Yine de çıkmak
istiyor musunuz?").execute();
              Log.d("JsonResponNotification"," "+ String.valueOf(error));
              returnResult=false;
              pDialog.hide();}});
        Volley.newRequestQueue(context).add(jsonObjReq);
        return returnResult; }
     private void openNotification(String PageTitle,String URL)
     {
        Intent intent = new Intent(DoctorScreen.this,NotificationPage.class);
        Bundle info=new Bundle();
        info.putString("PageTitle", PageTitle);
        info.putBoolean("UserType",true);
```

```
        info.putString("URL", URL);
        intent.putExtras(info);
        startActivity(intent);
    }
    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event)
    {if (keyCode == KeyEvent.KEYCODE_BACK && event.getRepeatCount()
== 0) {  return false; }
        return super.onKeyDown(keyCode, event); }
    class Confirm extends AsyncTask<String,Void,String>
    {
        boolean isConfirm=false;
        String message="";
        public Confirm(String message)
        {this.message=message; }
        @Override
        protected String doInBackground(String... params) {
            return null; }
        @Override
        protected void onPostExecute(String result) {
            AlertDialog.Builder dialog;
            dialog = new AlertDialog.Builder(context);
            dialog.setCancelable(true);
            dialog.setTitle("Uyarı");
            dialog.setMessage(message);
            dialog.setNegativeButton("HAYIR", new
DialogInterface.OnClickListener() {
                @Override
    public void onClick(DialogInterface dialog, int which) {}});
            dialog.setPositiveButton("EVET", new
DialogInterface.OnClickListener() {
                @Override
     public void onClick(DialogInterface dialog, int which) {
                    logout(true); }});
```

```
            dialog.show();}}}
    public class MakeMeasurementPage extends AppCompatActivity {
      Button ConductivityButton;
      Button HeatButton;
      Button GlucoseButton;
      Button BloodPressureButton;
      Button PulseoximeterButton;
      Button AirflowButton;
      Button ECGButton;
      Button EMGButton;
      String phoneId;
      Context context=this;
      ArrayList<String>testType;
      @Override
      protected void onCreate(Bundle savedInstanceState) {
         super.onCreate(savedInstanceState);
         setContentView(R.layout.activity_make_measurement_page);
         teachTools();
         toolsAction();}
      private void teachTools()
      {phoneId= Settings.Secure.getString(this.getContentResolver(),
Settings.Secure.ANDROID_ID);
         ConductivityButton=(Button)findViewById(R.id.ConductivityButton);
         HeatButton=(Button)findViewById(R.id.HeatButton);
         GlucoseButton=(Button)findViewById(R.id.GlucoseButton);
         BloodPressureButton=(Button)findViewById(R.id.BloodPressureButton);
         PulseoximeterButton=(Button)findViewById(R.id.PulseoximeterButton);
         AirflowButton=(Button)findViewById(R.id.AirflowButton);
         ECGButton=(Button)findViewById(R.id.ECGButton);
         EMGButton=(Button)findViewById(R.id.EMGButton);
         testType=new ArrayList<>();
         testType.add("Conductivity");
         testType.add("Heat");
         testType.add("Glucose");
```

```
    testType.add("BloodPressure");
    testType.add("Pulseoximeter");
    testType.add("Airflow");
    testType.add("ECG");
    testType.add("EMG");}
private void toolsAction()
{ConductivityButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            JsonPostRequest(0,true); }});
    HeatButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            JsonPostRequest(1,true); }});
    GlucoseButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            JsonPostRequest(2,true); }});
    BloodPressureButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            JsonPostRequest(3,true); }
    PulseoximeterButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            JsonPostRequest(4,true); }});
    AirflowButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            JsonPostRequest(5,false); }});
    ECGButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            JsonPostRequest(6,false); }});
```

```
        EMGButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                JsonPostRequest(7,false); }});}
    private void JsonPostRequest(final int test, final boolean type)
    {
        int timeout=0;
        String url="http://beybalik.com.tr/asptest/api/values/PostQueue";
        final ProgressDialog pDialog = new ProgressDialog(this);
        pDialog.setMessage(testType.get(test)+" ölçümleri yapılıyor...");
        pDialog.show();
        JSONObject object = new JSONObject();
        try {
            object.put("PhoneID", phoneId);
            object.put("Text", testType.get(test));
            if (type)
            {timeout=1;
                object.put("Type","Single");}
            else
            {timeout=3;
                object.put("Type","Chart");}
        } catch (JSONException e) {
            e.printStackTrace();
        }Log.d("gidenVeri", "" + String.valueOf(object));
        JsonObjectRequest jsonObjReq = new
JsonObjectRequest(Request.Method.POST,
                url, object,
                new Response.Listener<JSONObject>() {
                    @Override
                    public void onResponse(JSONObject response) {
                        JsonParse parse = new JsonParse();
                        Log.d("JsonResponse", ""+String.valueOf(response));
                        openPopop(parse.getJsonArray(response),test);
                        pDialog.hide();}
```

```
            }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
    new AlertHelper(context).getErrorMessage("İnternet bağlantı hatası!!",
"Uyarı!");
            Log.d("JsonResponNotification"," "+ String.valueOf(error));
            pDialog.hide();}});
    jsonObjReq.setRetryPolicy(new DefaultRetryPolicy(
            timeout*20000,
            DefaultRetryPolicy.DEFAULT_MAX_RETRIES,
            DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
    Volley.newRequestQueue(context).add(jsonObjReq); }
  private void openChart(ArrayList<String> datas,String testType)
  {
    Intent intent=new Intent(context,ChartPage.class);
    Bundle info=new Bundle();
    info.putStringArrayList("dataArray",datas);
    info.putString("testType",testType);
    intent.putExtras(info);
    startActivity(intent);
  } private void openPopop(JSONArray array,int testType)
  {ArrayList<String>testResults=new ArrayList<>();
    testResults=new JsonParse(array).getArrayList("ListItem");
    String resulItem="";
    if(testType<6)
    {for (int i = 0; i < testResults.size(); i++) {
            resulItem += "\n" + testResults.get(i); }
   new AlertHelper(context).getErrorMessage(resulItem, "Test Sonuçları");
    }else{
        openChart(testResults,this.testType.get(testType));}}}
   public class PatientDetailsPage extends AppCompatActivity {
  String patientID="";
  String phoneID="";
  Context context=this;
```

```
    TextView tcno;
    TextView patName;
    TextView patYas;
    TextView patAdres;
    TextView patPhone;
    TextView relName;
    TextView relDegree;
    TextView relPhone;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_patient_details_page);
        teachTools();
        getInfo();
        JsonPostRequest();}
    private void teachTools()
    {phoneID= Settings.Secure.getString(this.getContentResolver(),
Settings.Secure.ANDROID_ID);
        tcno=(TextView)findViewById(R.id.tcnoView);
        patName=(TextView)findViewById(R.id.hasIsimView);
        patYas=(TextView)findViewById(R.id.yasView);
        patAdres=(TextView)findViewById(R.id.adresView);
        patPhone=(TextView)findViewById(R.id.hastelView);
        relName=(TextView)findViewById(R.id.akrIsimView);
        relDegree=(TextView)findViewById(R.id.akrYakView);
        relPhone=(TextView)findViewById(R.id.akrTelView); }
    private void getInfo()
    {
        Intent intent = getIntent();
        Bundle info=intent.getExtras();
        patientID=info.getString("identityNo");}
    private void JsonPostRequest ()
    {
        String url="http://beybalik.com.tr/asptest/api/values/PatientInformation";
```

```
        final ProgressDialog pDialog = new ProgressDialog(this);
        pDialog.setMessage("Loading...");
        pDialog.show();
        JSONObject object = new JSONObject();
        try {
           object.put("PatientID", patientID); } catch (JSONException e) {
           e.printStackTrace();
        }JsonObjectRequest jsonObjReq = new
JsonObjectRequest(Request.Method.POST,
              url, object,
              new Response.Listener<JSONObject>() {
                 @Override
                 public void onResponse(JSONObject response) {
                    if(new AlertHelper(context).isConnect())
                    {
                       Log.d("JsonPostOnResponse", "" +
String.valueOf(response));
                       JsonParse parse=new JsonParse();
                       pDialog.hide();
                       listele(response); }}}, new Response.ErrorListener() {
           @Override
           public void onErrorResponse(VolleyError error) {
              new AlertHelper(context).getErrorMessage("Bir Hata Oluştu. Bu
yüzden verileri görüntüleyemiyoruz","Uyarı!" );
              Log.d("JsonResponNotification"," "+ String.valueOf(error));
              pDialog.hide();}});
        Volley.newRequestQueue(context).add(jsonObjReq); }
     private void listele(JSONObject object)
     {
        JsonParse parse=new JsonParse(object);
        tcno.setText(patientID);
        patName.setText(parse.getString("Patient_Name"));
        patYas.setText(parse.getString("Patient_Age"));
        patAdres.setText(parse.getString("Patient_Adress"));
```

```
        patPhone.setText(parse.getString("Patient_Phone"));
        relName.setText(parse.getString("Relative_Name"));
        relDegree.setText(parse.getString("Relative_kinship"));
        relPhone.setText(parse.getString("Relative_Phone"));}}
public class PatientScreen extends AppCompatActivity {
    Button makeMeasureButton;
    Button myMessagesButton;
    Button myMeasurementButton;
    Button messageToDoctorButton;
    Button logoutButton;
    Context context=this;
    String userName="";
    String phoneId="";
    boolean returnResult=true;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_patient_screen);
        getPageInfo();
        teachTools();
        toolsAction();}
    private void teachTools()
    {phoneId= Settings.Secure.getString(this.getContentResolver(),
Settings.Secure.ANDROID_ID);
        makeMeasureButton=(Button)findViewById(R.id.makeMeasureButton);
        myMessagesButton=(Button)findViewById(R.id.myMessagesButton);

myMeasurementButton=(Button)findViewById(R.id.myMeasurementButton);
messageToDoctorButton=(Button)findViewById(R.id.messageToDoctorButto
n);
        logoutButton=(Button)findViewById(R.id.logoutPatientButton);  }
    private void toolsAction()
    {
```

```
        makeMeasureButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                openPage(new Intent(context,MakeMeasurementPage.class),""); }
});
        myMessagesButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
            openPage(new
Intent(context,NotificationPage.class),"MESAJLAR");}});
        myMeasurementButton.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View v) {
                openPage(new Intent(context,MeasurementPage.class),userName);
}});
        messageToDoctorButton.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                openPage(new Intent(context,SendMessagePage.class),""); }});
        logoutButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if(JsonPostRequest())
                {logout(true); }}});}
    private void openPage(Intent intent,String title)
    {Bundle info=new Bundle();
        info.putString("PageTitle", title);
        info.putBoolean("UserType", false);info.putString("URL",
"http://beybalik.com.tr/asptest/api/values/MessageToPatient");
        intent.putExtras(info);
        startActivity(intent); }
    private void getPageInfo()
```

```
    {Intent intent=getIntent();
        Bundle info=intent.getExtras();
        userName=info.getString("Username");}
    private boolean JsonPostRequest()
    {
        String url="http://beybalik.com.tr/asptest/api/values/SafeExit";
        final ProgressDialog pDialog = new ProgressDialog(this);
        pDialog.setMessage("Loading...");
        pDialog.show();
        JSONObject object = new JSONObject();
        try {
            object.put("PhoneID", phoneId);
        } catch (JSONException e) {
            e.printStackTrace();}
        JsonObjectRequest jsonObjReq = new
JsonObjectRequest(Request.Method.POST, url, object,
                new Response.Listener<JSONObject>() {
                    @Override
                    public void onResponse(JSONObject response) {
                        if(new AlertHelper(context).isConnect())
                        {Log.d("JsonPostOnResponse", "" + String.valueOf(response));
                            returnResult =true; }
                        pDialog.hide();}}, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                new Confirm("İnternet bağlantınızda bir sorun var. Yine de çıkmak
istiyor musunuz?").execute();
                Log.d("JsonResponNotification"," "+ String.valueOf(error));
                returnResult=false;
                pDialog.hide();}});
        Volley.newRequestQueue(context).add(jsonObjReq);
        return returnResult; }
    public void logout(boolean isConfirm)
    {if (isConfirm)
```

```
        {
            DataBase dataBase=new DataBase(context);
            dataBase.logout();
            Intent intent=new Intent(context,SigninPage.class);
            startActivity(intent); }}
    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event) {
  return !(keyCode == KeyEvent.KEYCODE_BACK &&
event.getRepeatCount() == 0) && super.onKeyDown(keyCode, event); }
    class Confirm extends AsyncTask<String,Void,String>
    {
        String message="";
        public Confirm(String message)
        {this.message=message; }
        @Override
        protected String doInBackground(String... params) {
            return null; }
        @Override
        protected void onPostExecute(String result) {
            AlertDialog.Builder dialog;
            dialog = new AlertDialog.Builder(context);
            dialog.setCancelable(true);
            dialog.setTitle("Uyarı");
            dialog.setMessage(message);
            dialog.setNegativeButton("HAYIR", new
DialogInterface.OnClickListener() {@Override
                public void onClick(DialogInterface dialog, int which) {}});
            dialog.setPositiveButton("EVET", new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    logout(true); }});
            dialog.show();}}}
```

```
public class ViewPatientPage extends AppCompatActivity
{
    Button measurementButton;
    Button criticalValuesButton;
    Button patientDetailsButton;
    Button sendMesageButton;
    TextView titleView;
    String patientID;
    String pageTitle="";
    Context context=this;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_view_patient_page);
        getPageInfo();
        teachTool();
        toolsActions();}
    private void getPageInfo(){
        Intent intent=getIntent();
        Bundle info = intent.getExtras();
        patientID=info.getString("identityNo");
        pageTitle=info.getString("patientName");}
    private void teachTool()
    {
        measurementButton=(Button)findViewById(R.id.measurementButton);
        criticalValuesButton=(Button)findViewById(R.id.criticalValuesButton);
        patientDetailsButton=(Button)findViewById(R.id.patientDetailsButton);
        sendMesageButton=(Button)findViewById(R.id.sendMesageButton);
        titleView=(TextView)findViewById(R.id.hastaAdi);
        titleView.setText(pageTitle); }
    private void toolsActions()
    {
        measurementButton.setOnClickListener(new View.OnClickListener() {
            @Override
```

```
            public void onClick(View v) {
             goPage(new Intent(context,MeasurementPage.class),pageTitle); }});
        criticalValuesButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
           goPage(new Intent(context,CriticalValuesPage.class),pageTitle); }});
        patientDetailsButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
            goPage(new Intent(context,PatientDetailsPage.class),pageTitle); }});
        sendMesageButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
           goPage(new Intent(context,SendMessagePage.class),pageTitle); } });}
    private void goPage(Intent intent,String title)
    {
        Bundle info=new Bundle();
        info.putString("identityNo", patientID);
        info.putString("PageTitle", title);
        info.putBoolean("UserType",true);
        intent.putExtras(info);
        startActivity(intent); }}
public class ViewPatientPage extends AppCompatActivity
{
    Button measurementButton;
    Button criticalValuesButton;
    Button patientDetailsButton;
    Button sendMesageButton;
    TextView titleView;
    String patientID;
    String pageTitle="";
    Context context=this;
    @Override
```

```
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_view_patient_page);
   getPageInfo();
   teachTool();
   toolsActions();}
private void getPageInfo()
{
   Intent intent=getIntent();
   Bundle info = intent.getExtras();
   patientID=info.getString("identityNo");
   pageTitle=info.getString("patientName");}
private void teachTool()
{
   measurementButton=(Button)findViewById(R.id.measurementButton);
   criticalValuesButton=(Button)findViewById(R.id.criticalValuesButton);
   patientDetailsButton=(Button)findViewById(R.id.patientDetailsButton);
   sendMesageButton=(Button)findViewById(R.id.sendMesageButton);
   titleView=(TextView)findViewById(R.id.hastaAdi);
   titleView.setText(pageTitle); }
private void toolsActions()
{measurementButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
     goPage(new Intent(context,MeasurementPage.class),pageTitle); }});
   criticalValuesButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
    goPage(new Intent(context,CriticalValuesPage.class),pageTitle); }});
   patientDetailsButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
    goPage(new Intent(context,PatientDetailsPage.class),pageTitle); }});
   sendMesageButton.setOnClickListener(new View.OnClickListener() {
```

```
        @Override
        public void onClick(View v) {
        goPage(new Intent(context,SendMessagePage.class),pageTitle); }});
    }
    private void goPage(Intent intent,String title)
    {
      Bundle info=new Bundle();
      info.putString("identityNo", patientID);
      info.putString("PageTitle", title);
      info.putBoolean("UserType",true);
      intent.putExtras(info);
      startActivity(intent); }}

public class NotificationPage extends AppCompatActivity {
    String phoneId;
    Context context=this;
    String pageTitle="";
    String doctorTcNo="";
    TextView notificationTextView;
    ArrayList<ViewStructure> viewList;
    ViewStructure viewStructure;
    ArrayList<String>patientName;
    ArrayList<String>identityNo;
    ArrayList<String>message;
    ArrayList<String>recievedDate;
    String url;
    boolean usertype=false;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_listviews_page);
      getPageInfo();
      phoneId= Settings.Secure.getString(this.getContentResolver(),
Settings.Secure.ANDROID_ID);
```

```
        JsonPostRequest();}
    private void getPageInfo()
    {
        Intent intent=getIntent();
        Bundle info = intent.getExtras();
        pageTitle=info.getString("PageTitle");
        usertype=info.getBoolean("UserType");
        url=info.getString("URL");

notificationTextView=(TextView)findViewById(R.id.notificationTextView);
        notificationTextView.setText(pageTitle); }
    private void JsonPostRequest ()
    {
        final ProgressDialog pDialog = new ProgressDialog(this);
        pDialog.setMessage("Loading...");
        pDialog.show();
        JSONObject object = new JSONObject();
        try {
            object.put("PhoneID", phoneId); } catch (JSONException e) {
            e.printStackTrace();}
        JsonObjectRequest jsonObjReq = new
JsonObjectRequest(Request.Method.POST,
                url, object,
                new Response.Listener<JSONObject>() {
                    @Override
                    public void onResponse(JSONObject response) {
                        if(new AlertHelper(context).isConnect())
                        {Log.d("JsonPostOnResponse",""+ String.valueOf(response));
                            JsonParse parse=new JsonParse();
                            listele(parse.getJsonArray(response)); }
                        pDialog.hide();}}, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                new AlertHelper(context).getErrorMessage("Bir Hata Oluştu. Bu
```

```
yüzden verileri görüntüleyemiyoruz","Uyarı!");
                Log.d("JsonResponNotification"," "+ String.valueOf(error));
                pDialog.hide();}});
        Volley.newRequestQueue(context).add(jsonObjReq); }
    private void listele(JSONArray liste)
    {
        JsonParse parse=new JsonParse(liste);
        patientName=parse.getArrayList("PatientName");
        identityNo=parse.getArrayList("PatientTcNo");
        message=parse.getArrayList("Message");
        recievedDate=parse.getArrayList("ReceivedDate");
        viewList=new ArrayList<>();
        ListView listView=(ListView)findViewById(R.id.notificationListView);
if((message!=null&&message.size()>0)||(patientName!=null&&patientName.si
ze()>0))
        {switch (pageTitle)
            {case "MESAJLAR":
                    messageList();
                    break;
                case "BİLDİRİMLER":
                    messageList();
                    break;
                case "HASTALARIM":
                    patientList();
                    break; }
            CustomAdapter customAdapter = new CustomAdapter(this,viewList);
            listView.setAdapter(customAdapter);
            listView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
                @Override
                public void onItemClick(AdapterView<?> parent, View view, int
position, long id) {
                    if (usertype)
{openPatientView(identityNo.get(position),patientName.get(position));}}});
```

```
    }
    else
    {notificationTextView.setText(pageTitle+"\n \n \nListe boş");}}
private void openPatientView(String idNo,String patientN)
{
    Intent intent=new Intent(this,ViewPatientPage.class);
    Bundle info=new Bundle();
    info.putString("identityNo",idNo);
    info.putString("patientName",patientN);
    intent.putExtras(info);
    startActivity(intent); }
private void patientList()
{
    for(int i=0;i<patientName.size();i++)
    {
        viewStructure=new ViewStructure();
        String tittle="";
        String description="";
        tittle=patientName.get(i);
        description=identityNo.get(i);
        viewStructure.setTitle(tittle);
        viewStructure.setDescription(description);
        viewList.add(viewStructure); }}
private void messageList()
{
    if(usertype)
    {for(int i=0;i<patientName.size();i++)
        {viewStructure=new ViewStructure();
            String tittle="";
            String description="";
            tittle=patientName.get(i)+" ("+identityNo.get(i)+")";
            description=recievedDate.get(i)+"\n"+message.get(i);
            viewStructure.setTitle(tittle);
            viewStructure.setDescription(description);
```

```
        viewList.add(viewStructure); }}
else
{
    for (int i = 0; i < message.size(); i++) {
        viewStructure = new ViewStructure();
        String tittle = "";
        String description = "";
        tittle =recievedDate.get(i);
        description = message.get(i);
        viewStructure.setTitle(tittle);
        viewStructure.setDescription(description);
        viewList.add(viewStructure);}}}}
```

ÖZGEÇMİŞ

Ad-Soyad : Derya KARABAK

Doğum Yeri ve Tarihi : Trabzon/ TÜRKİYE

E-mail : deryakarabak@gmail.com

ÖĞRENİM DURUMU:

Lisans : 2011, Doğu Akdeniz Üniversitesi, Uygulamalı Matematik ve Bilgisayar

Yüksek Lisans: 2016, İstanbul Aydın Üniversitesi, Bilgisayar Mühendisliği