



GÖMÜLÜ SİSTEM ÇÖZÜMLEMELERİ

Dr. Öğr. Üyesi Fırat AYDEMİR

Uyku Apnesi Tespit Sitemi

2022-2023 Bahar Dönemi

22/05/2023-05/06/2023

7.Rapor

HASAN MÜNİR DOĞRUEL-201913171801

YUSUF SARUKAN-201813171058

EMİRHAN AKDİN-201813171049

GİRİŞ

Projenin bu rapor döneminde, bir önceki raporda bahsettiğimiz Web Socket serverını oluşturduk. Web Socket'in veri göndereceği hasta uygulamasının tasarımını yeniledik. Client kısmını oluşturduk. Apne Event'i geldiğinde verilerin realtime Database'e gönderilmesini sağladık.

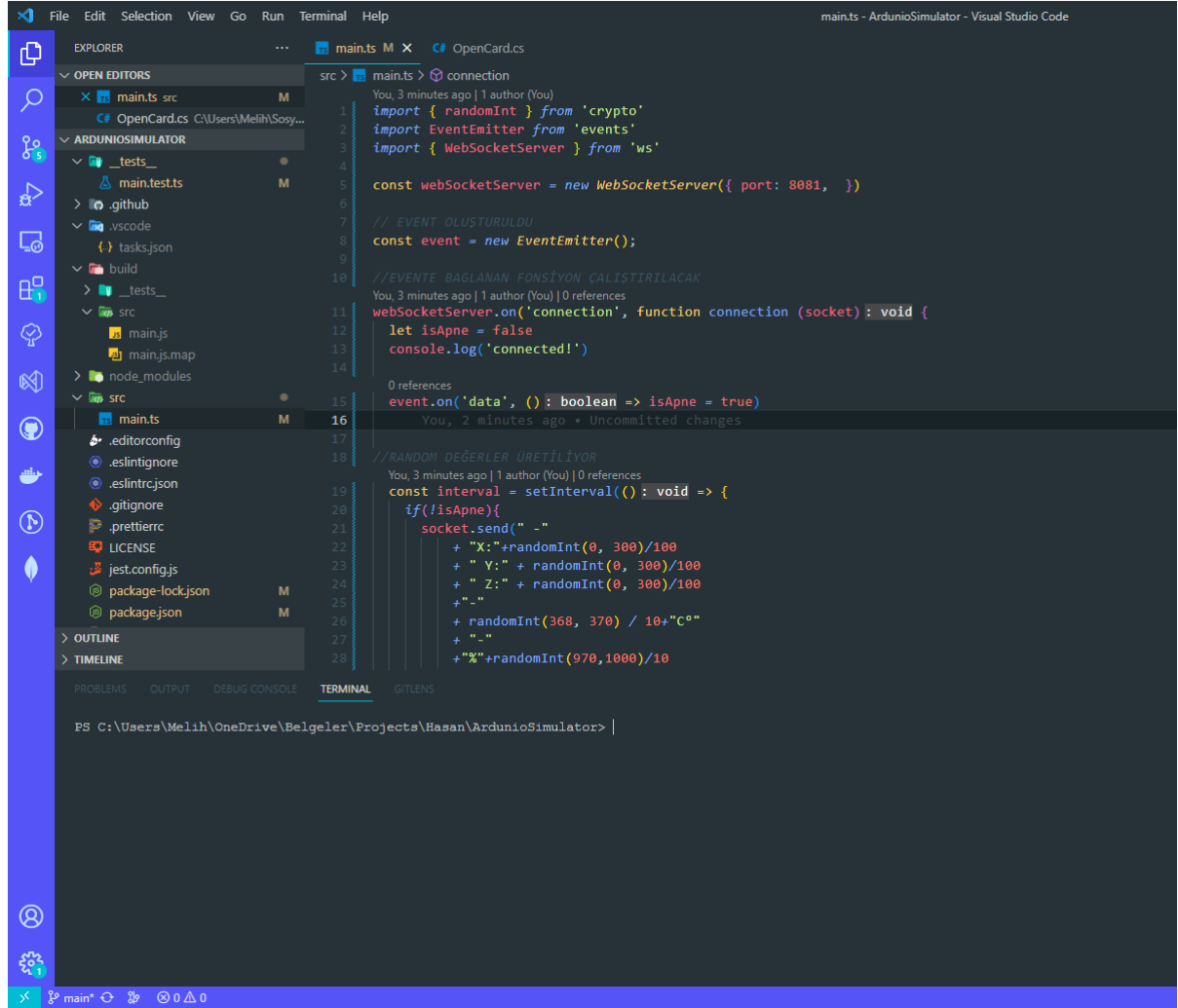
İŞ PLANI

Yapılacak iş	Rapor 1	Rapor 2	Rapor 3	Rapor 4	Rapor 5	Rapor 6	Rapor 7
Firestore Kurulumu, Normalizasyon	X						
Firestore, mobil bağlantısı		X					
Windows Form Tasarımı			X				
Windows Form Firestore bağlantısı				X			
Güncellemeler					X		
Veri taklit edecek server						X	
Verilerin yakalanması ve kullanıcı etkileşimi							X

DipNot: Rapor 5 ve Rapor 6 dönemleri değiştiği için 1 rapor dönemimizi önceki yapıtıklarımızı güncellemeye harcadık.

YAPILAN İŞLER

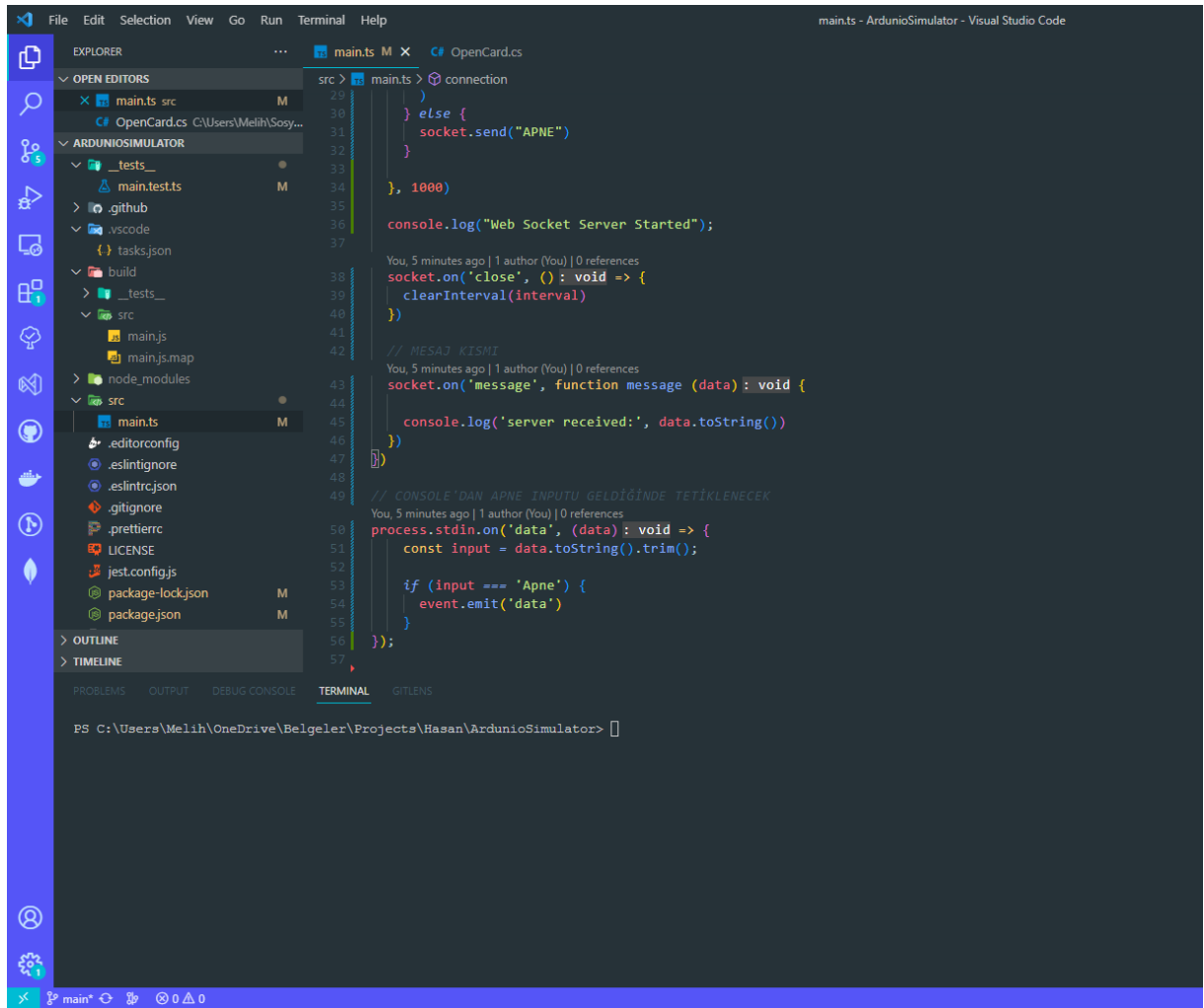
1.Web Socket



```
1  import { randomInt } from 'crypto'
2  import EventEmitter from 'events'
3  import { WebSocketServer } from 'ws'
4
5  const webSocketServer = new WebSocketServer({ port: 8081, })
6
7  // EVENT OLUŞTURULDU
8  const event = new EventEmitter();
9
10 //EVENTE BAĞLANAN FONKSİYON ÇALIŞTIRILACAK
11 You, 3 minutes ago | 1 author (You) | 0 references
12 webSocketServer.on('connection', function connection (socket) : void {
13   let isApne = false
14   console.log('connected!')
15
16   0 references
17   event.on('data', () : boolean => isApne = true)
18   You, 2 minutes ago • Uncommitted changes
19
20 //RANDOM DEĞERLER ÜRETİLİYOR
21 You, 3 minutes ago | 1 author (You) | 0 references
22 const interval = setInterval() : void => {
23   if(!isApne){
24     socket.send("-"
25       + "X:" + randomInt(0, 300)/100
26       + " Y:" + randomInt(0, 300)/100
27       + " Z:" + randomInt(0, 300)/100
28       + "-"
29       + randomInt(368, 370) / 10 + "C°"
30       + "-"
31       + "X:" + randomInt(970,1000)/10
```

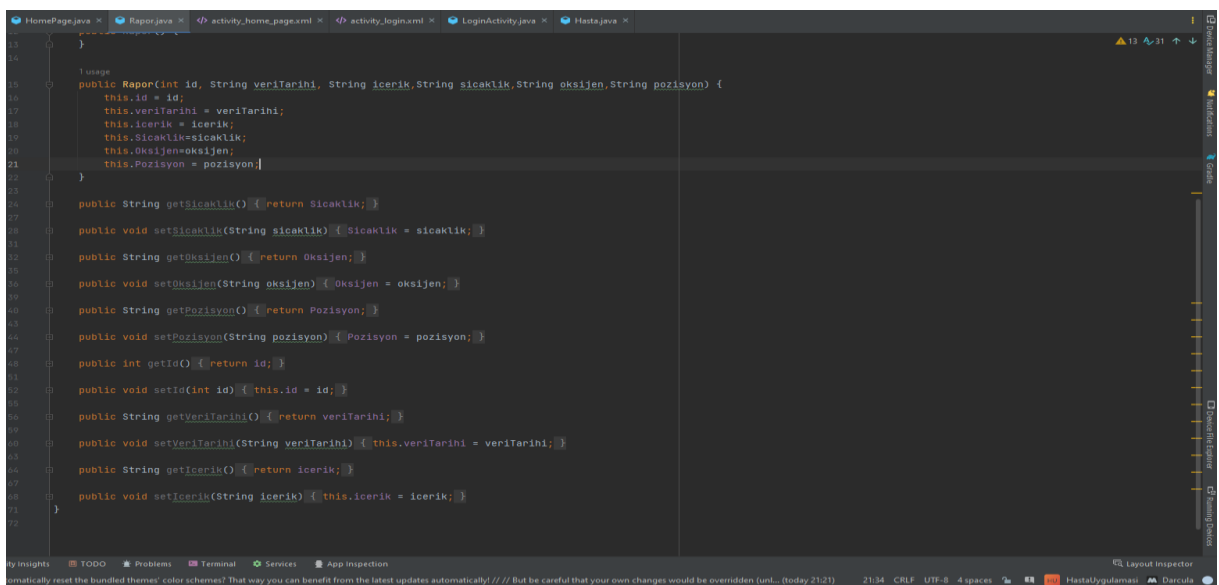
127.0.0.1 IP adresine 8081 portunda bir socket açtık. Bir event tanımladık (Konsol Dinleyici). Gerekli bağlantı kodlarını yazdık. Bir Client'in bağlanması halidne "connect" mesajı verdik.

Son olarak Clien'te göndermek üzere rastgele değerler ürettik.



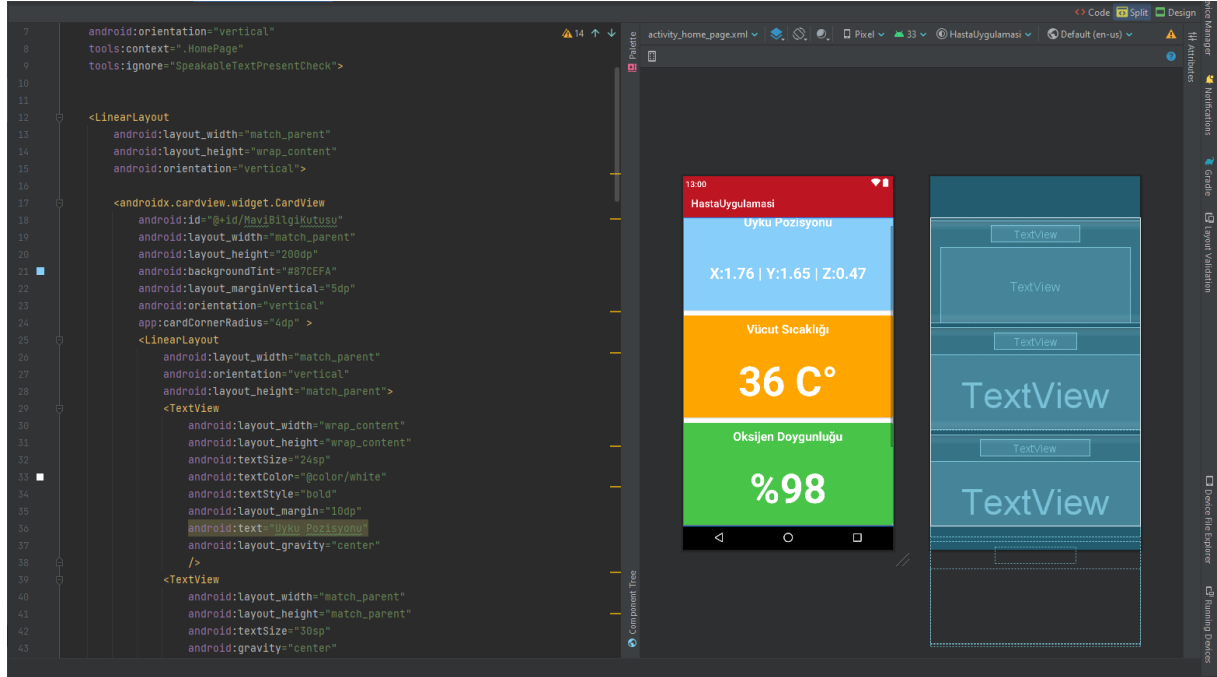
Terminale Apne yazıldığında bir önceki fotoğraftaki Event'i doldurduk. Burası bir tetikleyici görevi görüyor. Terminale Apne yazıldığında simüle edilen verilerin gönderilmesi duruyor. Son alınan veriler (Apne'nin gerçekleştiği anın verilerini) hem doktora göndermek üzere önce hastaya sonra ise hastadan doktora (Windows Form) gönderiyor.

2.Rapor Sınıf

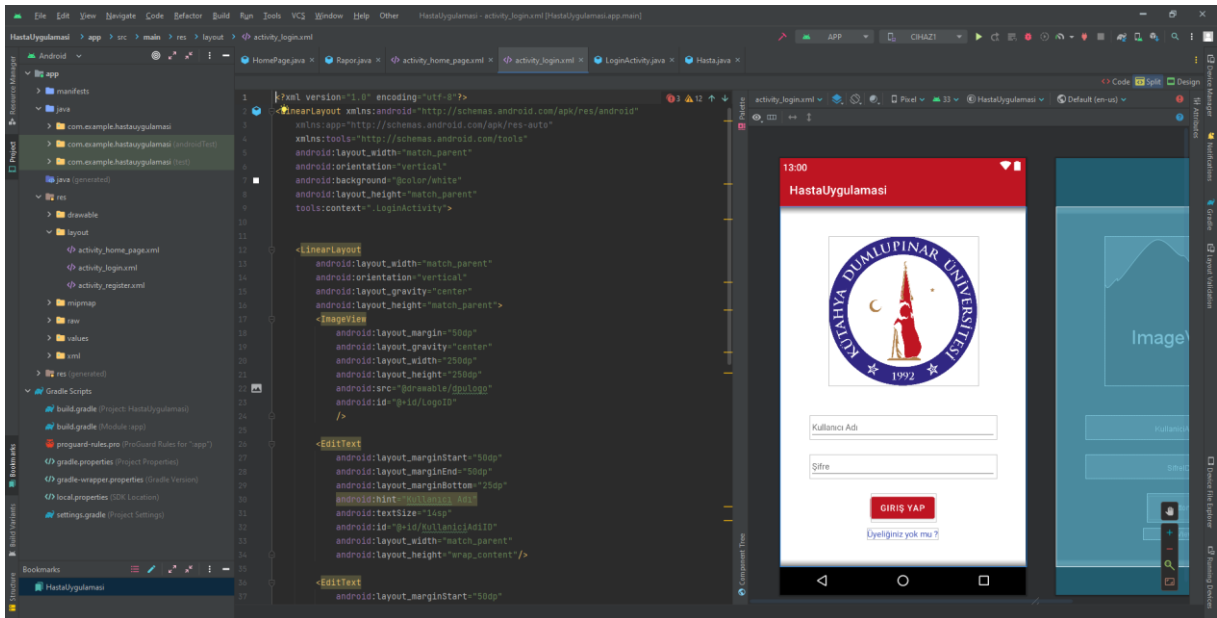


C# Windows Form uygulamasında tasarlanan rapor sınıfının veri tutarlılığı gereği aynı şekilde yazıldı.

2.1.Tasarım Güncellemeleri



Tasarım tekrardan değiştirildi ve hastanın o an ki durumunu gösteren İYİ, KÖTÜ CardView eklendi



Login ekranı üniversitenin logosundaki renklere göre tekrardan revize edildi.

2.2.LoginActivity Düzeltmesi ve Firebase Bağlantısı

```
25
26 </> public class LoginActivity extends AppCompatActivity {
27
28     1 usage
29     private final String path = "https://uykuapnesitespitsistemi-default-rtdb.firebaseio.com/";
30     2 usages
31     private EditText kullaniciAdiEdit, sifreEdit;
32     2 usages
33     private Button girisYapButton;
34     2 usages
35     private TextView uyelikTextView;
36     2 usages
37     private boolean control = false;
38
39     3 usages
40     private List<Hasta> hastaList = new ArrayList<>();
41     6 usages
42     private List<Users> usersList = new ArrayList<>();
43
44     private String hastaKey;
45     private String doktorID;
46
47     @Override
48     protected void onCreate(Bundle savedInstanceState) {
49         super.onCreate(savedInstanceState);
50         setContentView(R.layout.activity_login);
51         kullaniciAdiEdit = findViewById(R.id.kullaniciAdiID);
52         sifreEdit = findViewById(R.id.sifreID);
53         girisYapButton = findViewById(R.id.girisYapID);
54         uyelikTextView = findViewById(R.id.uyelikYokMuID);
55
56         FirebaseDatabase database = FirebaseDatabase.getInstance(path);
57         DatabaseReference myRef = database.getReference();
58
59         uyelikTextView.setOnClickListener(view ->{
60             Snackbar.make(view, text "Üyelik için doktorunuzla iletişime geçiniz!", Snackbar.LENGTH_LONG).show();
61         });
62     }
63 }
```

Gerekli listelerin, butonların, TextView'lerin ve veritabanından gelen geçici keylerin tutulacağı değişkenler tanımlandı.

```
girisYapButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        for (int i = 0; i < usersList.size(); i++) {
            DatabaseReference rootRef = FirebaseDatabase.getInstance().getReference();
            DatabaseReference usersRef =
                rootRef.child( pathString: "Users").child( pathString: usersList.get(i).getId() + "").child( pathString: "/HastaList");
            String index = usersList.get(i).getId();
            ValueEventListener valueEventListener = new ValueEventListener() {
                2 usages
                @Override
                public void onDataChange(DataSnapshot dataSnapshot) {
                    usersList.clear();
                    for (DataSnapshot ds : dataSnapshot.getChildren()) {
                        String ad = ds.child( path: "Name").getValue(String.class);
                        String soyad = ds.child( path: "Soyad").getValue(String.class);
                        String id = ds.child( path: "ID").getValue(String.class);
                        Hasta h = new Hasta();
                        h.setDoktorID(index);
                        h.setKey(ds.getKey());
                        h.setName(ad);
                        h.setSoyad(soyad);
                        h.setId(id);
                        hastaList.add(h);
                    }
                }
            };
            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {
                Log.d( tag: "TAG", databaseError.getMessage()); //Don't ignore errors!
            };
            usersRef.addListenerForSingleValueEvent(valueEventListener);
        }
        String userName = kullaniciAdiEdit.getText().toString();
        String password = sifreEdit.getText().toString();
        if (!validateData(userName, password)) return;
    }
});
```

Burada giriş yap butonuna tıklandığında gerçek zamanlı veritabanına bir istek atılıyor. Görsel nesnelere girilen kullanıcı adı ve parola hasta listesinde aranıyor. Hasta bulunduğu takdirde, hasta objesi dolduruluyor ve bir sonraki aktiviteye gönderilmek üzere hazır hale getiriliyor. Olası bir Request hatasında veritabanından gelen hata mesajı loglara kayıt ediliyor.

3.Web Socket ve Client

```
62 websocketClient = new WebSocketClient(uri) {
63     1 usage
64     @Override
65     public void onOpen() {
66         System.out.println("onOpen");
67         websocketClient.send("Telefon bağlandı!");
68     }
69
70     1 usage
71     @Override
72     public void onTextReceived(String message) {
73         System.out.println(message.toString());
74         if(message.equals("APNE")){
75             mediaPlayer.start();
76
77             DatabaseReference rootRef = FirebaseDatabase.getInstance().getReference();
78             DatabaseReference usersRef =
79                 rootRef.child( pathString: "Users").child(idData)
80                 .child( pathString: "HastaList").child(heyData).child( pathString: "RaporList").child(
81                     pathString: "0"
82                 );
83
84             usersRef.setValue(new Rapon( id: 0, venTarih: "12.06.2023", icerik: "Apne Yakalandı", sicaklik: "37.26", oksijen: "93", pozisyon: "X:0.2,Y:0.3,Z:0.6"));
85             saglikDurumu.setText("KÖTÜ");
86             websocketClient.close( timeout: 0, code: 0, reason: "");
87         }
88         return;
89
90         try {
91             String[] messages = message.split( regex: "[-]");
92             positions.setText(messages[1]);
93             temperatures.setText(messages[2]);
94             saturation.setText(messages[3]);
95         }catch (Exception ignoredException){}
96     }
97
98     1 usage
99     @Override
```

Web Socket nesnesi, web socket sınıfından türetiliyor. Dinlenecek olan IP ve Port numarası URI formatında WebSocketClient sınıfına parametre olarak veriliyor. Bağlantı durumunda onOpen methodu çalışıyor ve servera telefon bağlandı mesajı gönderiliyor. onTextReceiver methodu gelen mesajı (String array) Message.Split yardımı ile parçalıyor. Regex olarak ("[-]") belirlendiği için bu regexden parçalanmış mesaj TextView'lere yazdırılıyor.

```

81
82         usersRef.setValue(new Rapor( id: 0, venTarihi: "12.06.2023", icerik: "Apne Yakalandı", sicaklik: "37.2C", oksijen: "%93", pozisyon: "X:0.2,Y:0.3,
83         saglikDurumu.setText("KÖTÜ!");
84         websocketClient.close( timeout: 0, code: 0, reason: "");
85         return;
86     }
87
88     try {
89         String[] messages = message.split( regex: " " );
90         positions.setText(messages[1]);
91         temperatures.setText(messages[2]);
92         saturation.setText(messages[3]);
93     } catch (Exception ignoredException){}
94 }
95
96
97 1 usage
98 @Override
99 public void onBinaryReceived(byte[] data) { System.out.println("onBinaryReceived"); }
100
101 1 usage
102 @Override
103 public void onPingReceived(byte[] data) { System.out.println("onPingReceived"); }
104
105 1 usage
106 @Override
107 public void onPongReceived(byte[] data) { System.out.println("onPongReceived"); }
108
109 2 usages
110 @Override
111 public void onException(Exception e) { System.out.println(e.getMessage()); }
112
113 1 usage
114 @Override
115 public void onCloseReceived(int reason, String description) {
116     System.out.println("onCloseReceived");
117 }
118
119 };

```

Serverdan gelen tetikleyici algılandığında, serverdan gelen veriler HardText halinde Rapor sınıfının Constructor'a (Kurucu Method) ekleniyor. Sağlık durumunu tutan CardView üzerindeki TextView görsel nesnesi KÖTÜ değerini alıyor, Web Socket kapatılıyor ve veriler gerçek zamanlı veritabanına yazılıyor.