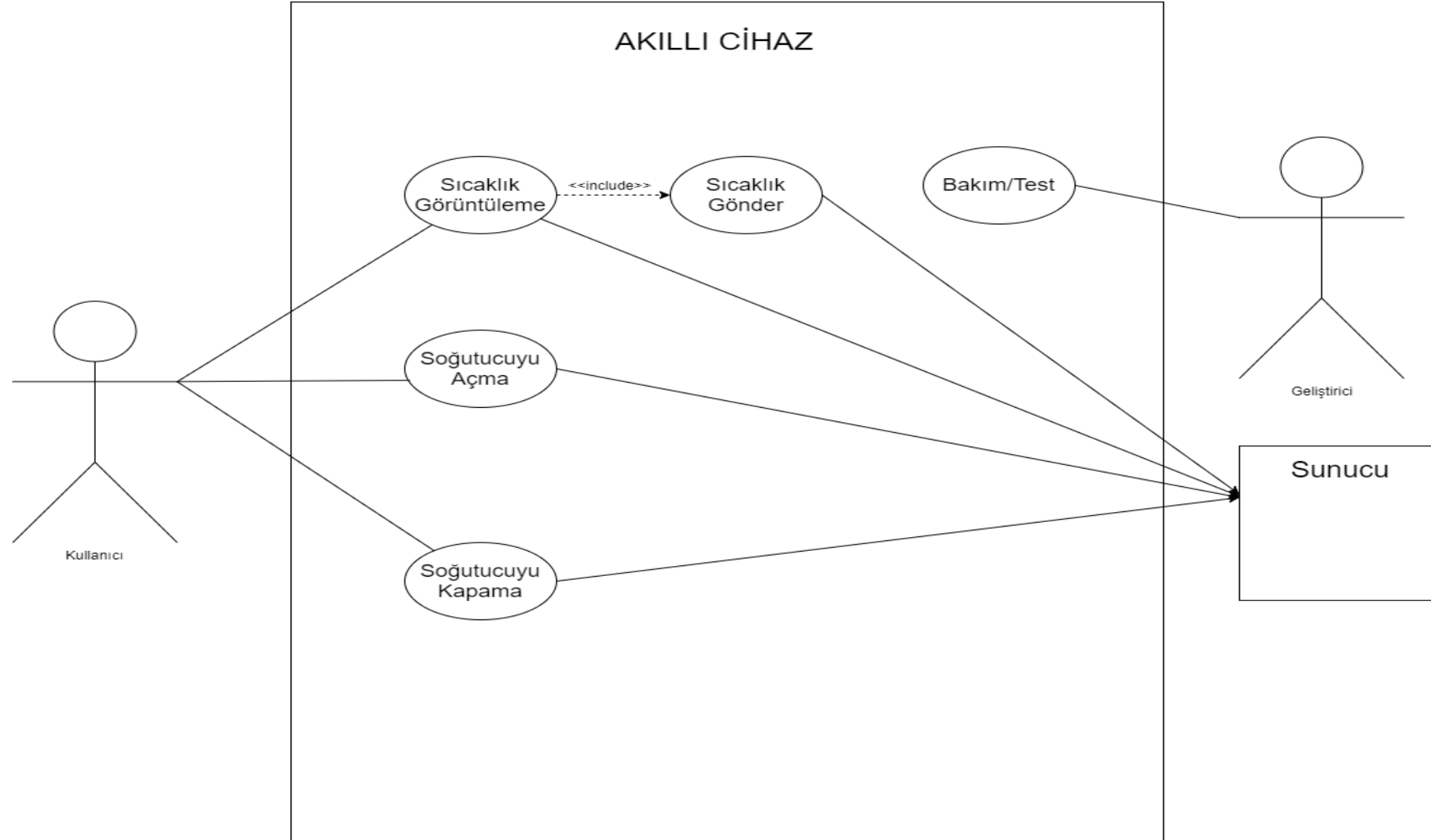


# Nesne Yönelimli Analiz ve Tasarım Proje Ödevi

Tahir Uzelli 1B  
G181210005  
2. Öğretim

[tahir.Uzelli@ogr.sakarya.edu.tr](mailto:tahir.Uzelli@ogr.sakarya.edu.tr)

# Use Case Diyagramı



# Metinsel Kullanım Durumları

- Eşsiz bir ad: Sıcaklık Görüntüleme
- Ağ Arayüzünden sıcaklık görüntüleme işlemini tanımlar.
- 08.05.2020 v1.8 kullanıcı54
- ilgili aktörler: Kullanıcı, Akıllı Cihaz
- Giriş koşulu: Kullanıcı ağ arayüzünden sıcaklık görüntüleme seçeneğini seçer
- Çıkış koşulu: Kullanıcı sıcaklık görüntüle penceresini kapatır
- Olay akışı: Kullanıcı, Kullanıcı Ağ arayüzünden "Sıcaklığı Görüntüle" seçeneğine basar. İşlem birimine sıcaklığı görüntüleme isteği iletilir. İşlem birimi algılayıcıya sıcaklığı algılama komutu verir. Sıcaklık algılayıcı tarafından algılanır ve bu bilgi işlem birimi aracılığı ile arayüze iletilir.
- Özel gereksinimler: UI gereksinimleri

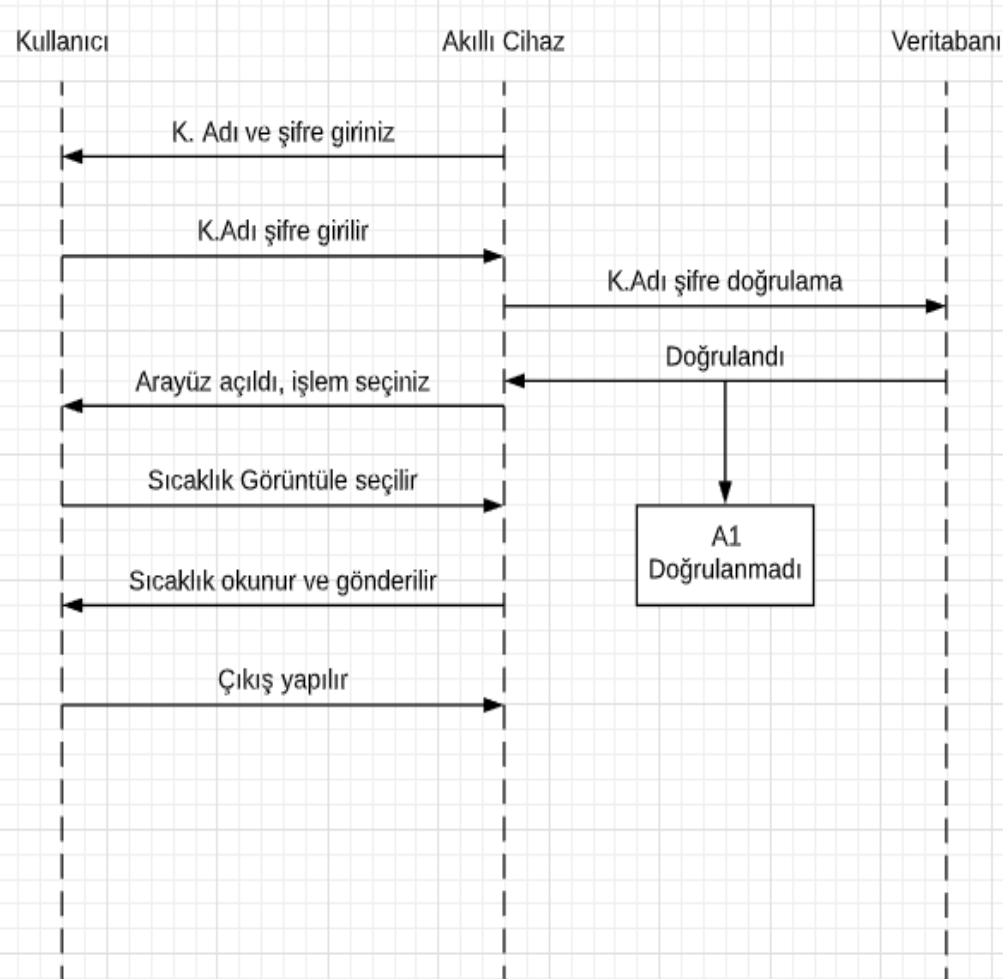
# (Devamı)

- \*"Soğutucuyu Açma" Kullanım Durumu
- \*Eşsiz Bir ad: Soğutucuyu Açma
- \*Kullanıcı ağ arayüzünden soğutucuyu açma işlemini tanımlar
- \*08.05.2020 v1.8 kullanıcı54
- \*İlgili aktörler: Soğutucu kullanıcısı
- \*Giriş Koşulu: Kullanıcı ağ arayüzünden "Soğutucuyu Açma" seçeneğini seçer
- \*Çıkış Koşulu: Soğutucu açılır
- \*Olay akışı: Kullanıcı, Kullanıcı ağ arayüzünden "Soğutucuyu Açma" seçeneğine basar. İşlem birimine soğutucuyu açma isteği iletilir. İşlem birimi Eyleyiciye soğutucuyu açma komutu verir. Soğutucu açılır ve bu bilgi işlem birimi aracılığı ile arayüze iletilir.
- \*Özel gereksinimler: UI gereksinimleri, soğutucunun kapalı olması

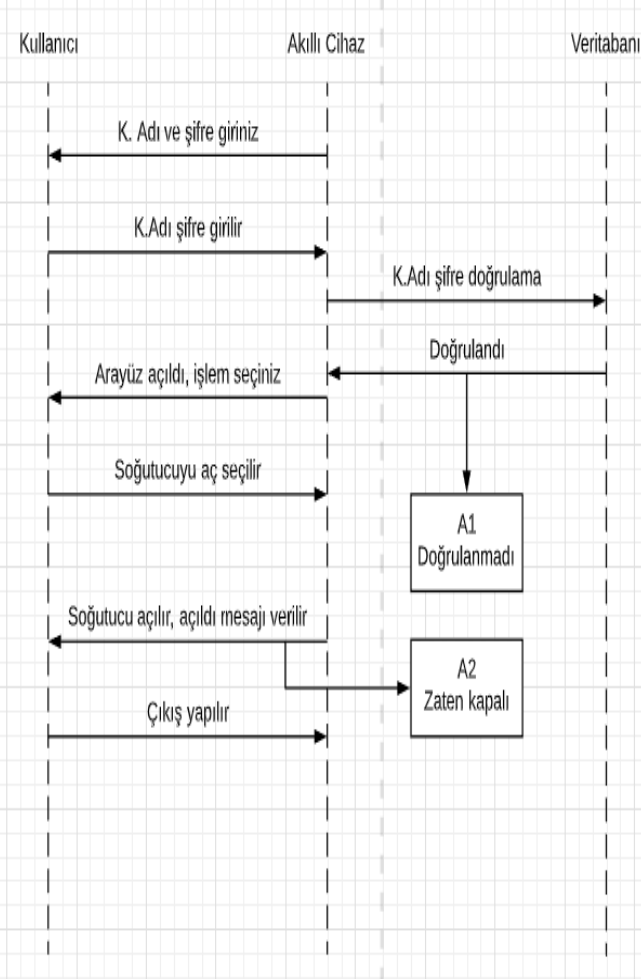
- \*"Soğutucuyu Kapama" Kullanım Durumu
- \*Eşsiz Bir ad: Soğutucuyu Kapama
- \*Kullanıcı ağ arayüzünden soğutucuyu kapama işlemini tanımlar
- \*08.05.2020 v1.8 kullanıcı54
- \*İlgili aktörler: Soğutucu kullanıcısı
- \*Giriş Koşulu: Kullanıcı ağ arayüzünden "Soğutucuyu Kapama" seçeneğini seçer
- \*Çıkış Koşulu: Soğutucu kapanır
- \*Olay akışı: Kullanıcı, Kullanıcı ağ arayüzünden "Soğutucuyu Kapama" seçeneğine basar. İşlem birimine soğutucuyu kapama isteği iletilir. İşlem birimi Eyleyiciye soğutucuyu kapama komutu verir. Soğutucu kapanır ve bu bilgi işlem birimi aracılığı ile arayüze iletilir.
- \*Özel gereksinimler: UI gereksinimleri, soğutucunun açık olması

# Sıralama Şemaları

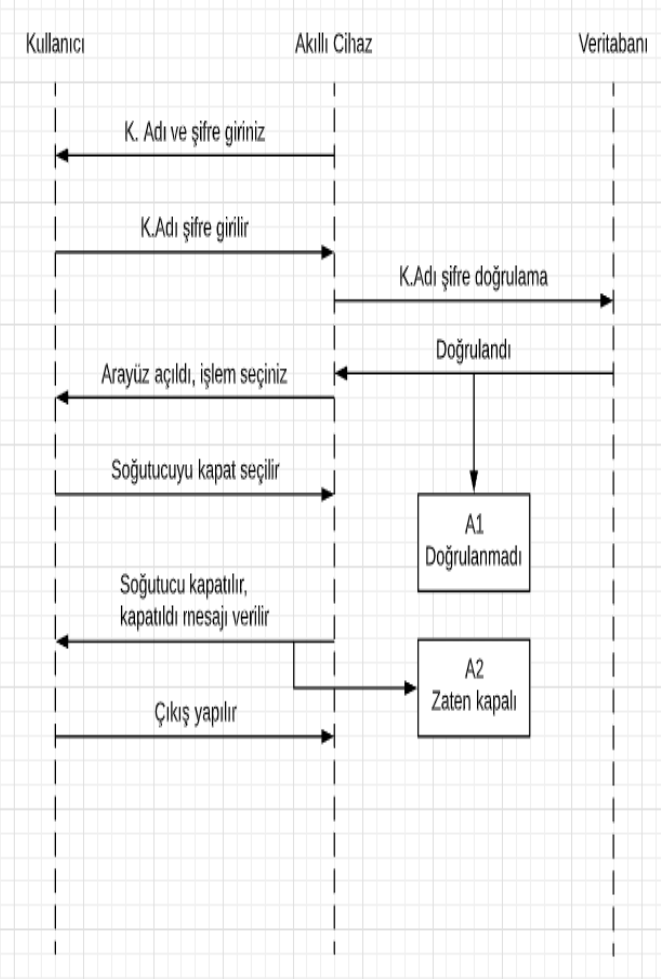
Sıcaklık görüntüle



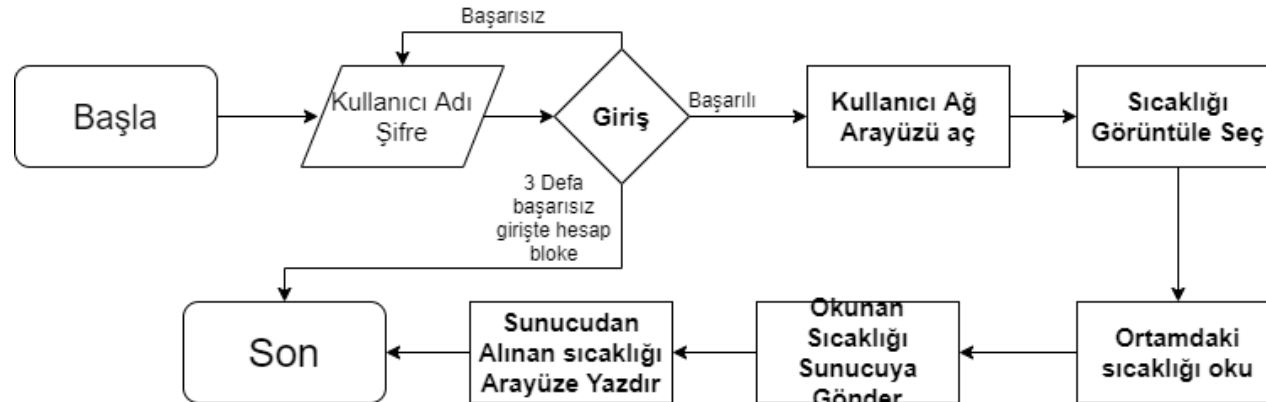
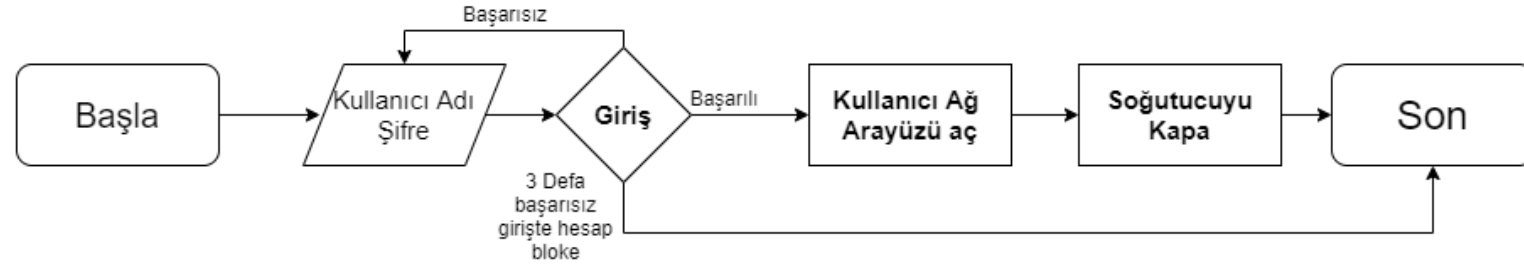
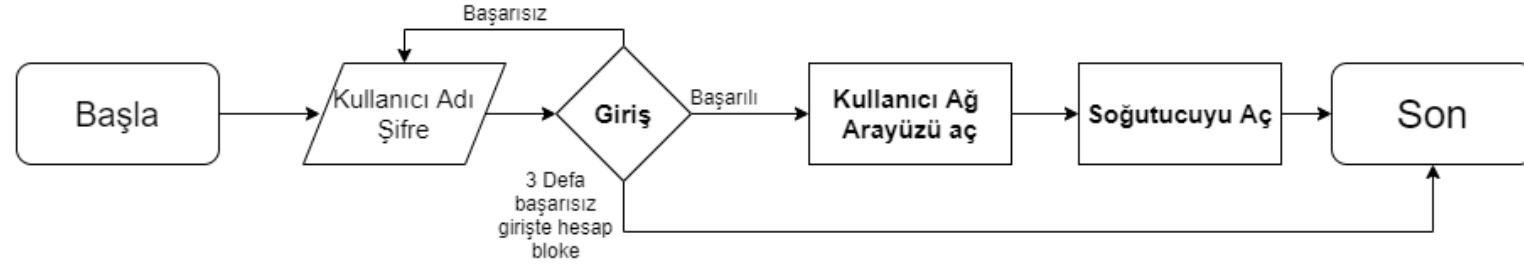
Soğutucuyu aç



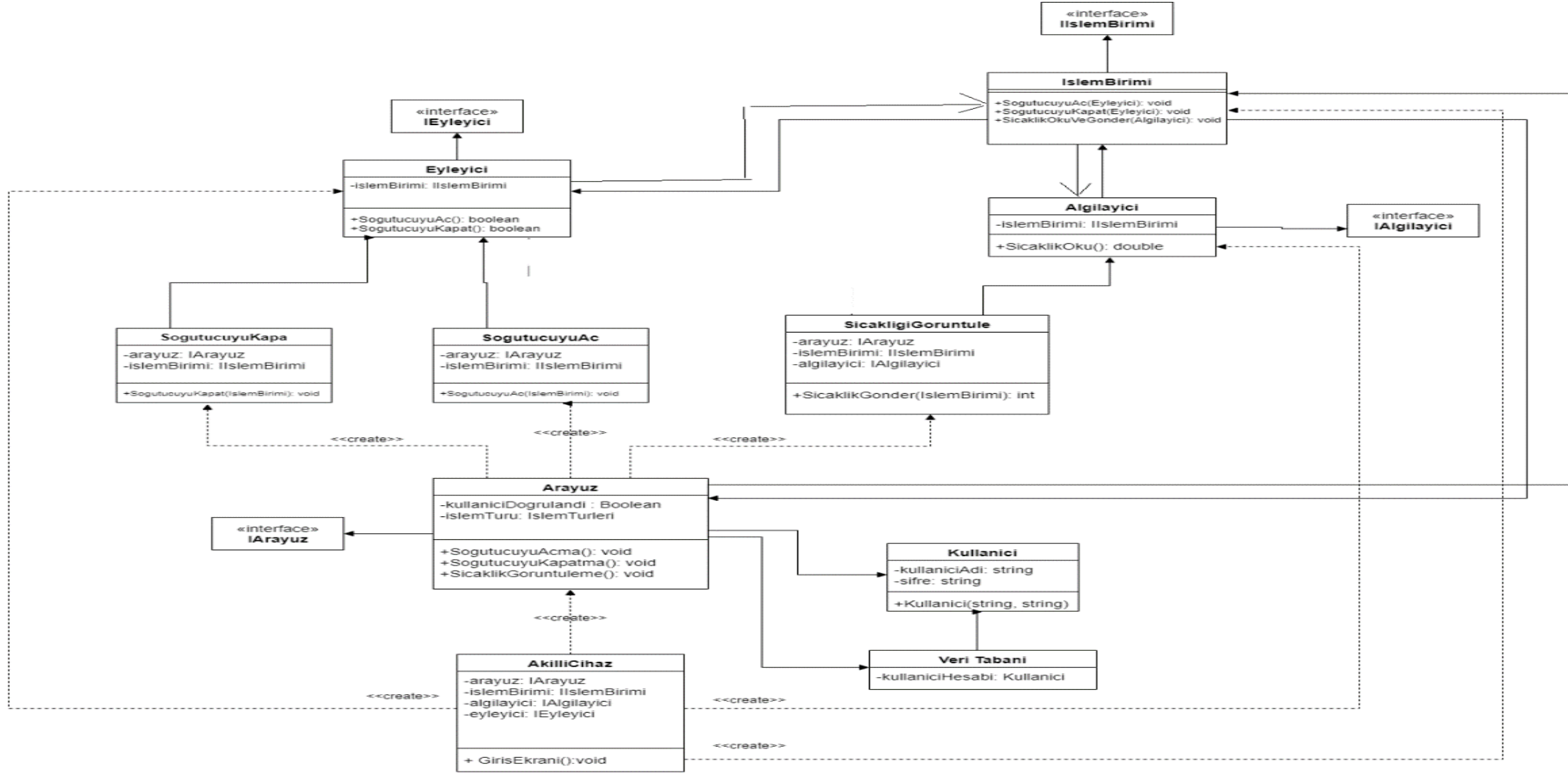
Soğutucuyu kapat



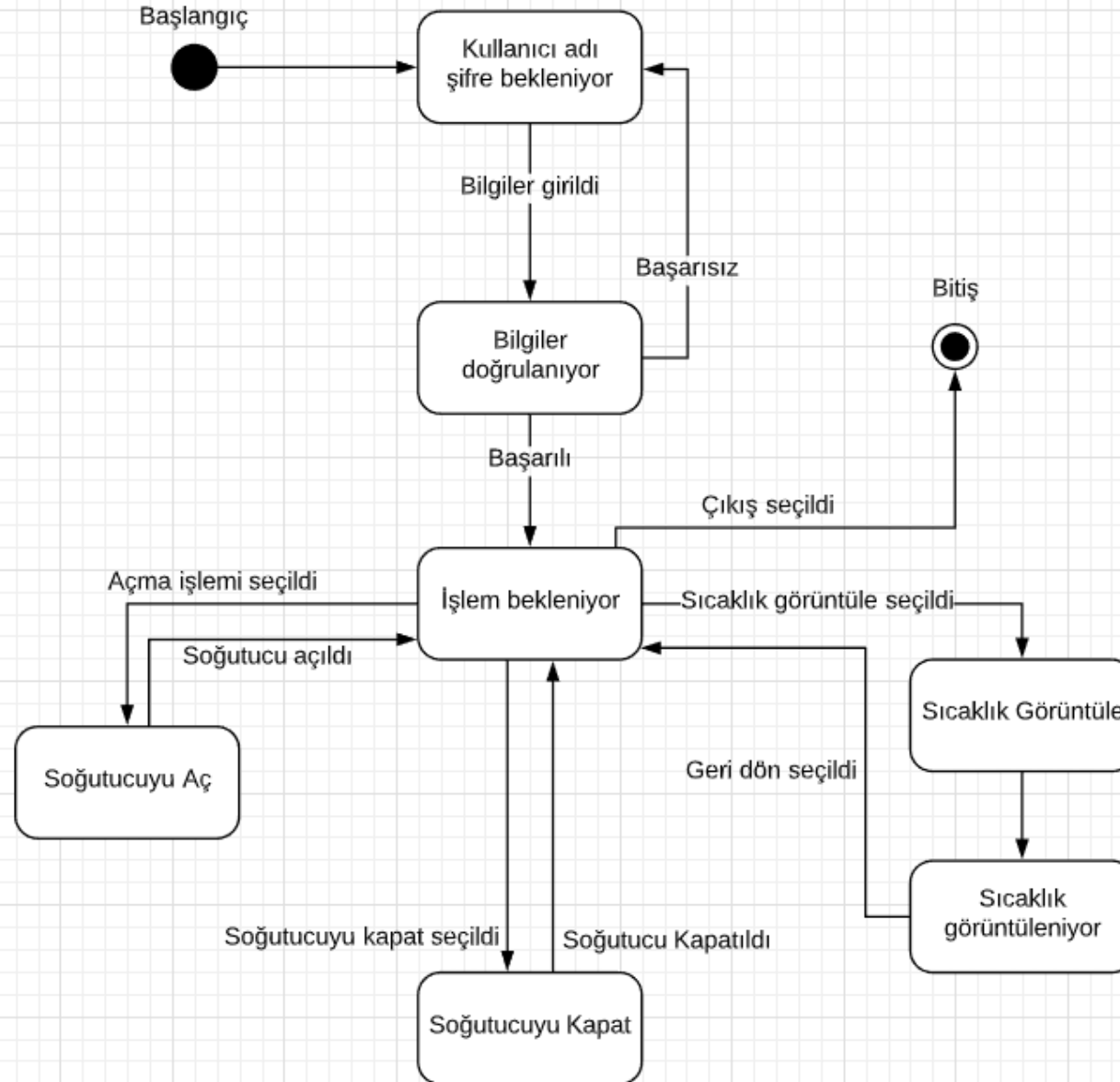
# Etkinlik Şeması



# Sınıf Şeması



# Durum Diyagramı





Veritabanına bağlandı!

Kullanıcı adı giriniz: *tahiruzelli*

Sifre giriniz: *zorsifre123*

Giris basarili!

1-)Sogutucuyu Ac

2-)Sogutucuyu Kapa

3-)Sicakligi Goruntule

4-)Cikis

Veritabanına bağlandı!

Kullanıcı adı giriniz: *dsafadsf*

Sifre giriniz: *asdfasd*

Bilgiler hatali, yeniden baslatiliyor!

Veritabanına bağlandı!

Kullanıcı adı giriniz:

Uygulamamız çalıştığı an bir sorunla karşılaşmaz ise Veritabanına bağlandığını bildiren bir mesaj yollar ve bizden kullanıcı bilgileri ister İstenilen kullanıcı bilgilerinde hata varsa tekrar sorar hata yok ise arayüze bağlanıp istenilen işlemi yaptırır

# Sıcaklık Görüntülenmesi

```
public class SicakligiGoruntule{  
    public int SicakligiGonder(IslemBirimi islemBirimi){  
        System.out.println("Mevcut Sicaklik:");  
        return Algilayici.getInstance("Log.txt").SicaklikOku();  
    }  
}
```

```
public class IslemBirimi implements IIslemBirimi {  
    public int SicaklikOkuVeGonder(Algilayici algilayici){  
        return algilayici.SicaklikOku();  
    }  
}
```

```
import java.io.FileWriter;  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.time.LocalDateTime;  
  
public class Algilayici implements IAlgilayici {  
    private static Algilayici instance;  
    private PrintWriter out;  
    private Algilayici(String logDosyasi){  
        try {  
            out = new PrintWriter(new FileWriter(logDosyasi, append: true), autoFlush: true);  
        } catch (IOException e) {e.printStackTrace();}  
    }  
    public static synchronized Algilayici getInstance(String logDosyasi){  
        if(instance==null)  
            instance = new Algilayici(logDosyasi);  
        return instance;  
    }  
    public int SicaklikOku(){  
        Random r=new Random(); //random sınıfı  
        int sayi = r.nextInt( bound: 40);  
        return sayi;  
    }  
}
```

# Sogutucuyu Aç

```
public class SogutucuyuAc implements Observer {  
    private Observable observable;  
    @Override  
    public void notify(String message) {  
        System.out.println(message);  
    }  
    Eyleyici eyleyici = new Eyleyici();  
    public void SogutucuyuAc(IslemBirimi islemBirimi){  
        if(islemBirimi.SogutucuyuAc(eyleyici)==true)  
            notify("Sogutucu Acilmistir.");  
    }  
}
```

```
public class IslemBirimi implements IIslemBirimi {  
    public boolean SogutucuyuAc(Eyleyici eyleyici){  
        return eyleyici.SogutucuyuAc();  
    }  
}
```

```
public class Eyleyici implements IEyleyici {  
    public boolean SogutucuyuAc(){  
        return true;  
    }  
}
```

# Sogutucuyu Kapa

```
public class SogutucuyuKapa implements Observer {  
    private Observable observable;  
    @Override  
    public void notify(String message) { System.out.println(message); }  
    Eyleyici eyleyici = new Eyleyici();  
    public void SogutucuyuKapa(IslemBirimi islemBirimi){  
        if(islemBirimi.SogutucuyuKapa(eyleyici)==true)  
            notify("Sogutucu Kapanmistir.");  
    }  
}
```

```
public class IslemBirimi implements IIslemBirimi {  
    public boolean SogutucuyuKapa(Eyleyici eyleyici){  
        return eyleyici.SogutucuyuKapa();  
    }  
}
```

```
public class Eyleyici implements IEyleyici {  
    public boolean SogutucuyuKapa(){  
        return true;  
    }  
}
```

# Singleton Deseni

```
import java.util.Random;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.time.LocalDateTime;

public class Algilayici implements IAlgilayici {
    private static Algilayici instance;
    private PrintWriter out;
    private Algilayici(String logDosyasi){
        try {
            out = new PrintWriter(new FileWriter(logDosyasi, append: true), autoFlush: true);
        } catch (IOException e) {e.printStackTrace();}
    }
    public static synchronized Algilayici getInstance(String logDosyasi){
        if(instance==null)
            instance = new Algilayici(logDosyasi);
        return instance;
    }
    public int SicaklikOku(){
        Random r=new Random(); //random sinifi
        int sayi = r.nextInt( bound: 40);
        return sayi;
    }
}
```

```
public class SicakligiGoruntule{
    public int SicakligiGonder(IslemBirimi islemBirimi){
        System.out.println("Mevcut Sicaklik:");
        return Algilayici.getInstance("Log.txt").SicaklikOku();
    }
}
```

# Observer


```
import java.util.ArrayList;
import java.util.List;

public class NoticeObservable implements Observable {
    private List<Observer> observerList = new ArrayList<>();
    private String message = "Notice... !";

    @Override
    public void addObserver(Observer observer) { observerList.add(observer); // Kullanıcıları duyuruya eklemek için. }

    @Override
    public void removeObserver(Observer observer) {
        observerList.remove(observer); // Kullanıcıları duyurudan silmek için.
    }

    @Override
    public void notifyObserver() {
        for (Observer observer : observerList) {
            observer.notify(message); // Duyuru ya kayıtlı kullanıcılara mesaj göndermek için.
        }
    }
}
```

```
public interface Observer {
     void notify(String message);
}
```


## Gerçeklemesi

↓

```
public class SogutucuyuAc implements Observer {
    private Observable observable;

    @Override
    public void notify(String message) {
        System.out.println(message);
    }

    Eyleyici eyleyici = new Eyleyici();
    public void SogutucuyuAc(IslemBirimi islemBirimi){
        if(islemBirimi.SogutucuyuAc(eyleyici)==true)
            notify("Sogutucu Acilmistir.");
    }
}
```

```
public interface Observable {
    void addObserver(Observer observer);
    void removeObserver(Observer observer);
     void notifyObserver();
}
```

# Kaynak Kodlarım

<https://drive.google.com/file/d/1YpXXHLRffYp-JcLOOL8EHPNqXJCDDIzK/view?usp=sharing>

# Tanıtım Videom

- <https://youtu.be/mlpUjq-rZ1M>