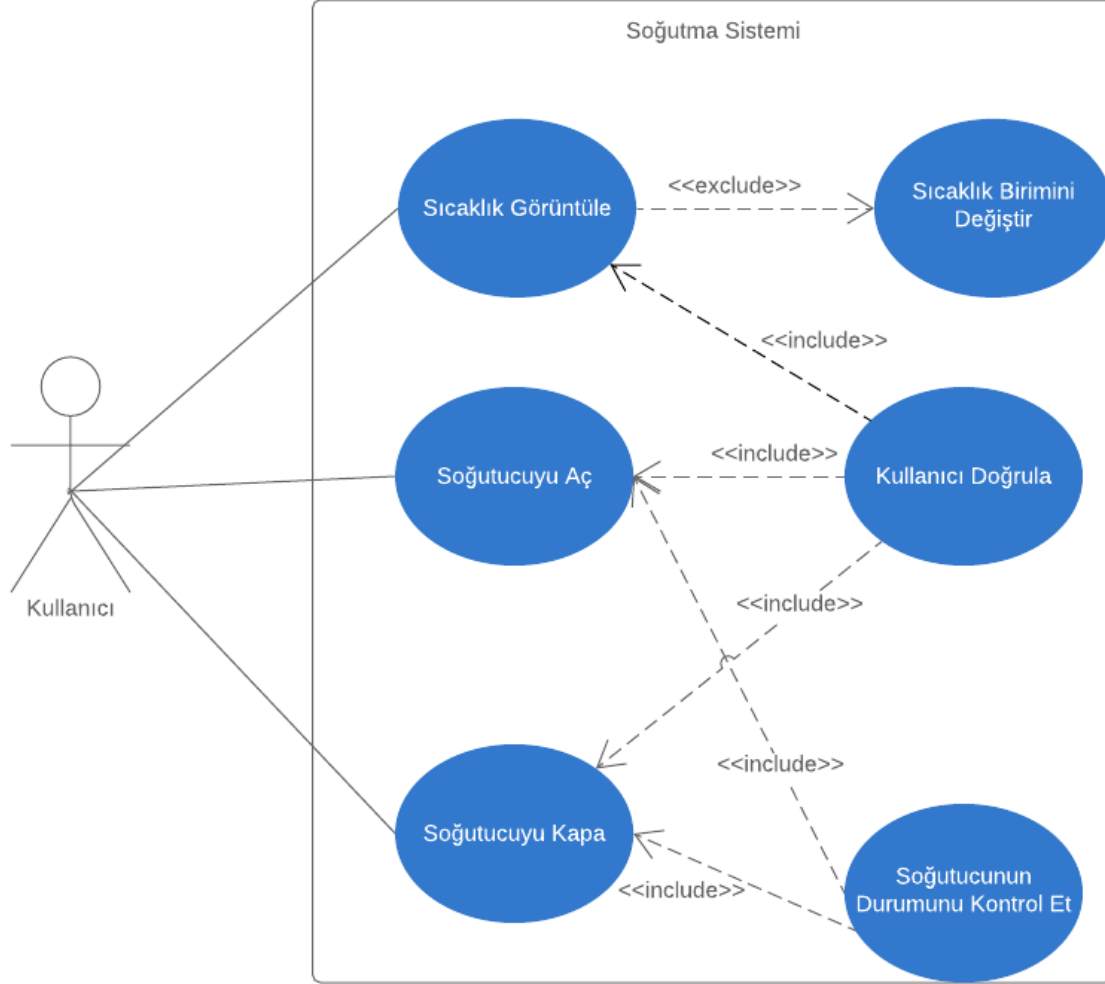


**NESNEYE DAYALI  
ANALİZ  
VE TASARIM  
PROJE**

**HÜSEYİN BERA BULUT  
B181210104  
2. Sınıf 2A**

**[bera.bulut@ogr.sakarya.edu.tr](mailto:bera.bulut@ogr.sakarya.edu.tr)**

## KULLANIM DURUMU (USE-CASE)



## USE-CASE “ SOĞUTUCU AÇMA / KAPAMA ”

### Ana Senaryo:

1. Kullanıcı internet arayüzüne girer.
2. Önceden belirlenmiş olan kullanıcı bilgilerini girer.
3. Sistem kullanıcı doğrulama yapar.
4. Soğutucu sistemi durum sorgulaması yapar. (Açık veya kapalı olma durumu)
5. Kullanıcı durumu değiştirebilir. ( Açık veya kapalı)
6. Oturum sonlandırılır.

### A1. Kullanıcı doğrulanamadı

1. Arayüz kullanıcı bilgileri tekrar girilmesi ister.

### A2. Kullanıcının internet bağlantısında sıkıntı oldu

1. Sistem 15 saniye bekler.
2. İnternet bağlantısı karşılıklı sağlanamamışsa oturum sonlanır.

### A3. Sistemin internet bağlantısında sıkıntı oldu

1. Arayüz kullanıcıya sistemin internet bağlantısında sıkıntı olduğu mesajını gösterir.
2. Sıcaklık gösterme veya durum değiştirme fonksiyonları tepki vermez.
3. Bağlantı düzelmişse uyarı kaybolur ve fonksiyonlar beklendiği gibi çalışır.

### A4. Eyleyicide arıza oldu

1. Arayüz kullanıcıya eyleyicide arıza olduğunu bildirir.
2. Arıza giderilmişse uyarı kaybolur.

#### ❖ Soğutucu Aç Kullanım Durumu

- **Eşsiz bir ad:** Soğutucu Aç,
  - Soğutucuyu açma işlemini tanımlar
  - 20.03.2020, V1.0, kullanıcıadi
- **İlgili Aktörler:** Soğutucu sahibi
- **Giriş koşulu:** Kullanıcı arayüze girer ve doğrulanır
- **Çıkış koşulu:** Kullanıcı işlemini tamamlar
- **Olay akışı:**
  - Ana Senaryo
  - A1. Kullanıcı doğrulanamadı
  - A2. Kullanıcının internet bağlantısında sıkıntı oldu
  - A3. Sistemin internet bağlantısında sıkıntı oldu
  - A4. Eyleyicide arıza oldu
- **Özel gereksinimler:** Kullanıcı arayüzü gereksinimleri, Kullanıcının internet bağlantısı, Sistemin internet bağlantısı, 24 saat çalışma

#### ❖ Soğutucu Kapat Kullanım Durumu

- **Eşsiz bir ad:** Soğutucu Kapat,
  - Soğutucuyu kapatma işlemini tanımlar
  - 20.03.2020, V1.0, kullanıcıadi
- **İlgili Aktörler:** Soğutucu sahibi
- **Giriş koşulu:** Kullanıcı arayüze girer ve doğrulanır
- **Çıkış koşulu:** Kullanıcı işlemini tamamlar
- **Olay akışı:**
  - Ana Senaryo
  - A1. Kullanıcı doğrulanamadı
  - A2. Kullanıcının internet bağlantısında sıkıntı oldu
  - A3. Sistemin internet bağlantısında sıkıntı oldu
  - A4. Eyleyicide arıza oldu
- **Özel gereksinimler:** Kullanıcı arayüzü gereksinimleri, Kullanıcının internet bağlantısı, Sistemin internet bağlantısı, 24 saat çalışma

## USE-CASE “ SICAKLIK GÖRÜNTÜLENMESİ ”

### Ana Senaryo:

1. Kullanıcı internet arayüzüne girer.
2. Önceden belirlenmiş olan kullanıcı bilgilerini girer.
3. Sistem kullanıcı doğrulama yapar.
4. Kullanıcı sıcaklık sorgulaması yapar.
5. Sistem sıcaklığı gösterir.
6. Kullanıcı sıcaklık birimini değiştirebilir. (Celsius, Fahrenheit vb.)
7. Oturum sonlandırılır.

### A1. Kullanıcı doğrulanamadı

1. Arayüz kullanıcı bilgileri tekrar girilmesi ister.

### A2. Kullanıcının internet bağlantısında sıkıntı oldu

1. Sistem 15 saniye bekler.
2. İnternet bağlantısı karşılıklı sağlanamamışsa oturum sonlanır.

### A3. Sistemin internet bağlantısında sıkıntı oldu

1. Arayüz kullanıcıya sistemin internet bağlantısında sıkıntı olduğu mesajını gösterir.
2. Sıcaklık gösterme veya durum değiştirme fonksiyonları tepki vermez.
3. Bağlantı düzelmişse uyarı kaybolur ve fonksiyonlar beklendiği gibi çalışır.

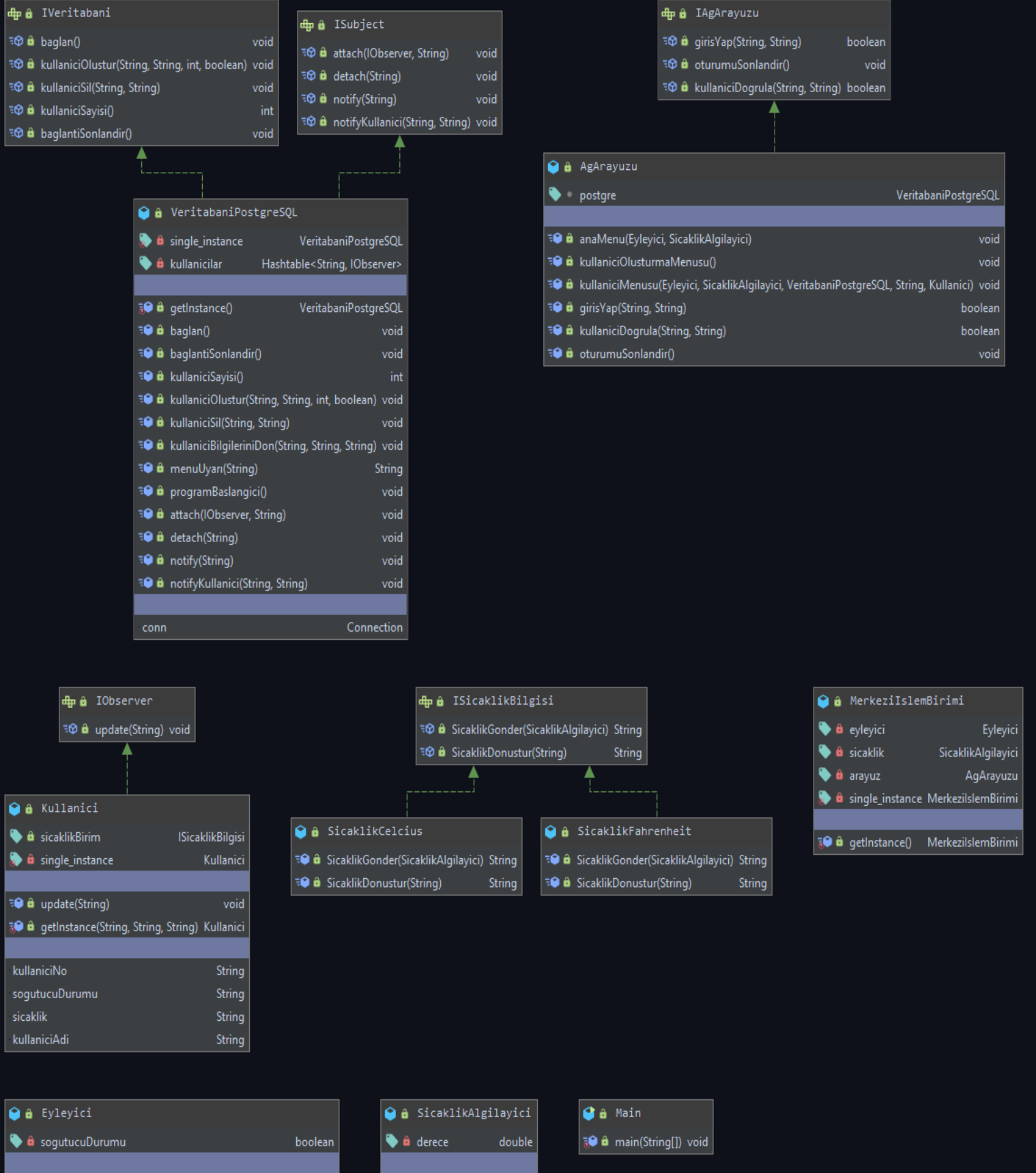
### A4. Sıcaklık algılayıcıda arıza oldu

1. Arayüz kullanıcıya sıcaklık algılayıcıda arıza olduğunu bildirir.
2. Arıza giderilmişse uyarı kaybolur.

❖ **Sıcaklık Göster Kullanım Durumu**

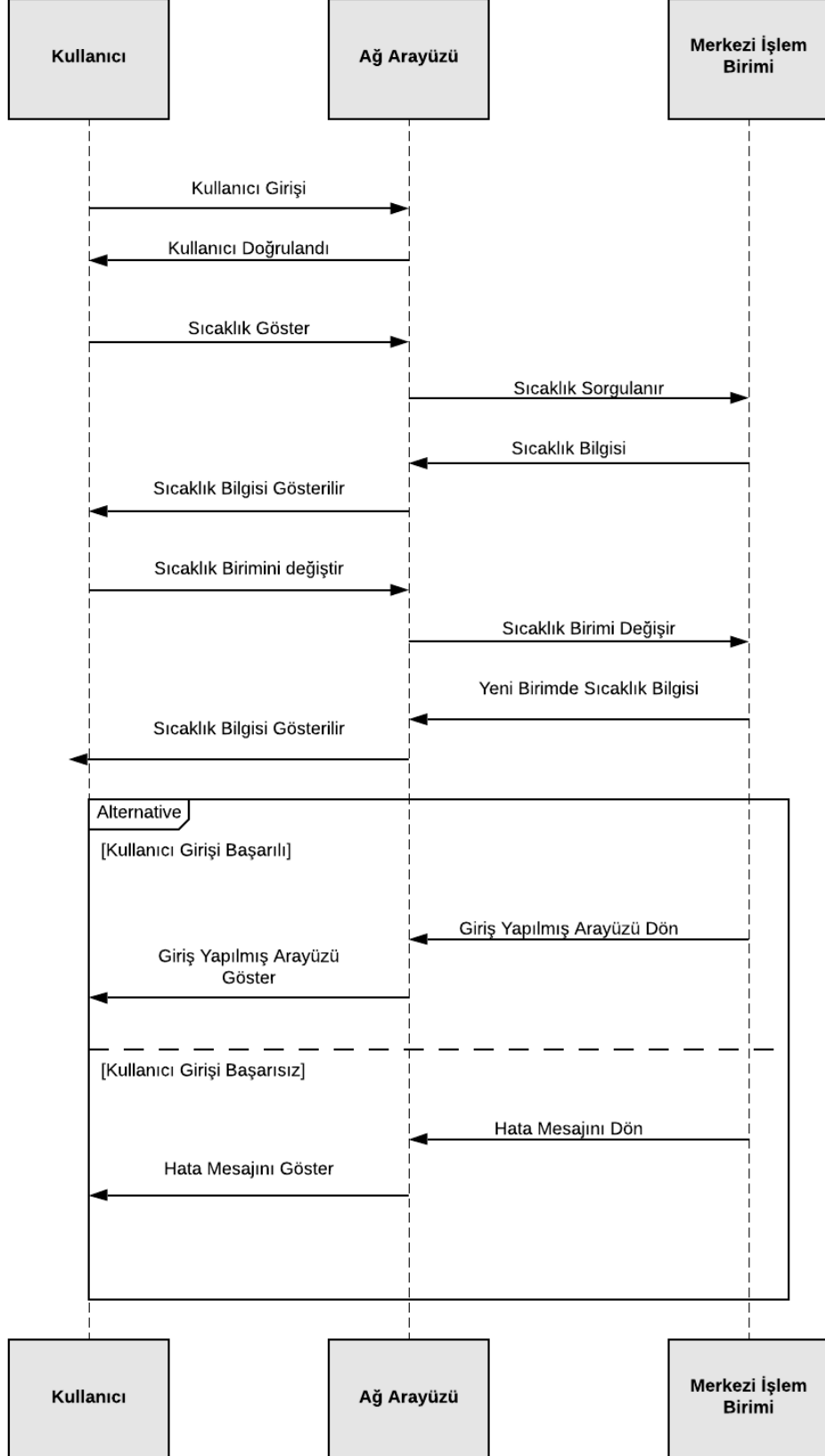
- **Eşsiz bir ad:** Sıcaklık Göster,
  - Arayüzde sıcaklık gösterme işlemini tanımlar
  - 20.03.2020, V1.0, kullanıcıadı
- **İlgili Aktörler:** Soğutucu sahibi
- **Giriş koşulu:** Kullanıcı arayüze girer ve doğrulanır
- **Çıkış koşulu:** Kullanıcı işlemini tamamlar
- **Olay akışı:**
  - Ana Senaryo
  - A1. Kullanıcı doğrulanamadı
  - A2. Kullanıcının internet bağlantısında sıkıntı oldu
  - A3. Sistemin internet bağlantısında sıkıntı oldu
  - A4. Sıcaklık algılayıcıda arıza oldu
- **Özel gereksinimler:** Kullanıcı arayüzü gereksinimleri, Kullanıcının internet bağlantısı, Sistemin internet bağlantısı, 24 saat çalışma

# SINIF ŞEMASI



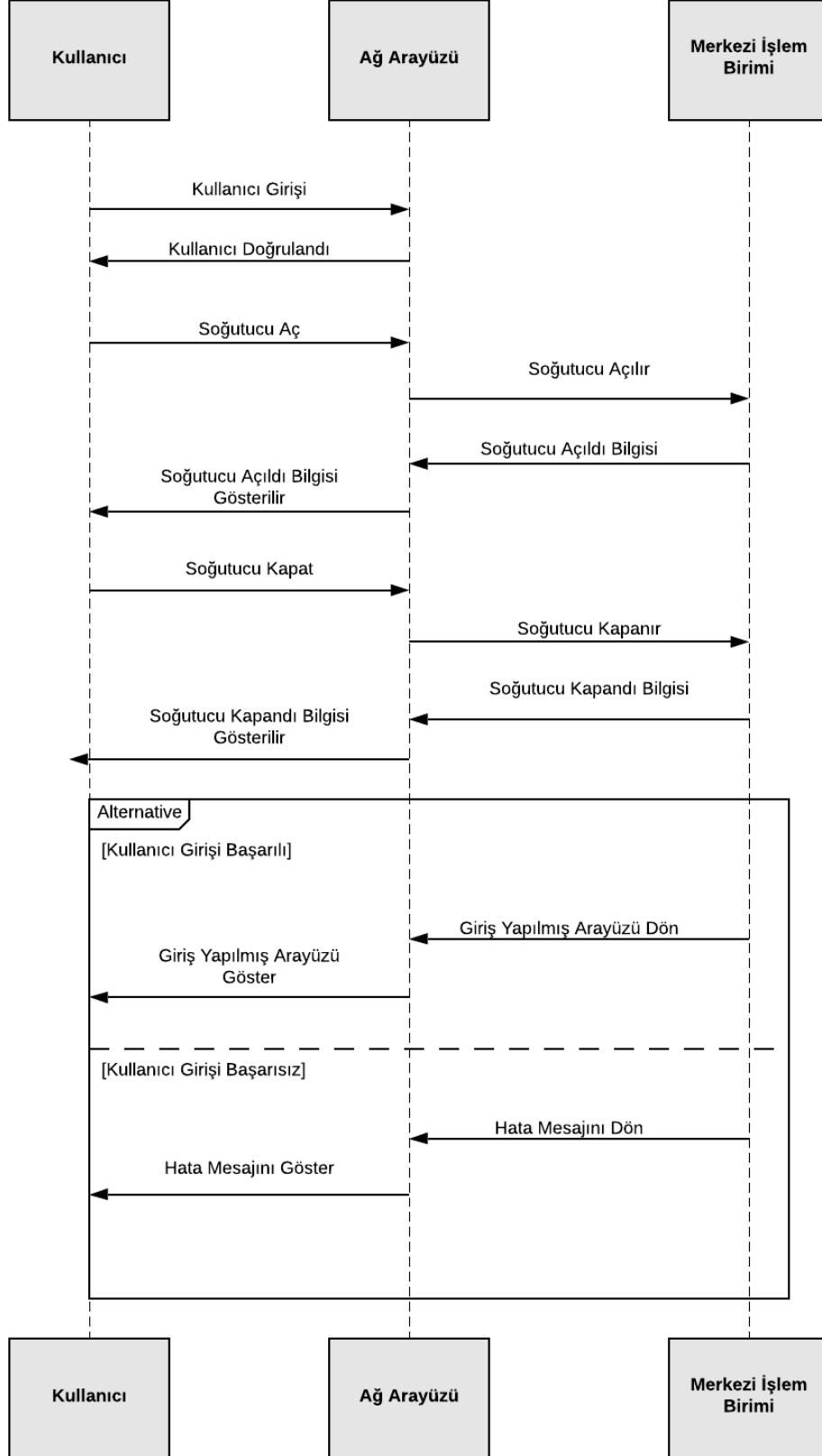
# SICAKLIK BİLGİSİ SIRALAMA ŞEMASI

## (SEQUENCE DIAGRAM)

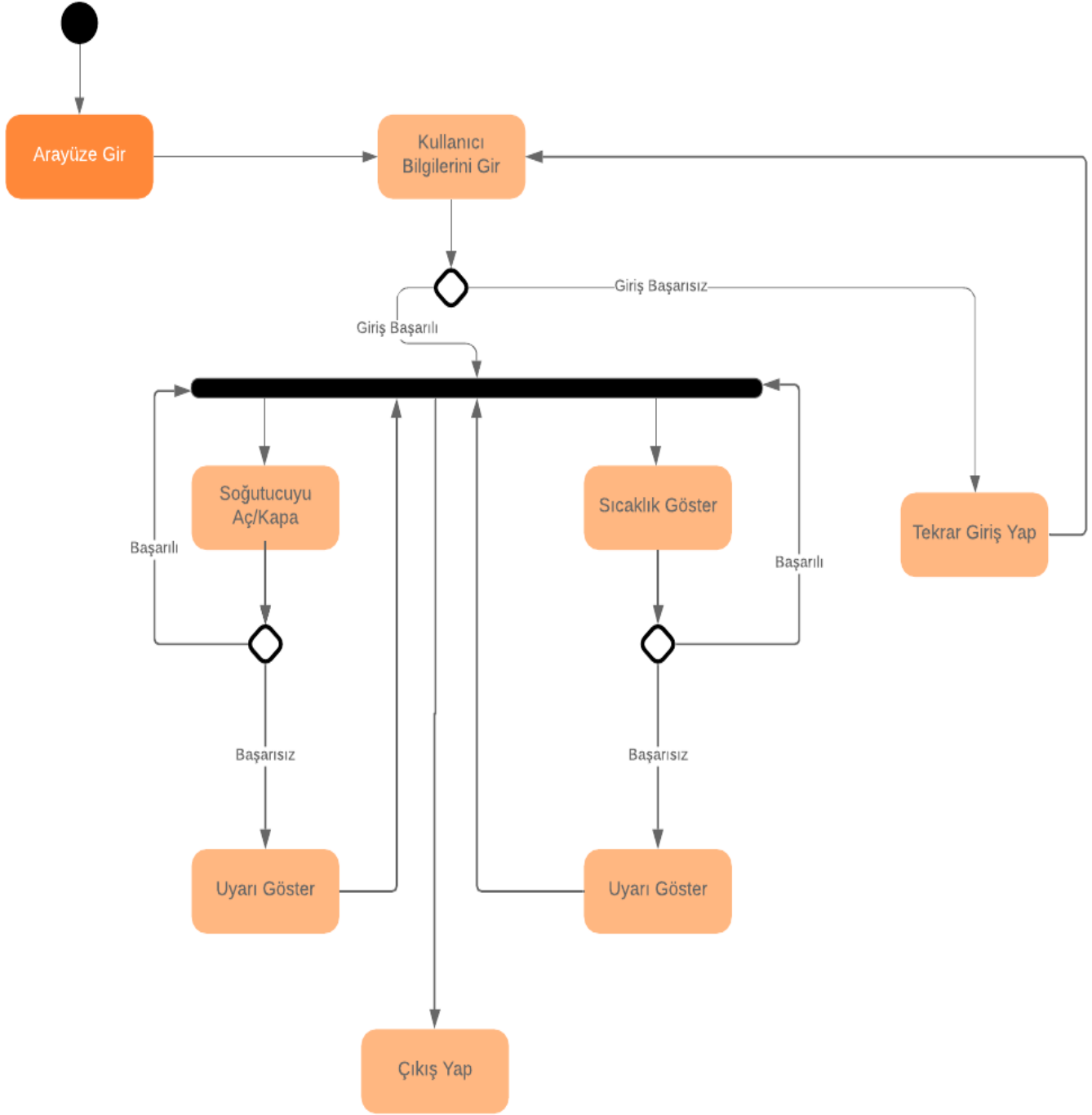




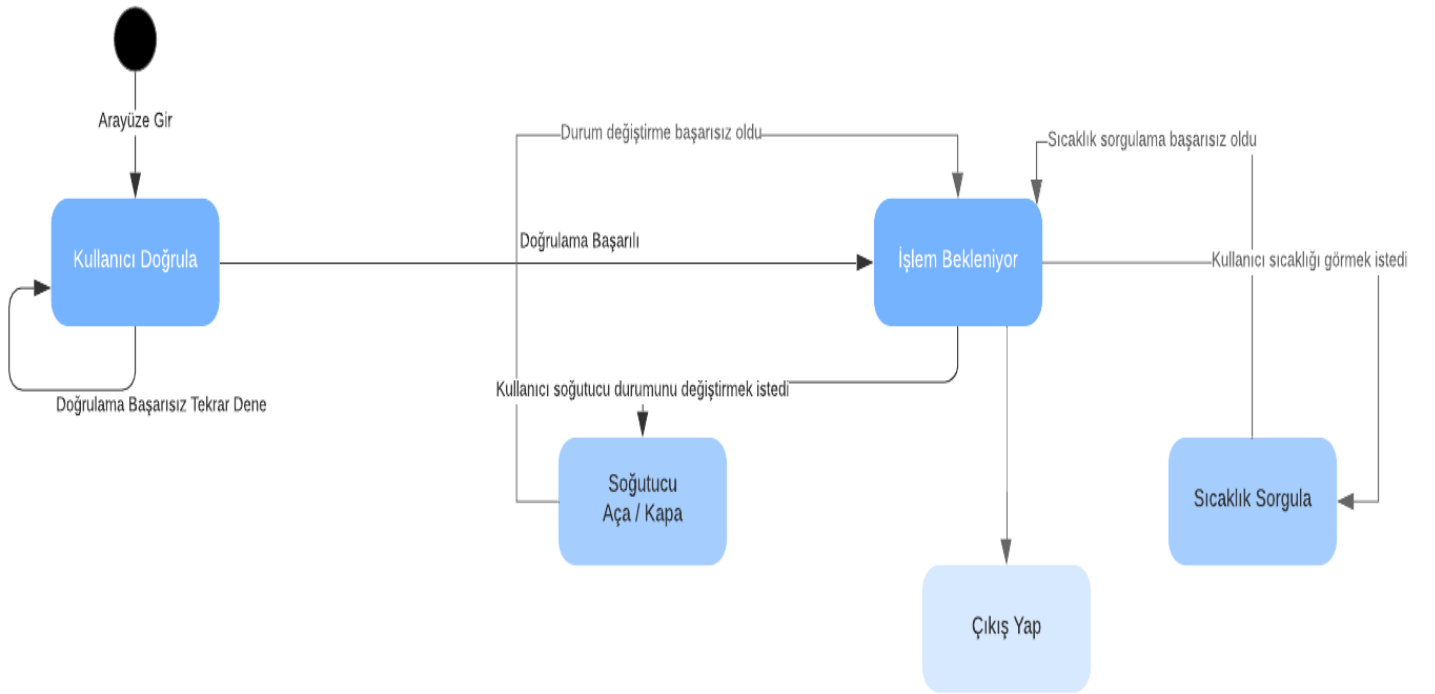
# SOĞUTUCU AÇ / KAPAT SIRALAMA ŞEMASI (SEQUENCE DIAGRAM)



# SICAKLIK GÖSTER VE SOĞUTUCU AÇ/KAPAT ETKİNLİK DİYAGRAMI (ACTIVITY DIAGRAM)



## DURUM DİYAGRAMI ( STATE MACHINE DIAGRAM)



## KULLANICI DOĞRULAMA EKRANI

```
"C:\Program Files\Java\jdk-11.0.6\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2019.3.2\lib\idea_rt.jar=5000:C:\Program Files\JetBrains\IntelliJ IDEA 2019.3.2\bin" -Dfile.encoding=UTF-8
Ağa bağlandı!
1 - GIRIS YAP!
2 - KULLANICI OLUSTUR!
3 - PROGRAMDAN CIK
1
Kullanici adinizi giriniz :
bera
Sifrenizi giriniz :
bulut
```

Kullanıcı doğrulama ekranında var olan hesabımıza giriş yapabilir veya kullanıcı oluşturabiliriz.

Yukarıdaki resimde var olan hesabımıza giriş yapıyoruz.

```
"C:\Program Files\Java\jdk-11.0.6\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2019.3.2\lib\idea_rt.jar=5000:C:\Program Files\JetBrains\IntelliJ IDEA 2019.3.2\bin" -Dfile.encoding=UTF-8
Ağa bağlandı!
1 - GIRIS YAP!
2 - KULLANICI OLUSTUR!
3 - PROGRAMDAN CIK
1
Kullanici adinizi giriniz :
bera
Sifrenizi giriniz :
bulut
bera giris yapti!

Kullanici : bera          Sicaklik : 4.724 C
Sevgili bera UYARI : sicaklik goster menusunu kullanmayın!

1 - Sicaklik Goster
2 - Sogutucuyu Ac
3 - Sogutucuyu Kapa
4 - Sicaklik birimini degistir
5 - Cikis Yap
```

Giriş yaptıktan sonra karşımıza kullanıcıya göre özelleştirilmiş ana menü çıktı.

```





Ağa bağlandı!
1 - GIRIS YAP!
2 - KULLANICI OLUSTUR!
3 - PROGRAMDAN CIK
2
Kullanici adi giriniz!
deneme
Sifre girin
deneme
deneme adli kullanıcı oluşturuldu!
1 - GIRIS YAP!
2 - KULLANICI OLUSTUR!
3 - PROGRAMDAN CIK
1
Kullanici adinizi giriniz :
deneme
Sifrenizi giriniz :
deneme
deneme giriş yaptı!

Kullanici : deneme          Sicaklik : 26.32 C
Sevgili deneme UYARI : sicaklik goster menusunu kullanmayın!

1 - Sicaklik Goster
2 - Sogutucuyu Ac
3 - Sogutucuyu Kapa
4 - Sicaklik birimini degistir
5 - Cikis Yap

```

Yukarıdaki resimde ise yeni bir kullanıcı oluşturuyoruz.

Database:  NesneOdev <span>▼</span> Schema:  public <span>▼</span> Table:  Kullanicilar <span>▼</span>					
 <input type="checkbox"/>	kullaniciAdi	sifre	kullaniciNo	SogutucuDurumu	
1	bera	bulut	1	false	
2	deneme	deneme	2	false	

```
"C:\Program Files\Java\jdk-11.0.6\bin\java.exe" "-javaagent:C:\Pro
Ağa bağlandı!
1 - GIRIS YAP!
2 - KULLANICI OLUSTUR!
3 - PROGRAMDAN CIK
1
Kullanici adinizi giriniz :
deneme
Sifrenizi giriniz :
ddeneme
Sistem deneme adli kullaniciyi dogrulayamadi!
1 - GIRIS YAP!
2 - KULLANICI OLUSTUR!
3 - PROGRAMDAN CIK
```

Yukarıdaki resimde yeni oluşturduğumuz kullanıcıya giriş yaparken yanlış şifre girdim ve veritabanındaki şifre ile girilen şifre uyuşmadığı için giriş yapamadık.

## SICAKLIK GÖSTER

```
Kullanici : bera          Sicaklik : 33.19 C

1 - Sicaklik Goster
2 - Sogutucuyu Ac
3 - Sogutucuyu Kapa
4 - Sicaklik birimini degistir
5 - Cikis Yap
1
-----Hava 33.03 C derece!-----

Kullanici : bera          Sicaklik : 33.03 C

1 - Sicaklik Goster
2 - Sogutucuyu Ac
3 - Sogutucuyu Kapa
4 - Sicaklik birimini degistir
5 - Cikis Yap
```

1 numaralı seçeneği seçtiğimizde 0-35 arasında rastgele bir sayı oluşturuluyor ve sıcaklık olarak ekrana yazılıyor. Aynı zamanda ana menüde de sıcaklık göstergesi var o da yeni değer ile güncelleniyor.

## SOĞUTUCU AÇ / KAPAT

```
2
-----Sogutucu acildi!-----
Kullanici : bera          Sicaklik : 33.03 C

1 - Sicaklik Goster
2 - Sogutucuyu Ac
3 - Sogutucuyu Kapa
4 - Sicaklik birimini degistir
5 - Cikis Yap
2
-----Sogutucu zaten acik!-----

Kullanici : bera          Sicaklik : 33.03 C

1 - Sicaklik Goster
2 - Sogutucuyu Ac
3 - Sogutucuyu Kapa
4 - Sicaklik birimini degistir
5 - Cikis Yap
3
-----Sogutucu kapatildi!-----

Kullanici : bera          Sicaklik : 33.03 C

1 - Sicaklik Goster
2 - Sogutucuyu Ac
3 - Sogutucuyu Kapa
4 - Sicaklik birimini degistir
5 - Cikis Yap
```

2 numaralı seçeneği seçtiğimizde soğutucuyu açma isteği gönderiyoruz ilk seferde soğutucu kapalı olduğu için açıldı. İkinci sefer açık olan soğutucuyu açmaya çalıştığımızda soğutucu açık olduğu için “Soğutucu zaten açık uyarısı geldi”.

3 numaralı seçeneği seçtiğimizde ise açık olan soğutucu kapatıldı.

# OPEN / CLOSED

SOLID prensiplerinden biri olan Open/Closed prensibi sürdürülebilir ve tekrar kullanılabilir bir yapıda kod yazmamıza olanak sağlar. Gelişime **açık**, değişime **kapalı**.

**Open** : Sınıfa yeni davranışlar eklememizi sağlar. Gereksinimler değişirse yeni gereksinimlerin giderilmesi için sınıfa yeni veya farklı davranışlar ekleyebilmeliyiz.

**Closed**: Sınıfın temel özellikleri değişmemelidir.

Projemde bu prensibi sıcaklık bilgisini kullanıcının tercihiye göre Celsius veya Fahrenheit olarak göstererek sağladım.

```
public interface ISicaklikBilgisi {  
    public String SicaklikGonder(SicaklikAlgilayici sicaklik);  
    public String SicaklikDonustur(String sicaklik);  
}
```

```
public class SicaklikCelsius implements ISicaklikBilgisi {  
    @Override  
    public String SicaklikGonder(SicaklikAlgilayici sicaklik) {  
        sicaklik.sicaklikOku();  
        return sicaklik.sicaklikGonder() + " C";  
    }  
    @Override  
    public String SicaklikDonustur(String sicaklik) {  
        if(sicaklik.contains("C"))  
            return sicaklik;  
        else  
            return String.valueOf((Double.parseDouble(sicaklik.substring(0,4)) - 32) / 1.8).substring(0,4) + "C";  
    }  
}
```

```
public class SicaklikFahrenheit implements ISicaklikBilgisi {  
    @Override  
    public String SicaklikGonder(SicaklikAlgilayici sicaklik) {  
        sicaklik.sicaklikOku();  
        return (sicaklik.sicaklikGonder() * 1.8 + 32) + " F";  
    }  
    @Override  
    public String SicaklikDonustur(String sicaklik) {  
        if(sicaklik.contains("F"))  
            return sicaklik;  
        else  
            return (String.valueOf(Double.parseDouble(sicaklik.substring(0,5)) * 1.8 + 32)).substring(0,4) + "F";  
    }  
}
```



```

else if(secenek == 4) {
    System.out.println("1 - Celcius");
    System.out.println("2 - Fahrenheit");
    secenek = scan.nextInt();

    if(secenek == 1) {
        kullanici.sicaklikBirim = new SicaklikCelcius();
        derece = kullanici.sicaklikBirim.SicaklikDonustur(derece);
        postgre.notifyKullanici( mesaj: " sicaklik biriminiz Celcius'a cevrigildi", kullaniciAdi);
    }
    else if(secenek == 2) {
        kullanici.sicaklikBirim = new SicaklikFahrenheit();
        derece = kullanici.sicaklikBirim.SicaklikDonustur(derece);
        postgre.notifyKullanici( mesaj: " sicaklik biriminiz Fahrenheit'e cevrigildi", kullaniciAdi);
    }
}
}

```

```

if(secenek == 1) {
    derece = kullanici.sicaklikBirim.SicaklikGonder(sicaklik);
    System.out.println("-----Hava " + derece + " derece!-----");
}

```

Sıcaklık gönder metodu Celsius ve Fahrenheit sınıfları için ortak. Ama kullanıcının terciğine göre hangisinin kullanılacağı seçiliyor. SicaklikDonustur metodu ise sıcaklıkların birbirlerine dönüştürmemize yarıyor.

```

Kullanici : bera          Sicaklik : 33.03 C

1 - Sicaklik Goster
2 - Sogutucuyu Ac
3 - Sogutucuyu Kapa
4 - Sicaklik birimini degistir
5 - Cikis Yap
4
1 - Celcius
2 - Fahrenheit
2
Sevgili bera  sicaklik biriminiz Fahrenheit'e cevrigildi

Kullanici : bera          Sicaklik : 91.4F

```

# SINGLETON DESENİ

Singleton tasarım deseni bir sınıftan sadece bir nesne oluşturulmasını sağlar, yeni bir nesne oluşturmasını engeller ve nesneye ihtiyaç olduğunda önceden oluşturulan örneği çağırır.

Singleton tasarım desenini uygulayabilmemiz için sınıfın bir kurucusunun olması gerekir ve o kurucuyu private veya protected yaparak sınıfın dışından “new Sınıf()” kelime öbeğiyle nesne oluşturmalarını engelleriz.

Singleton Tasarım Desenini veritabanı gibi program boyunca tek bir tane nesneye ihtiyacımız olacak sınıflarda kullanmak mantıklı olabilir.

```
public class VeritabaniPostgreSQL implements ISubject, IVeritabani {
    private Connection conn;
    private static VeritabaniPostgreSQL single_instance = null;
    private Hashtable<String, IObserver> kullanicilar = new Hashtable<>();
    private VeritabaniPostgreSQL() { baglan(); }
    public static VeritabaniPostgreSQL getInstance()
    {
        if (single_instance == null)
            single_instance = new VeritabaniPostgreSQL();

        return single_instance;
    }
}
```

```
public class AgArayuzu implements IAgArayuzu {
    VeritabaniPostgreSQL postgre = VeritabaniPostgreSQL.getInstance();
}
```

Yukarıdaki resimde Singleton tasarım desenini veritabanı sınıfıma uyguladım.

Aşağıdaki resimde ise MerkeziİşlemBirimi sınıfıma.

```
private static MerkeziİşlemBirimi single_instance = null;

private MerkeziİşlemBirimi() { arayuz.anaMenu(eyleyici, sicaklik); }
public static MerkeziİşlemBirimi getInstance()
{
    if (single_instance == null)
        single_instance = new MerkeziİşlemBirimi();

    return single_instance;
}
```

# OBSERVER DESENİ

Observer davranışsal desenlerden biridir. Amacımız çok sayıda nesneye gözlemledikleri nesnede meydana gelen olayı bildirmektir. Yani bir nesnenin değişikliğinden birden fazla nesne etkileniyor.

Programda Observer desenini kullanıcılara uyarı gönderirken kullanıyorum.

```
public interface IObserver {  
    public void update(String m);  
}
```

```
public class Kullanici implements IObserver{  
    private String kullaniciAdi;  
    private String kullaniciNo;  
    private String sogutucuDurumu;  
    private String sicaklik;  
    public ISicaklikBilgisi sicaklikBirim = new SicaklikCelcius();  
    private static Kullanici single_instance = null;
```

```
public interface ISubject {  
    public void attach(IObserver kullanici, String kullaniciAdi);  
    public void detach(String kullaniciAdi);  
    public void notify(String mesaj);  
    public void notifyKullanici(String mesaj, String kullaniciAdi);  
}
```

```
public class VeritabaniPostgreSQL implements ISubject, IVeritabani {  
    private Connection conn;  
    private static VeritabaniPostgreSQL single_instance = null;  
    private Hashtable<String, IObserver> kullanicilar = new Hashtable<>();  
    private VeritabaniPostgreSQL() { baglan(); }  
    public static VeritabaniPostgreSQL getInstance()  
    {  
        if (single_instance == null)  
            single_instance = new VeritabaniPostgreSQL();  
  
        return single_instance;  
    }  
}
```

```

@Override
public void attach(IObserver kullanici, String kullaniciAdi) { kullanici.put(kullaniciAdi, kullanici); }
@Override
public void detach(String kullaniciAdi) { kullanici.remove(kullaniciAdi); }
@Override
public void notify(String mesaj) { kullanici.forEach((key, value) -> value.update(mesaj)); }
@Override
public void notifyKullanici(String mesaj, String kullaniciAdi) {
    kullanici.forEach((key, value) -> {
        if(key.equals(kullaniciAdi)) {
            value.update(mesaj);
        }
    });
}
}
}

```

```

@Override
public void update(String mesaj) {
    if(!(mesaj.length() > 0))
        return;
    else
        System.out.println("Sevgili " + kullaniciAdi + " " + mesaj);
}

```

```

while(true) {
    String uyari = "";
    uyari = postgre.menuUyari(uyari);
    System.out.print("\nKullanici : " + kullaniciAdi);
    System.out.println("      Sicaklik : " + derece);
    postgre.notify(uyari);

    System.out.println("\n1 - Sicaklik Goster");
    System.out.println("2 - Sogutucuyu Ac");
    System.out.println("3 - Sogutucuyu Kapa");
    System.out.println("4 - Sicaklik birimini degistir");
    System.out.println("5 - Cikis Yap");
}

```

```

Kullanici : bera      Sicaklik : 30.12 C
Sevgili bera UYARI : Observer deseni uyar

```

---

- 1 - Sicaklik Goster
- 2 - Sogutucuyu Ac
- 3 - Sogutucuyu Kapa
- 4 - Sicaklik birimini degistir
- 5 - Cikis Yap

**KAYNAK KODLARI:** <https://github.com/berabulut/NesneyeDayaliAnalizProjeOdevi>

**VIDEO ADRESİ:** <https://www.youtube.com/watch?v=m0ZJrmq3FA8>