

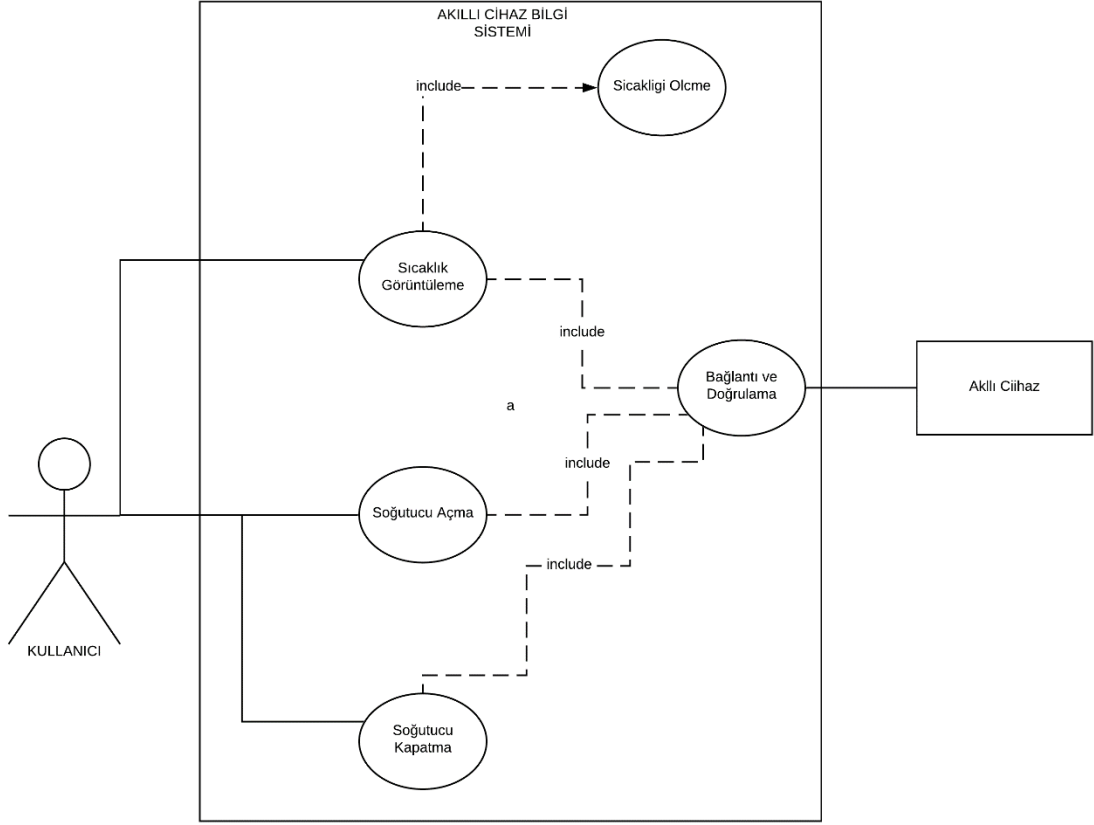


**T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ
FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
Nesne Yönelimli Analiz ve Tasarım**

PROJE/TASARIM

**Hazırlayan:
B171210084 – Serhat Erenel 2. Sınıf 1. Öğretim A
Grubu**



SICAKLIĞIN GÖRÜNTÜLENMESİ KULLANIM DURUMU

- Eşsiz bir ad: Sıcaklığın Görüntülenmesi,
- Akıllı cihazın sıcaklığının görüntülenmesi sürecini tanımlar.
- 8.05.2020 v1.1 Serhat Erenel.

İlgili Aktörler: İnternet Kullanıcıları

Giriş Koşulu: Sıcaklığın görüntülenmesi isteği.

Çıkış Koşulu: Sıcaklığın görüntülenmesi.

Ana Olay Akışı;

- 1.Kullanıcı ağ arayüzü giriş ekranına gelir.
- 2.Ekrana kullanıcı adı ve şifre ekranı gelir.
- 3.Kullanıcı şifreyi ve kullanıcı adını girer.
- 4.Ağ arayüzü erişimi onaylar ve seçim menüsünü gösterir.
- 5.Kullanıcı sıcaklığın görüntülenmesi işlemini seçer .
- 6.Sistem seçimi algılar ve seçimi çalıştırır.

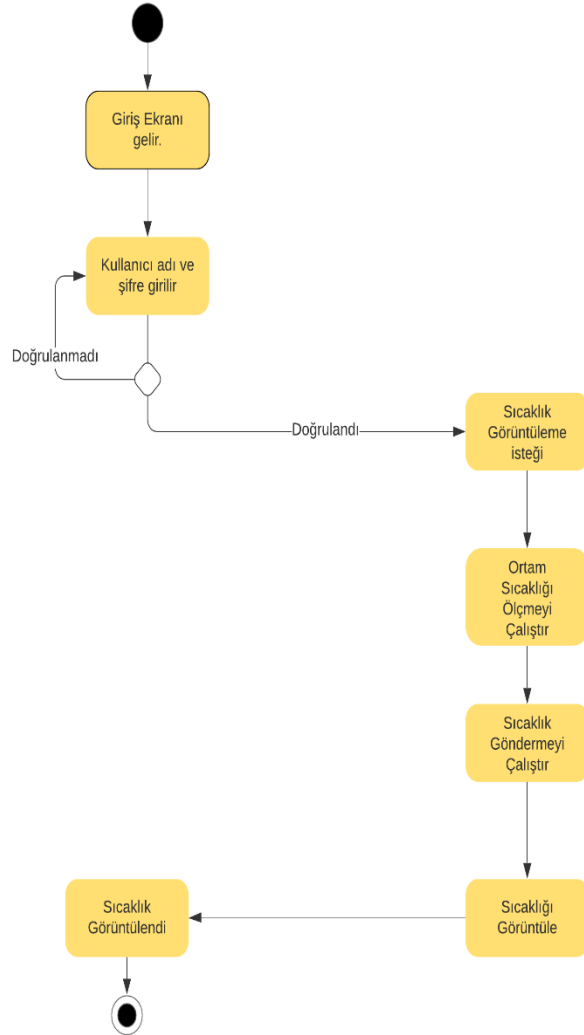
Alternatif Olay Akışı;

A1. Kullanıcı adı ve şifre yanlış.(3)

4. Giriş ekranına dönülür.

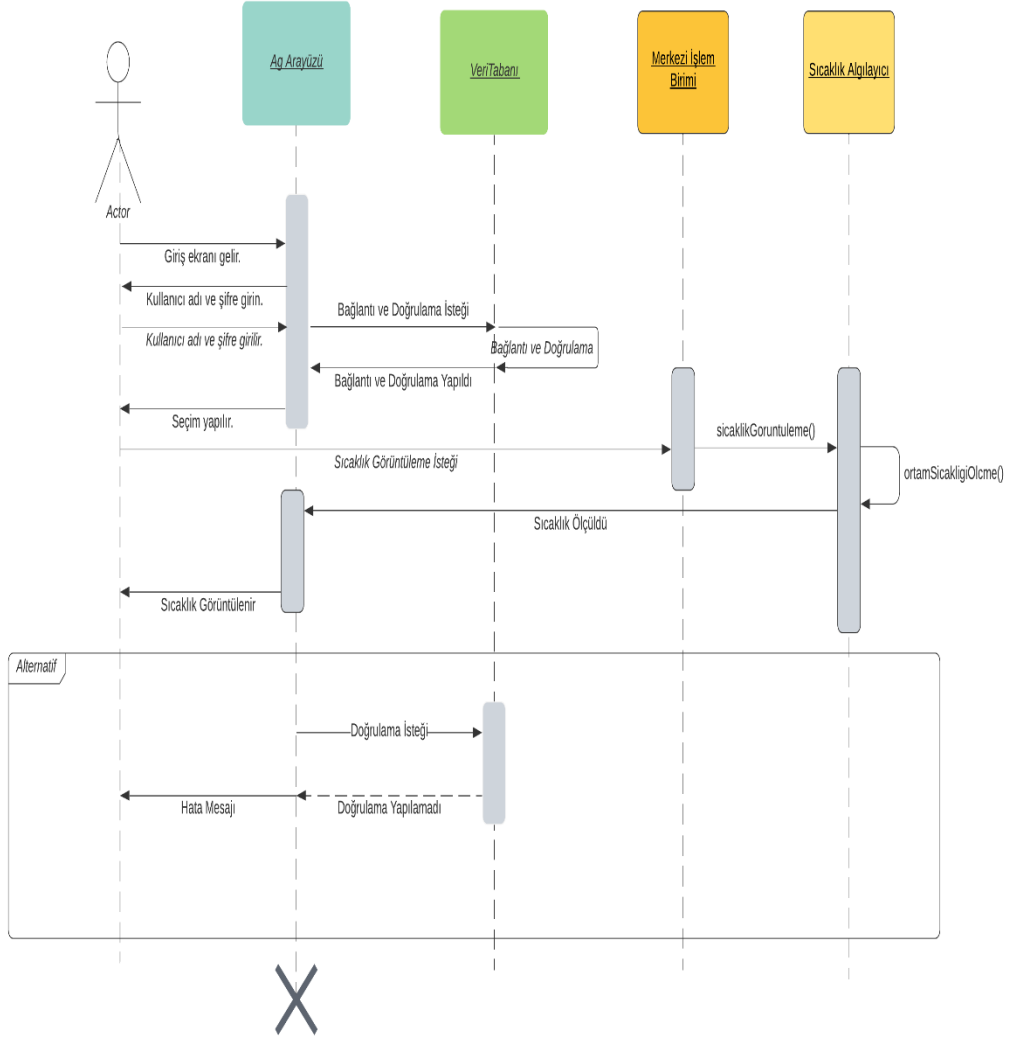
SICAKLIĞIN GÖRÜNTÜLENMESİ ACTIVITY DIAGRAM

Sıcaklığın Görüntülenmesi Activity Diagram



SICAKLIĞIN GÖRÜNTÜLENMESİ SEQUENCE DIAGRAM

Sıcaklığın Görüntülenmesi Sequence Diagram



SOĞUTUCUNUN ÇALIŞTIRILMASI KULLANIM DURUMU

- Eşsiz bir ad: Soğutucunun Çalıştırılması
- Akıllı cihazın soğutucusunun başlatılması sürecini tanımlar.
- 8.05.2020 v1.1 Serhat Erenel.

İlgili Aktörler: İnternet Kullanıcıları

Giriş Koşulu: Soğutucunun çalıştırılması isteği.

Çıkış Koşulu: Soğutucunun çalıştırılması.

Ana Olay Akışı;

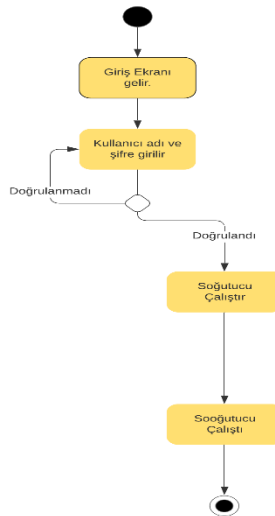
- 1.Kullanıcı ağ arayüzü giriş ekranına gelir.
- 2.Ekrana kullanıcı adı ve şifre ekranı gelir.
- 3.Kullanıcı şifreyi ve kullanıcı adını girer.
- 4.Ağ arayüzü erişimi onaylar ve seçim menüsünü gösterir.
- 5.Kullanıcı soğutucu açma işlemini seçer .
- 6.Sistem seçimi algılar ve seçimi çalıştırır.

Alternatif Olay Akışı;

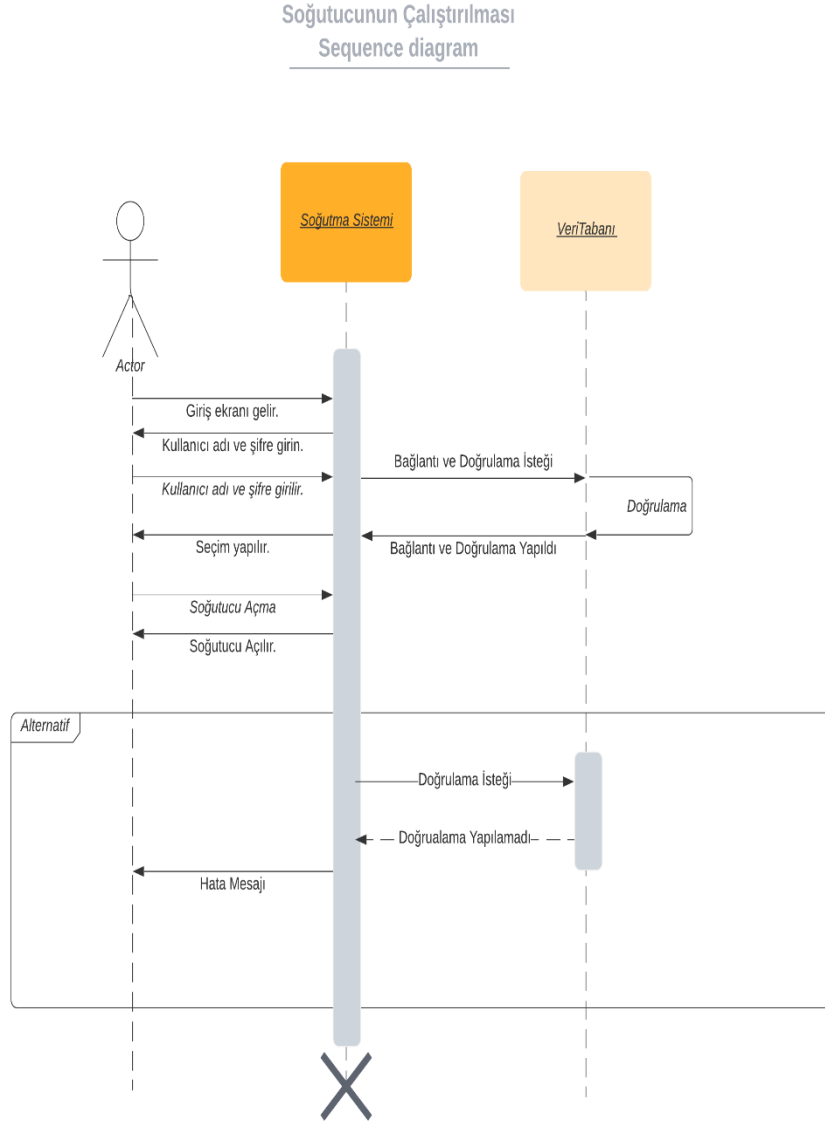
- A1. Kullanıcı adı ve şifre yanlış.(3)
4. Giriş ekranına dönülür.

SOĞUTUCUNUN ÇALIŞTIRILMASI ACTIVITY DIAGRAM

Soğutucunun Çalıştırılması Activity Diagram

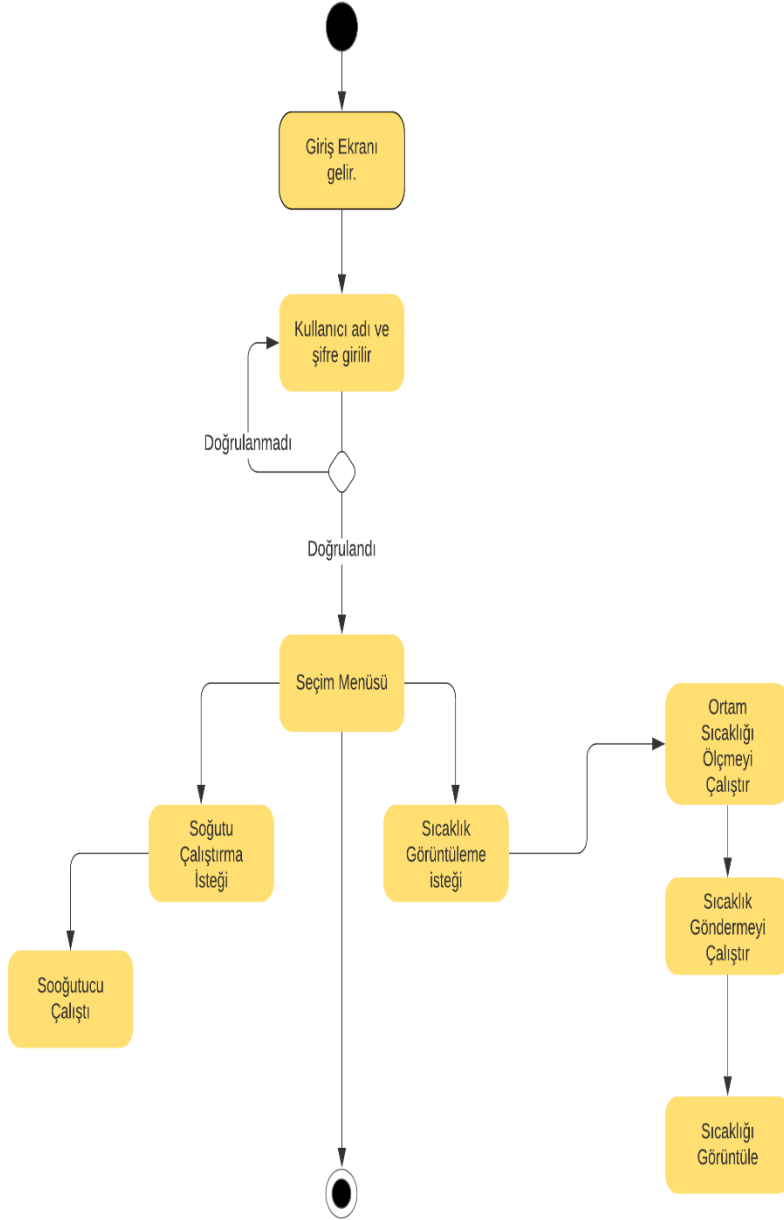


SOĞUTUCUNUN ÇALIŞTIRILMASI SEQUENCE DIAGRAM



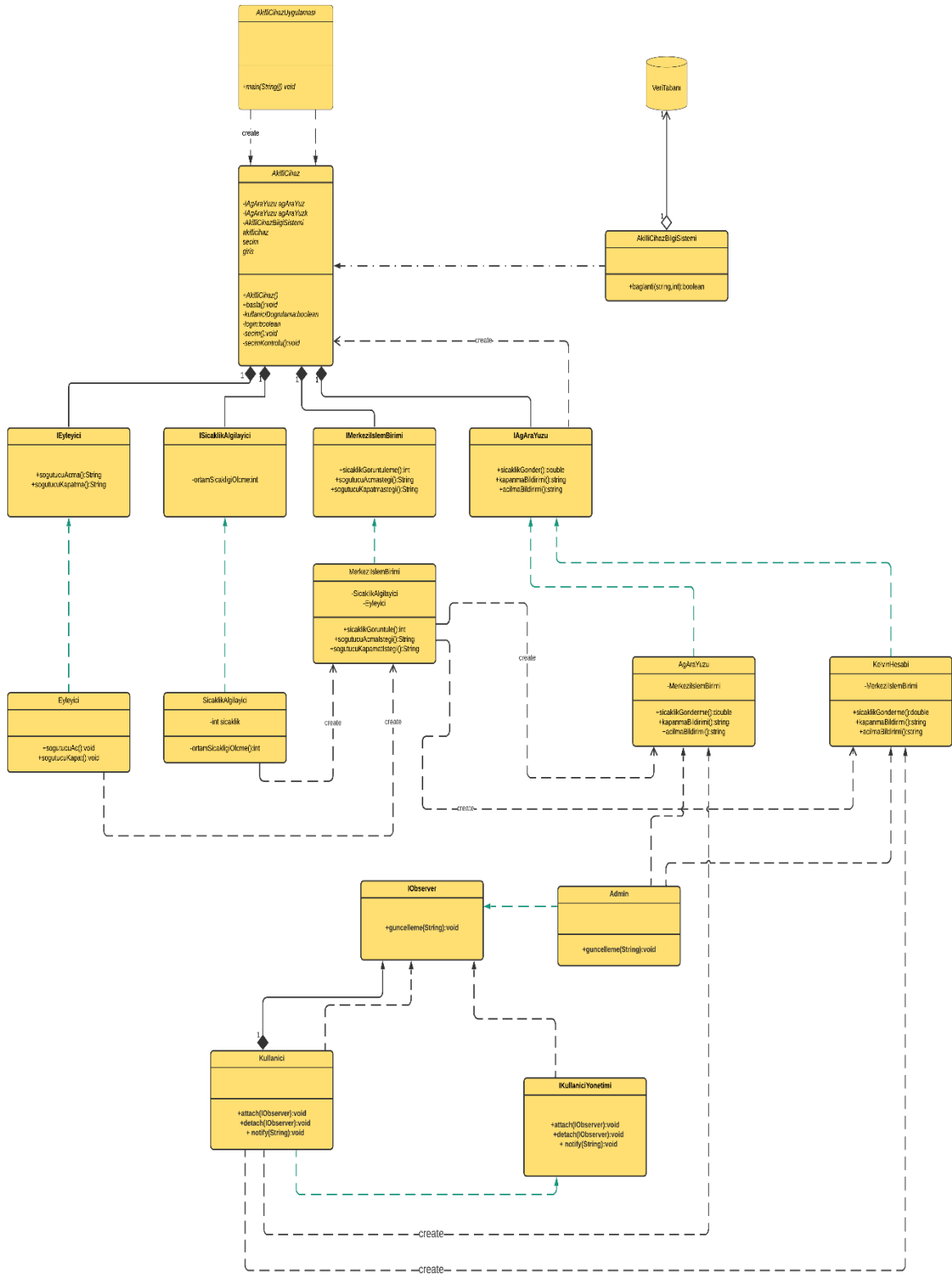
DURUM STATE MACHINE DIAGRAM

UML State Diagram



CLAS DIAGRAM

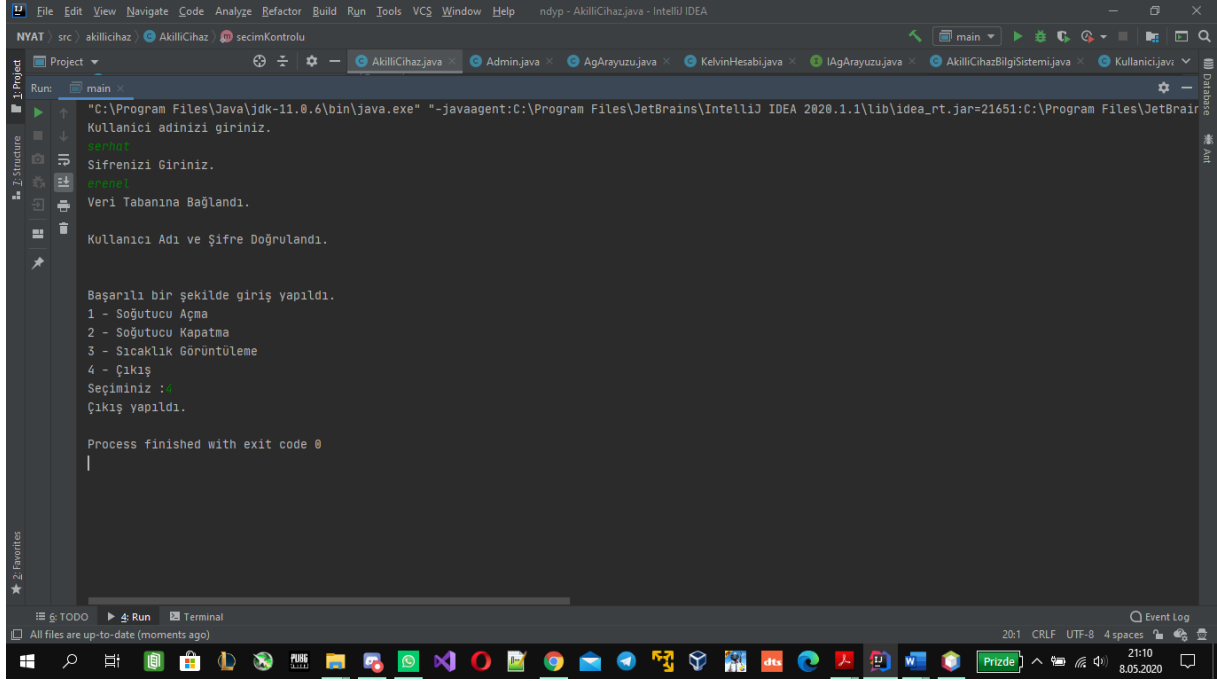
Class Diagram



KULLANICI DOĞRULAMA EKRANI

Kullanıcının sisteme giriş yapabilmesi için ilk önce kullanıcı adı ve şifre ekranından veritabanında bulunan bilgileri doğru bir şekilde girerek giriş yapması gerekiyor. Aksi takdirde giriş yapamaz ve o ekranda kalır. Giriş yaptıktan çıkmak isterse çıkış seçeneği ile çıkış yapılabilir.

BAŞARILI GİRİŞ

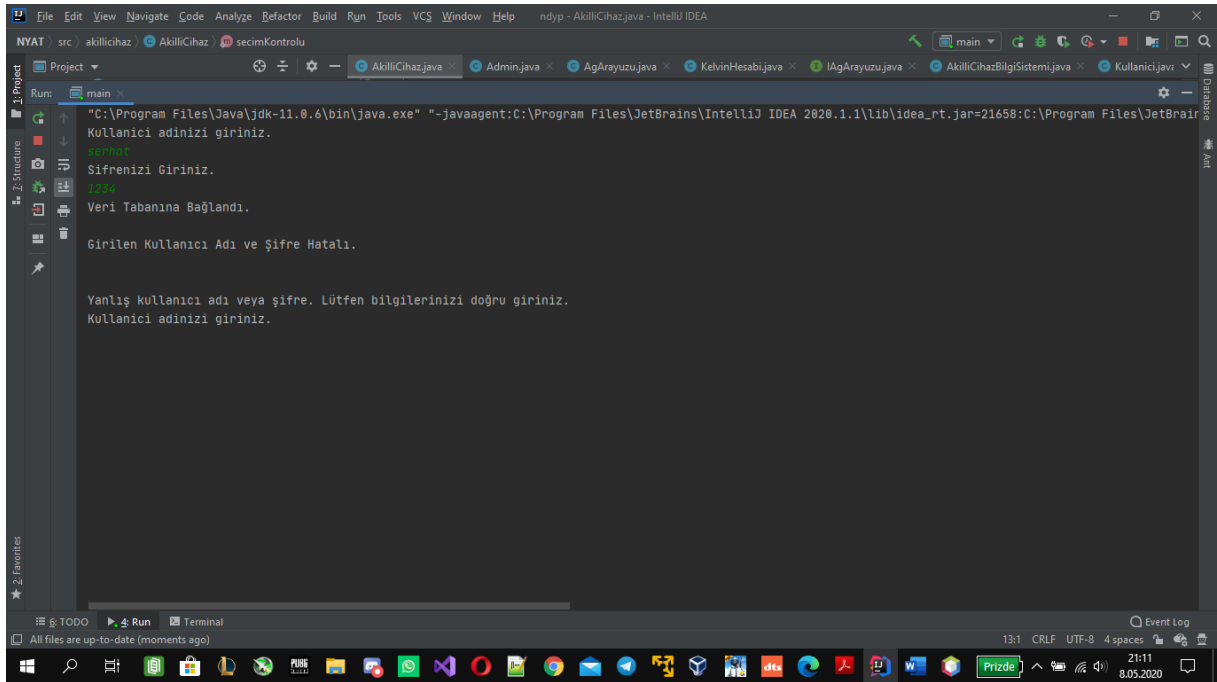


```
"C:\Program Files\Java\jdk-11.0.6\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2020.1.1\lib\idea_rt.jar=21651:C:\Program Files\JetBrains\IntelliJ IDEA 2020.1.1\bin" -Dfile.encoding=UTF-8
Kullanıcı adınızı giriniz.
Sifrenizi Giriniz.
Veri Tabanına Bağlandı.
Kullanıcı Adı ve Şifre Doğrulandı.

Başarılı bir şekilde giriş yapıldı.
1 - Soğutucu Açma
2 - Soğutucu Kapatma
3 - Sıcaklık Görüntüleme
4 - Çıkış
Seçtiniz : 4
Çıkış yapıldı.

Process finished with exit code 0
```

BAŞARISIZ GİRİŞ

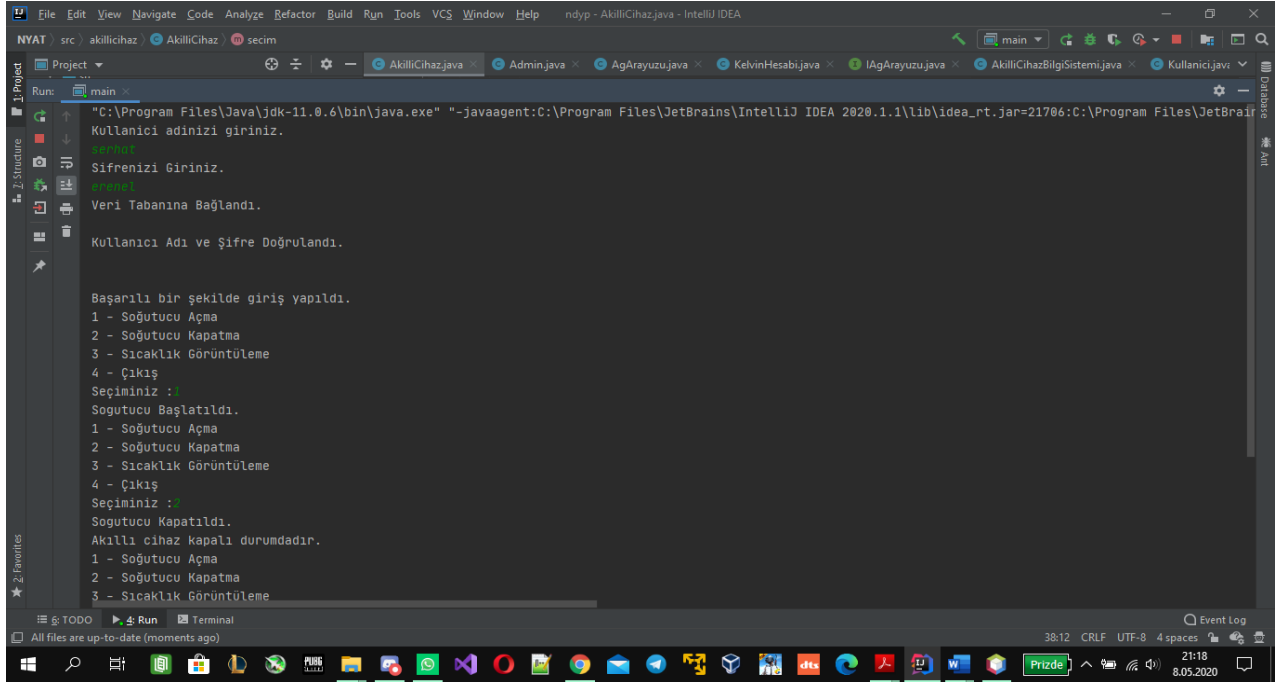


```
"C:\Program Files\Java\jdk-11.0.6\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2020.1.1\lib\idea_rt.jar=21658:C:\Program Files\JetBrains\IntelliJ IDEA 2020.1.1\bin" -Dfile.encoding=UTF-8
Kullanıcı adınızı giriniz.
Sifrenizi Giriniz.
Veri Tabanına Bağlandı.
Girilen Kullanıcı Adı ve Şifre Hatalı.

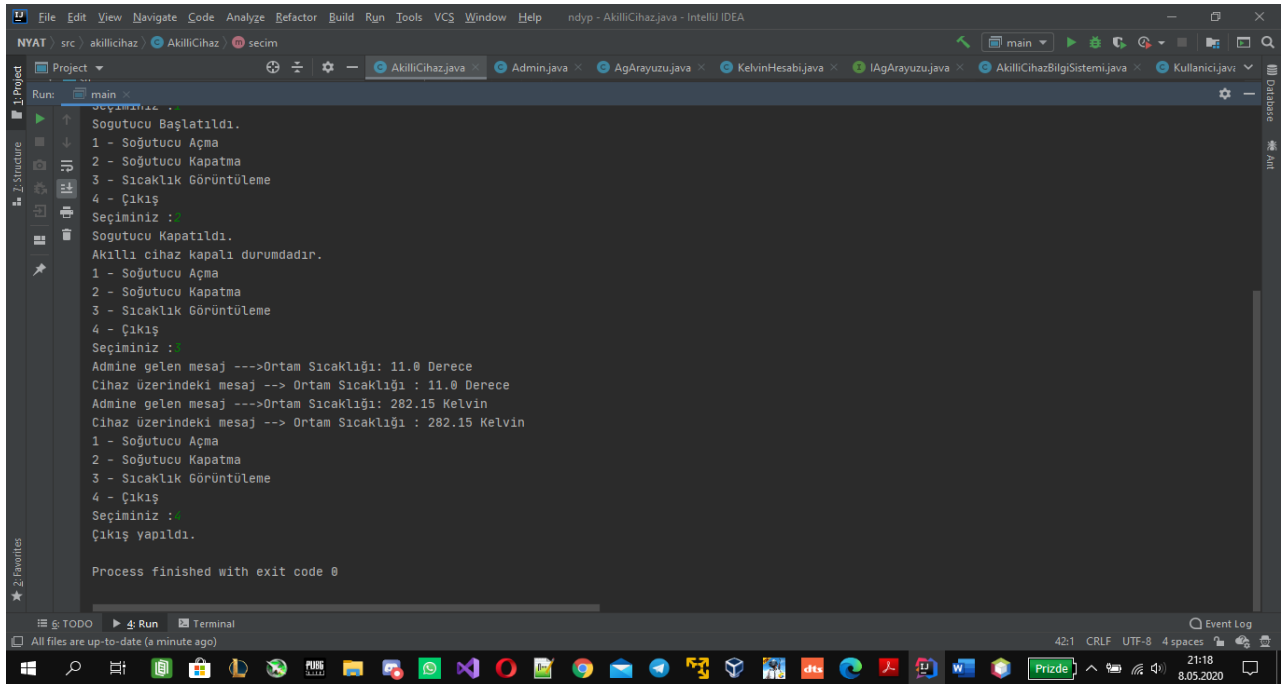
Yanlış kullanıcı adı veya şifre. Lütfen bilgilerinizi doğru giriniz.
Kullanıcı adınızı giriniz.
```

SICAKLIĞIN GÖRÜNTÜLENMESİ VE SOĞUTUCUNUN AÇILIP KAPANMASI

Sıcaklığın görüntülenmesi ve soğutucunun açılması işlemleri için ilk önce sisteme başarılı bir şekilde giriş yapıyoruz. Daha sonra giriş yaptıktan sonra çıkan menüde gerekli numaralara basarak yapmak istediğimiz işlemi seçiyoruz. 1 numaralı işlemi seçersek soğutucu açılıyor. 2 numaralı işlemi seçersek soğutucu kapanıyor. 3 numaralı işlemi seçersek sıcaklık görüntüleniyor derece ve kelvin cinsinden. 4 numaralı işlemi seçersek sistemden çıkış yapıyoruz.



```
Run: main
"C:\Program Files\Java\jdk-11.0.6\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2020.1.1\lib\idea_rt.jar=21706:C:\Program Files\JetBrains\IntelliJ IDEA 2020.1.1\bin" -Dfile.encoding=UTF-8
Kullanıcı adınızı giriniz.
Sifrenizi Giriniz.
Veri Tabanına Bağlandı.
Kullanıcı Adı ve Şifre Doğrulandı.
Başarılı bir şekilde giriş yapıldı.
1 - Soğutucu Açma
2 - Soğutucu Kapatma
3 - Sıcaklık Görüntüleme
4 - Çıkış
Seçiminiz : 1
Soğutucu Başlatıldı.
1 - Soğutucu Açma
2 - Soğutucu Kapatma
3 - Sıcaklık Görüntüleme
4 - Çıkış
Seçiminiz : 2
Soğutucu Kapatıldı.
Akıllı cihaz kapalı durumdadır.
1 - Soğutucu Açma
2 - Soğutucu Kapatma
3 - Sıcaklık Görüntüleme
```

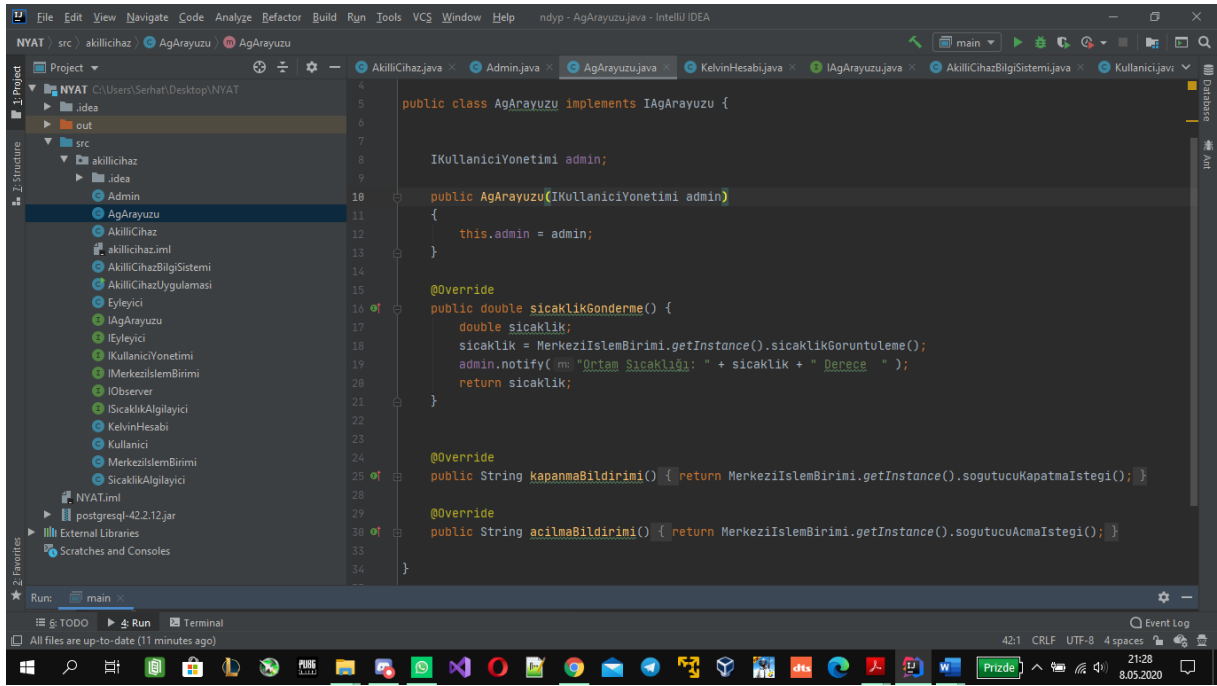


```
Run: main
Soğutucu Başlatıldı.
1 - Soğutucu Açma
2 - Soğutucu Kapatma
3 - Sıcaklık Görüntüleme
4 - Çıkış
Seçiminiz : 3
Soğutucu Kapatıldı.
Akıllı cihaz kapalı durumdadır.
1 - Soğutucu Açma
2 - Soğutucu Kapatma
3 - Sıcaklık Görüntüleme
4 - Çıkış
Seçiminiz : 3
Admin gelen mesaj --->Ortam Sıcaklığı: 11.0 Derece
Cihaz üzerindeki mesaj --> Ortam Sıcaklığı : 11.0 Derece
Admin gelen mesaj --->Ortam Sıcaklığı: 282.15 Kelvin
Cihaz üzerindeki mesaj --> Ortam Sıcaklığı : 282.15 Kelvin
1 - Soğutucu Açma
2 - Soğutucu Kapatma
3 - Sıcaklık Görüntüleme
4 - Çıkış
Seçiminiz : 4
Çıkış yapıldı.
Process finished with exit code 0
```

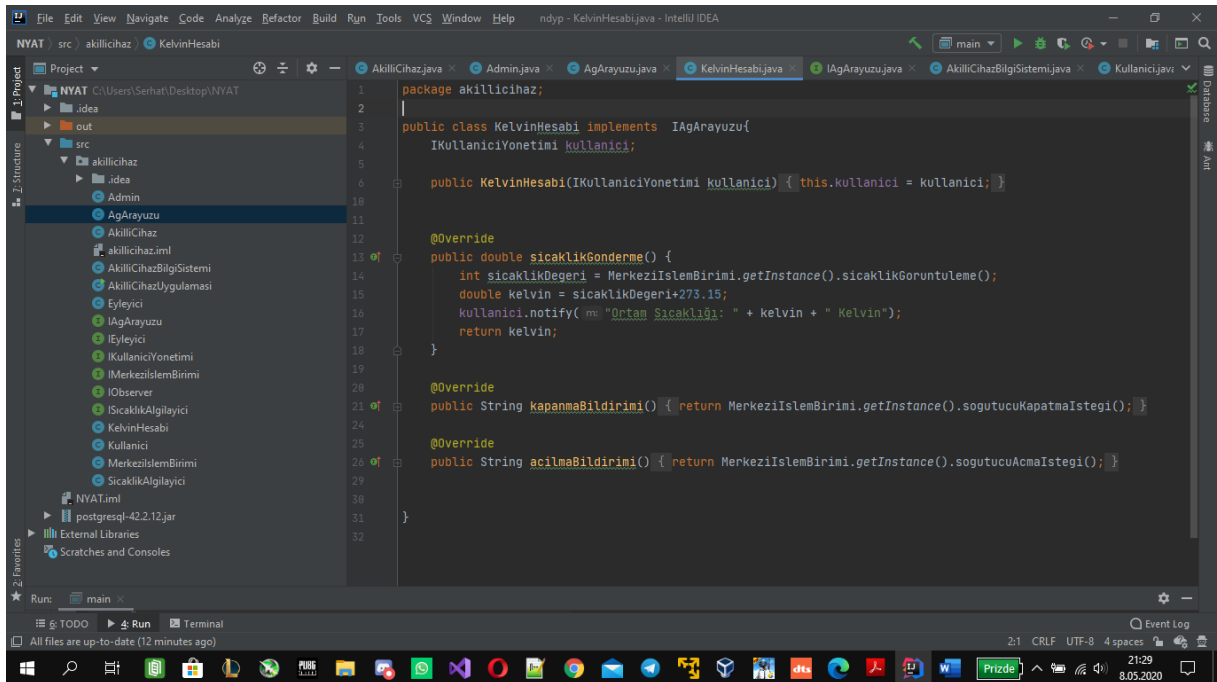
OPEN/CLOSED İLKESİ

Müşteriden gelen isteklerin veya sistemimizde değişiklik yapmak istediğimizde, sistemimizin rahatlıkla geliştirilebiliyor olması ve başka geliştirmelere de uzun kodlar yazmadan imkanı sağlanması adına, uygulamamızı Open/Closed prensibine göre geliştirmek, bizler için hayat kurtaran bir özelliktir.

Bende programımda hem sınıflar için arayüz yapıp hem de aynı arayüzden kalıtım alan iki sınıf oluşturdum. Bu sınıflar AgArayuzu ve KelvinHesabi sınıflarıdır. Kalıtım aldıkları arayüz ise IAgArayuzu interface i dir. İki sınıfta sıcaklık görüntülenmesi işlemlerini yapan methodlara sahip bu methodlar aynı ama işlevleri farklı.



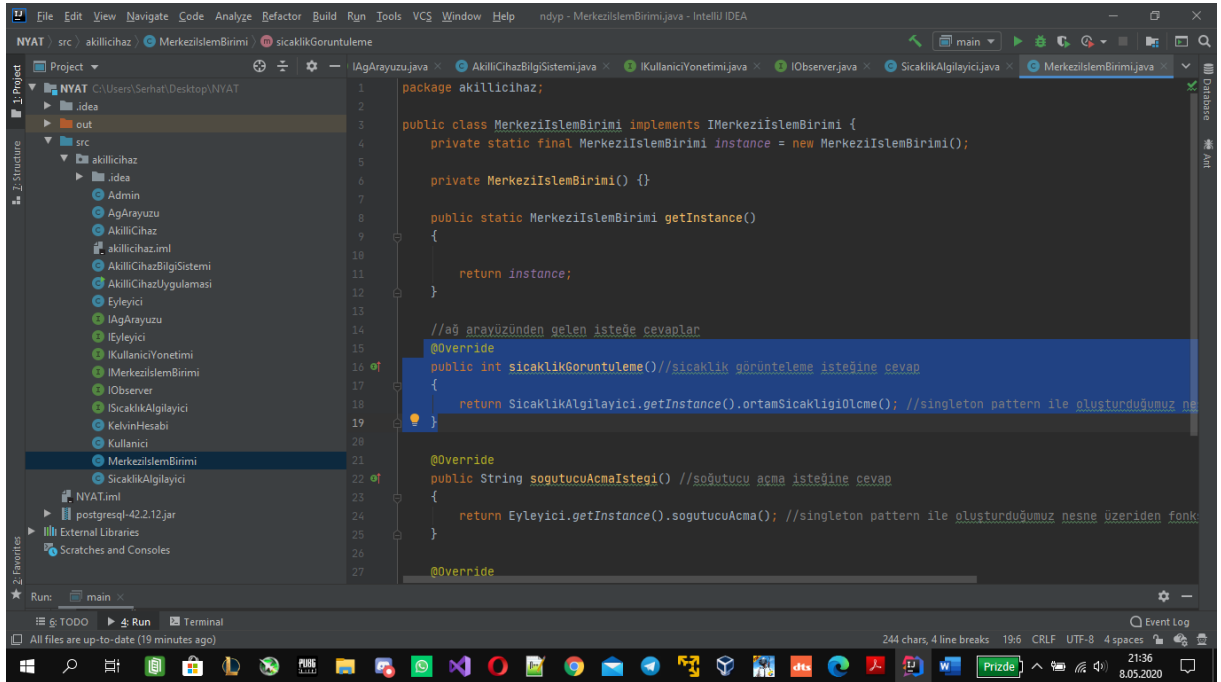
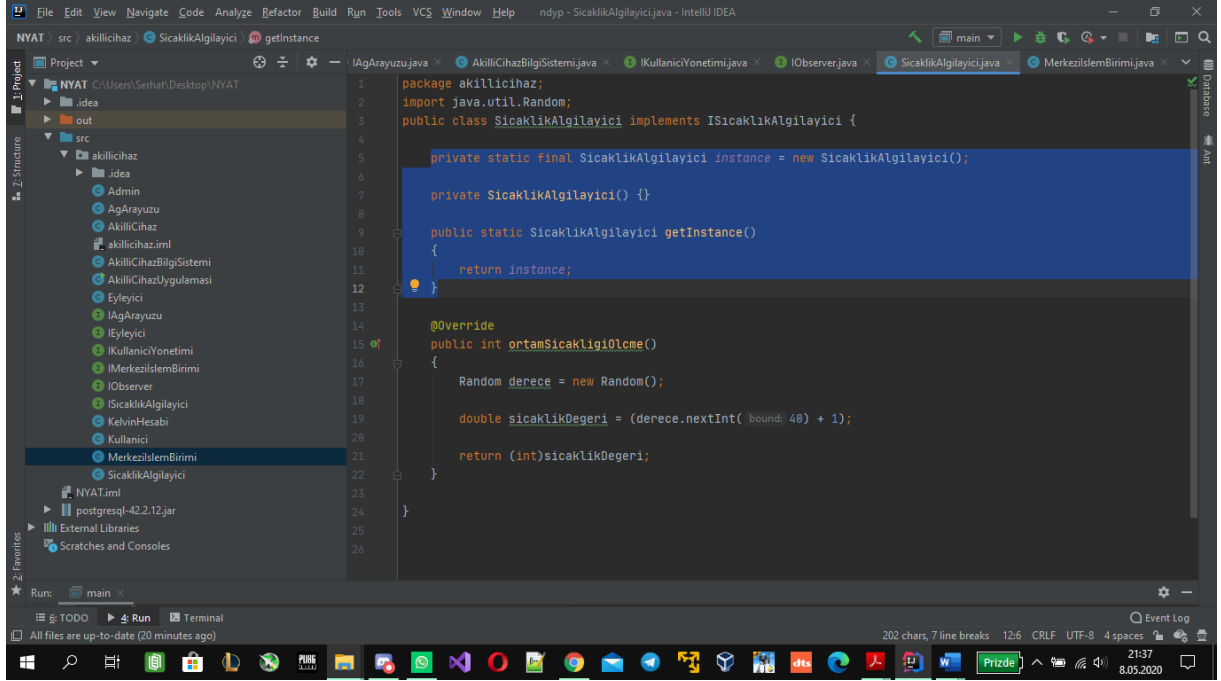
```
1 public class AgArayuzu implements IAgArayuzu {
2
3     IKullaniciYonetimi admin;
4
5     public AgArayuzu(IKullaniciYonetimi admin) {
6         this.admin = admin;
7     }
8
9     @Override
10    public double sicaklikGonderme() {
11        double sicaklik;
12        sicaklik = MerkeziIslemBirimi.getInstance().sicaklikGoruntuleme();
13        admin.notify(msg: "Ölüm Sıcaklığı: " + sicaklik + " Derece ");
14        return sicaklik;
15    }
16
17    @Override
18    public String kapanmaBildirim() { return MerkeziIslemBirimi.getInstance().sogutucuKapatmaIstegi(); }
19
20    @Override
21    public String acilmaBildirim() { return MerkeziIslemBirimi.getInstance().sogutucuAcmaIstegi(); }
22
23 }
```



```
1 package akillicihaz;
2
3 public class KelvinHesabi implements IAgArayuzu{
4     IKullaniciYonetimi kullanici;
5
6     public KelvinHesabi(IKullaniciYonetimi kullanici) { this.kullanici = kullanici; }
7
8     @Override
9     public double sicaklikGonderme() {
10        int sicaklikDegeri = MerkeziIslemBirimi.getInstance().sicaklikGoruntuleme();
11        double kelvin = sicaklikDegeri+273.15;
12        kullanici.notify(msg: "Ölüm Sıcaklığı: " + kelvin + " Kelvin");
13        return kelvin;
14    }
15
16    @Override
17    public String kapanmaBildirim() { return MerkeziIslemBirimi.getInstance().sogutucuKapatmaIstegi(); }
18
19    @Override
20    public String acilmaBildirim() { return MerkeziIslemBirimi.getInstance().sogutucuAcmaIstegi(); }
21
22 }
```

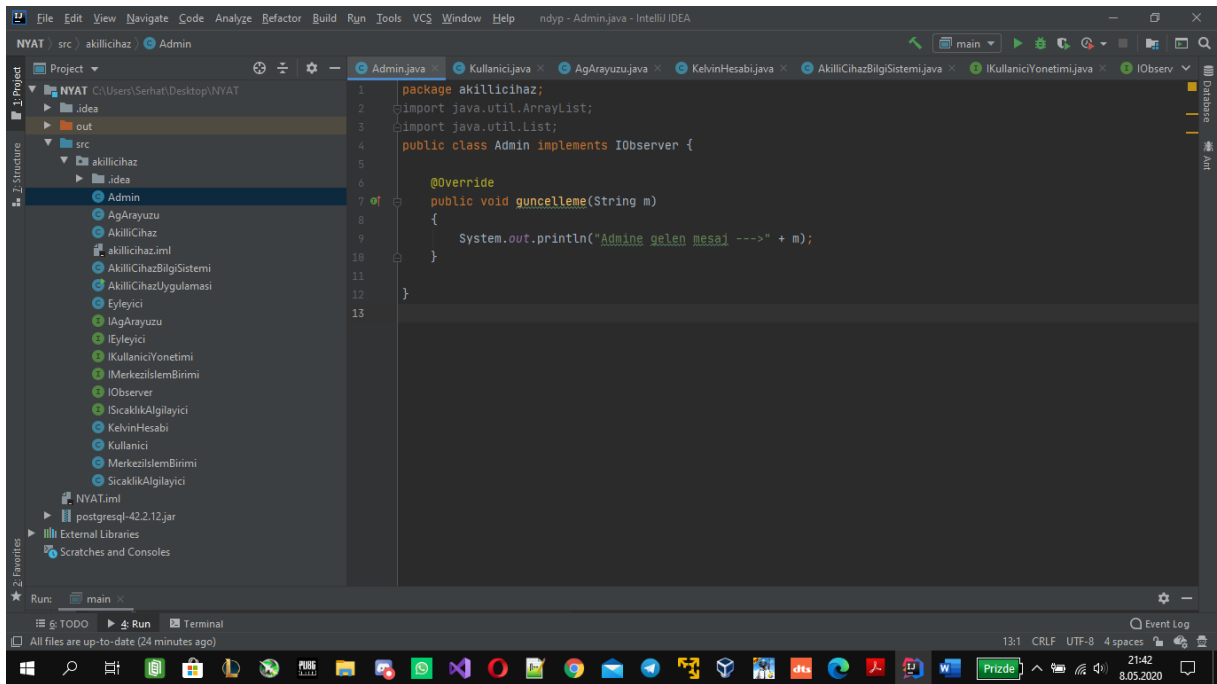
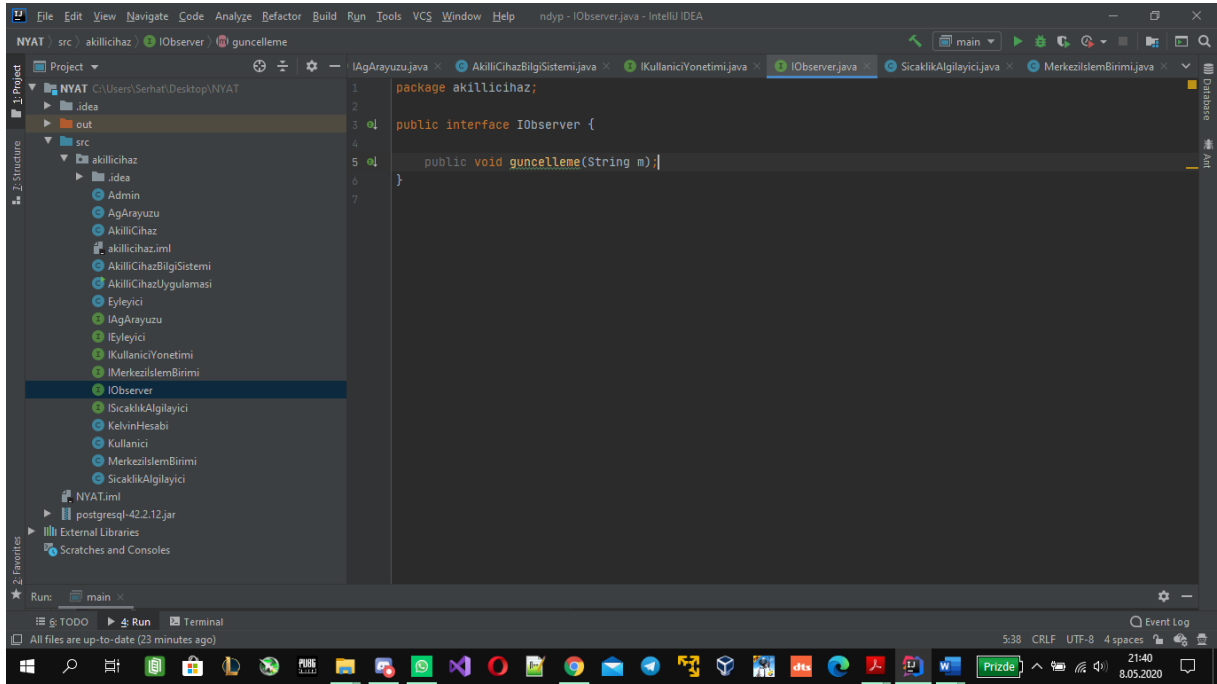
SINGLETON VE OBSERVER DESENLERİ

Singleton deseni, hazırlayacağımız sınıftan sadece bir örneğinin oluşturulmasını sağlar. Bu sayede nesnenin kopyalanmasını ya da yeni bir tane oluşturmasını engeller ve nesneye ihtiyaç duyulduğunda o nesnenin daha önceden oluşturulan örneğini çağırır.



Yukardaki fotoğraflarda mavili olan kısımlar bu desenin kullanımını gösteriyor. İlk resimde nesnemizi üretiyoruz ikinci resim yani alttaki resimde nesnemizi başka sınıfta kullanıyoruz.

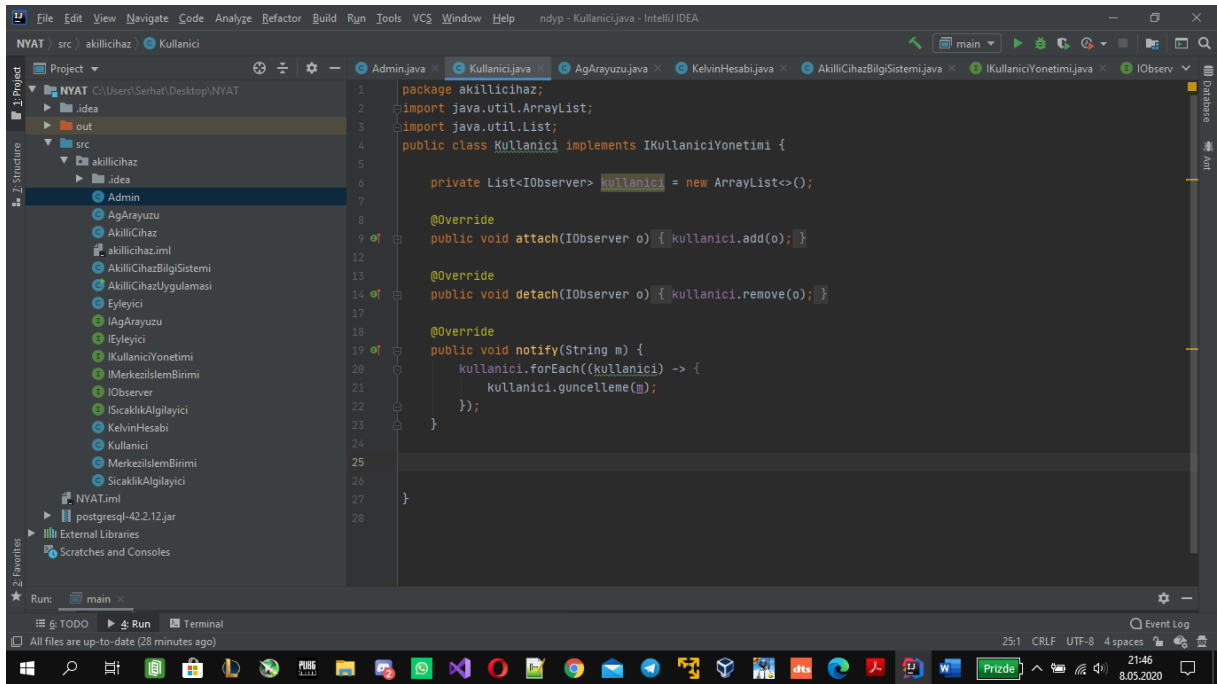
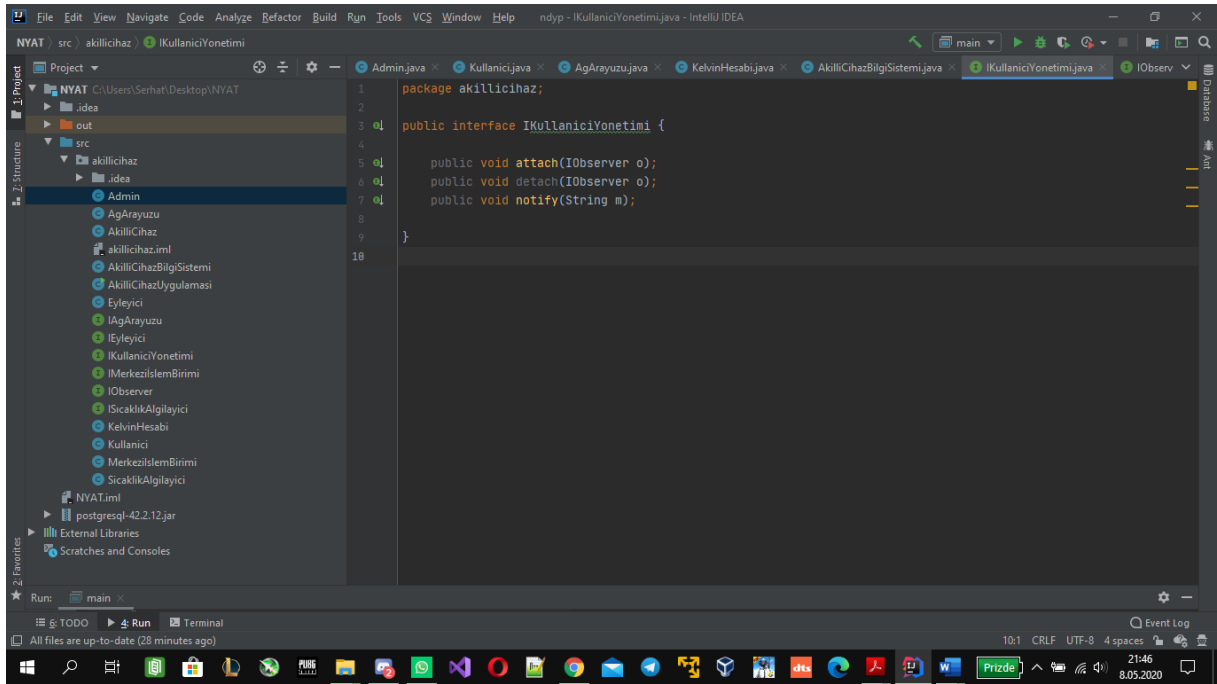
Observer deseni, behavioral tasarım desenlerinden biridir. Nesneler arasında one-to-many ilişki sağlar. Bir nesne durumunu değiştirdiğinde, ona bağlı diğer tüm nesneler uyarılır ve otomatik olarak güncellenir.



İlk fotoğrafta IObservable interface i değişiklik olduğunda nesneyi uyaracaktır.

İkinci fotoğrafta ise Admin sınıfı IObservable interface inden kalıtım alarak değişimi yazdırıyor.

Değişim işlemleri ise aşağıdaki fotoğraflardaki sınıflarda gerçekleşiyor.



VIDEO LİNKİ:

https://drive.google.com/file/d/1KADPaRRU0ywgEfij29QEXJKNd_eVEjK9/view?usp=sharing