T.C.
SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

NESNE YÖNELİMLİ ANALİZ ve TASARIM ÖDEV RAPORU

NESNELERİN İNTERNETİ SİSTEMLERİ İÇİN AKILLI CİHAZ TASARIMI

B201210400 - Umut Özgür DELİMEHMETOĞLU

2.Oğretim A Grubu

ozgur.delimehmetoglu@ogr.sakarya.edu.tr

SAKARYA

Nisan, 2022

Nesne Yönelimli Analiz ve Tasarım Dersi

NESNELERİN İNTERNETİ SİSTEMLERİ İÇİN AKILLI CİHAZ TASARIMI

Umut Özgür DELİMEHMETOĞLU

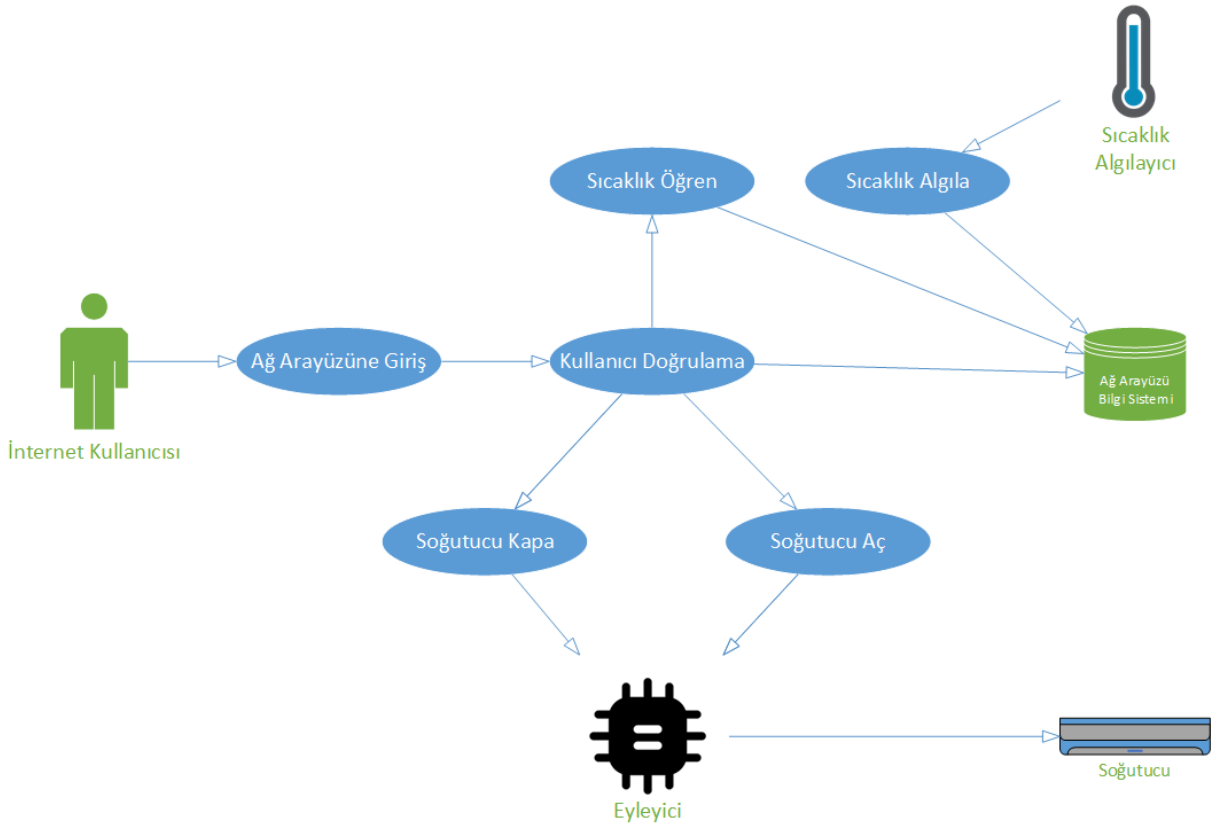
[a] *B201210400 2.Öğretim A Grubu*

Özet

Bu ödevde bizden nesnelerin interneti alanında kullanılacak bir sistemin yazılım tasarımını gerçekleştirmemiz istenmiş. Nesne yönelimli programlama kullanarak gerçekleştirilen projede bizden tasarım deseni olarak observer ve fabrika desenleri istendi. Ayrıca Dependency Inversion prensibiyle gerçekleştirildi.

PROJE TASARIMI

İnternet Kullanıcısı için Use Case Diyagramı



Şekil 1: Use Case Diyagramı

İnternet kullanıcısının use case diyagramı Şekil 1'de verilmiştir. Aktör olarak kullanıcı, algılayıcı, eyleyici,soğutucu ve Ağ Arayüzü bilgi sistemi olarak belirlendi. Tasarım buna göre gerçekleştirildi.

Kullanım Durumlarının Metinsel Tanımları
Soğutucunun Çalıştırılması Kullanım Durumu

Soğutucunun Çalıştırılması

- Akıllı Cihaz tarafından soğutucunun açılma sürecini tanımlar.
- 25/03/2022 v1.0 Umut Özgür DELİMEHMETOĞLU
- **İlgili Aktörler:** İnternet Kullanıcısı, Ağ Arayüzü Bilgi Sistemi, Eyleyici, Soğutucu

Giriş Koşulu: İnternet Kullanıcısı ağ arayüzüne bilgilerini yazar ve kullanıcı doğrulanır ve soğutucuyu çalıştırı seçerse gerçekleşir.
Çıkış Koşulu: İnternet Kullanıcısı işlemini bitirirse gerçekleşir.
Özel Gereksinimler: Elektrik Enerjisi, İnternet Bağlantısı, Mobil veya Web uygulaması, 7/24 çalışmalı, İşlemlerin gecikme süresi 5 saniyenin altında olmalı, Soğutucunun gücü aktif olmalı

Soğutucunun Çalıştırılması Ana Olay Akışı

1) Kullanıcı ağ arayüz uygulamasını açar.
2) Kullanıcıya giriş ekranı gösterilir.
3) Kullanıcı giriş için gerekli bilgileri yazar.
4) Ağ arayüzü bilgi sistemi, veri tabanından kullanıcı bilgilerini kontrol eder.
5) Ağ arayüzü bilgi sistemi kullanıcı bilgilerini doğrular.
6) Kullanıcı soğutucuyu aç seçeneğini seçer.
7) Ağ arayüzü bilgi sistemi ana işlem platformuna soğutuyucu aç sinyali gönderir.
8) Eyleyici soğutucuya sinyal gönderir.
9) Soğutucu açılır.
10) Soğutucunun açıldığı doğrulanır.
11) Ağ arayüzü bilgi sistemi, kullanıcı uygulamasına soğutucunun açıldığı mesajını gönderir.
12) Mesaj kullanıcıya gösterilir.

Soğutucunun Çalıştırılması Alternatif Olay Akışı

A1) Kullanıcı Doğrulanmadı (5)
6) Kullanıcıya hata mesajı gönderilir.
7) Giriş ekranına yönlendirilir.
A2) Ağ arayüzü soğutucuya aç sinyali gönderemez (7)
8) Kullanıcıya hata mesajı gönderilir.
9) İşlem sonlandırılır.
A3) Eyleyici soğutucuya sinyal gönderemez (8)
9) Ağ arayüzüne hata mesajı gönderilir.
10) Ağ arayüzü kullanıcı uygulamasına hata mesajı gönderir
11) Kullanıcıya hata mesajı gösterilir.
12) İşlem sonlandırılır

Yukarıdaki şekillerde sırasıyla soğutucunun çalıştırılmasının kullanım durumunun ayrıntıları verilmiştir.

Sıcaklığın Görüntülenmesi Kullanım Durumu

Sıcaklığın Görüntülenmesi

- Akıllı Cihaz tarafından Sıcaklığın görüntülenme sürecini tanımlar.
- 25/03/2022 v1.0 Umut Özgür DELİMEHMETOĞLU
- **İlgili Aktörler:** İnternet Kullanıcısı, Ağ Arayüzü Bilgi Sistemi, Sıcaklık algılayıcı

Giriş Koşulu: İnternet Kullanıcısı ağ arayüzüne bilgilerini yazar ve kullanıcı doğrulanır ve sıcaklığı görüntüle seçeneğini seçerse gerçekleşir.
Çıkış Koşulu: İnternet Kullanıcısı işlemini bitirirse gerçekleşir.
Özel Gereksinimler: Elektrik Enerjisi, İnternet Bağlantısı, Mobil veya Web uygulaması, 7/24 çalışmalı, İşlemlerin gecikme süresi 5 saniyenin altında olmalı

Sıcaklığın Görüntülenmesi Ana Olay Akışı

1) Kullanıcı ağ arayüz uygulamasını açar.
2) Kullanıcıya giriş ekranı gösterilir.
3) Kullanıcı giriş için gerekli bilgileri yazar.
4) Ağ arayüzü bilgi sistemi, veritabanından kullanıcı bilgilerini kontrol eder.
5) Ağ arayüzü bilgi sistemi kullanıcı bilgilerini doğrular.
6) Kullanıcı sıcaklığı görüntüle seçeneğini seçer.
7) Merkezi işlem platformu sıcaklık algılayıcıdan verileri alır.
8) Merkezi işlem platformu sıcaklık verilerini ağ arayüzü bilgi sistemine gönderir.
9) Ağ arayüzü bilgi sistemi sıcaklık verilerini kullanıcı uygulamasına gönderir.
10) Sıcaklık kullanıcıya gösterilir.

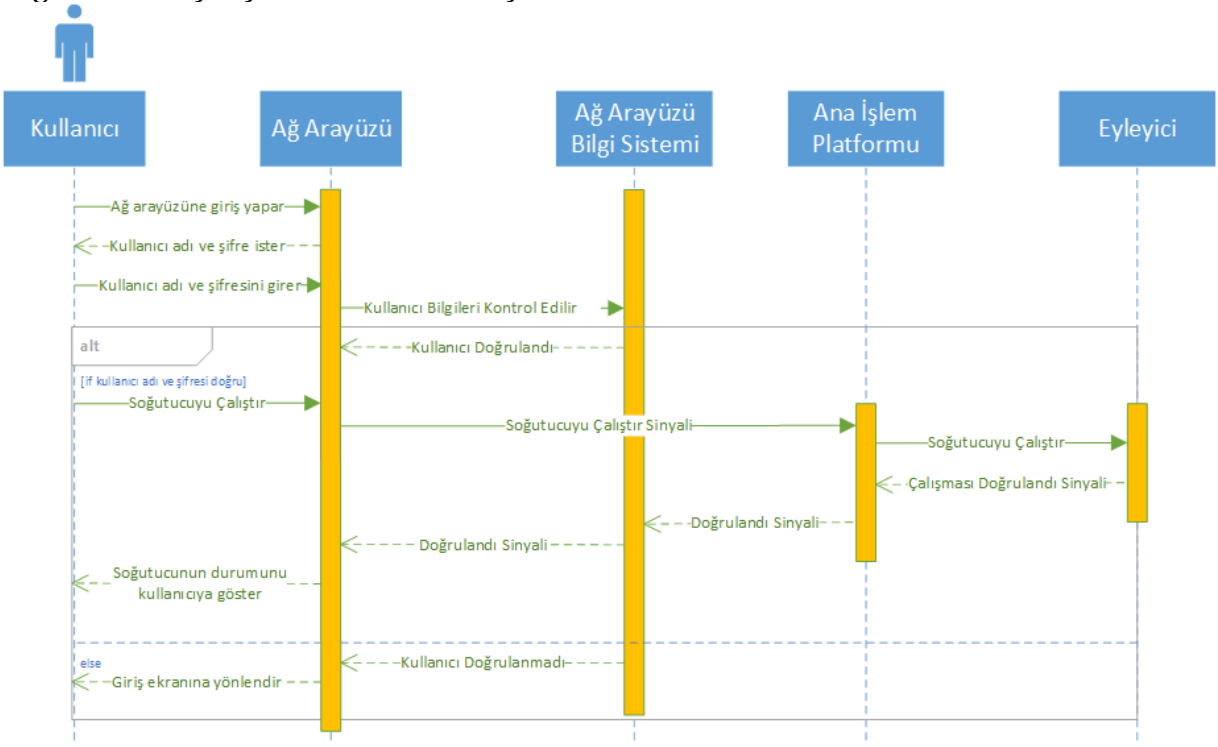
Sıcaklığın Görüntülenmesi Alternatif Olay Akışı

A1) Kullanıcı Doğrulanmadı (5)
6) Kullanıcıya hata mesajı gönderilir.
7) Giriş ekranına yönlendirilir.
A2) Sıcaklık Algılayıcından veri alınamaz (7)
8) Ağ arayüzü sıcaklık verisine hata değerini yazar.
9) Ağ arayüzü uygulamaya hata mesajı gönderir.
10) Kullanıcıya hata mesajı gönderilir.
11) İşlem sonlandırılır.
A3) Sıcaklık verileri kullanıcı uygulamasına gönderilemez (9)
10) Uygulama ağ arayüzüne bağlanamaz.
11) Kullanıcıya hata mesajı gönderilir.
12) İşlem sonlandırılır.

Yukarıdaki şekillerde sırasıyla sıcaklığın görüntülenmesi kullanım durumunun ayrıntıları verilmiştir.

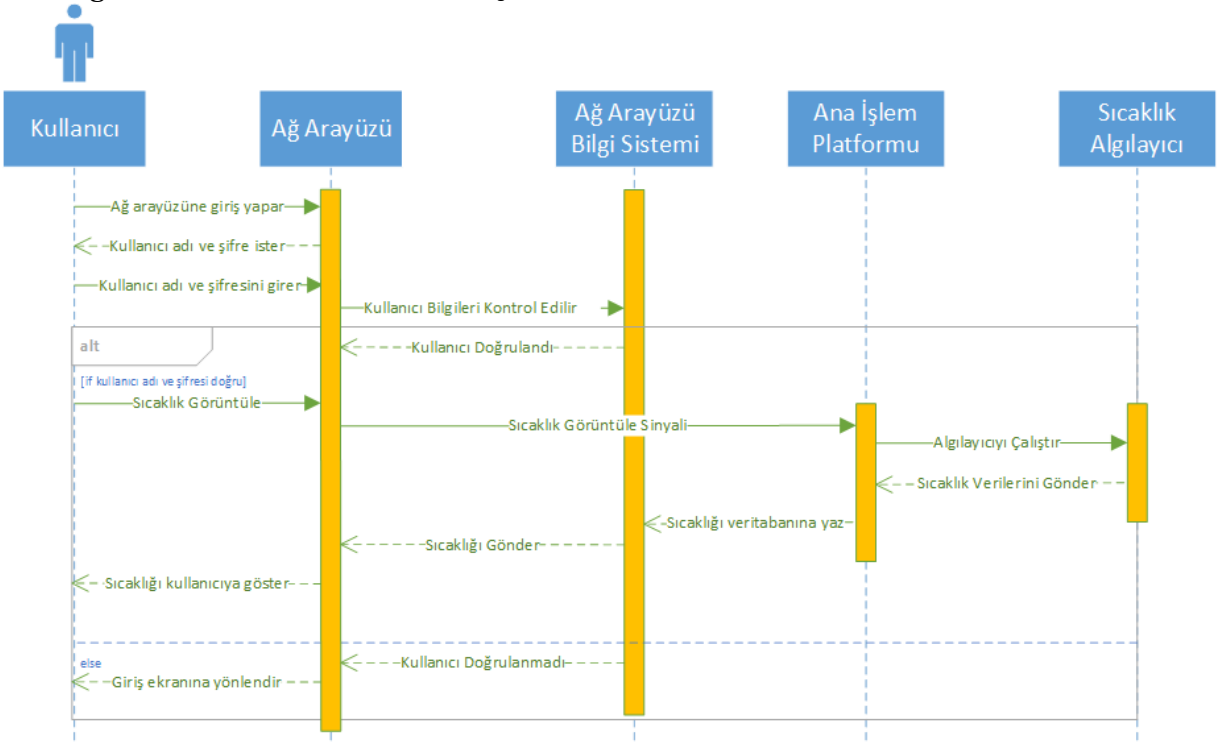
Sıralama Şemaları

Soğutucunun Çalıştırılması Sıralama Şeması



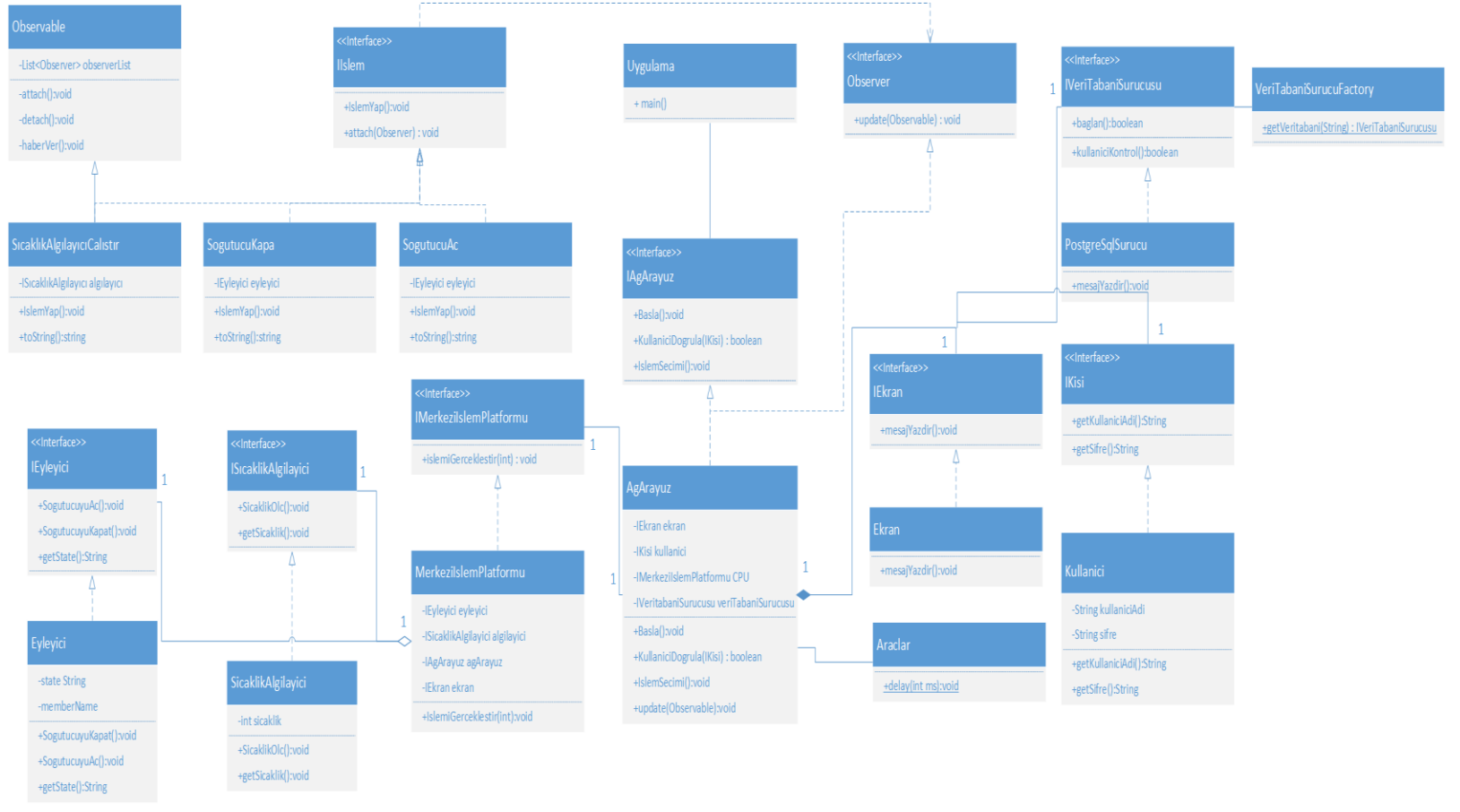
Şekil 2: Soğutucunun Çalıştırılması Sıralama Şeması

Sıcaklığın Görüntülenmesi Sıralama Şeması



Şekil 3: Sıcaklığın Görüntülenmesi Sıralama Şeması

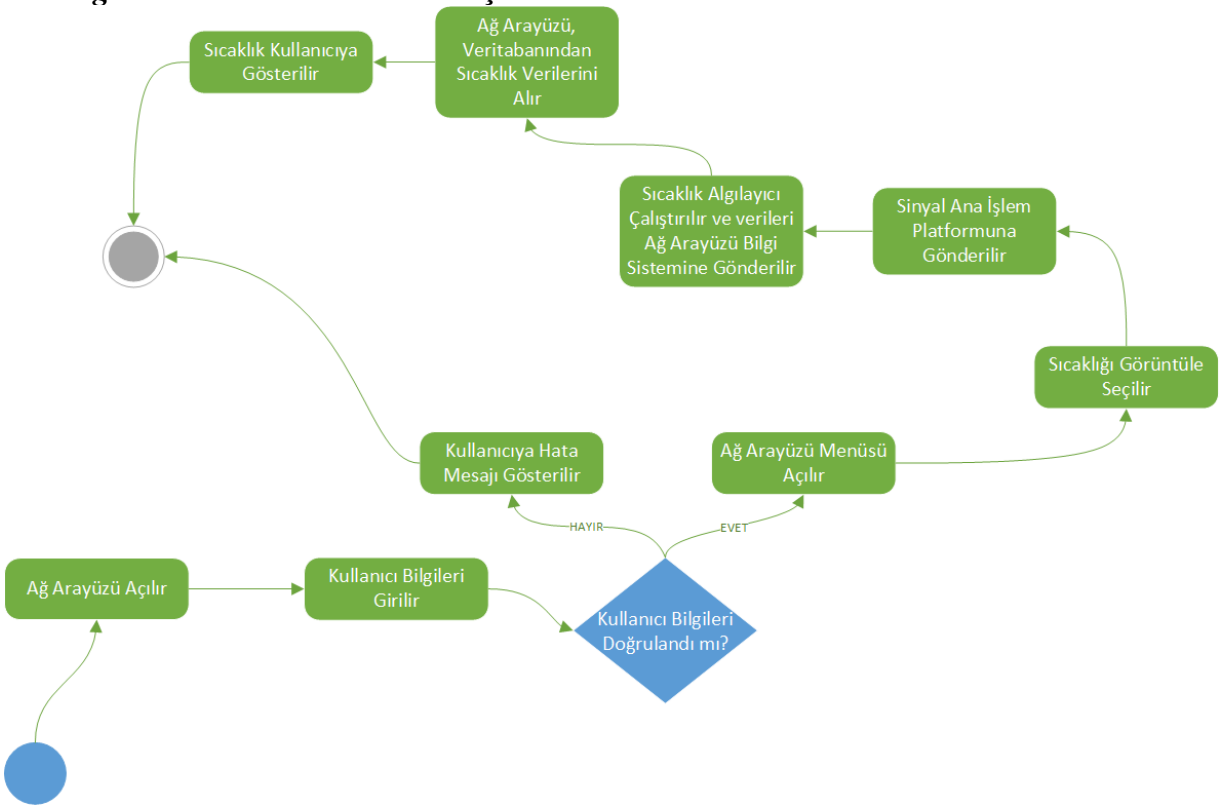
Sınıf Şeması



Şekil 4: Sistemin Sınıf Diyagramı

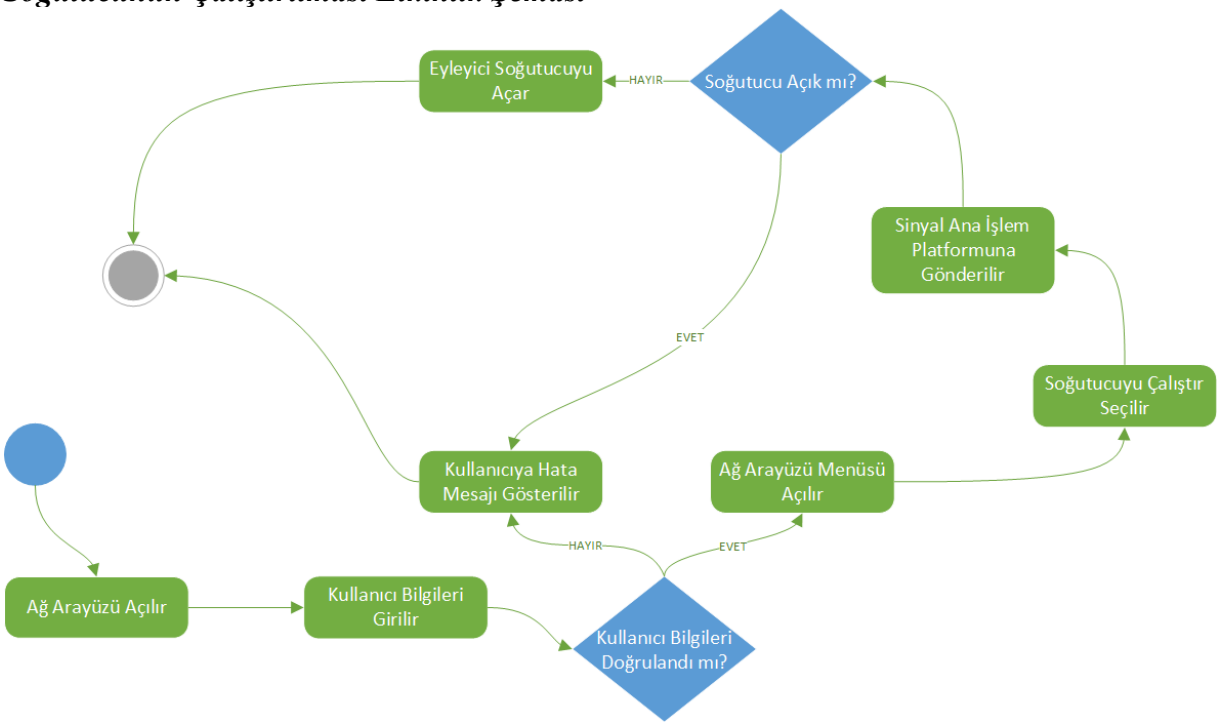
Etkinlik Şemaları

Sıcaklığın Görüntülenmesi Etkinlik Şeması



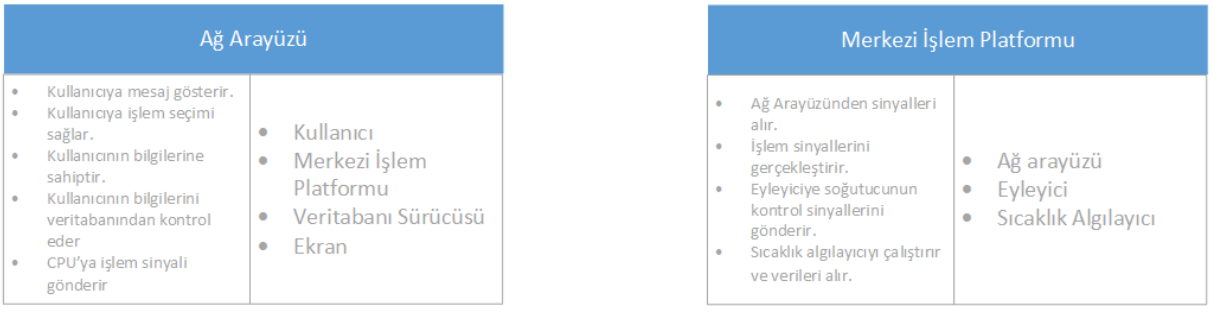
Şekil 5: Sıcaklığın Görüntülenmesi Etkinlik Şeması

Soğutucunun Çalıştırılması Etkinlik Şeması



Şekil 6: Soğutucunun Çalıştırılması Etkinlik Şeması

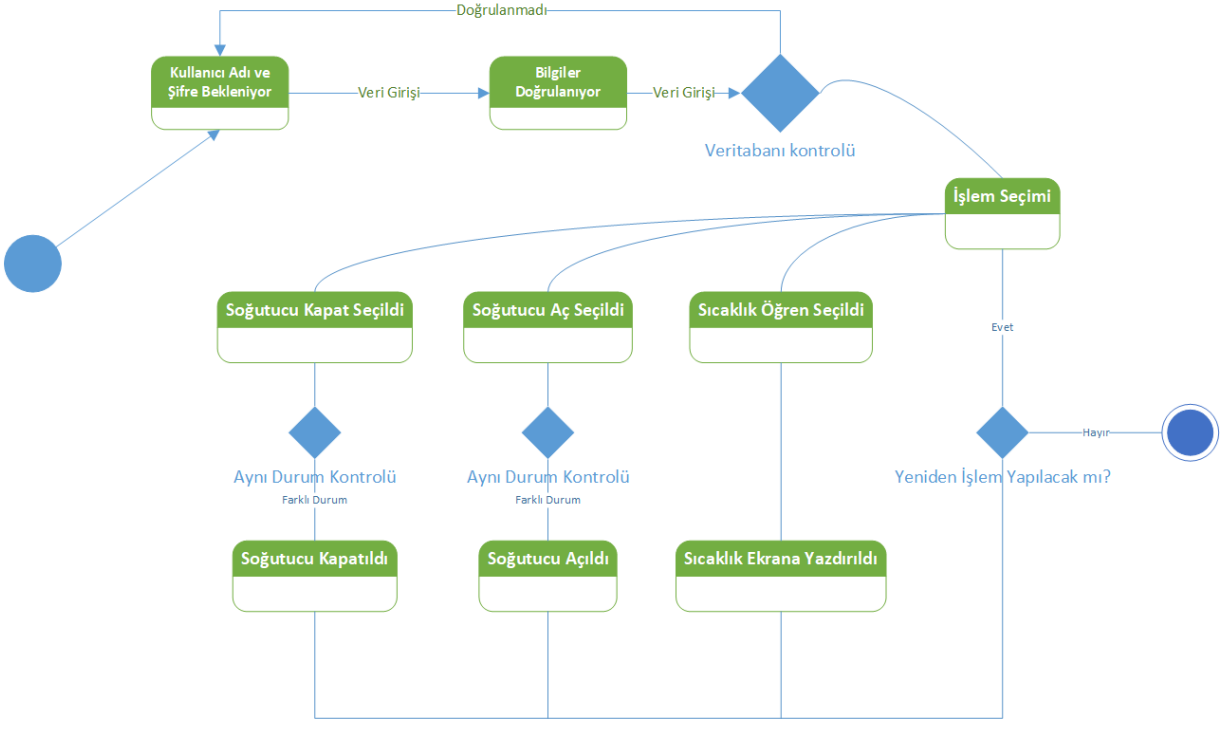
CRC Kartları



Şekil 7: Ağ Arayüzü ve Merkezi İşlem Platformunun CRC Kartları

Ağ arayüzü ve Merkezi İşlem Platformunun CRC kartları yukarıda verilmiştir.

Durum Makine Diyagramı

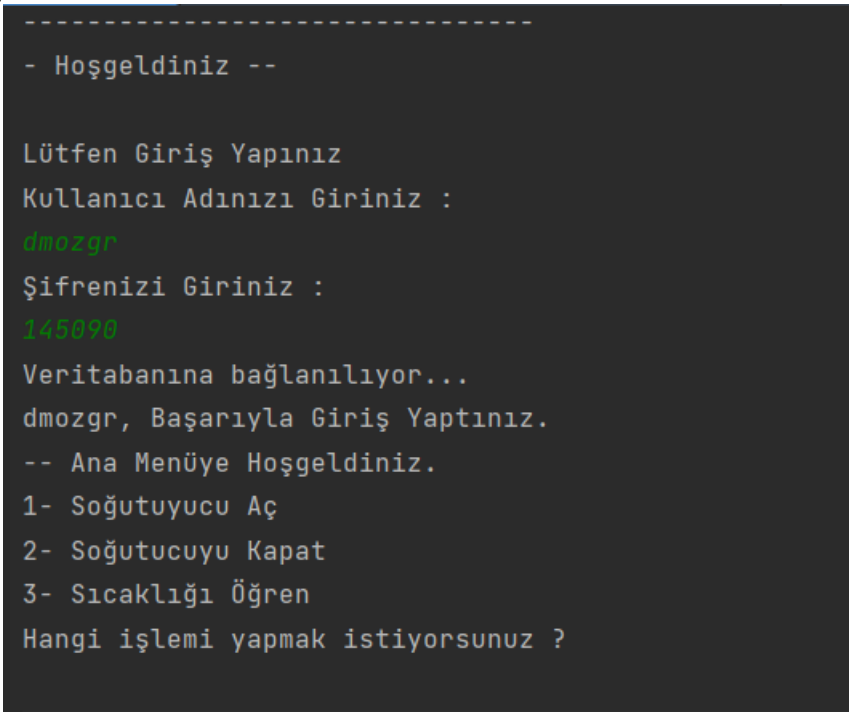


Şekil 8: Sistemin Durum Makine Diyagramı

TASARIMIN GERÇEKLENMESİ

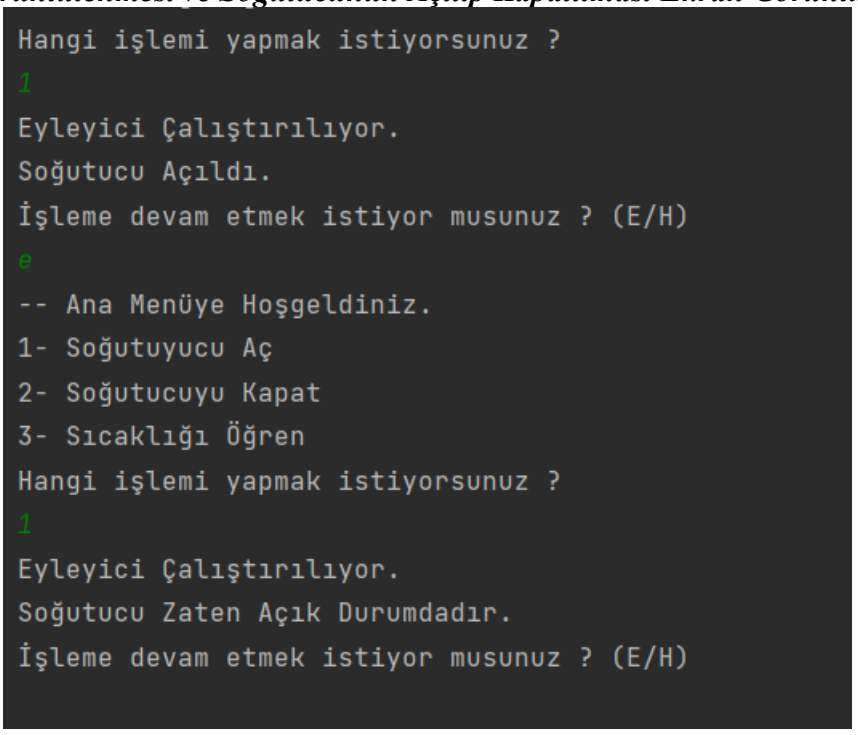
Aşağıdaki bölümlerde uygulamanın bazı anlarındaki ekran görüntüleri verilmiştir. Ayrıca rapor sonunda Github linki verilmiştir. Kaynak kodlarına oradan ulaşabilirsiniz.

Kullanıcı Doğrulaması



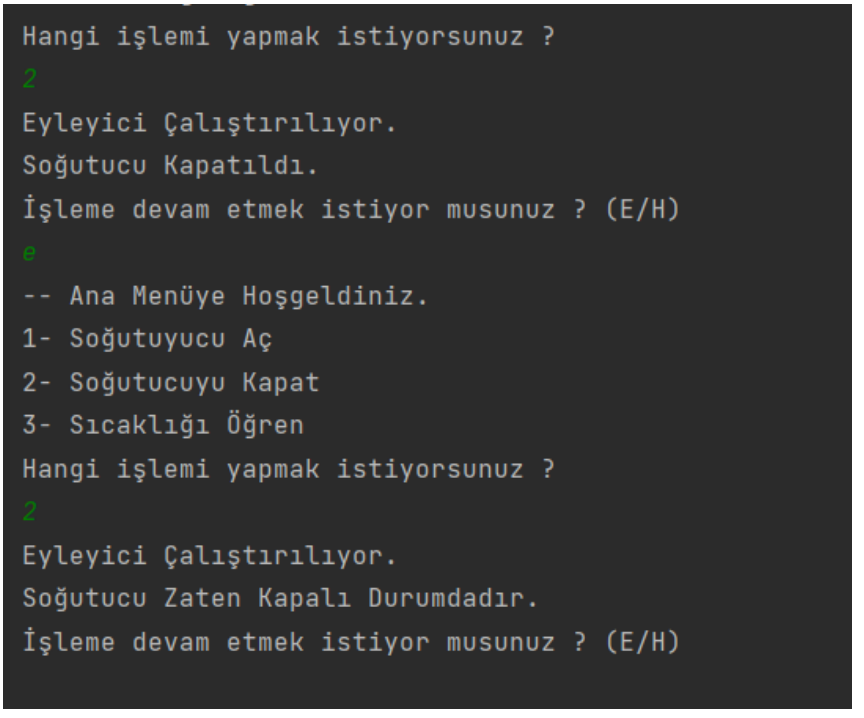
Şekil 9: Kullanıcı Doğrulama

Sıcaklığın Görüntülenmesi ve Soğutucunun Açılıp Kapatılması Ekran Görüntüleri



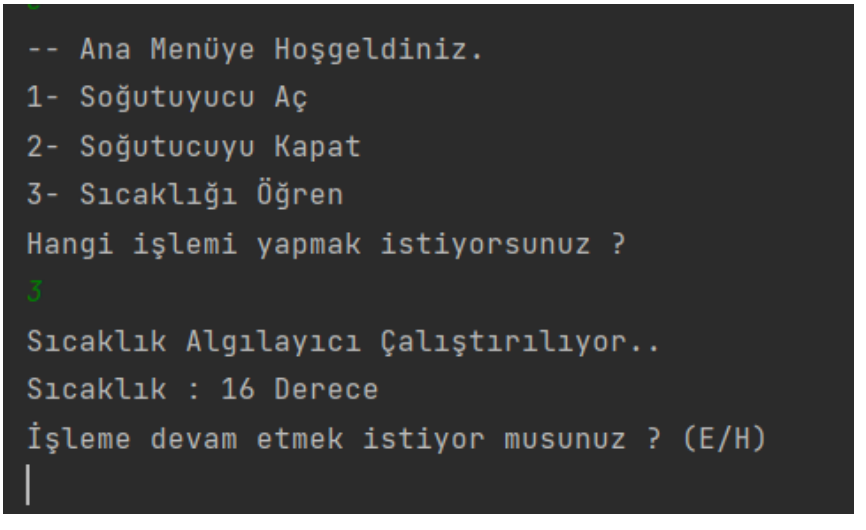
Şekil 10: Soğutucunun Açılması

Şekil 10'da soğutucu açılmasının ekran görüntüsü verilmiştir. Ayrıca soğutucu açıkken tekrardan açılması istenirse "Soğutucu Zaten Açık Durumdadır" uyarısı kullanıcıya gösterilmiştir.



Şekil 11: Soğutucunun Kapatılması

Şekil 10'da soğutucu kapatılmasının ekran görüntüsü verilmiştir. Ayrıca soğutucu kapalıyken tekrardan kapatılması istenirse "Soğutucu Zaten Kapalı Durumdadır" uyarısı kullanıcıya gösterilmiştir.



Şekil 12: Sıcaklığın Gösterilmesi

Veri Tabanı

	id	kullaniciAdi	sifre
✓	3	dmozgr	145090
✓	4	ozgrdmo	12345
✓	5	gamzecyln	1080q1450

Şekil 13: Veri Tabanı Ekran Görüntüsü

Tasarım Desenlerinin Kullanımı

Observer Tasarım Deseni

Observer tasarım deseni, birden fazla nesneyi takip ettikleri başka bir nesnede gerçekleşen olaylarla ilgili bilgilendirmeyi sağlayan bir abonelik mekanizması oluşturmayı amaçlar. Bu projede IIslem interface'nden implemente edilen SogutucuAc, SogutucuKapa ve SicaklikAlgilayiciCalistir sınıfları **observable** soyut sınıfından extends edilerek gözlemlenen olarak tanımlanmıştır. İşlem yapıldığında **Observer** olan AğArayüzü sınıfına haber vererek kullanıcıya mesaj verilmiştir.

```
public void IslemiGerceklestir(int secim){
    IIslem islem;
    switch (secim)
    {
        case 1:
            islem = new SogutucuAc(eyleyici);
            islem.attach((Observer) agArayuz);
            ekran.mesajYazdir("Eyleyici Çalıştırılıyor.");
            Araclar.delay( millisecond: 1000);
            islem.IslemYap();
            break;
```

Şekil 14: Attach Methodu

Attach methodu ile hangi işlem seçildiyse o işlemin Observable listesine agArayüzünü eklendi.

```
public class SogutucuKapat extends Observable implements IIslem {
    private IEyleyici eyleyici;
    public SogutucuKapat(IEyleyici eyleyici) { this.eyleyici = eyleyici; }

    @Override
    public void IslemYap() {
        haberVer();
        eyleyici.SogutucuyuKapat();
    }
```

Şekil 15: haberVer Methodu

HaberVer() – notify- method ile o işlem gerçekleştiğini ağarayüzüne haber verildi.

```
    @Override
    public void update(Observable observable) { ekran.mesajYazdir(observable.toString()); }
}
```

Şekil 16: Update Methodu

Update method'u ile hangi observable bu fonksiyona gelmiş ise o sınıfa ait toString() methodunu kullanıcıya gösteriyoruz.

- Sıcaklığın Gösterilmesi
- Soğutucunun Açılması
- Soğutucunun Kapatılması

Kullanıcı ekranında, yukarıdaki işlemlerin bilgilerini observer deseni ile görebiliyoruz.

Factory Method Deseni

Kelime anlamı "Fabrika Metodu" olan Factory Method, üst sınıfta nesneler oluşturmak için bir arabirim sağlayan, ancak alt sınıfların oluşturulacak bu nesne türünü değiştirmesine izin veren bir yaratımsal desen (creational pattern) türüdür. Bu projede VeriTabanıSürücüsü oluşturmak için factory method kullanılmıştır. Kullanıcı hangi sürücüyü istediyse fabrika o sürücüyü kullanıcıya üretecektir.

```java
public class VeriTabaniSurucuFactory {
    public static IVeriTabaniSurucusu getVeritabani(String surucu)
    {
        IVeriTabaniSurucusu veriTabaniSurucusu = null;
        if(surucu.equalsIgnoreCase( anotherString: "postresql"))
        {
            veriTabaniSurucusu = new PostreSqlSurucu();
        }
        return veriTabaniSurucusu;
    }
}
```

Şekil 17: Veritabanı Factory Sınıfı

```java
    public AgArayuz() {
        this.ekran = new Ekran();
        this.veriTabaniSurucusu=VeriTabaniSurucuFactory.getVeritabani( surucu: "postresql");
        this.CPU=new MerkeziIslemPlatform(ekran, agArayuz: this);
    }
```

Şekil 18: Factory Method kullanımı

Yukarıda AgArayuz sınıfının yapıcı methodunda istenen sürücüyü fabrikanın static get methodu ile alabiliyoruz.

S

Dependency Inversion Prensibi

Maddeleri basitleştirecek olursak; sınıflar arası bağımlılıkların minimal seviyeye indirgenmesi ve bağımlılıkların sınıflar ile değil arayüzler (**interface**) ile kurulması gerektiğine dayanır. Sistemimizi bu şekilde tasarlamazsak yüksek seviyeli bileşenler, düşük seviyeli bileşenlere bağımlı kalacak ve düşük seviyeli bir bileşen içerisinde yapılacak olan değişikliğin zincirleme olarak bağımlı olan tüm yüksek seviye bileşenleri de değişikliğe zorlayacaktır. Dependency Inversion tam olarak da bu bağımlılığın tersine çevrilmesini amaçlamaktadır. Bu projede kullanılan bütün bağıntılar “Loosely Coupling” olacak şekilde yapılmıştır.

```java
public class AgArayuz implements IAgArayuz,Observer{
    private IEkran ekran;
    private IKisi kullanici;
    private IVeriTabaniSurucusu veriTabaniSurucusu;
    private IMerkeziIslemPlatform CPU;
    public AgArayuz() {
        this.ekran = new Ekran();
        this.veriTabaniSurucusu=VeriTabaniSurucuFactory.getVeritabani( surucu: "postresql");
        this.CPU=new MerkeziIslemPlatform(ekran, agArayuz: this);
    }
```

Şekil 19: Dependency Inversion Prensibi

Şekil 19’da verilen şekil’de Dependency Inversion ilkesinin kullanımı görülmüştür. Bütün projede bu ilke benimsenmiş ve uygulanmıştır.

SONUÇ

Ödev olarak verilen proje kullanıcı isteklerininin hepsinin yerine getirilmesiyle bitirilmiştir. Hata durumları göz önüne alınarak gerçekleştirilen projede hatalar ayıklanmıştır. Code Reuse fazlaca olmakla beraber “Loosely Coupled” tasarım gerçeklenmiştir. Ayrıca Code Smells’den uzak bir proje yapımı amaçlanmıştır. Fonksiyon ve değişken isimlerine dikkat edilmiş, methodlar uzun yazılmamış, böylece yorum satırlarının olabildiğince az olması sağlanmıştır. Yazılım tasarımı ve OOP konusunda oldukça katkı sağlayan bu ödev için ilgili hocalarıma teşekkür ederim.

Kaynak kodlarını aşağıda verilen Github linkinde bulabilirsiniz.

https://github.com/dmozgr/NYATSogutucuKontrol