**1fd75c3872a94e26ad68c7fa7667bdc82c07dd4ba85f4a6793f7a2b4e943b8fd**

**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.

© 2022 Sakarya Üniversitesi.

Bu rapor benim özgün çalışmamdır. Faydalanmış olduğum kaynakları içeresinde belirttim. Her hangi bir kopya işleminde sorumluluk bana aittir.

## 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



Yukarıdaki şekillerde sırasıyla soğutucunun çalıştırılmasının kullanım durumunun ayrıntıları verilmiştir.

Sıcaklığın Görüntülenmesi Kullanım Durumu





Yukarıdaki şekillerde sırasıyla sıcaklığın görüntülenmesi kullanım durumunun ayrıntıları verilmiştir.

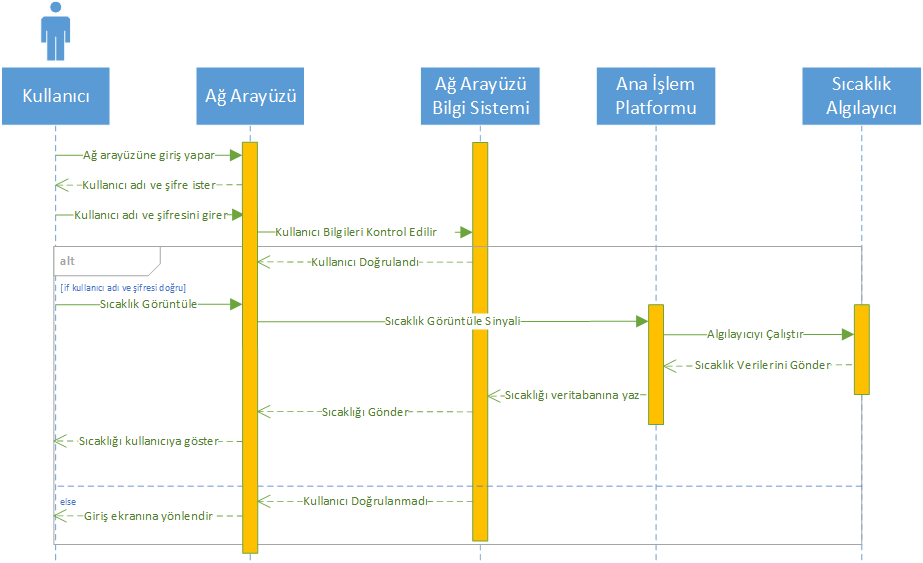
##### Sıralama Şemaları

Soğutucunun Çalıştırılması Sıralama Şeması



Şekil 2: Soğutucunun Çalıştırılması Sıralama Şeması

Sıcaklığın Görüntülenmesi Sıralama Şeması



Şekil 3: Sıcaklığın Görüntülenmesi Sıralama Şeması

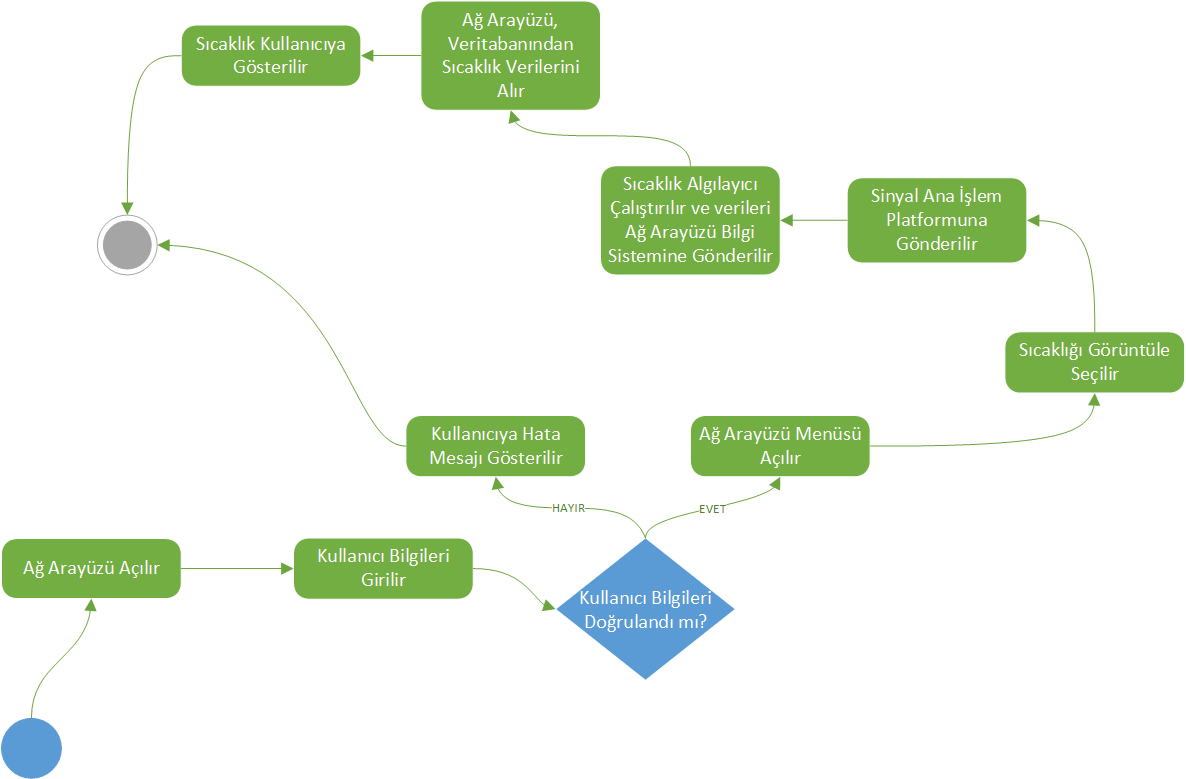
##### Sınıf Şeması



Şekil 4: Sistemin Sınıf Diyagramı

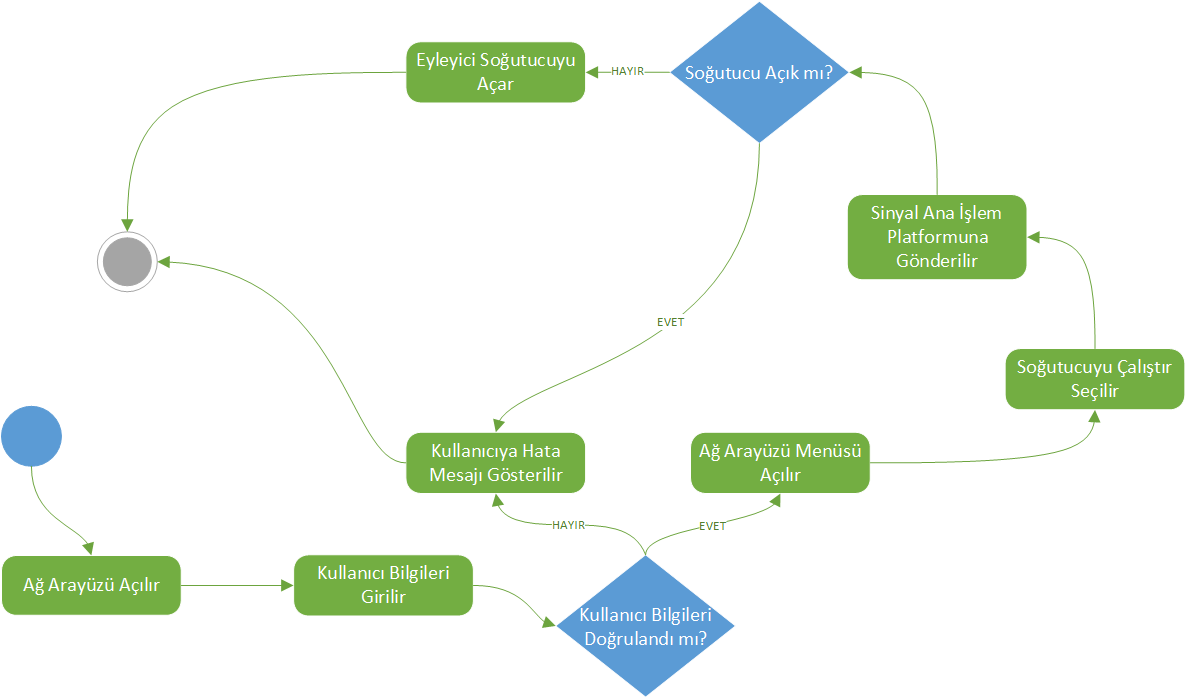
##### Etkinlik Şemaları

Sıcaklığın Görüntülenmesi Etkinlik Şeması



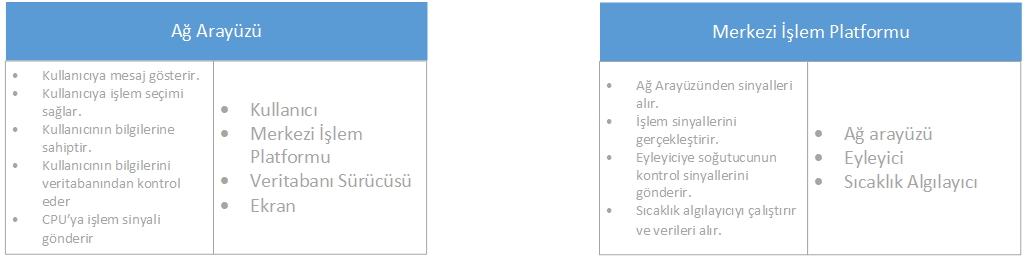
Şekil 5: Sıcaklığın Görüntülenmesi Etkinlik Şeması

Soğutucunun Çalıştırılması Etkinlik Şeması



Şekil 6: Soğutucunun Çalıştırılması Etkinlik Şeması

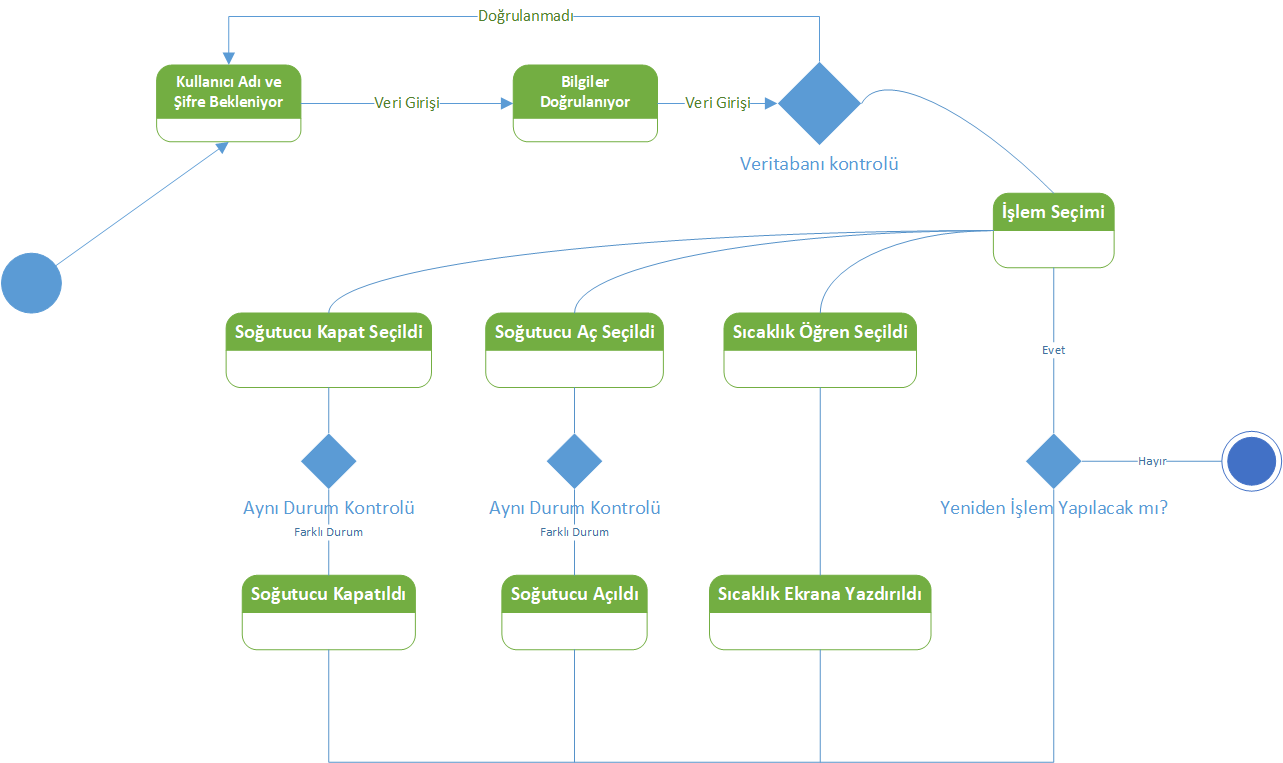
##### CRC Kartları



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

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 9: Kullanıcı Doğrulama

Sıcaklığın Görüntülenmesi ve Soğutucunun Açılıp Kapatılması Ekran Görüntüleri

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

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.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

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.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 12: Sıcaklığın Gösterilmesi

Veri Tabanı

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

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.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 14: Attach Methodu

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

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 15: haberVer Methodu

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

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

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.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 17: Veritabanı Factory Sınıfı

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

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.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

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>