ONDOKUZ MAYIS ÜNİVERSİTESİ
BİLGİSAYAR MÜH.

YAZILIM MÜHENDİSLİĞİ
PROJE RAPORU

RAPOR HAZIRLAYAN :

MOHAMED ALIWI	**15060937**
LAİTH ATİK	**14060878**
AHMAD ALSHAIKH	**14060873**

2018 - 2019

1. PROBLEM AÇIKLAMASI

1.1 ÖNSÖZ

Şuan mevcut pizza lokantalardan özel bir pizza istendiği zaman onlara arayıp hem müşterinin hem de görevlinin zamanı alır ve bazen yanlış anlaşılmalar yüzünden yanlış sipariş meydana gelebilmektedir, Bizim amacımız müşteri istediği pizza online isteyebilsin ki hem görevli kendi işinden başka işlerle uğraşmasın hem de müşteriye güzel sunduğumuz hizmetten faydalansın.

1.2 BİZİM AMACIMIZ

Amacımız müşterilerin kullanabileceği bir kullanıcı arayüzü (User Interface) ona bağlı bir veritabanı (Database) sunmamız olup müşteri kendi pizzaları için çeşitli malzemeleri seçebilsin ve siparişler verebilsin, ondan sonra da verilen siparişlerin yapılacağı bir mutfağa gönderilir. Bizim odakladığımız nokta "Kullanımı kolay olan (Easy to use)" web tabanlı bir uygulama geliştirmemizdir

1.3 MEVCUT ÇÖZÜM

Bir çok farklı web tabanlı sipariş sistemi vardır, fakat bu sistemler bir kullanıcı için o kadar farklı ve benzersiz (Unique) işlevsellik sunulmamaktadır, genelde bu sistemler sadece kullanıcılara ürün ekleme/silme olanağı verir ve çoğu da müşterilerine var olan pizzalara ek olarak kendi malzemelerini seçerek özelleştirme imkanı sunuyor.

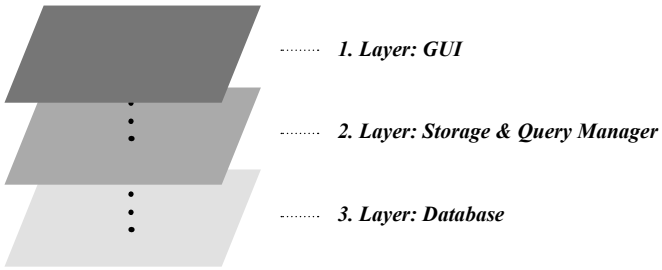
1.4 BİZİM ÇÖZÜMÜMÜZ

İstenen pizzalar ve yan ürün (Non pizza products) birbirinden ayıran bir sipariş sistemi geliştirmektir
Sipariş sistemi kullanıcıya üç temel bölüm sunmaktadır:
• Kendi pizzaları özelleştirme bölümü
• Yan ürün (Non pizza products) seçme bölümü
• Teslimat detayları bölümü
• Kendi siparişi için takip bölümü.
Kullanıcı istediği zaman herhangi birini geçebilir. Kullanıcı istediği pizza özelleştirildikten sonra isterse yan-ürün (Non pizza products) seçtikten sonra gerekli adres bilgileri girdiği zaman sipariş bizim mutfağımıza gelip gerekli işlemler görevli tarafında yürütülür.

2. SİSTEM ÖZELLİKLERİ (SYSTEM SPECİFİCATİON)

2.1 SİSTEM KATMANLARI

Sistem üç katmandan oluşur. Üstte Grafik Kullanıcı Arayüzü GUI (Graphical User Interface) katmanı vardır. Orta katman depolama ve sorgu yöneticisidir (Storage & Query Manager). Alt katmanı ise veritaban (Database) katmanıdır.



Resim 2.1 Sistem Katmanları (System Layers)

2.1.1 ARAYÜZ KATMANI

Arayüz katmanı, kullanıcıların sisteme erişmesine izin verir. Sistemin tüm işlevleri GUI aracılığıyla erişilebilir olmalıdır. İki ayrı arayüzler var. Biri müşterilerin pizza siparişleri oluşturmasıdır. Diğeri, siparişleri ve yönetim amaçlarını işlemek için yöneticiler içindir. GUI, giriş hatalarını önlemeli ve önlenemeyen hatalar durumunda, net hata mesajları vermelidir.

2.1.2 DEPOLAMA VE SORGU YÖNETME KATMANI

Depolama ve sorgu yöneticisi katmanı bilgi depolama, alma ve hata kontrolünden sorumludur. Bu katman, farklı sorguları kullanarak veritabanındaki varlık ve ilişkileri seçmeyi, eklemeyi, güncellemeyi, silmeyi sağlar.

2.1.3 VERİTABAN KATMANI

Veri tabanı katmanı, varlıkların ve bunların ilişkilerinin tüm verilerini içerir.

2.2 TEKNİK ÖZELLİKLERİ (TECHNİCAL SPECİFİCATİON)

Süreç boyunca kullanılacak araç ve yöntemleri seçme sürecinde, mevcut uzmanlığımıza ve projenin gerekliliklerine bir göz attık.

Backend programlama için ASP.NET en iyi seçim olurdu, ancak bunun açık kaynak ortamı olmaması ve veritabanının mysql tabanlı bir veritabanı olacağından, PHP doğru seçim oldu. PHP mysql ile mükemmel bir entegrasyona sahiptir.

Mysql veritabanını yönetmek için phpmyadmin kullanıldı. Bu mantıklı bir seçim idi çünkü hem ücretsiz hem de kullandığımız kaynaklar ile çok iyi entegrasyona sahiptir.

Frontend'da çalışmamızı gerçekleştirmek için JS framework'lerden Angular ve jQuery de kullandık.

3. GEREKSİNİMLER (REQUİREMENTS)

Bu bölümde sistemin farklı kullanıcı tipleri açıklanmaktadır. Ayrıca her bir kullanıcı tipi için fonksiyonel gereksinimler listelenir.

3.1 KULLANICILAR

İki tip kullanıcı sistemi kullanabilmelidir: Müşteri ve Yönetici. Müşteriler, web sitesini ziyaret eden ve pizzaları özelleştirerek sipariş oluşturabilen kullanıcılar. Yönetici sistemin nihai kontrolüne sahiptir, melzeme ve diğer ürünleri ekleyebilir, değiştirebilir veya silebilir.

3.2 FONKSİYONEL GEREKSİNİMLERİ (FUNCTİONAL REQUİREMENTS)

1- Müşteri

1.1 Müşteri yeni bir sipariş oluşturabilmelidir.
1.2 Müşteri bir pizzayı aşağıdaki şekilde özelleştirilmelidir:
    1.2.1 Müşteri, mevcut malzemelerin bir listesini görüntüleyebilmelidir.
    1.2.2 Müşteri, özel bir pizzaya malzeme ekleyebilmelidir
    1.2.3 Müşteri, bir malzemeyi özel bir pizzadan çıkarabilmelidir
    1.2.4 Müşteri, seçerek grafiksel geri bildirim alabilmelidir.
1.3 Müşteri, siparişe özel bir pizza ekleyebilmelidir.
1.4 Müşteri, mevcut yan ürünlerinin (Non pizza products) bir listesini görebilmesi gerekir.
1.5 Müşteri, siparişe yan ürünler (Non pizza products) ekleyebilmelidir.
1.6 Müşteri, özel pizzaların ve yan ürünler (Non pizza products) listesini görebilmelidir ve siparişe eklenir.
1.7 Müşteri, özel bir pizza miktarını değiştirebilmelidir.
1.8 Müşteri, yan ürün (Non pizza products) miktarını değiştirebilmelidir.
1.9 Müşteri, siparişten özel bir pizzayı silebilmelidir.
1.10 Müşteri, yan ürün (Non pizza products) siparişten silmeli.
1.11 Müşteri, bir siparişin toplam ücretini görebilmelidir.
1.12 Müşteri, müşterinin adını ve adresini ekleyebilmelidir.
1.13 Müşteri, mevcut siparişi yeni bir tane başlatmak için temizleyebilmelidir.
1.14 Müşteri, ödeme türü seçebilmeli (Nakit, Kredi kart).
1.15 Müşteri, siparişi onaylayabilmelidir.
1.16 Müşteri, kendi siparişi takibi yapabilir.

2- Yönetici - Admin

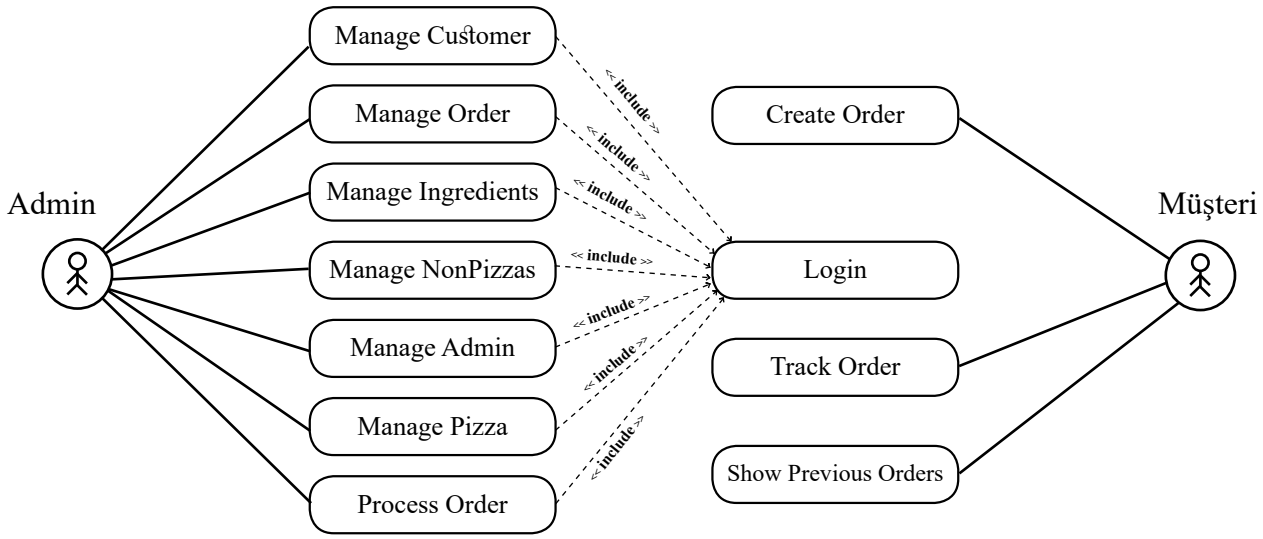
2.1 Yönetici, giriş ve çıkış yapabilmelidir.
2.2 Yönetici, mevcut siparişlerin ve onların özelleştirilmiş pizzaların listesini görebilmeli
2.3 Yönetici, gelen siparişler için “alındı” olarak işaretleyebilmelidir.
2.4 Yönetici, alınan siparişler için “hazırlandı” veya “hazırlanamadı” olarak işaretleyebilmelidir.
2.5 Yönetici, hazırlanan siparişler için “teslim edildi” veya “teslim edilemedi” olarak işaretleyebilmelidir.
2.6 Yönetici, siparişleri silebilmeli / güncelleyebilmeli.
2.7 Yönetici, malzemeleri ekleyebilmeli / silebilecek / güncelleyebilmeli.
2.8 Yönetici, yan ürünleri (Non pizza products) ekleyebilmeli / silebilmeli / güncelleyebilmeli.
2.9 Yönetici, yönetici ekleyebilmeli / silebilmeli / güncelleyebilmeli.
2.10 Yönetici, customer ekleyebilmeli / silebilmeli / güncelleyebilmeli.
2.11 Yönetici, pizza silebilmeli
2.12 Yönetici, sipariş logu görüntüleyebilmelidir.

3.3 FONKSİYONEL OLMAYAN GEREKSİNİMLERİ (NON FUNCTİONAL REQUİREMENTS)

Operasyonel bir gereklilik olarak, sistem bir web sitesi ile kullanıcı arayüzü olarak çalışacaktır. Performans gereksinimi olarak, sistem haftada yedi gün, günde 24 saat erişilebilir olmalıdır. Sipariş vermek için siteyi ziyaret eden müşteriler ziyaretleri için bir oturum (Session) oluşturulacaktır. Belirli bir süre sonra aktif olmayan oturum, ilgili sipariş bilgileriyle birlikte silinecektir.

4. KULLANIM DURUMLARI (USE CASES)

Genel Kullanim Durumların UML diyagramı aşağıdaki gibidir:



Şekil 4.1 Kullanım Durumların UML Diyagramı

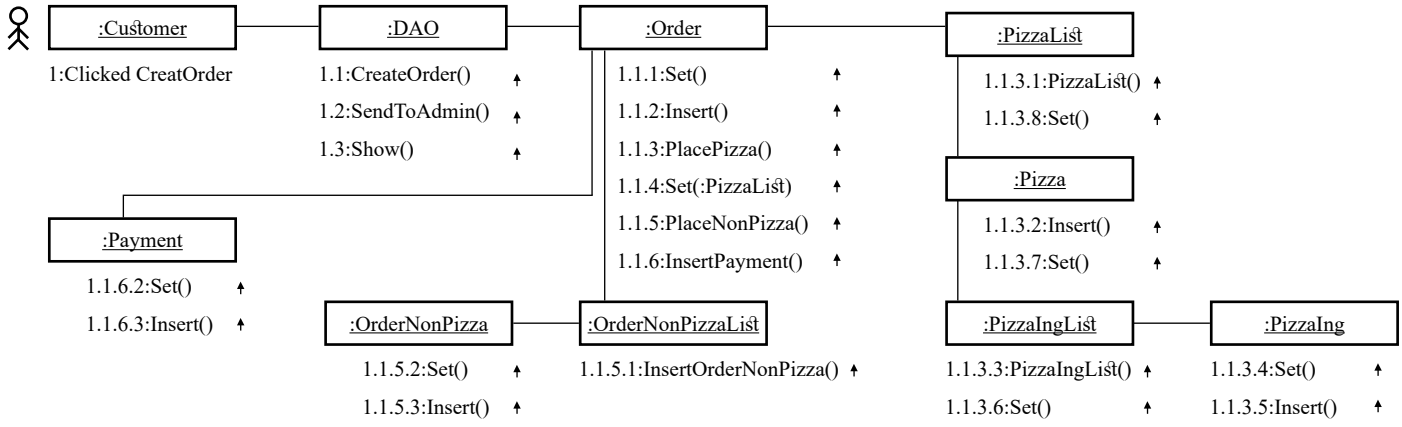
4.1 MÜŞTERİNİN KULLANIM DURUMLARI (CUSTOMER USE CASES)

A. CREATE ORDER KULLANIM DURUMU

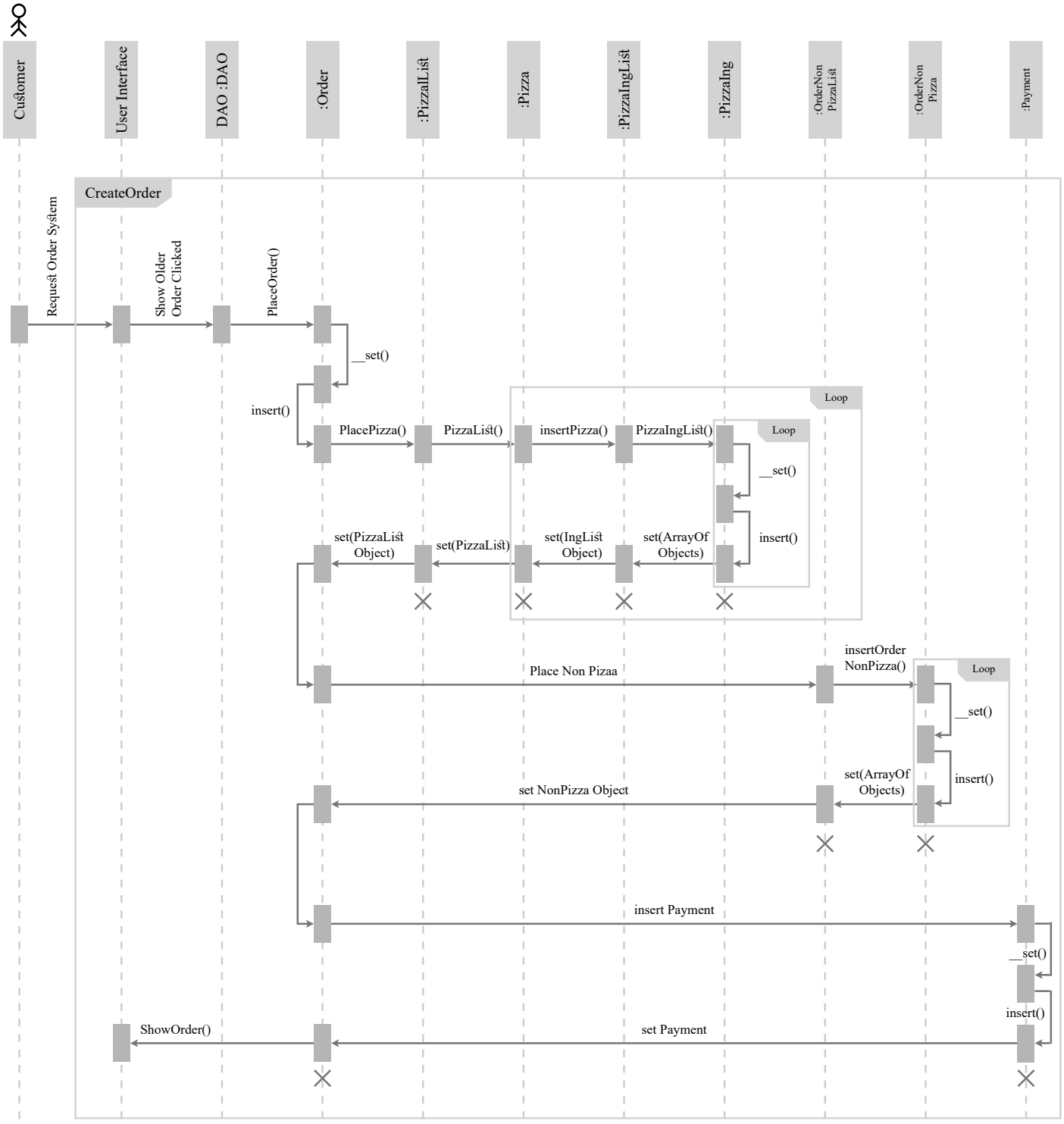
- Kullanım Durum Adı: Create Order
- Katılımcı Aktörler: Müşteri
- İş Akışı:

1. Malzemeler arasından seçip özelleştirilmiş bir pizza oluşturabilsin
2. Eklediği malzemeler Çıkarabilir
3. Özelleştirdiği pizza sepete Ekleyebilir, ekledikten sonra isterse çıkarabilir
4. Yeni bir pizza özelleştirilebilir
5. Non pizza ürün sepete ekleyebilir
6. Eklenen non pizza urunu miktari arttirilabilir ve sepeteden çıkarabilir
7. Bütün özelleştireceği pizzalar özelleştirildikten sonra teslimat için gerekli bilgiler girer
8. Ödeme türü seçer ve onaylar
9. Müşterinin nesnesi oluşturulur ve database bir sorgu gonderır
10. Gönderilen müşteri bilgileri bir kaydımız uyuşmuyor ise yeni müşteri kaydı yapılır değilse müşterinin ID'si get() yapilir
11. Müşterinin ID'sine bağlı bir sipariş oluşturulur
12. Sipariş içindeki pizza ve non pizza sipariş ID'sine bağlı kayıtlar yapılır (DATABASE)
13. Pizzayı oluşturan malzemeler Pizza ID'sine bağlı kayıtlar yapılır
14. Siparişin yapılacağı mutfağa gönderilir aynı anda kullanıcı bir bildirim alir "siparis iletildi"

- Başlangıç Koşullu: Bir Müşteri Sistemi girdiği zaman
- Bitiş Koşullu: İlgli müşteri siparişini onayladığı zaman
- Kalite Gereksinimler: -



Şekil 4.1.1 CreateOrder Kullanım Durumun Haberleşme Diyagramı



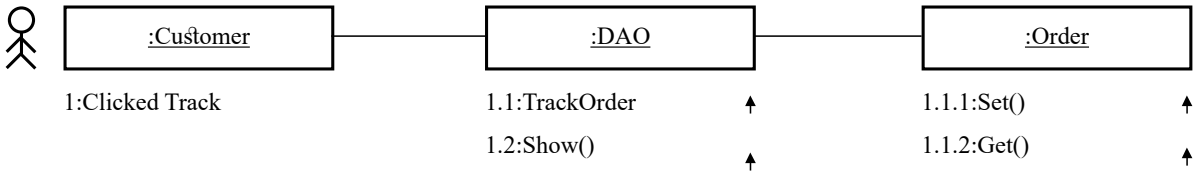
Şekil 4.1.2 CreateOrder Kullanım Durumun Sekans Diyagramı

B. TRACK ORDER KULLANIM DURUMU

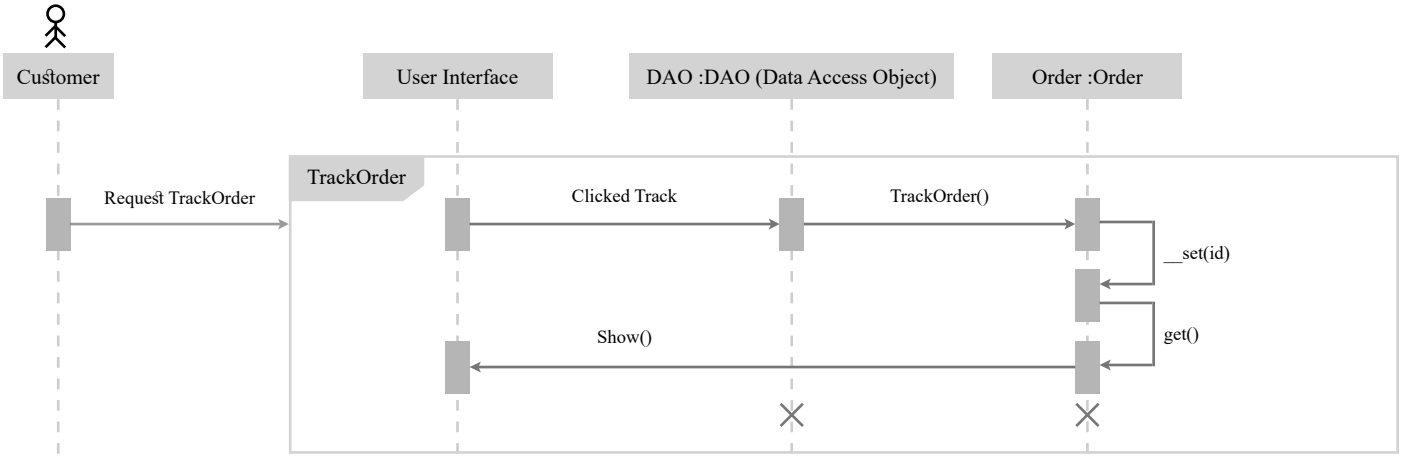
- Kullanım Durum Adı: Track Order
- Katılımcı Aktörler: Müşteri
- İş Akışı:

1. Müşteri Sparisin Id girer
    2. Database sorgu gonderilir
3. Gonderilen siparis id kayitlarimizda birini uyuşmuyor ise cikip hata mesaj verir aksi halde gönderilen sipariş ID'sine göre bütün siparis bilgileri getirilir
    4. Siparis ID'sine göre bu siparişe ait bütün pizza ve varsa non pizza ürünleri getirilir
5. Gelen pizza id'lerine göre her pizzanin malzemeleri getirilir
    6. Son olarak butun siparişin bilgileri musteriye goruntulenir ve siparis statusu gosterilir

- Başlangıç Koşullu: Müşteri bir siparişi takibi yapmak istediği zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: -



Şekil 4.1.3 Track Order Kullanım Durumun Haberleşme Diyagramı



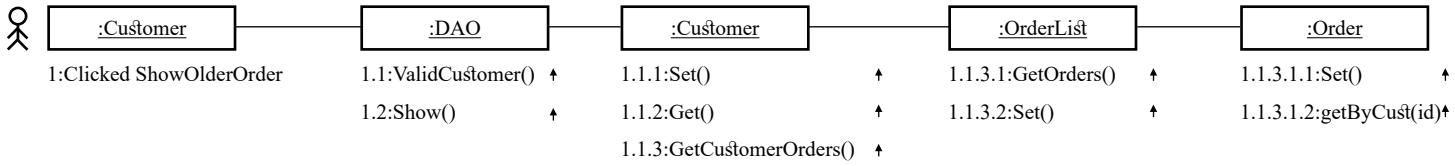
Şekil 4.1.4 Track Order Kullanım Durumun Sekans Diyagramı

C. SHOW PREVİOUS ORDER KULLANIM DURUMU

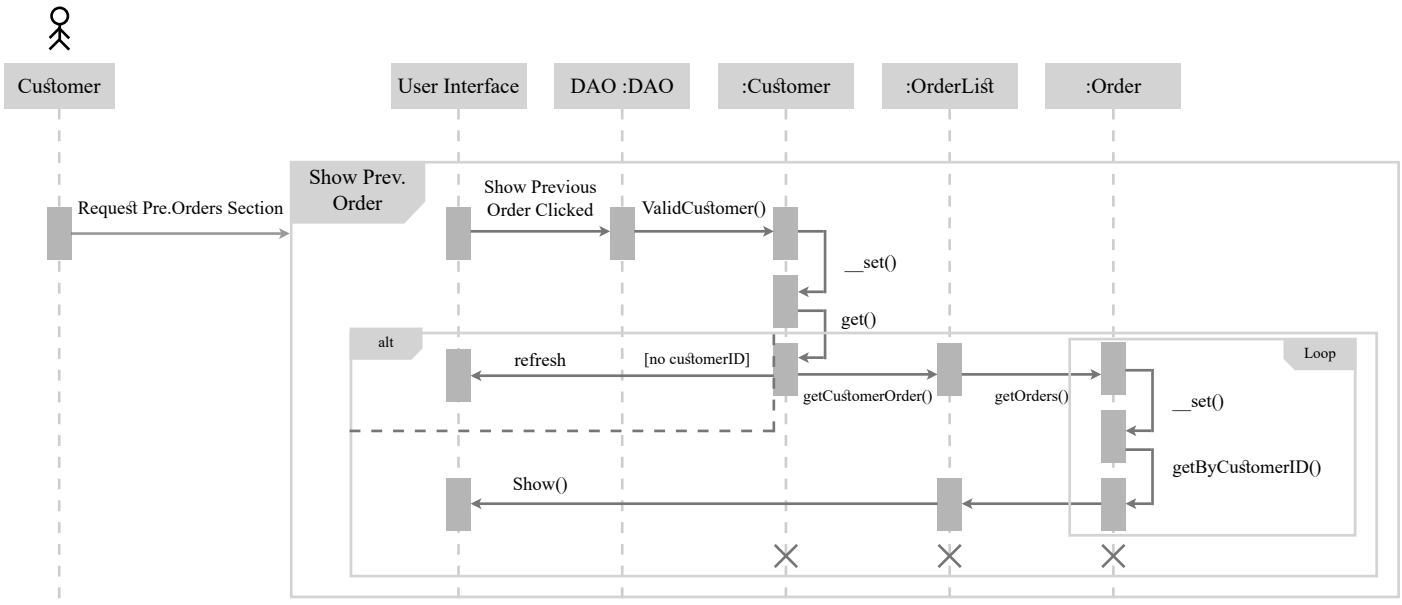
- Kullanım Durum Adı: Show Previous Order
- Katılımcı Aktörler: Müşteri
- İş Akışı:

1. Müşterinin bilgileri girer
   2. Veritabanına sorgu gonderilir
3. Gönderilen müşterinin bilgiler kayıtlarımızda bulunmuyor ise çıkıp hata mesaj verir, aksi halde gönderilen müşterinin bilgilere göre bütün siparisleri getirilir

- Başlangıç Koşullu: Müşteri eskiden istemis oldugu siparislar gormek istedigi zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: listeleyen siparişler yeniden istenilmesi



Şekil 4.1.5 Show Previous Order Kullanım Durumun Haberleşme Diyagramı



Şekil 4.1.6 Show Previous Order Kullanım Durumun Sekans Diyagramı

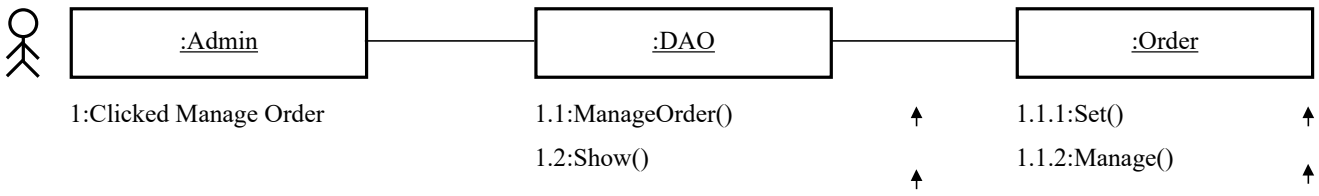
4.2 YÖNETİCİNİN KULLANIM DURUMLARI (ADMİN USE CASES)

A. MANAGE ORDER KULLANIM DURUMU

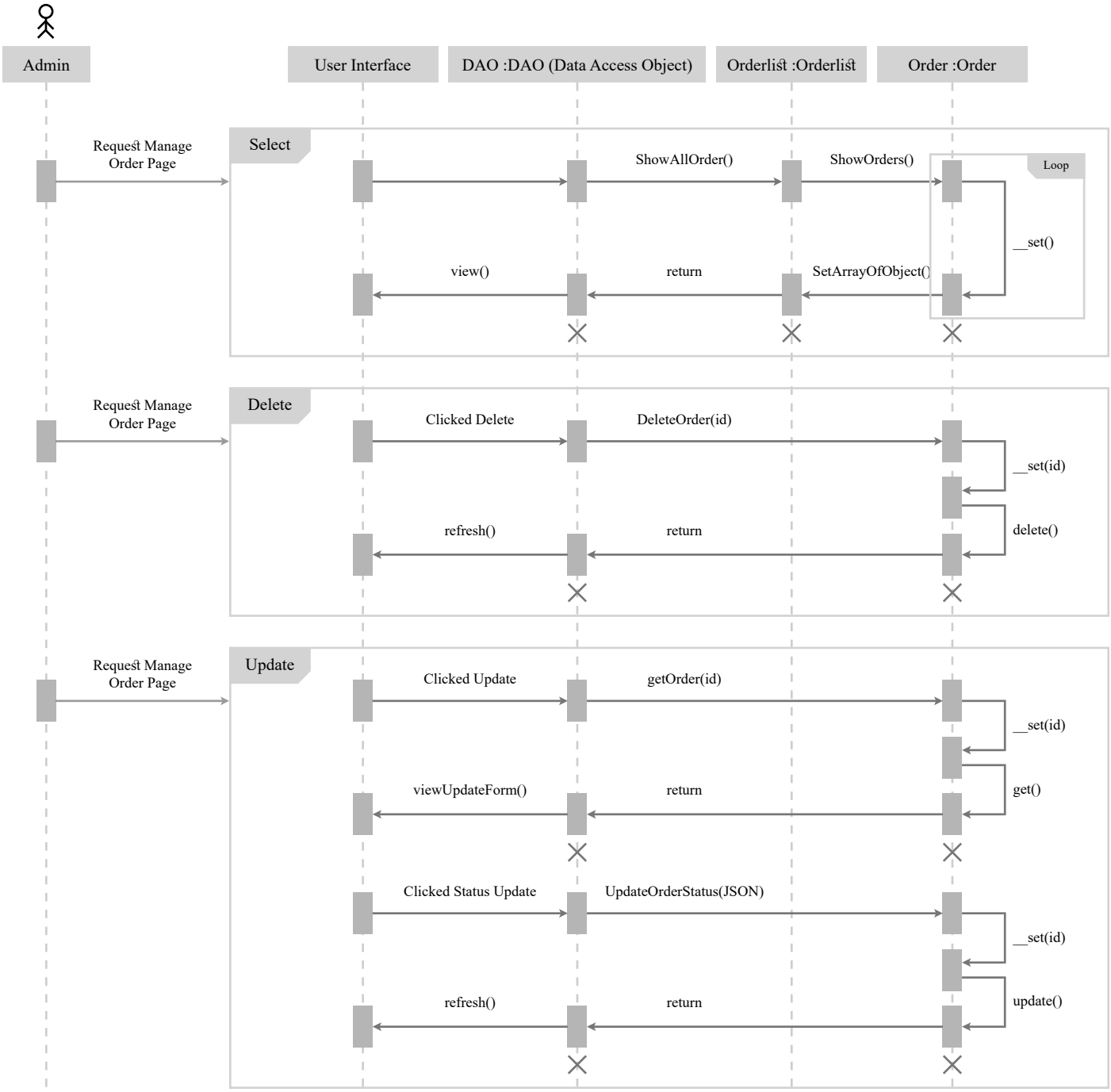
- Kullanım Durum Adı: Manage Order
- Katılımcı Aktörler: Admin
- İş Akışı:

1. Login kullanım durumu
    2. Login kullanım durumu başarı ile bittikten sonra admine hemen görünecek sayfa bütün siparişler içerir ve bu siparisler üzerinde delete / update status işlemleri yapabilir
3. Eğer delete button'a tıklanır ise bir sipariş nesnesi oluşturup sadece onun ID içerir, Yeni bir pizza özelleştirilebilir
    4. Ondan sonra siparis ID içeren bir sorgu veritabanına sorgular, delete işlemi gerçekleştiği zaman sayfa yeniden yuklenir,
5. Eğer Update Status button'a tiklarsa siparis nesnesi oluşturup sadece onun ID içerir
    6. Bu ID sayesinde butun siparis bilgileri getirir ve order update arayüzüne siparis bilgileri yükler
7. Order status'una Gerekli değişiklik yaptıktan sonra update button'a tıklanır ise bütün girilen bilgileri alir ve veritabanındaki kaydı güncellenip sayfa yeniden yüklenir

- Başlangıç Koşullu: Siprais üzerinde islemleri yapmayı istediği zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: -



Şekil 4.2.1 Manage Order Kullanım Durumun Haberleşme Diyagramı



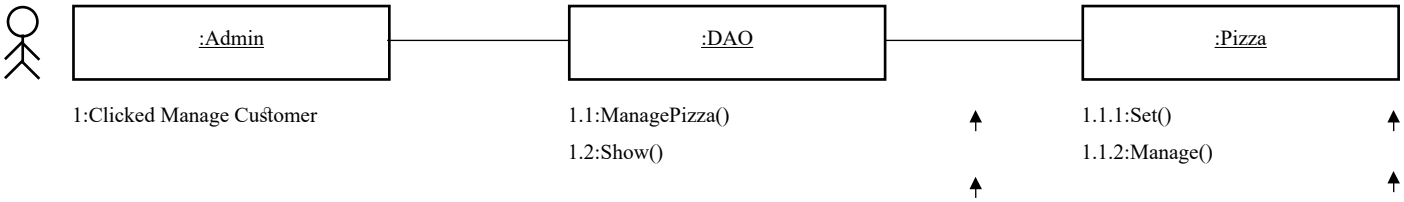
Şekil 4.2.2 Manage Order Kullanım Durumun Sekans Diyagramı

B. MANAGE PİZZA KULLANIM DURUMU

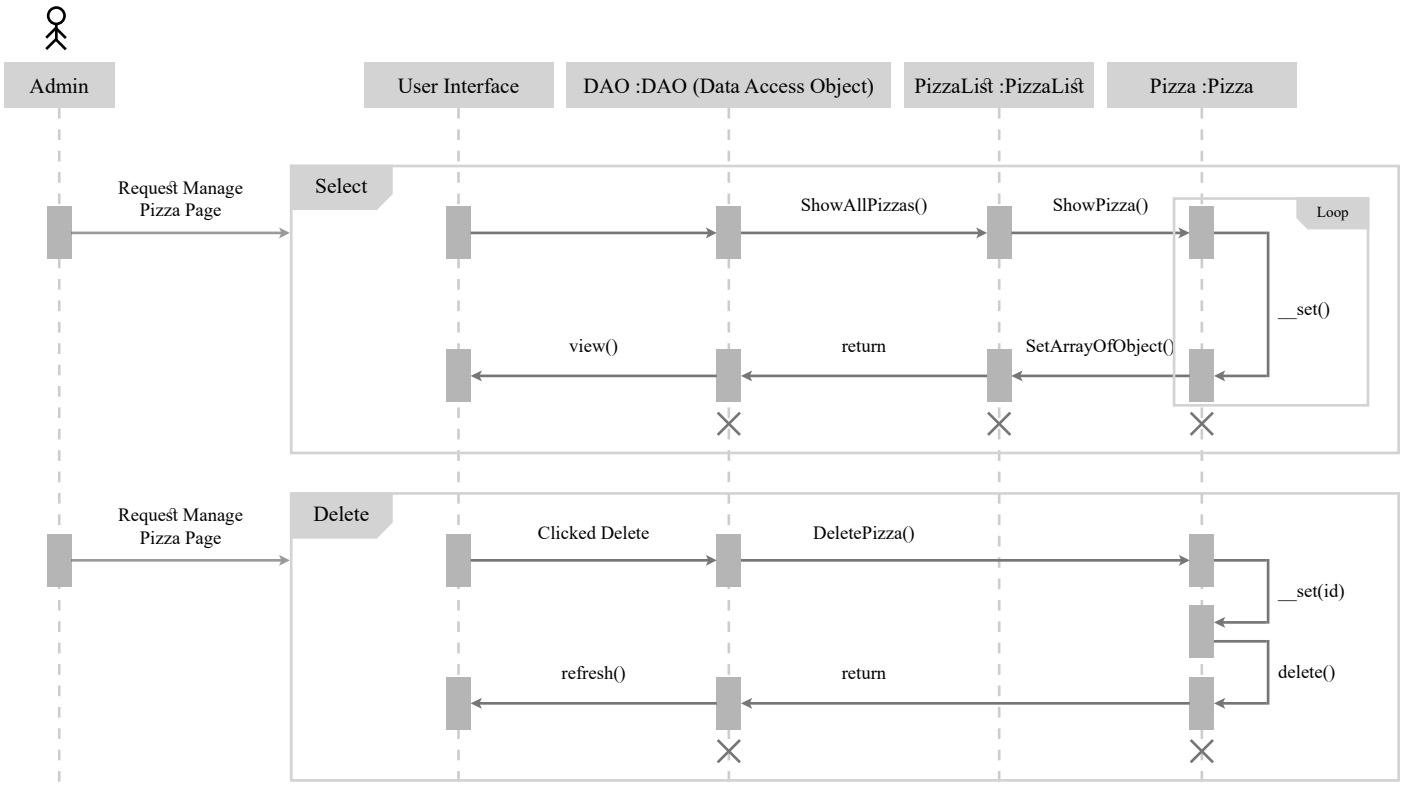
- Kullanım Durum Adı: Manage Pizza
- Katılımcı Aktörler: Admin
- İş Akışı:

1. Login kullanım durumu
    2. Login kullanım durumu başarı ile bittikten sonra admine hemen görünecek sayfa bütün siparişler içerir.
3. Eğer üst sağdaki Pizza button'a tıklanır ise hemen görünecek sayfa bütün Pizza nesneler içerir ve bu Pizzalar üzerinde delete işlemi yapabilir.
4. Eğer delete button'a tıklanır ise bir Pizza nesnesi oluşturup sadece onun ID içerir
    5. Ondan sonra Pizza ID içeren bir sorgu veritabanına sorgular, delete işlemi gerçekledigi zaman sayfa yeniden yüklenir

- Başlangıç Koşullu: Pizza üzerinde işlemleri yapmayı istediği zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: -



Şekil 4.2.3 Manage Pizza Kullanım Durumun Haberleşme Diyagramı



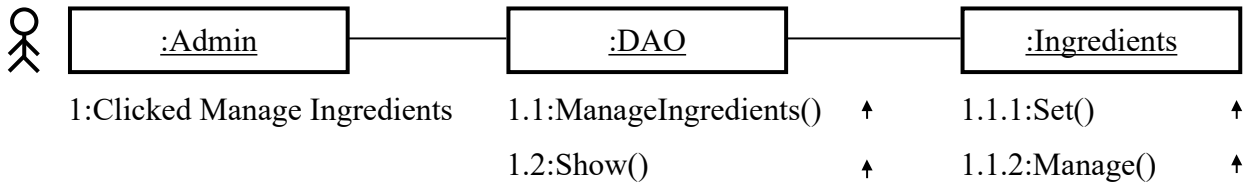
Şekil 4.2.4 Manage Pizza Kullanım Durumun Sekans Diyagramı

C. MANAGE İNGREDİENTS KULLANIM DURUMU

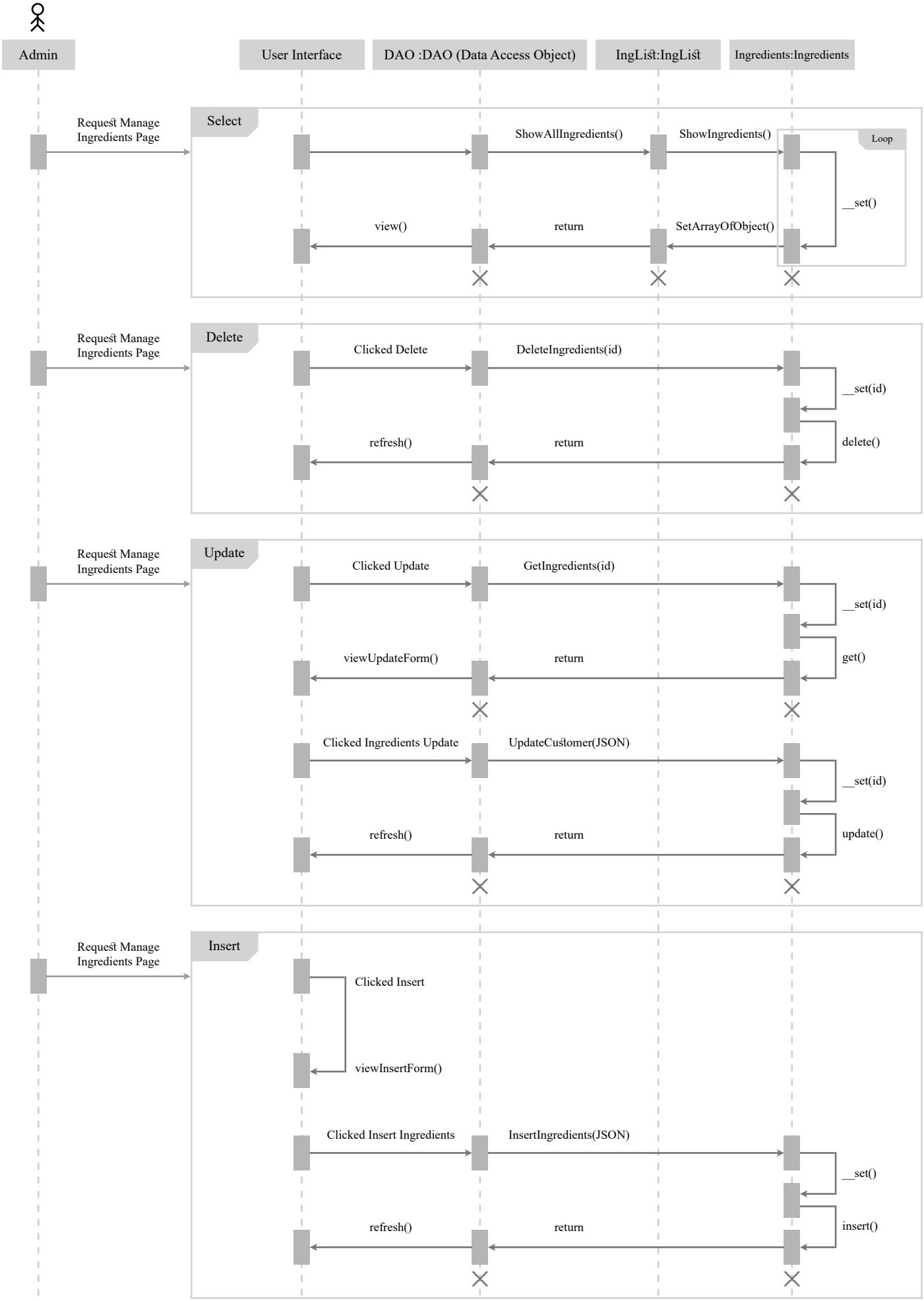
- Kullanım Durum Adı: Manage İngredients
- Katılımcı Aktörler: Admin
- İş Akışı:

1. Login kullanım durumu
    2. Login kullanım durumu başarı ile bittikten sonra admine hemen görünecek sayfa bütün siparişler içerir.
3. Eğer üst sağdaki ingredients button'a tıklanır ise hemen görünecek sayfa bütün ingredients nesneler içerir ve bu ingredientsler üzerinde insert / delete / update işlemleri yapabilir.
4. Eğer delete button'a tıklanır ise bir ingredients nesnesi oluşturup sadece onun ID içerir
    5. Ondan sonra ingredients ID içeren bir sorgu veritabanına sorgular, delete işlemi gerçekleştiği zaman sayfa yeniden yüklenir
6. Eğer Update button'a tıklanır ise ingredients nesnesi oluşturup sadece onun ID içerir
    7. Bu ID sayesinde bütün ingredients bilgileri getirir ve ingredients update arayüzüne ingredients bilgileri yükler
8. Gerekli değişiklik yaptıktan sonra update button'a tıklanır ise bütün girilen bilgileri alır ve veritabanındaki kaydı güncellenip sayfa yeniden yüklenir
    9. Eğer insert button'a tıklanır ise insert formu karşımıza çıkar
10. Gerekli bilgiler girip insert button'a tıklanır ise girilen ingredients'in bilgiler alınır ve veritabanındaki kaydı oluşturur ve sayfayı yeniden yüklenir

- Başlangıç Koşullu: ingredients üzerinde işlemleri yapmayı istediği zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: -



Şekil 4.2.5 Manage İngredients Kullanım Durumun Haberleşme Diyagramı



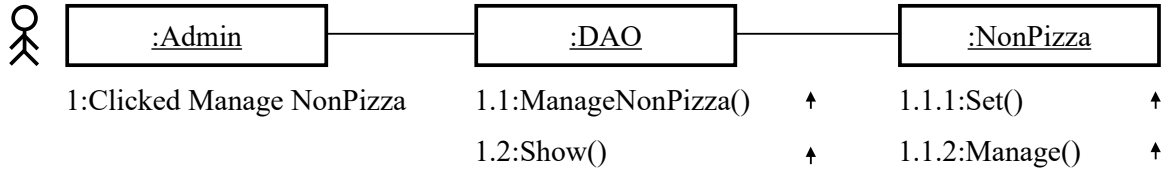
Şekil 4.2.6 Manage İngredients Kullanım Durumun Sekans Diyagramı

D. MANAGE NON-PİZZA KULLANIM DURUMU

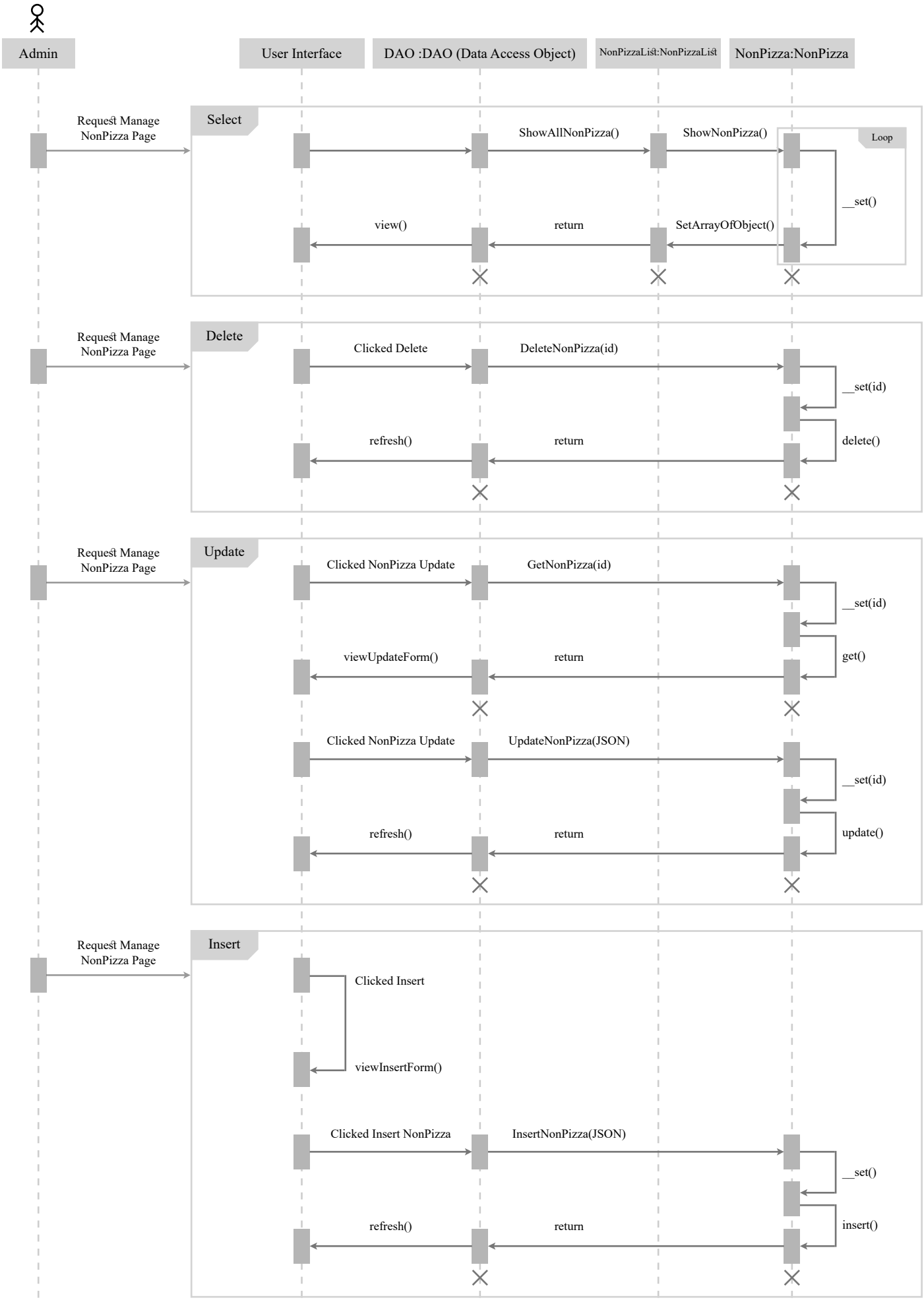
- Kullanım Durum Adı: Manage Non-Pizza
- Katılımcı Aktörler: Admin
- İş Akışı:

1. Login kullanım durumu
    2. Login kullanım durumu başarı ile bittikten sonra admine hemen görünecek sayfa bütün siparişler içerir.
3. Eğer üst sağdaki non-pizza button'a tıklanır ise hemen görünecek sayfa bütün non-pizza nesneler içerir ve bu non-pizzalar üzerinde *insert / delete / update* işlemleri yapabilir.
4. Eğer delete button'a tıklanır ise bir non-pizza nesnesi oluşturup sadece onun ID içerir
    5. Ondan sonra non-pizza ID içeren bir sorgu database sorgular, delete işlemi gerçekleştiği zaman sayfa yeniden yüklenir
6. Eğer Update button'a tıklanır ise non-pizza nesnesi oluşturup sadece onun ID içerir
    7. Bu ID sayesinde bütün non-pizza bilgileri getirir ve non-pizza update arayüzüne ingredients bilgileri yükler
8. Gerekli değişiklik yaptıktan sonra update button'a tıklanır ise bütün girilen bilgileri alır ve veritabanındaki kaydı güncellenip sayfa yeniden yüklenir
    9. Eğer insert button'a tıklanır ise insert formu karşımıza çıkar
10. Gerekli bilgiler girip insert button'a tıklanır ise girilen non-pizza bilgiler alınır ve veritabanındaki kaydı oluşturur ve sayfayı yeniden yüklenir

- Başlangıç Koşullu: Non-pizza üzerinde işlemleri yapmayı istediği zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: -



Şekil 4.2.7 Manage Non-Pizza Kullanım Durumun Haberleşme Diyagramı



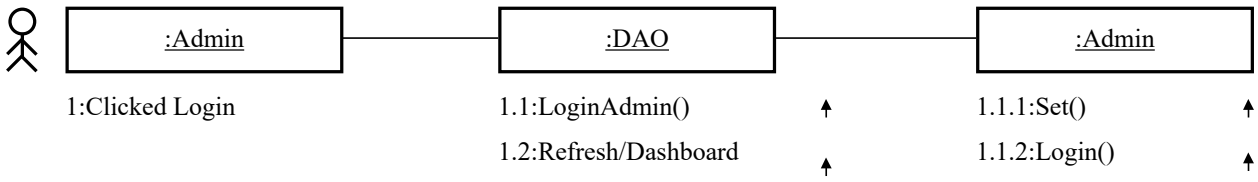
Şekil 4.2.8 Manage Non-Pizza Kullanım Durumun Sekans Diyagramı

E. LOGIN KULLANIM DURUMU

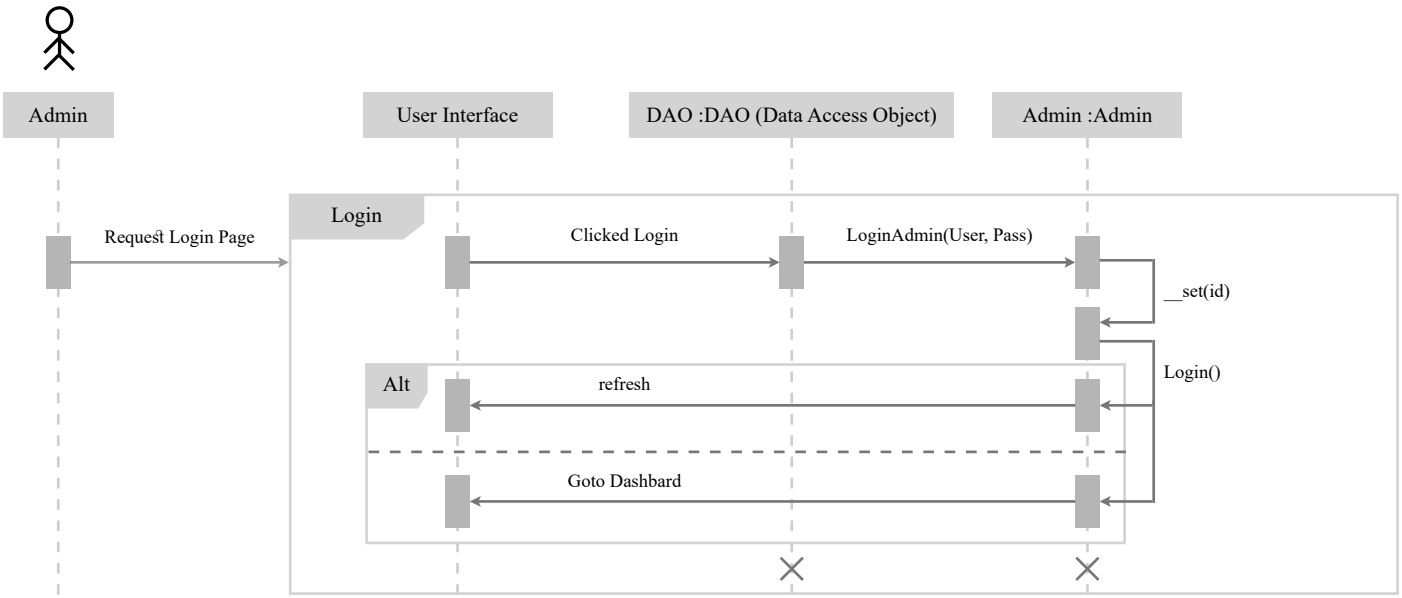
- Kullanım Durum Adı: Login
- Katılımcı Aktörler: Admin
- İş Akışı:

1. Admin username ve password ile ilgili kısımları doldurur
2. Gelen bilgiler veritabanına bir sorgu gönderilir
3. Eğer sorgudan gelen değer true ise admin dashboard sayfasina yollar aksi takdirde (false) hata mesajı verir ve yine login sayfasina yollar

- Başlangıç Koşullu: Admin dashboard sayfasina giriş yapmak istediği zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: -



Şekil 4.2.9 Login Kullanım Durumun Haberleşme Diyagramı



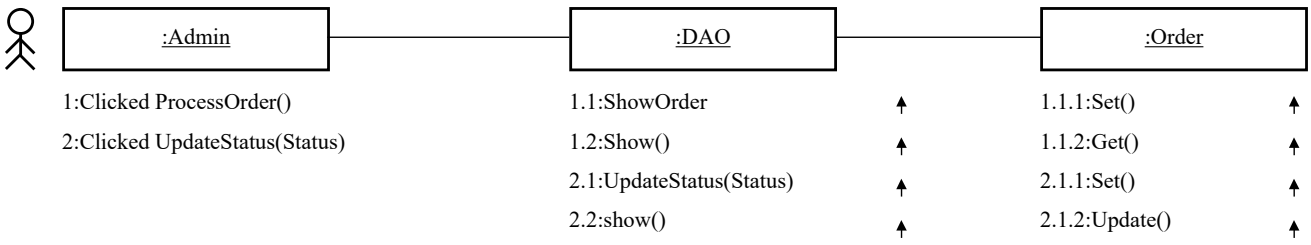
Şekil 4.2.10 Login Kullanım Durumun Sekans Diyagramı

F. PROCESS ORDER KULLANIM DURUMU

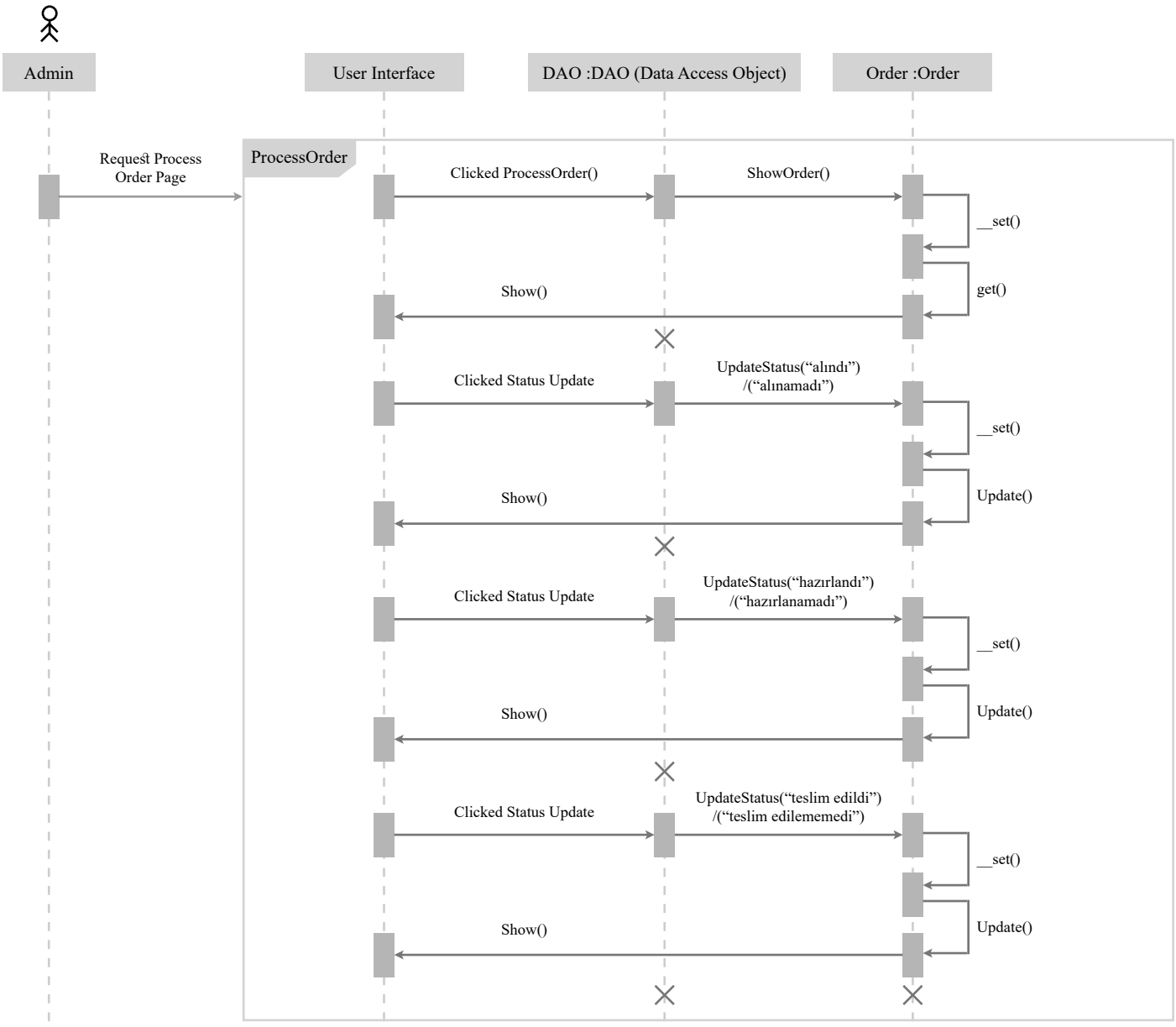
- Kullanım Durum Adı: Process Order
- Katılımcı Aktörler: Admin
- İş Akışı:

1. Müşteri Sipariş onaylayip mutfaga gelir
2. Eklediği malzemeler Çıkarabilir
3. Öncelikle adrese bakar eğer müşterinin adresi cok uzaksa siparis alinamadi işaretler ve musteri haber vermek icin ona arar aksi takdirde sipariş alındı olarak işaretler
4. Sonra pizza listesine bakip yapilacak malzeme yeterli değilse siparis hazirlanamadi olarak isaretler ve musteri haber vermek icin ona arar aksi takdirde sipariş hazir oldugu zaman sipariş hazırlandı olarak işaretler
5. Ondan sonra müşterinin adresine gider eğer teslimat basarili bir sekilde tamamlanmış ise sipariş teslim edildi olarak işaretler aksi takdirde sipariş teslim edilemedi olarak işaretler

- Başlangıç Koşullu: Müşteri kendi siparişi onayladiği zaman ve siparış detayları mutfağa geldiği zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: -



Şekil 4.2.11 Process Order Kullanım Durumun Haberleşme Diyagramı



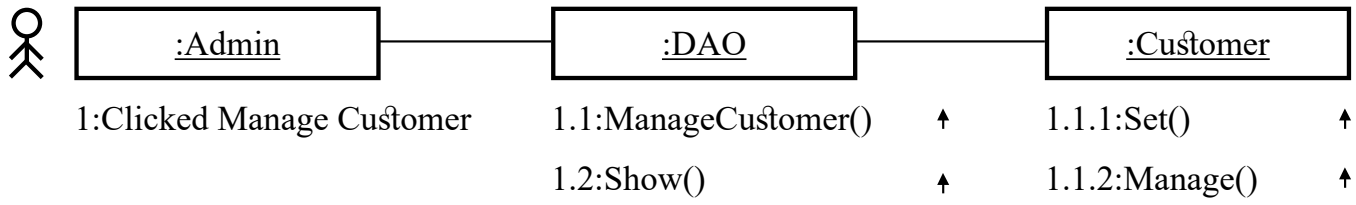
Şekil 4.2.12 Process Order Kullanım Durumun Sekans Diyagramı

G. MANAGE CUSTOMER KULLANIM DURUMU

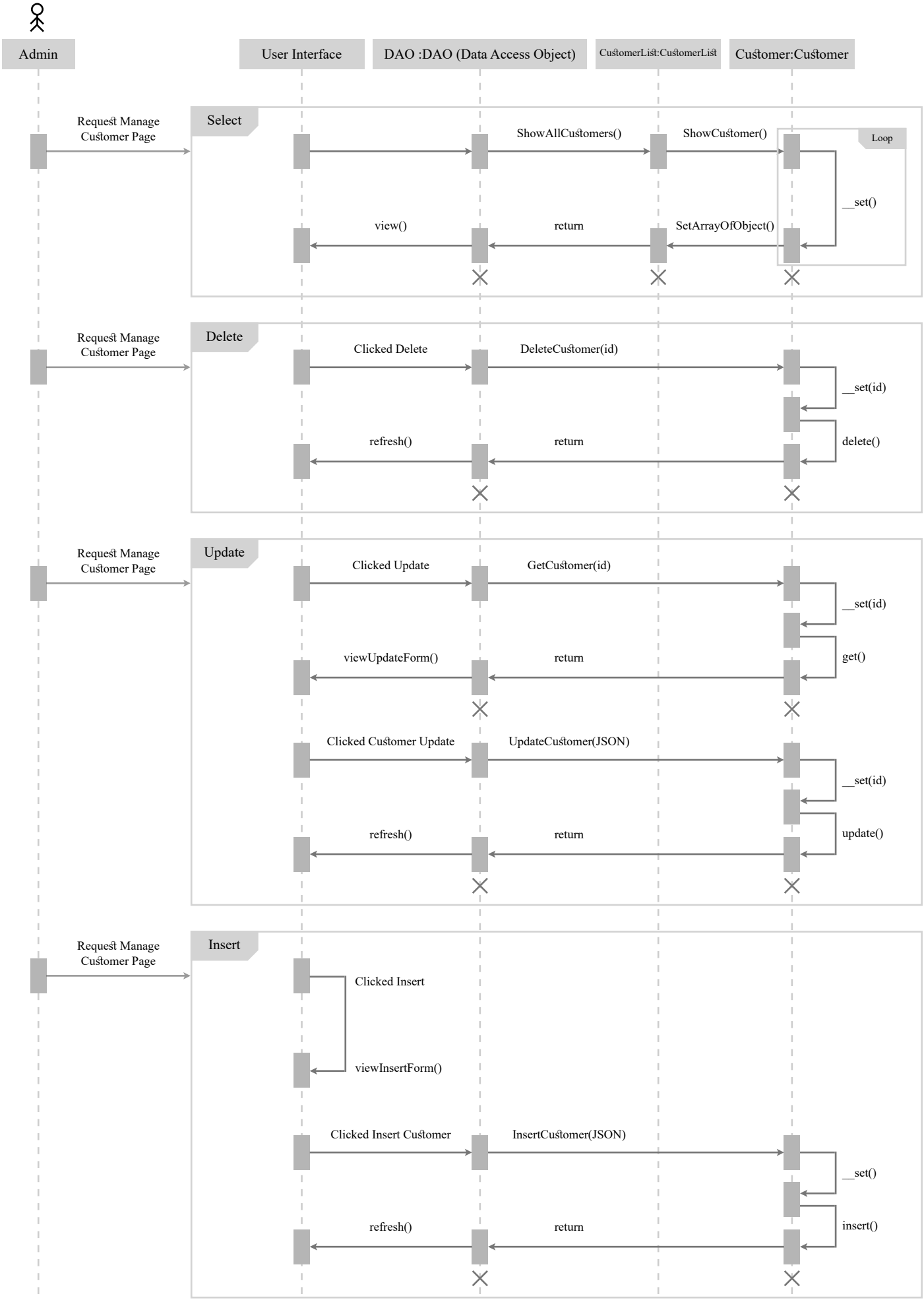
- Kullanım Durum Adı: Manage Customer
- Katılımcı Aktörler: Admin
- İş Akışı:

1. Login kullanım durumu
    2. Login kullanım durumu başarı ile bittikten sonra admine hemen görünecek sayfa bütün siparişler içerir.
3. Eğer üst sağdaki customer button'a tıklanır ise hemen görünecek sayfa bütün customer nesneler içerir ve bu customerler üzerinde insert / delete / update işlemleri yapabilir.
4. Eğer delete button'a tıklanır ise bir customer nesnesi oluşturup sadece onun ID içerir
    5. Ondan sonra customer ID içeren bir sorgu veritabanına sorgular, delete işlemi gerçekleştiği zaman sayfa yeniden yüklenir
6. Eğer Update button'a tıklanır ise customer nesnesi oluşturup sadece onun ID içerir :)
    7. Bu ID sayesinde bütün customer bilgileri getirir ve customer update arayüzüne customer bilgileri yükler
8. Gerekli değişiklik yaptıktan sonra update button'a tıklanır ise bütün girilen bilgileri alır ve veritabanındaki kaydı güncellenip sayfa yeniden yüklenir
    9. Eğer insert button'a tıklanır ise insert formu karşımıza çıkar
10. Gerekli bilgiler girip insert button'a tıklanır ise girilen customer'in bilgiler alınır ve veritabanındaki kaydı oluşturur ve sayfayı yeniden yüklenir

- Başlangıç Koşullu: Customer üzerinde işlemleri yapmayı istediği zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: -



Şekil 4.2.13 Manage Customer Kullanım Durumun Haberleşme Diyagramı



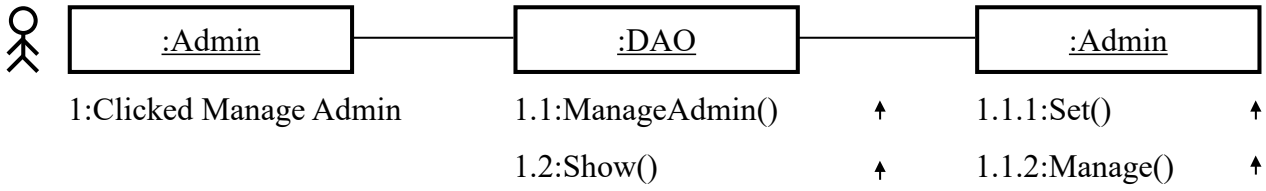
Şekil 4.2.14 Manage Customer Kullanım Durumun Sekans Diyagramı

H. MANAGE ADMİN KULLANIM DURUMU

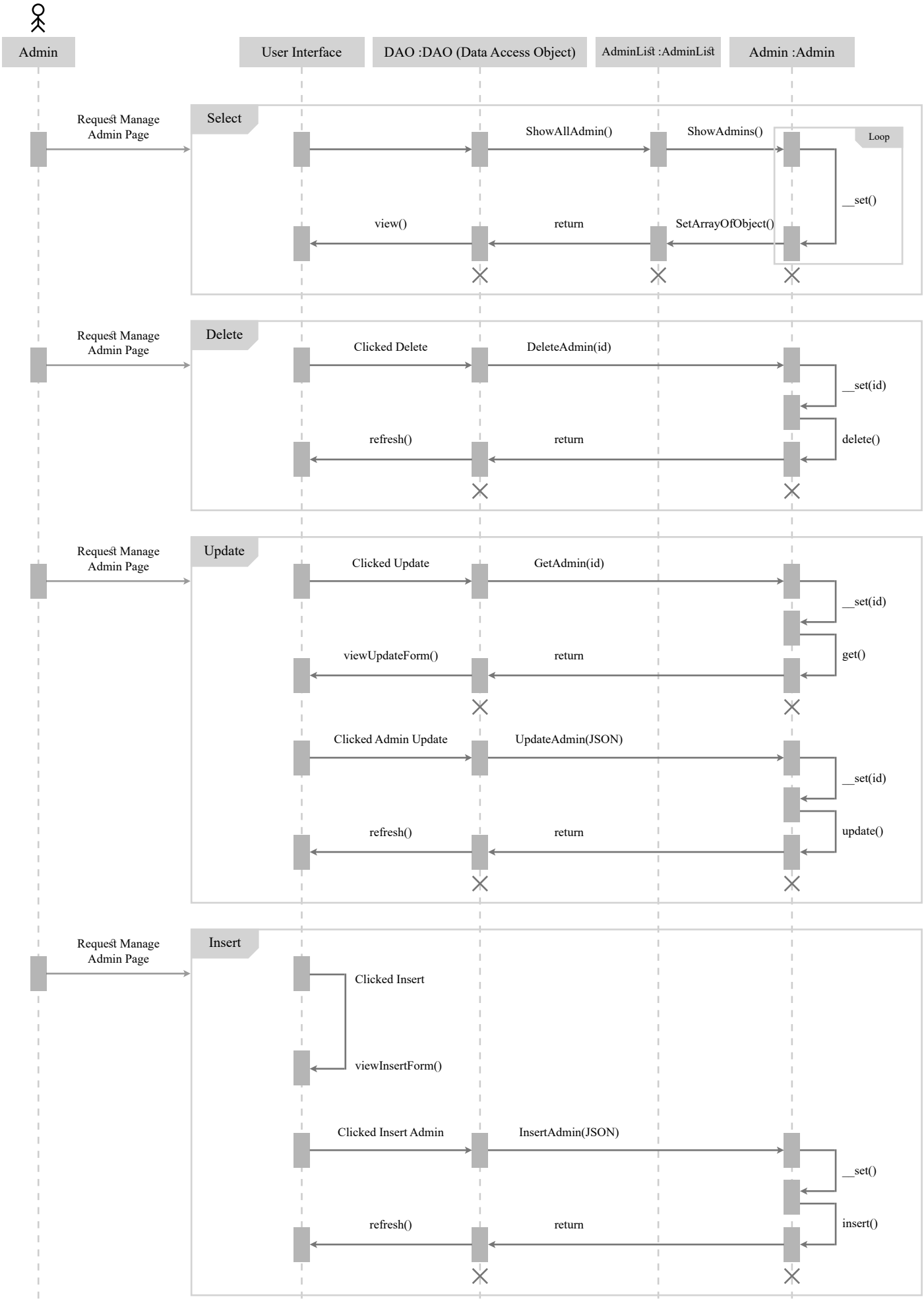
- Kullanım Durum Adı: Manage Admin
- Katılımcı Aktörler: Admin
- İş Akışı:

1. Login kullanım durumu
    2. Login kullanım durumu başarı ile bittikten sonra admine hemen görünecek sayfa bütün siparişler içerir.
3. Eğer üst sağdaki Admin button'a tıklanır ise hemen görünecek sayfa bütün Admin nesneler içerir ve bu Adminler üzerinde insert / delete / update işlemleri yapabilir.
4. Eğer delete button'a tıklanır ise bir Admin nesnesi oluşturup sadece onun ID içerir
    5. Ondan sonra Admin ID içeren bir sorgu veritabanına sorgular, delete işlemi gerçekleştiği zaman sayfa yeniden yüklenir
6. Eğer Update button'a tıklanır ise Admin nesnesi oluşturup sadece onun ID içerir
    7. Bu ID sayesinde bütün Admin bilgileri getirir ve Admin update arayüzüne Admin bilgileri yükler
8. Gerekli değişiklik yaptıktan sonra update button'a tıklanır ise bütün girilen bilgileri alır ve veritabanındaki kaydı güncellenip sayfa yeniden yüklenir
    9. Eğer insert button'a tıklanır ise insert formu karşımıza çıkar
10. Gerekli bilgiler girip insert button'a tıklanır ise girilen Admin'in bilgiler alınır ve veritabanındaki kaydı oluşturur ve sayfayı yeniden yüklenir

- Başlangıç Koşullu: Admin üzerinde işlemleri yapmayı istediği zaman
- Bitiş Koşullu: bahsedilen işlemlerden birini gerçekleştiği zaman
- Kalite Gereksinimler: -



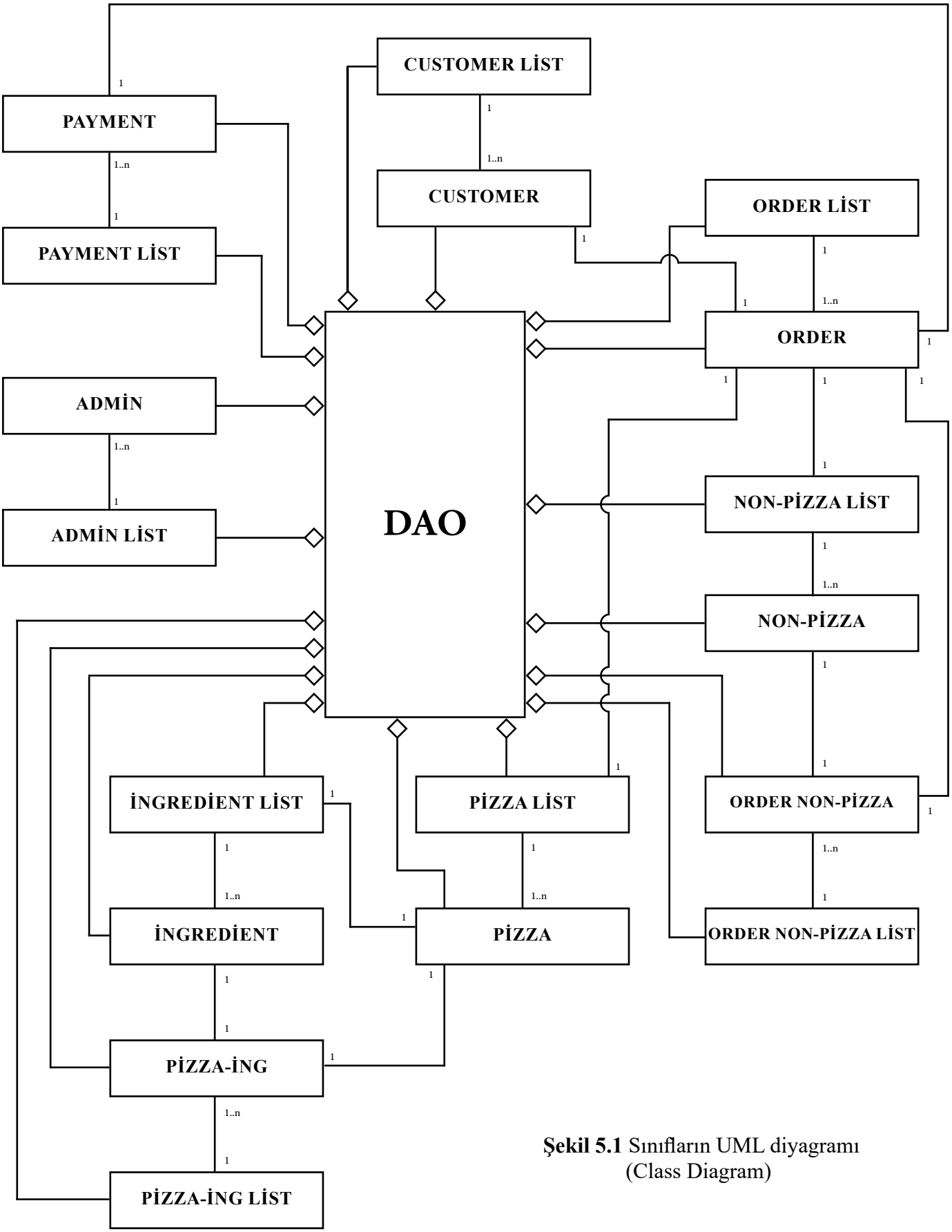
Şekil 4.2.15 Manage Admin Kullanım Durumun Haberleşme Diyagramı



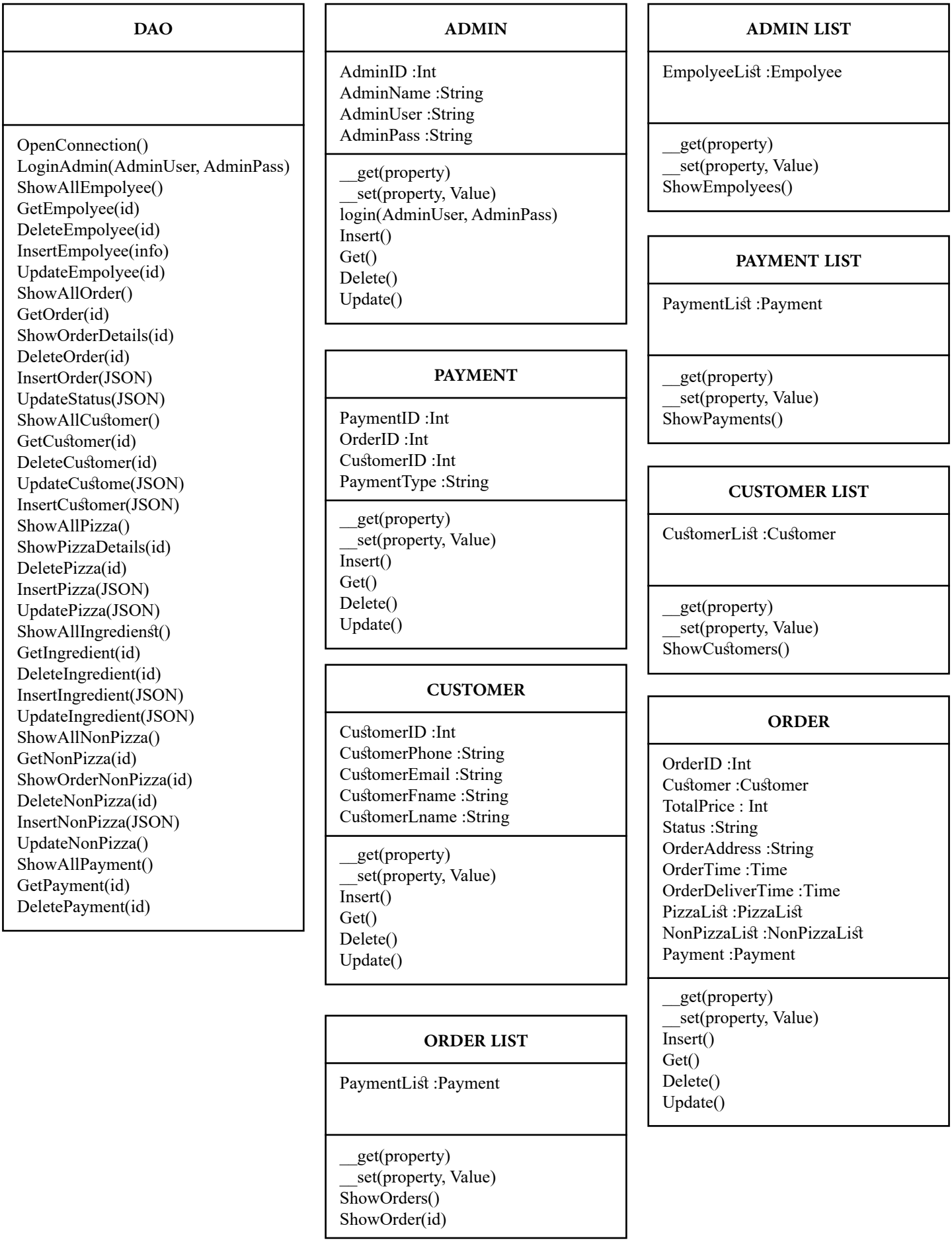
Şekil 4.2.16 Manage Admin Kullanım Durumun Sekans Diyagramı

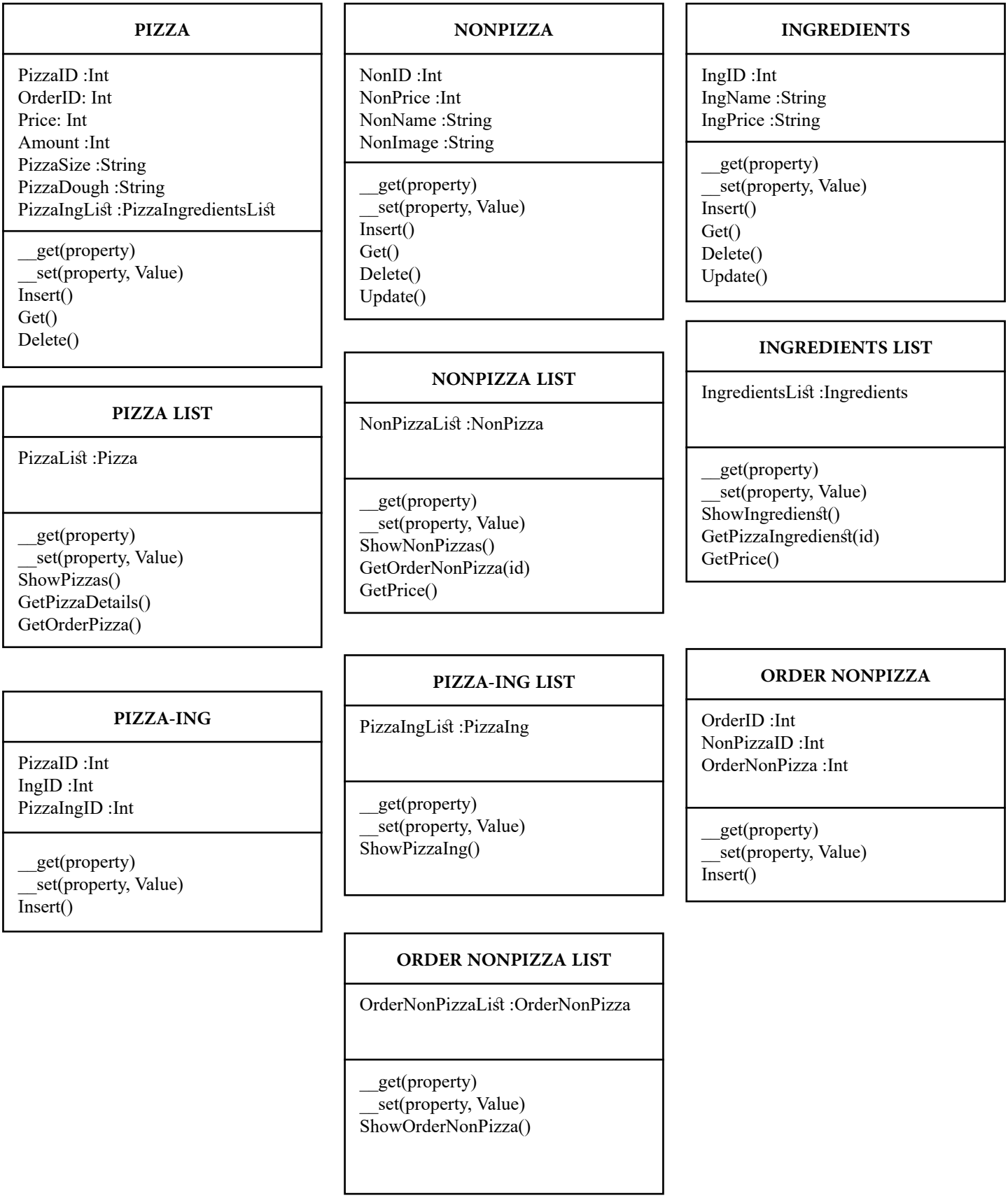
5. SINIFLARIN DİYAGRAMI (CLASSES' UML DİAGRAM)

Object Design faaliyetinden çıkan İş Çıktısı aşağıdaki UML sınıf diyagramı olarak üretilmiştir:

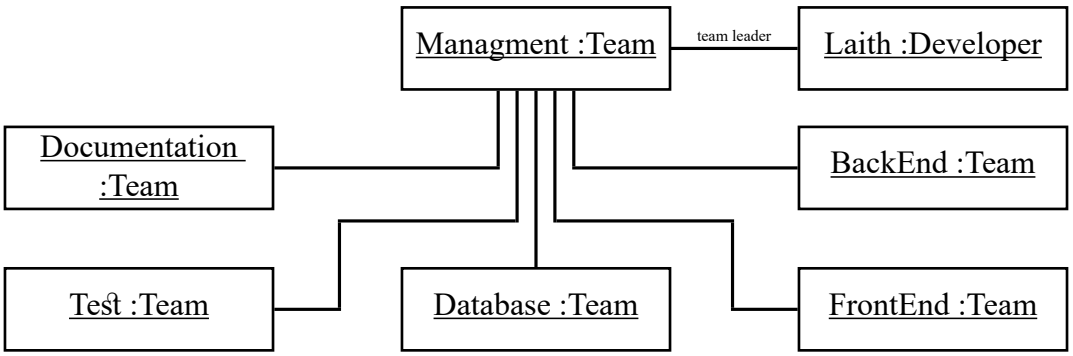


Şekil 5.1 Sınıfların UML diyagramı (Class Diagram)

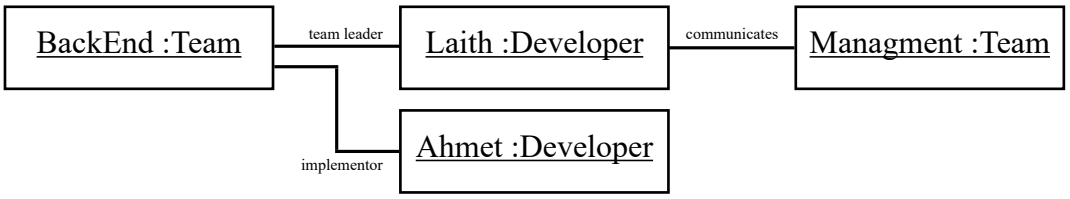

DAO
OpenConnection()
LoginAdmin(AdminUser, AdminPass)
ShowAllEmpolyee()
GetEmpolyee(id)
DeleteEmpolyee(id)
InsertEmpolyee(info)
UpdateEmpolyee(id)
ShowAllOrder()
GetOrder(id)
ShowOrderDetails(id)
DeleteOrder(id)
InsertOrder(JSON)
UpdateStatus(JSON)
ShowAllCustomer()
GetCustomer(id)
DeleteCustomer(id)
UpdateCustome(JSON)
InsertCustomer(JSON)
ShowAllPizza()
ShowPizzaDetails(id)
DeletePizza(id)
InsertPizza(JSON)
UpdatePizza(JSON)
ShowAllIngredienst()
GetIngredient(id)
DeleteIngredient(id)
InsertIngredient(JSON)
UpdateIngredient(JSON)
ShowAllNonPizza()
GetNonPizza(id)
ShowOrderNonPizza(id)
DeleteNonPizza(id)
InsertNonPizza(JSON)
UpdateNonPizza()
ShowAllPayment()
GetPayment(id)
DeletePayment(id)
ADMIN
AdminID :Int
AdminName :String
AdminUser :String
AdminPass :String
__get(property)
__set(property, Value)
login(AdminUser, AdminPass)
Insert()
Get()
Delete()
Update()
PAYMENT
PaymentID :Int
OrderID :Int
CustomerID :Int
PaymentType :String
__get(property)
__set(property, Value)
Insert()
Get()
Delete()
Update()
CUSTOMER
CustomerID :Int
CustomerPhone :String
CustomerEmail :String
CustomerFname :String
CustomerLname :String
__get(property)
__set(property, Value)
Insert()
Get()
Delete()
Update()
ORDER LIST
PaymentList :Payment
__get(property)
__set(property, Value)
ShowOrders()
ShowOrder(id)
ADMIN LIST
EmpolyeeList :Empolyee
__get(property)
__set(property, Value)
ShowEmpolyees()
PAYMENT LIST
PaymentList :Payment
__get(property)
__set(property, Value)
ShowPayments()
CUSTOMER LIST
CustomerList :Customer
__get(property)
__set(property, Value)
ShowCustomers()
ORDER
OrderID :Int
Customer :Customer
TotalPrice : Int
Status :String
OrderAddress :String
OrderTime :Time
OrderDeliverTime :Time
PizzaList :PizzaList
NonPizzaList :NonPizzaList
Payment :Payment
__get(property)
__set(property, Value)
Insert()
Get()
Delete()
Update()


PIZZA
PizzaID :Int
OrderID: Int
Price: Int
Amount :Int
PizzaSize :String
PizzaDough :String
PizzaIngList :PizzaIngredientsList
__get(property)
__set(property, Value)
Insert()
Get()
Delete()
PIZZA LIST
PizzaList :Pizza
__get(property)
__set(property, Value)
ShowPizzas()
GetPizzaDetails()
GetOrderPizza()
PIZZA-ING
PizzaID :Int
IngID :Int
PizzaIngID :Int
__get(property)
__set(property, Value)
Insert()
NONPIZZA
NonID :Int
NonPrice :Int
NonName :String
NonImage :String
__get(property)
__set(property, Value)
Insert()
Get()
Delete()
Update()
NONPIZZA LIST
NonPizzaList :NonPizza
__get(property)
__set(property, Value)
ShowNonPizzas()
GetOrderNonPizza(id)
GetPrice()
PIZZA-ING LIST
PizzaIngList :PizzaIng
__get(property)
__set(property, Value)
ShowPizzaIng()
ORDER NONPIZZA LIST
OrderNonPizzaList :OrderNonPizza
__get(property)
__set(property, Value)
ShowOrderNonPizza()
INGREDIENTS
IngID :Int
IngName :String
IngPrice :String
__get(property)
__set(property, Value)
Insert()
Get()
Delete()
Update()
INGREDIENTS LIST
IngredientsList :Ingredients
__get(property)
__set(property, Value)
ShowIngredienst()
GetPizzaIngredienst(id)
GetPrice()
ORDER NONPIZZA
OrderID :Int
NonPizzaID :Int
OrderNonPizza :Int
__get(property)
__set(property, Value)
Insert()

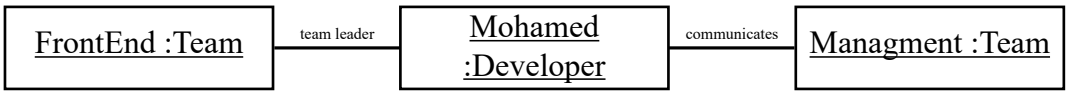
6. TAKIMLAR (TEAMS)



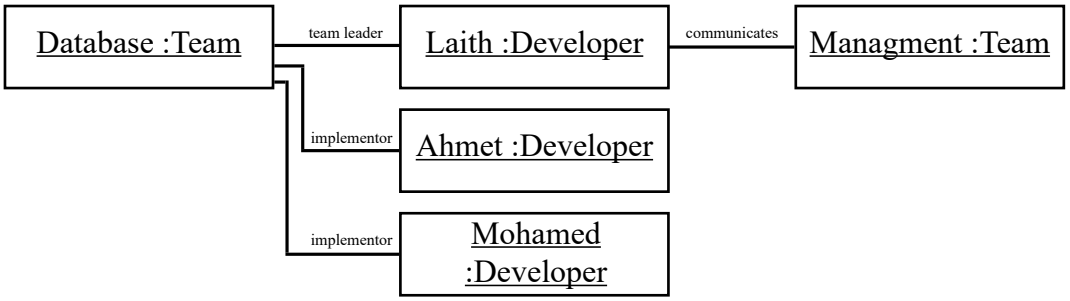
Şekil 6.1 Managment Team



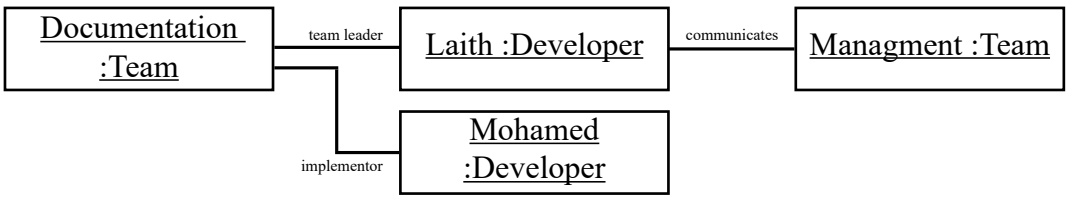
Şekil 6.2 BackEnd Team



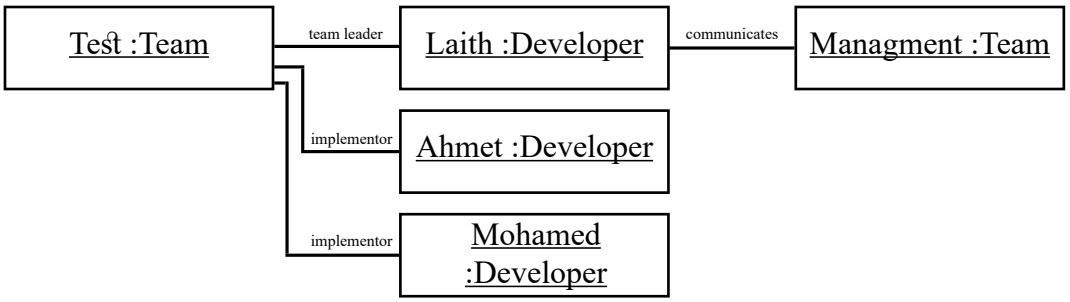
Şekil 6.3 FrontEnd Team



Şekil 6.4 Database Team



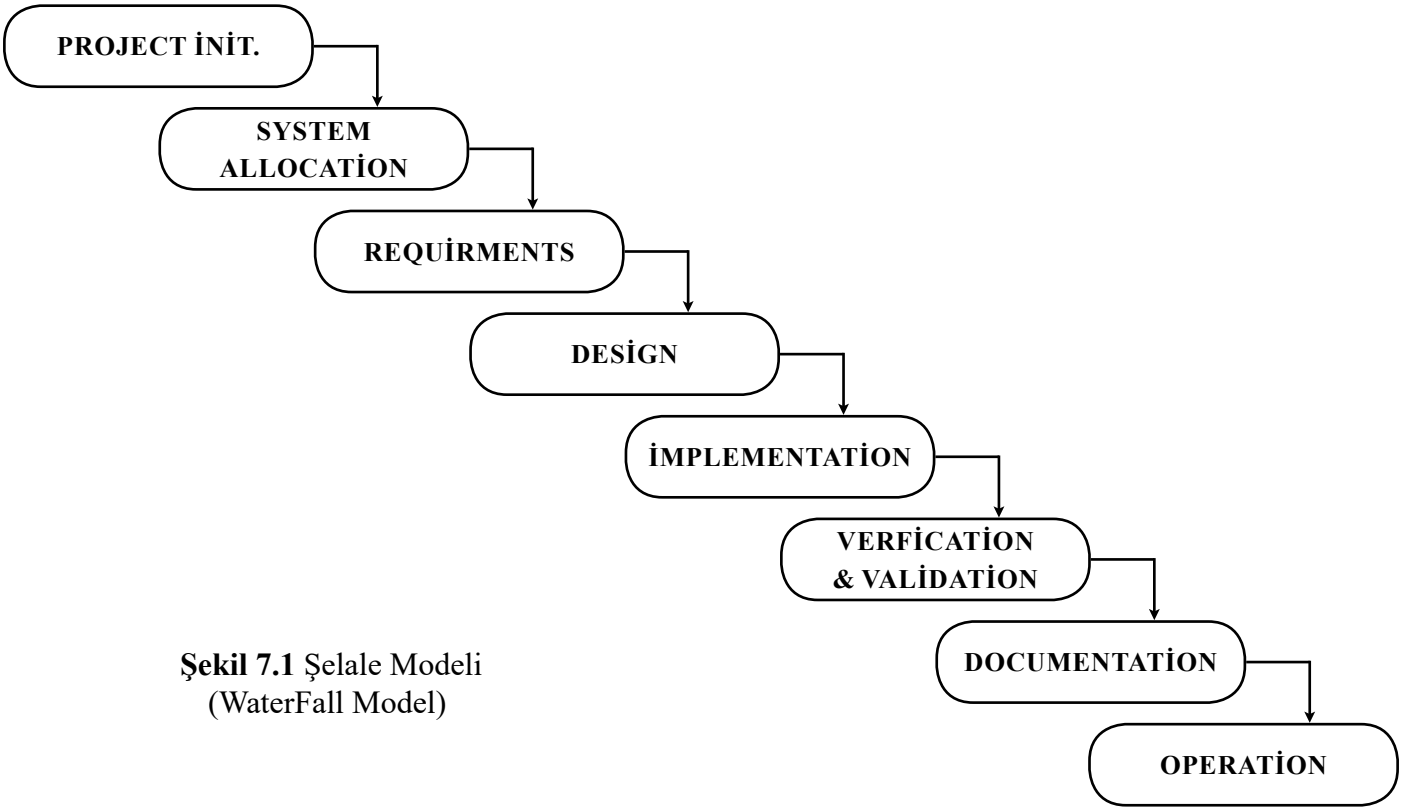
Şekil 6.5 Documentation Team



Şekil 6.6 Test Team

7. ŞELALE MODELİ (WATERFALL MODEL)

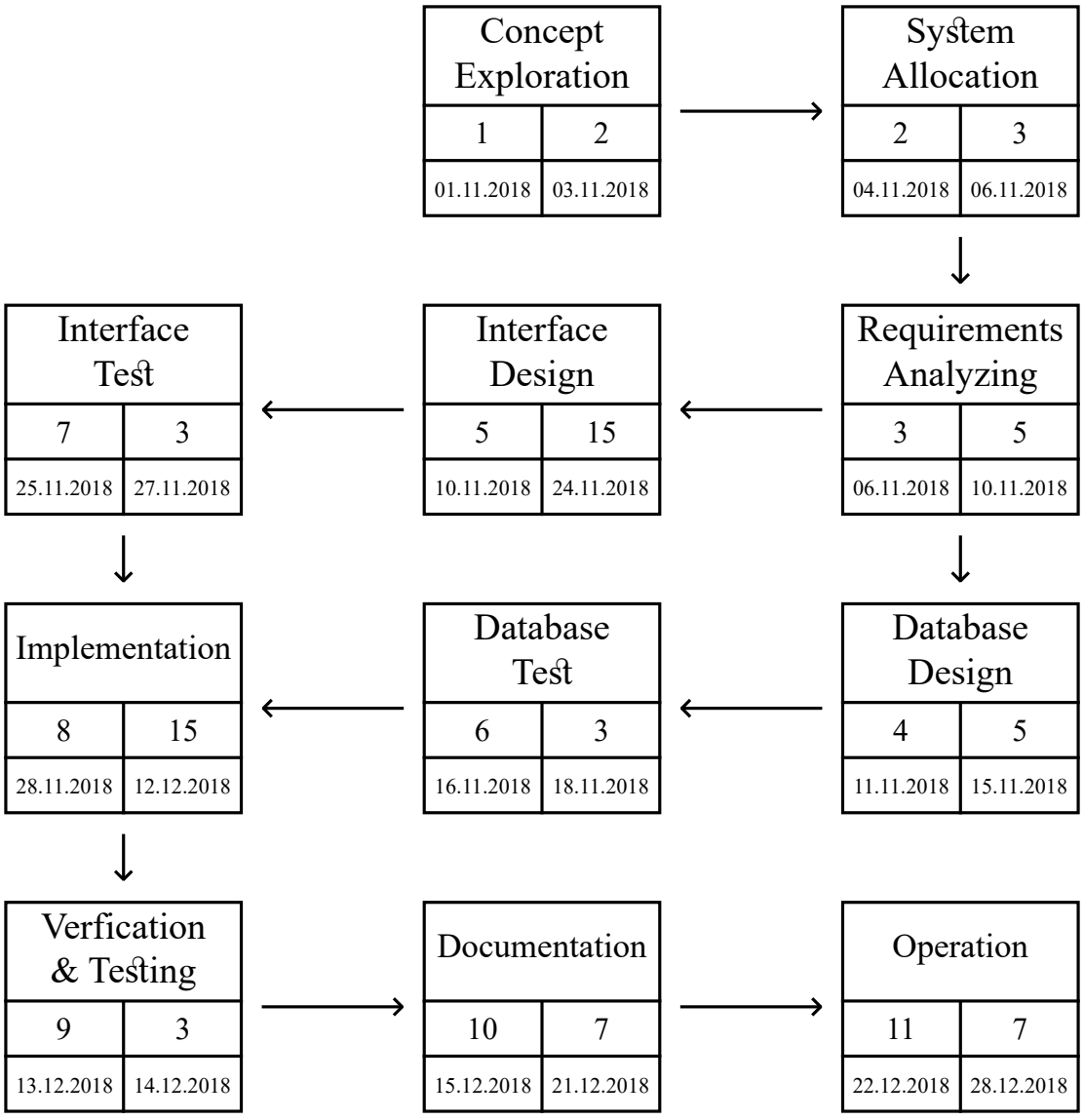
Şelale modeli, yazılım projelerinde uygulanan faaliyetlerin ardışık safhalar halinde icra edildiği, yazılım mühendisliğinin en eski ve en temel modelidir. Günümüzde hükümetler ve büyük şirketler tarafından her türlü projenin yönetim standardı olarak kabul görmektedir. Projemizde WATERFALL Modeli Kullanılıdı



Şekil 7.1 Şelale Modeli
(WaterFall Model)

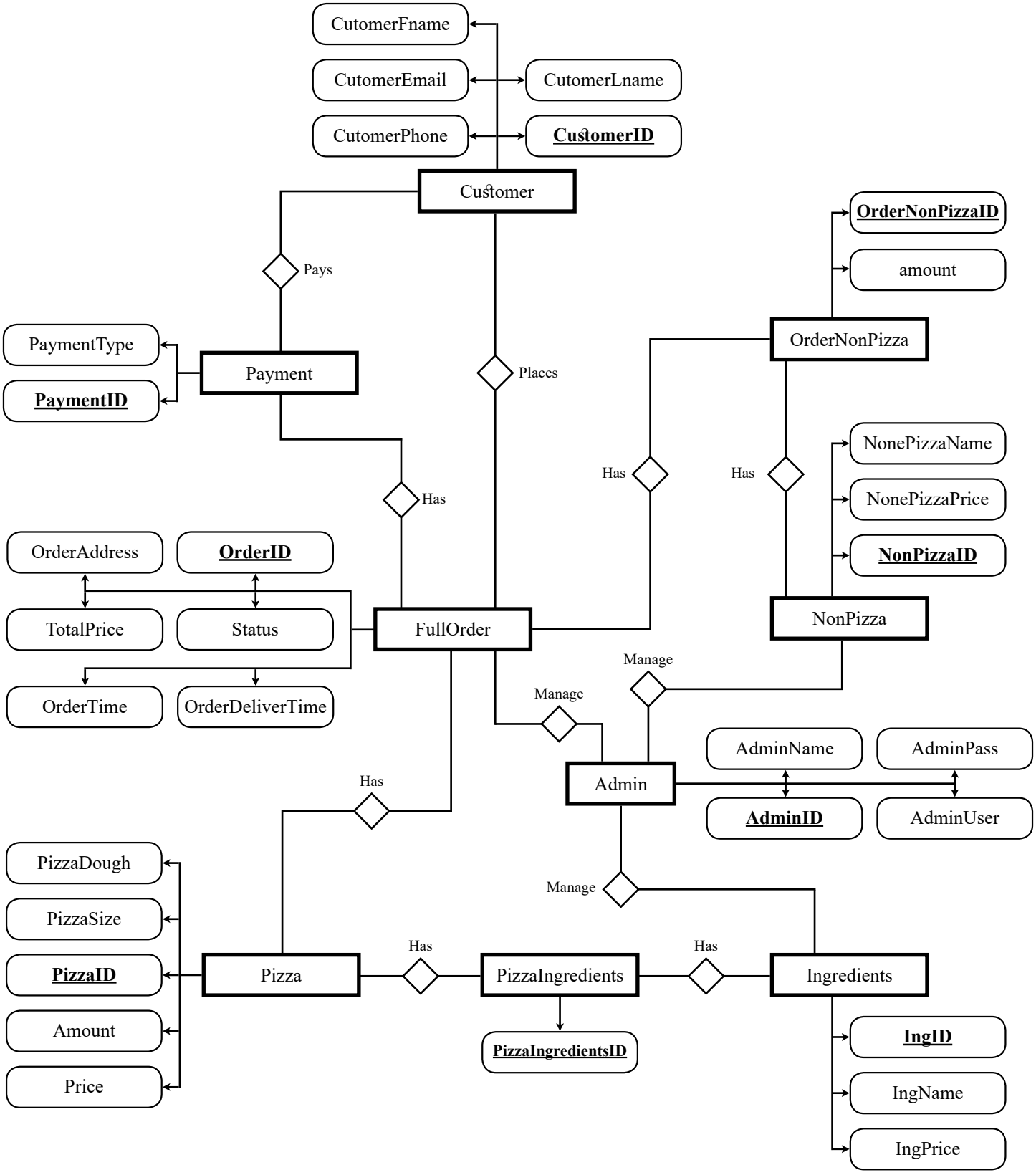
8. PROJE PLANI

Proje planı göstermek için PERT şeması kullandık. Projemizin planı aşağıdaki gibi:



Şekil 8.1 Proje Planı / Programı

9. VERİTABANIN ERD DİYAGRAMI



Şekil 9.1 ERD Diyagramı