CNC MAKİNA ASR CNC MAKİNA

ASR STOK
PROJE RAPORU

GRUP 6

Giriş:

1.1. Proje Ekibi

1030510316 Aysun Görpe
1030510320 Ayşe Nur Gürbulak
1030510503 Ayşem Sude Karagöz
1030510338 Duygu Gözde Kayabaşı
1030510529 Gizem Nur Çelik
1030510317 Hilal Toklu
1030510331 Melisa Demir
1030510278 Şerife Nefise Otlaklı
1030510267 Şule Otlaklı
1030519554 Yaren Koç

1.2. Proje Başlığı:

AsrStok.

1.3. Projeye Genel Bakış:

AsrStok uygulaması, bir işletmenin stok yönetimi süreçlerini etkili bir şekilde takip etmesine ve yönetmesine, gider hesaplamalarını daha kolay yapabilmesine yardımcı olan bir uygulamadır.

1.4. Amaç ve Hedefler:

1.4.1. Stok seviyelerinin optimize edilmesi:

Kullanıcı veri tabanına ürünlerini ekleyebilir, ürünlerin fiyatını stok miktarını fatura numaralarını güncelleyebilir ve veri tabanından ürün silebilir. Kullanıcı istediği zaman ürünlerin stok seviyelerini, fiyatlarını, stok güncelleme tarihlerine göre fiyat değişim oranlarını görüntüleyebilir.

1.4.2. Eski fatura verilerinin veri tabanında tutulması:

Her ürünün fatura numarası stok güncelleme tarihine göre veri tabanına kaydedilir. Kullanıcı istediği zaman fatura numarasına göre eski verileri görüntüleyebilir. Stok her güncellendiğinde eski fatura numaraları veri tabanında saklanmaya devam edilir.

1.4.3. Firmanın toplam gider hesabının yapılması:

Firmanın aylık ürün giderleri, elektrik, su, vergiler ve personel giderleri gibi verileri kullanıcı tarafından uygulamaya girilir. Bu veriler tarihe göre veri tabanına kaydedilir ve firmanın giderleri otomatik olarak aylık ve yıllık bazda hesaplanarak veri tabanında tutulur. Kullanıcı istediği zaman bu verileri sorgulayabilir.

1. Kurulum Kılavuzu:

2.1. Sistem Gereksinimleri

İşletim Sistemi: AsrStok uygulaması Windows işletim sisteminde çalışabilen bir uygulamadır.

İşlemci (CPU): Uygulama orta düzeyde bir işlemciyle çalışabilir. Eş zamanlı olarak sadece iki adet kullanıcıya hizmet vereceği için çok yüksek performanslı bir işlemci gerekmemektedir.

Bellek (RAM): Yeterli bellek, uygulamanın hızlı ve verimli bir şekilde çalışabilmesi için önemlidir. Uygulamanın verimli bir şekilde çalışabilmesi için en az 8 GB RAM gereklidir.

Depolama Alanı: Uygulamanın yanı sıra kullanıcı verileri, stok hareket kayıtları ve diğer ilgili bilgiler için yeterli depolama alanı. Büyük veri setleri için geniş depolama kapasitesi gereklidir.

2.2. Kurulum Adımları

2.2.1. Sistem Gereksinimlerini Kontrol Etme:

Firma öncelikle uygulamayı kullanacağı bilgisayarda sistem gereksinimlerini karşılayıp karşılamadığını kontrol eder. Sistem gereksinimleri işletim sistemi versiyonu, RAM, depolama ve işlemcidir.

2.2.2. Yazılımın İndirilmesi:

İkinci adım olarak uygulamanın kullanılacağı bilgisayara uygulama paketi veya kaynak kodu indirilir.

2.2.3. Kurulum Dosyalarını Çıkartma:

Uygulamanın içerdiği sıkıştırılmış dosyalar çıkartılır.

2.2.4. Yazılımın Kurulması:

Uygulama ilgili kurulum komutları çalıştırılarak firmanın bilgisayarına kurulur.

2.2.5. Test:

Uygulamanın gerekli gereksinimleri karşılayacak şekilde çalışıp çalışmadığı, firma yetkilileriyle birlikte test edilir.

2.2.6. Son Kontroller:

Kurulumun tamamlanmasıyla ilgili son bir kontrol yapılır.

3. Kullanıcı Kılavuzu:

3.1.Temel Sınıflar:

- AsrApplication {}
- AsrController{}
- db_connector{}
- GiderHesap {}
- login{}
- StokControl{}

3.2.Temel fonksiyonlar:

```
@FXML
protected void urunSorgulamaButtonAction() throws SQLException {/*…*/}
```

StokControl sınıfındaki sorgula() fonksiyonunu kullanarak ürün sorgulamayı sağlayan buton action metodudur.

```
@FXML
protected void InsertButtonAction(){/*…*/}
```

StokControl sınıfındaki insert_product() fonksiyonunu kullanarak ürün eklemeyi sağlayan buton action metodudur.

```
@FXML
protected void deleteFromDatabaseButtonAction(){/*…*/}
```

StokControl sınıfındaki delete_product() fonksiyonunu kullanarak ürün silmeyi sağlayan buton action metodudur.

```
@FXML
protected void aylikGiderHesapButonAction() throws SQLException {/*…*/}
```

GiderHesap sınıfındaki toplam_gider fonksiyonunu kullanarak aylık gider miktarını labela yazdıran buton action metodudur.

```
@FXML
protected void stokGuncellemeButtonAction() { /*…*/}
```

StokControl sınıfındaki stok_guncelle() fonksiyonunu kullanarak mevcut stoğu güncelleyen buton action metodudur.

```
@FXML
protected void giderAl(){ /*…*/}
```

GiderHesap sınıfındaki calculateDifference() fonksiyonunu kullanarak ürün başı gider hesabını sağlayan buton action metodudur.

```
@FXML
protected void atolyeGiderGir() throws SQLException { /*…*/}
```

GiderHesap sınıfındaki atolyeGiderAl() fonksiyonunu kullanarak gider bilgisini ekleyen buton action metodudur.

```
@FXML
public void initialize() {//Combobox /*…*/}
```

ComboBox oluşturmak için kullanılmıştır.

```
@FXML
public void handleLoginButtonAction(){/*…*/ }
```

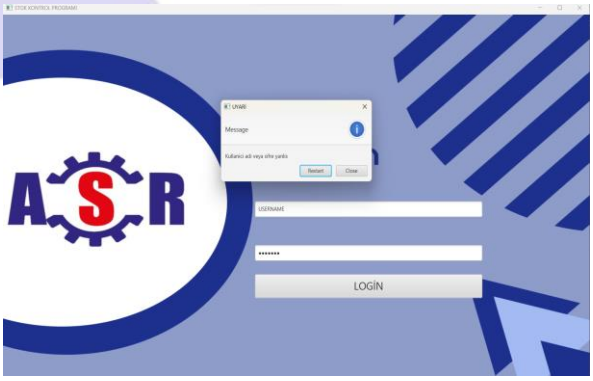
Login ekranında kullanıcıdan Username ve password alıp kontrol eden buton action metodudur.

```
@FXML
private void LogOutButtonAction() {/*…*/}
```

Her arayüzde çıkışı sağlayan logOut buton action metodudur.

3.3.Arayüzler:

Arayüz(1)



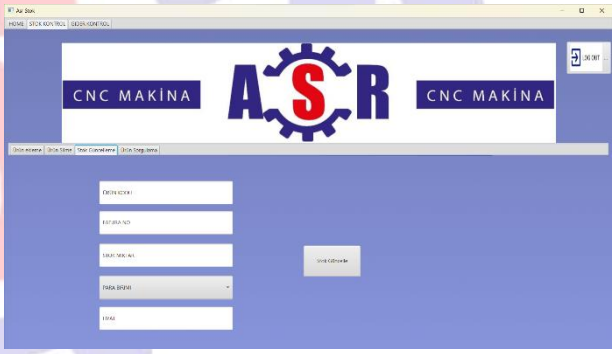
Arayüz(2)

Müşteri programı çalıştırdığında Arayüz(1) açılır ve kullanıcıdan giriş bilgileri alınır. Müşteri bilgilerini girip butona tıkladığında girilen bilgiler doğruysa Arayüz(3) açılır ; yanlışsa Arayüz(2)deki gibi ekranda uyarı mesajı verilir. Oluşan uyarı mesajında Restart seçilirse tekrar sorgu ekranı açılır close seçilirse program sonlanır.

Arayüz(3) **Arayüz(4)**

Arayüz(3) kullanıcıyı işlem seçimine yönlendirir. Yapılacak işlemler sol üst köşedeki tab bölümünden seçilir. Projenin iyileştirme aşamasında bu bölüme resimler ve farklı yönergeler eklenilebilir.

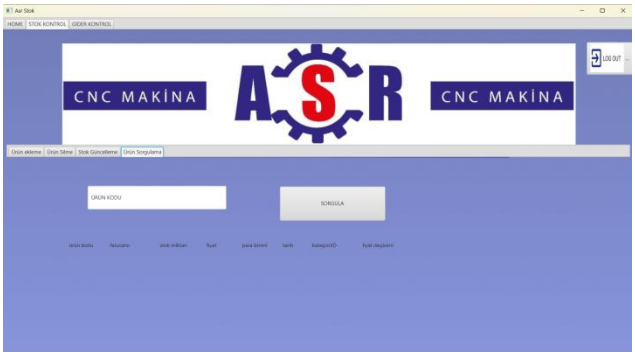
Arayüz(4) Stok Kontrol ürün işlemlerinin bulunduğu alandır. Logo altında bulunan sekmelerden kullanıcı işlemini seçer. Arayüz(4) ürün ekleme sayfasını göstermektedir. Kullanıcı gerekli alanları doldurup butona bastığında ürün veri tabanına eklenir.



Arayüz(5) **Arayüz(6)**

Arayüz(5) kullanıcıdan ürün kodu girdisi isteyip butona basınca belirtilen ürünü veri tabanından silmeyi sağlar.

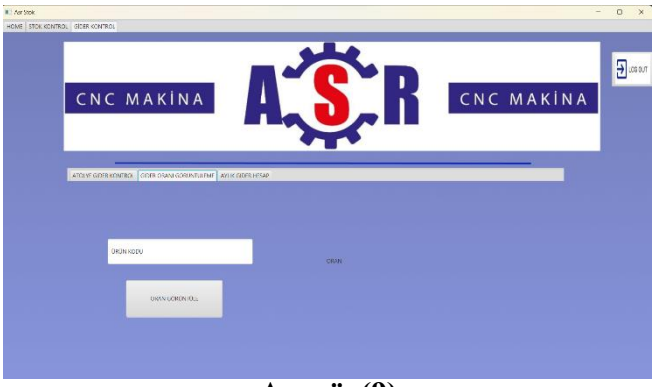
Arayüz(6) kullanıcıdan ürün bilgilerini alıp butona basıldığında veri tabanını günceller.



Arayüz(7) kullanıcıdan ürün kodu girdisi alıp butona basıldığında girilen ürünü sorgulayarak label alanlarda belirtilen ürün bilgilerini gösterir.

Arayüz(7)

Arayüz(8)

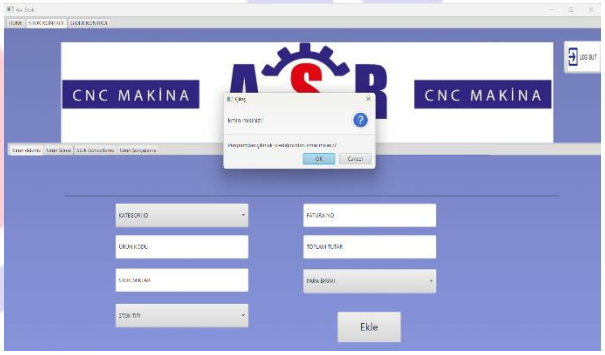


Arayüz(9)

Gider Kontrol sekmesinde firmanın gider hesap işlemlerini içeren sekmeler bulunmaktadır. Arayüz(8) kullanıcıdan alınan gider türü ve masraf tutarı bilgisi butona basıldığında veri tabanının ilgili yerine kaydedilir.

Arayüz(9) butona basıldığında kullanıcının girdiği ürün kodu doğrultusunda hesaplanan bilgi label üzerinde gösterilir.

Arayüz(10)

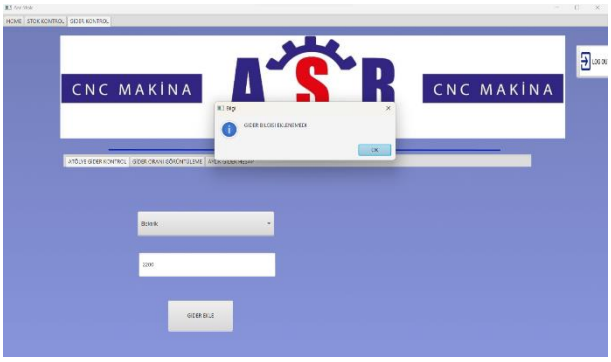


Arayüz(11)

Arayüz(10) belirlenen alanlardan mevcut kur bilgisini alıp butona basıldığında aylık gider labelında o ayın gider miktarını gösterir.

Arayüz(11) de görülen bilgi ekranı her sayfada sağ üst köşede bulunan LOGOUT butonuna basıldığında oluşur ve kullanıcı programı kapatmak için OK butonunu kullanır.

Arayüzde kullanıcının yaptığı her işlem için kullanıcıya işleminin başarıyla tamamlandığına dair işlem özelinde oluşturulan bir mesaj ekranı gösterilir. İşlem, karşılaşılan sorunlardan dolayı gerçekleştirilemiyorsa yine her işlem özelinde hazırlanmış bir mesaj ekranı ile kullanıcı bilgilendirilir.



Arayüz(12) **Arayüz(13)**

Arayüz(12) de görülen uyarı ekranı ürün silinirken yanlış ürün kodu girildiğinde veya veri tabanı bağlantısı kurulamadığında butona basılınca ekranda gösterilir.

Arayüz(13) de görülen uyarı ekranı gider bilgisi eklenirken veri tabanı bağlantısı kurulamazsa ekranda gösterilir.

3.4.Kullanıcı İşlemleri ve İpuçları

Ürün Ekleme: Veri tabanına doğru bir şekilde ekleme işlemi yapılabilmesi için kullanıcının arayüzde bulunan kategori id, ürün kodu, stok miktarı, stok tipi, fatura no, toplam tutar ve parabirimi alanlarındaki bilgileri doğru bir şekilde girmesi gerekmektedir.

Ürün Güncelleme: Veri tabanında doğru bir şekilde ürün güncelleme yapılabilmesi için kullanıcının arayüzde bulunan ürün kodu, fatura no, stok miktarı, para birimi ve fiyat bilgilerini doğru bir şekilde girmesi gerekmektedir.

Ürün silme: Veri tabanında mevcut ve doğru olan ürünü silebilmesi için kullanıcının ürün kodunu doğru bir şekilde girmesi gerekmektedir.

Ürün sorgulama: Veri tabanından doğru ürününün bilgilerinin kullanıya gösterilebilmesi için sorgulanmak istenen ürünün kodu doğru girilmelidir.

Atölye Gider Kontrol: Veri tabanına doğru bir gider ekleme yapılabilmesi için kullanıcının gider türü ve aylık masraf tutarı alanlarına bu bilgileri doğru girmesi gerekmektedir.

Gider Oranı Görüntüleme: doğru ürünle işlem yapılabilmesi için kullanıcının ürün kodunu doğru girmesi gerekmektedir.

Aylık Gider Hesap: Atölyenin mevcut bütün giderlerinin toplamlarının doğru hesaplanabilmesi için para birimlerini ortak bi paydada buluşması gerekir. Bu işlem para birimlerini türk lirası olarak çevirmeyle gerçekleşir. Bu sebeple doğru bir hesap yapılabilmesi için mevcut dolar ve euro kurunu doğru girmesi gerekmektedir.

Her işlem sonunda bu işlemlerin başarılı olup olmadığına dair arayüzde kullanıcıya mesaj kutusu aracılığıyla bilgi verilir. Bu işlem sayesinde kullanıcı yapılan işlemin düzgün gerçekleşip gerçekleşmediği ile ilgili haberdar edilir.

3.5. Hata Mesajları ve Çözümleri:

Login işlemi: Giriş işleminin gerçekleşememesi durununda mesaj kutusu aracılığıyla uyarı verilir. Bu uyarıyla karşılaşılması durumunda uyarıda belirtildiği gibi kullanıcı adı ve şifrenin kontrol edilmesi gerekmektedir.

Ürün Ekleme: Ürün ekleme işlemi sırasında mesaj kutusu aracılığıyla eklenemedi uyarısı alınıyorsa uygulamada veri tabanı ile bağlantı yapılırken bir problemle karşılaştığı anlamına gelir. Kullanıcı adı, veritabanı adı, şifre gibi bağlantılar kontrol edilir veya bağlantı kurulan ağ kontrolü yapılır.

Ürün Güncelleme: Ürün güncelleme işlemi sırasında mesaj kutucuğu aracılığıyla ürünün güncellenemediğine dair bir uyarı alınıyorsa ürün kodu yanlış girilmiş olabilir. Öncelikle ürün kodunun kontrol edilmesi ve veri tabanında mevcut bir ürün olduğundan emin olunması gerekmektedir. Eğer ürün kodu doğru ve veri tabanında mevcut bir ürün ise o zaman veritabanı ile bağlantıda bir problem ile karşılaşıldığı anlamına gelir. Bu durum sunucu veya ağ kaynaklı olabileceğinden gerekli kontroller yapılır.

Ürün Silme: Ürün silme işlemi sırasında ürünün silinemediğine dair uyarı alınması durumunda silinmek istene ürününün kodunun doğru girildiği ve veri tabanında mevcut olduğu kullanıcı tarafından teyit edilmelidir. Eğer ürün kodu doğru ve veri tabanında mevcut bir ürün ise o zaman veritabanı ile bağlantıda bir problem ile karşılaşıldığı anlamına gelir. Bu durumda sunucu ve ağ kontrolü yapılabilir.

Ürün Sorgulama: ürün sorgulama sırasında arayüzdeki labellarda bilgilerin görüntülenememesi durumunda sorgulanmak istenen ürünün kodunun doğruluğu ve ürünün veri tabanındaki mevcutluğu teyit edilmelidir. Eğer ürün kodu doğru ve ürünün veri tabanında mevcutluğu kesin ise o zaman veritananı ile bağlantıda bir problem ile karşılaşıldığı anlamına gelir. Bu durumda mevcut sql sorguları ve ağ kaynaklı problemler kontrol edilir.

Atölye Gider Kontrol: gider ekleme sırasında eklenemedi uyarısı alınması durumu veritabanı ile bağlantı sorunu olduğu anlamına gelir. Bu durumda veritabanı bağlantısı zaman aşımın, sunucu hataları gibi olası sorunlar kontrol edilir.

Gider Oranı Görüntüleme: Oranın görüntülenememesi durumunda ürün kodunun veritabanında bulunup bulunmadığı ve ürün kodunun doğruluğu teyit edilmelidir. Ürün veri tabanında bulunuyor ve kod doğruysa o zaman veritabanı ile bağlatıda bir problem yaşandığı anlamına gelir. Kullanıcı adı, veritabanı adında problem yoksa sunucu ve ağ hataları kontrol edilir.

Aylık Gider Hesap: arayüzde herkangi bir çıktıyla karşılaşılamaması durumunda veri tabanıyla bağlantı kurulma aşamasında problem yaşandığı anlamına gelir. Bu durumda veritabanı sürümü ve sunucu sunucu sürümü kontrol edilir.

4. Yönetim Bakım Kılavuzu:

4.1.Yönetim Arayüzü ve Kontrolleri:

Admin kullanıcı veritabanına doğrudan erişim sağlayabildiğinden veritabanında tutulan bilgileri değiştirme özelliğine sahiptir. Stok ekleme, güncelleme ve silme işlevleri ile veri tabanında doğrudan değişimler yapabilirken, sorgulama işlemi ile veritabanındaki bilgileri görüntüleyebilir. Ürünlerde olduğu gibi giderler bölümünde de atölyeye ait giderleri girebilme ve bu giderlerin maliyetlerini ürün bazında ve atölyenin aylık toplam gideri olacak şekilde görüntüleyebilir.

4.2. Rutin Bakım İşlemleri:

Veri tabanı bilgilerinin güncel ve geçerli olabilmesi için kullanıcının düzenli olarak ürün ve maliyet ile ilgili tüm bilgileri uygulama arayüzü aracılığıyla güncellemesi gerekmektedir.

Günlük rutin bakım işlemleri arasında, stok güncelleme işlemlerinin düzenli olarak yapılması, veritabanı yedeklemelerinin alınması ve sistem güvenliği için güncel güvenlik yamalarının uygulanması yer alır. Bu işlemler, sistemdeki verimliliği artırmaya ve olası sorunları önlemeye yardımcı olur.

4.3. Sorun Giderme ve Hata Ayıklama:

Hata (1): Stok Bilgilerinin Uyumsuzluğu: Atölyede mevcutta bulunan ürün miktarı ile veri tabanında tutulan ünün miktarı birbirini tutmuyor olabilir.

Çözüm (1): Stok güncelleme işlemlerinin doğru yapılıp yapılmadığını kontrol edilmelidir. Manuel olarak stok bilgilerini karşılaştırarak sistemdeki uyumsuzlukları düzeltilmelidir.

İpucu: Stok güncelleme işlemlerinden sonra sorgu işlemi ile işlemin doğruluğu kontrol edilebilir.

Hata (2): Arayüzde Bağlantı Problemi: Stok projesinde erişimde bağlantı problemi yaşanıyor olabilir.

Çözüm(2): Ağ bağlantısını kontrol edin ve varsa kesintilerin giderilmesi gerekmektedir.Sunucu tarafından sorun olup olmadığını kontrol edilmeli ve gerekirse sistem yöneticisiyle iletişime geçilmelidir.

Hata(3): Veritabanı Bağlantısı: Uygulama ile veritabanı bağlantısı kurulamıyor olabilir.

Çözüm(3): Veritabanı sunucusunun çalıştığından emin olunmalıdır.Bağlantı bilgileri kontrol edilerek doğru olduğundan emin olunmalıdır.Gerekirse veritabanı şifresi sıfırlanmalı ve bağlantı yeniden kurulmalıdır.

İpucu: Veritabanı bağlantısı ile ilgili hataları önlemek için düzenli yedekleme işlemleri gerçekleştirilmeli ve bağlantı bilgileri güvende tutulmalıdır.

5.API Dokümantasyonu:

5.1.API Endpointleri ve Açıklamaları

5.1.1. GET /api/products

Açıklama: Tüm ürünleri listeler.

Parametreler: Yok.

5.1.2. GET /api/products/{id}

Açıklama: Belirli bir ürünün detaylarını getirir.

Parametreler: {id} - Ürün kimliği.

5.1.3. POST /api/products

Açıklama: Yeni bir ürün ekler.

Parametreler: Ürün bilgileri (örneğin, ad, fiyat, stok miktarı).

5.1.4. PUT /api/products/{id}

Açıklama: Belirli bir ürünün bilgilerini günceller.

Parametreler: {id} - Güncellenecek ürünün kimliği, Güncellenmiş ürün bilgileri.

5.1.5. DELETE /api/products/{id}

Açıklama: Belirli bir ürünü siler.

Parametreler: {id} - Silinecek ürünün kimliği.

5.1.6. GET /api/inventory/{productId}

Açıklama: Belirli bir ürünün stok durumunu getirir.

Parametreler: {productId} - Ürün kimliği.

5.1.7. POST /api/inventory/{productId}/adjust

Açıklama: Belirli bir ürünün stok miktarını düzenler.

Parametreler: {productId} - Ürün kimliği, Düzeltilmiş stok miktarı.

5.1.8. GET/api/Outgoings

Açıklama: Giderleri listeler.

Parametreler: Yok.

5.2.İstek ve Yanıt Formatları

İstek Formatı:

```
POST /api/products
Content-Type: application/xml
<product>
  <ProductName>Ürün Adı</productName>
  <Price>29.99</price>
  <StockQuantity>100</stockQuantity>
</product>
```

Yanıt Formatı:

```
HTTP/1.1 200 OK
Content-Type: application/xml
<product>
  <productId>123</productId>
  <ProductName>Ürün Adı</productName>
  <Price>29.99</price>
  <StockQuantity>100</stockQuantity>
</product>
```

5.3.Yetkilendirme ve Kimlik Doğrulama

5.3.1. Kimlik Doğrulama:

Kullanıcı Adı ve Şifre Girişi:

Kullanıcı, uygulamanın giriş sayfasında önceden belirlenmiş olan kullanıcı adı ve şifre bilgilerini girecektir.

Kimlik Bilgilerinin Kontrolü:

Sunucu, gelen kimlik bilgilerini doğrulama işlemine tabi tutar.

Kullanıcı adı ve şifrenin geçerliliği kontrol edilir.

5.3.2. Yetkilendirme Süreçleri ve Politikalar:

Kullanıcı Rollerinin Belirlenmesi:

Uygulamada 2 adet kullanıcı olacaktır. Her iki kullanıcı da admin olarak işlem yapacaktır.

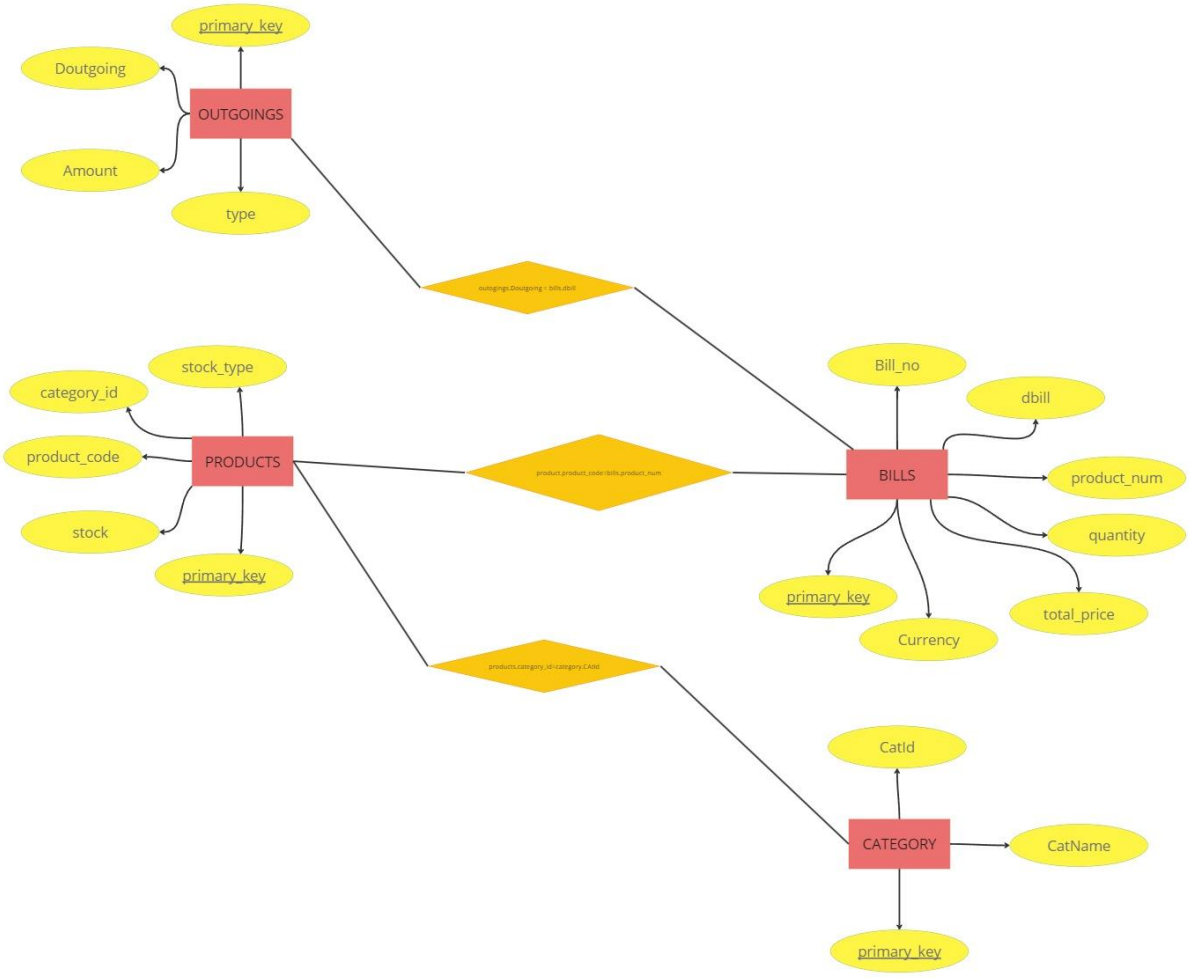
Yetkilendirme Politikalarının Oluşturulması:

Her iki kullanıcı da uygulamanın işlevleri olan ürün ekleme, ürün silme, ürün güncelleme, giderleri kontrol etme yetkilerine sahip olacaktır. Kullanıcılara tam yetki verilecektir.

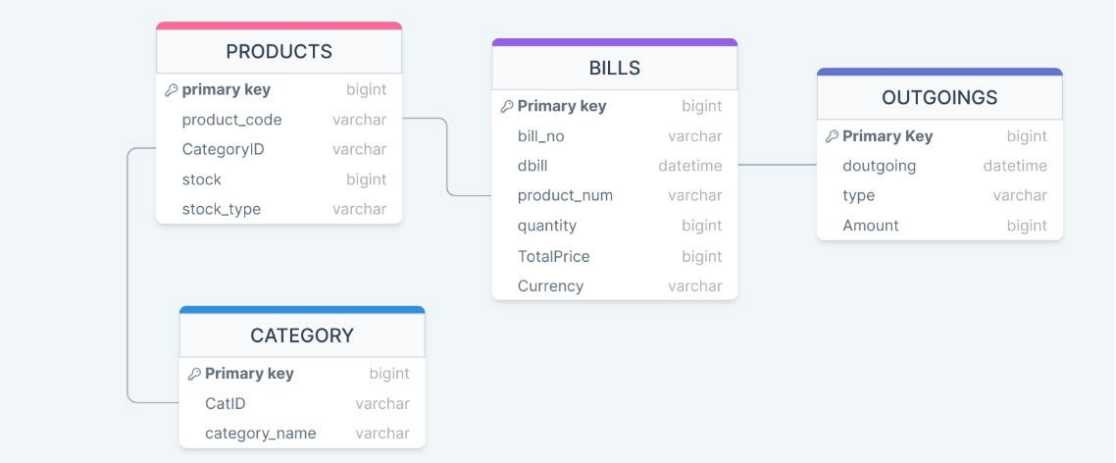
6.Veritabanı Dokümantasyonu:

6.1. Veri tabanı Şeması ve Tabloların Açıklamaları:

6.1.1. ER Diyagramı:



6.1.2. Veri tabanı Tabloları



6.2.İndeksleme ve Optimizasyon Stratejileri

Uygulamada bulunan "giderler," "ürünler," "faturalar," ve "kategoriler" tabloları için indeksleme ve optimizasyon stratejileri şu şekilde düşünülebilir:

Giderler Tablosu İçin:

- Giderler tablosunda tarih bazlı sorgular sıkça yapılacağından, tarih alanına indeks eklemek faydalı olacaktır.
- Gider türü sık filtrelenen bir alan olduğundan bu alana indeks eklemek, sorgu performansını artırabilir.
- İlgili diğer tablolarla ilişkilendirilen alanlara indeks eklemek, JOIN işlemlerini optimize edebilir.

Ürünler Tablosu İçin:

- Ürünler tablosunda sıkça kullanılan filtreleme veya sıralama alanlarına indeks eklemek, sorgu performansını artırabilir.
- Kategori ID'si gibi sık kullanılan sütunlara indeks eklemek, ilişkili kategori sorgularını hızlandırabilir.
- Stok miktarı gibi sık güncellenen alanlarda indeks kullanımını dikkatlice değerlendirmek önemlidir.

Faturalar Tablosu İçin:

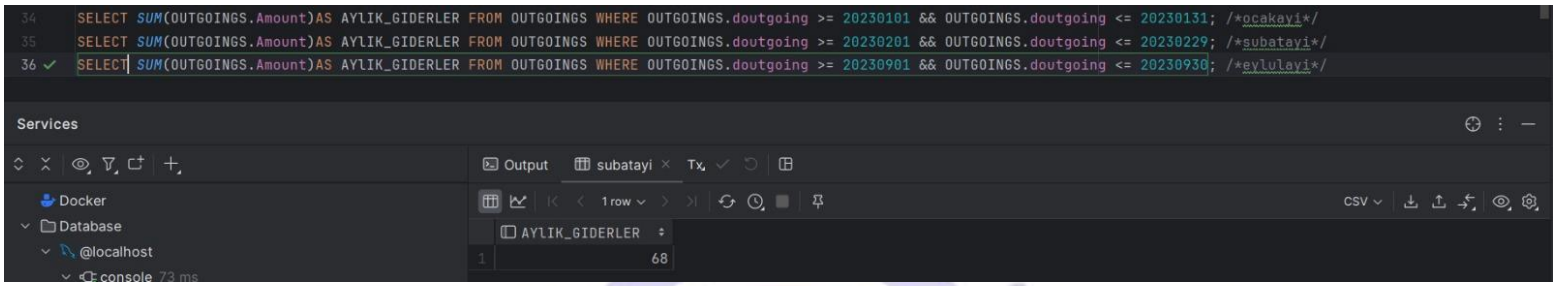
- Fatura tarihi veya fatura no gibi sık filtrelenen alanlara indeks eklemek, sorgu performansını artırabilir.
- Ürün ID'si gibi sık kullanılan alanlara indeks eklemek, ilişkili ürün sorgularını hızlandırabilir.

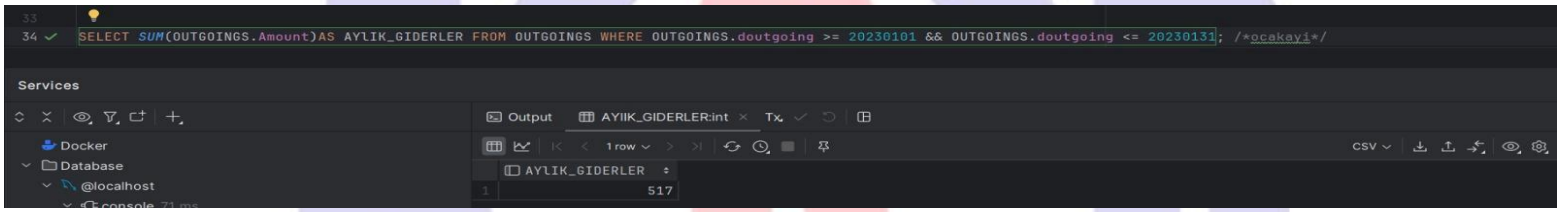
Kategoriler Tablosu İçin:

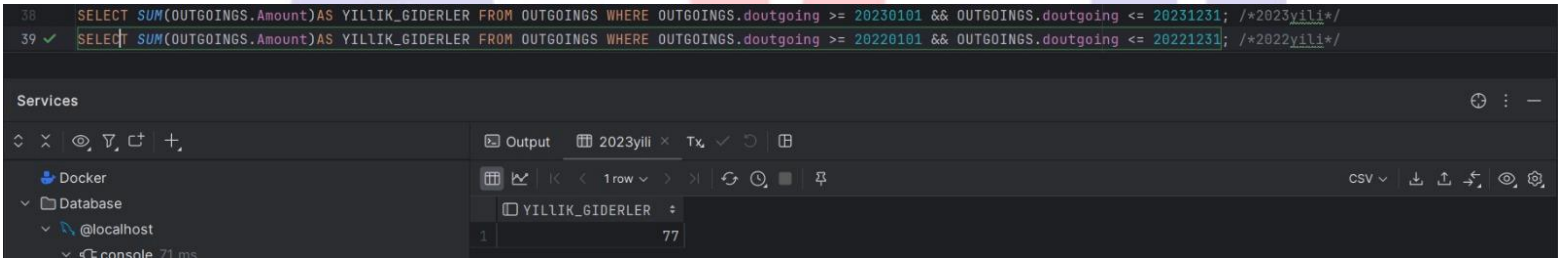
- Kategori adı gibi sıkça sorgulanan alanlara indeks eklemek, sorgu performansını artırabilir.
- Ürünler tablosuyla ilişkili olan Kategori ID'si gibi sütunlara indeks eklemek, JOIN işlemlerini optimize edebilir.

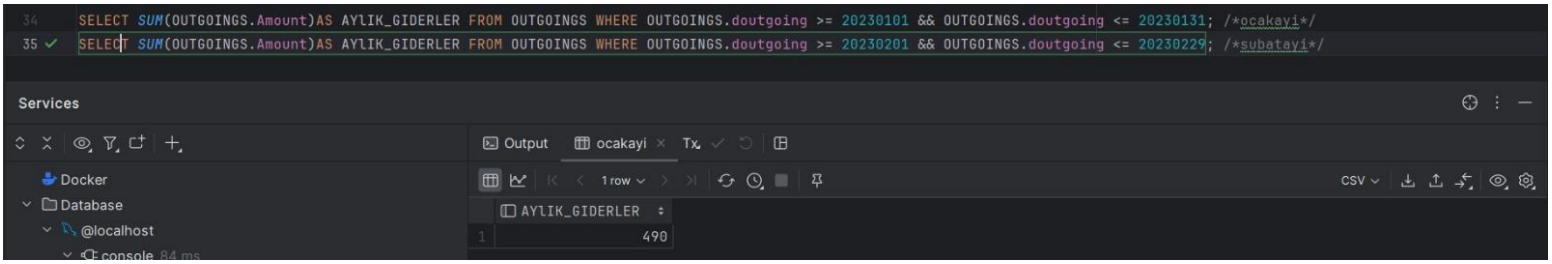
7. Performans Dokümantasyonu:

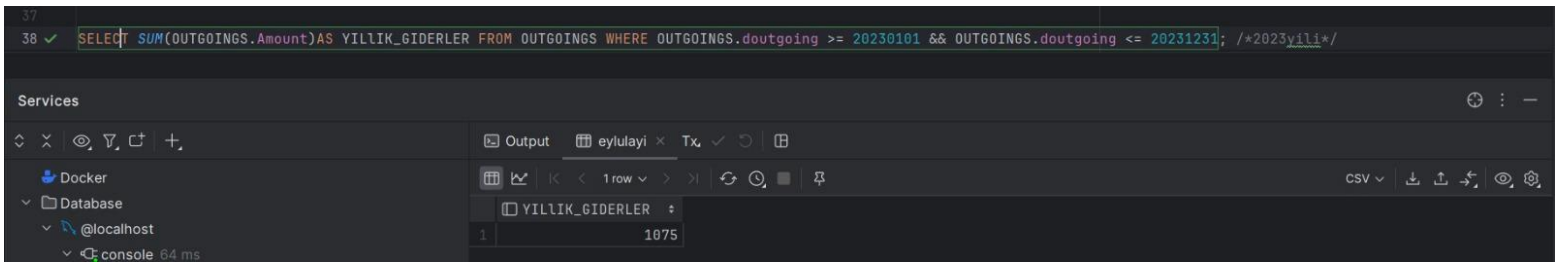
7.1.Performans Test Sonuçları:

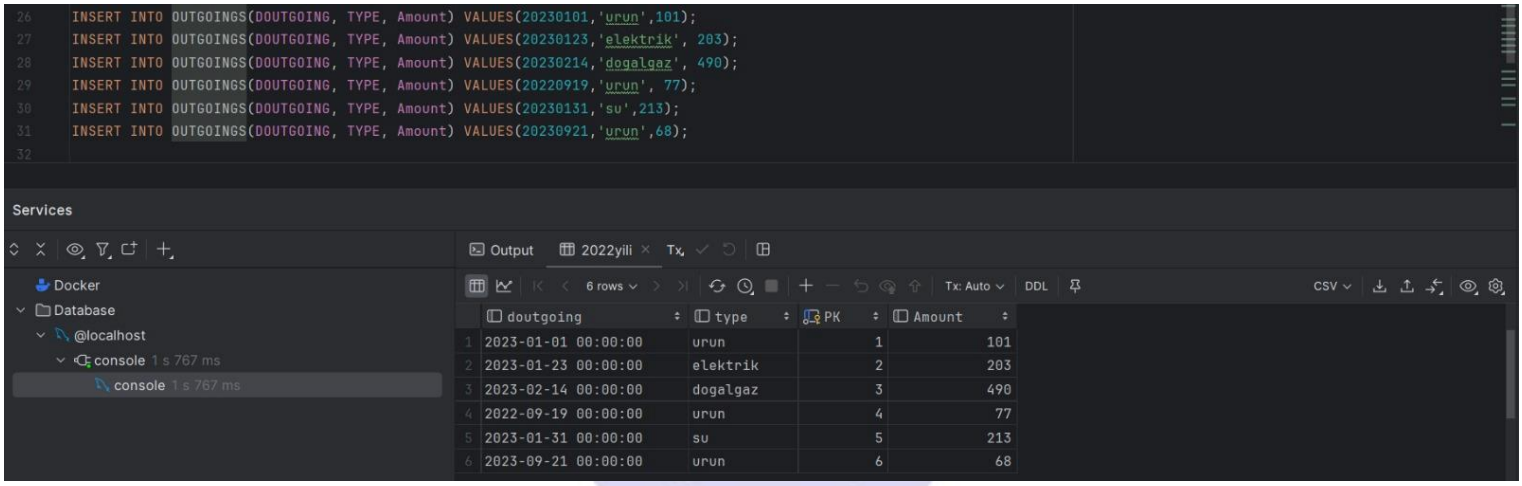












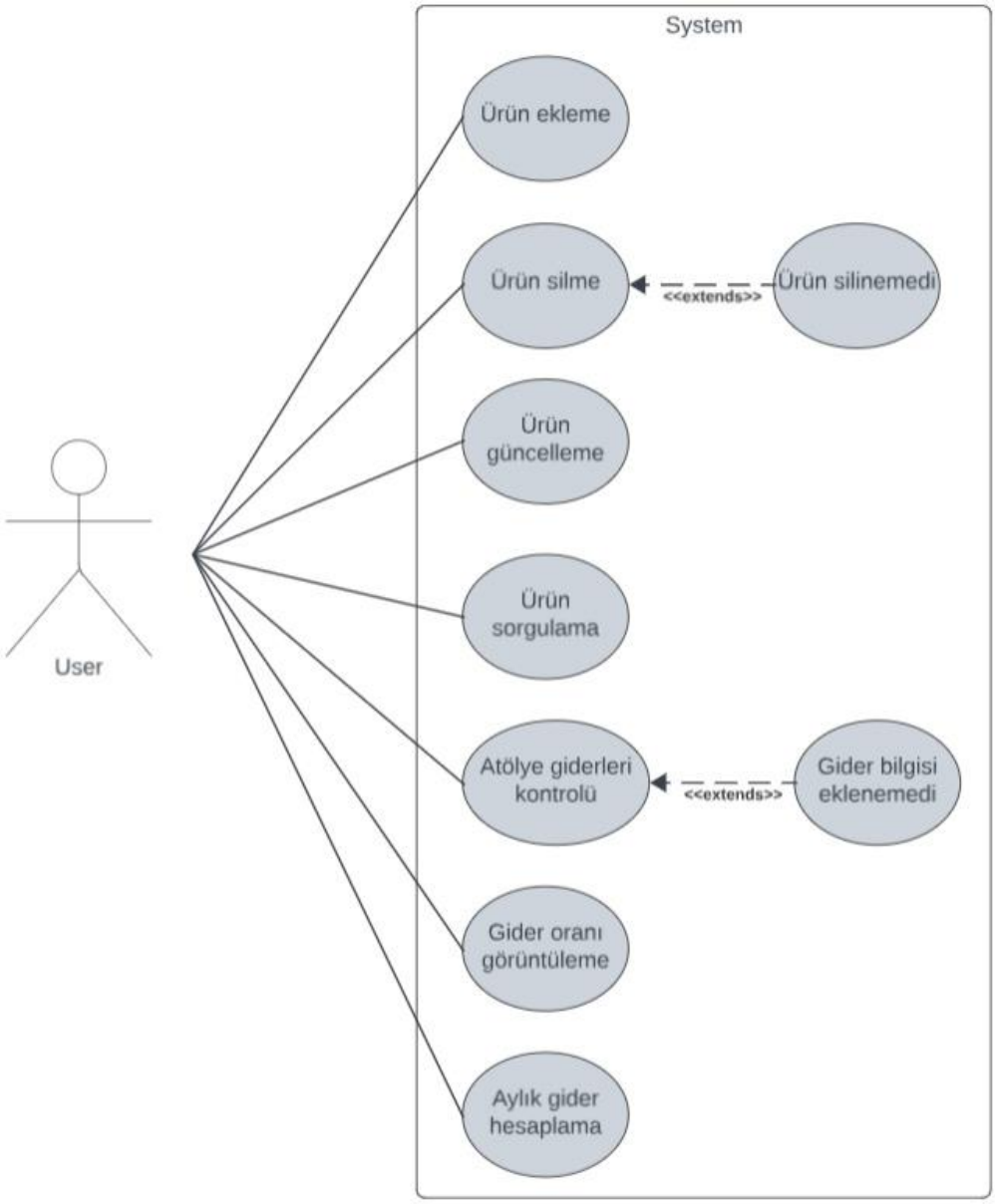
7.2.İyileştirme Stratejileri

- Uygulamada home arayüzü daha ilgi çekici olacak şekilde görseller ve diğer ilgili materyaller ile güçlendirilebilir.
- Uygulama kullanıcı ekleme özelliği ile birden çok sayıda kullanıcının kullanacağı şekilde ayarlanabilir.
- Uygulamada ekranda gösterilen veriler tablo benzeri daha ilgi çekici bir vasıta ile gösterilebilir.
- Uygulama daha ilgi çekici hale getirilmek için çeşitli görsel materyaller eklenebilir.
- Uygulamada oluşan problemleri çözmek için kullanıcının yararlanacağı ipucu ve help gibi sorun çözme amaçlı hazırlanmış yardımcı alt parçalar oluşturulabilir.
- Uygulamada fonksiyonlara belirli erişim seviyeleri atanabilir. Yalnızca özelliği erişim seviyesi uygun olanlar kullanacak şekilde tasarlanabilir.
- Dolar ve euro kuru internet vasıtasıyla çekilerek işlem yapılabilir.
- Maliyet değişimleri yıllık ve aylık bazda yüzdelik cinsinden artış veya azalış olacak şekilde gösterilebilir.

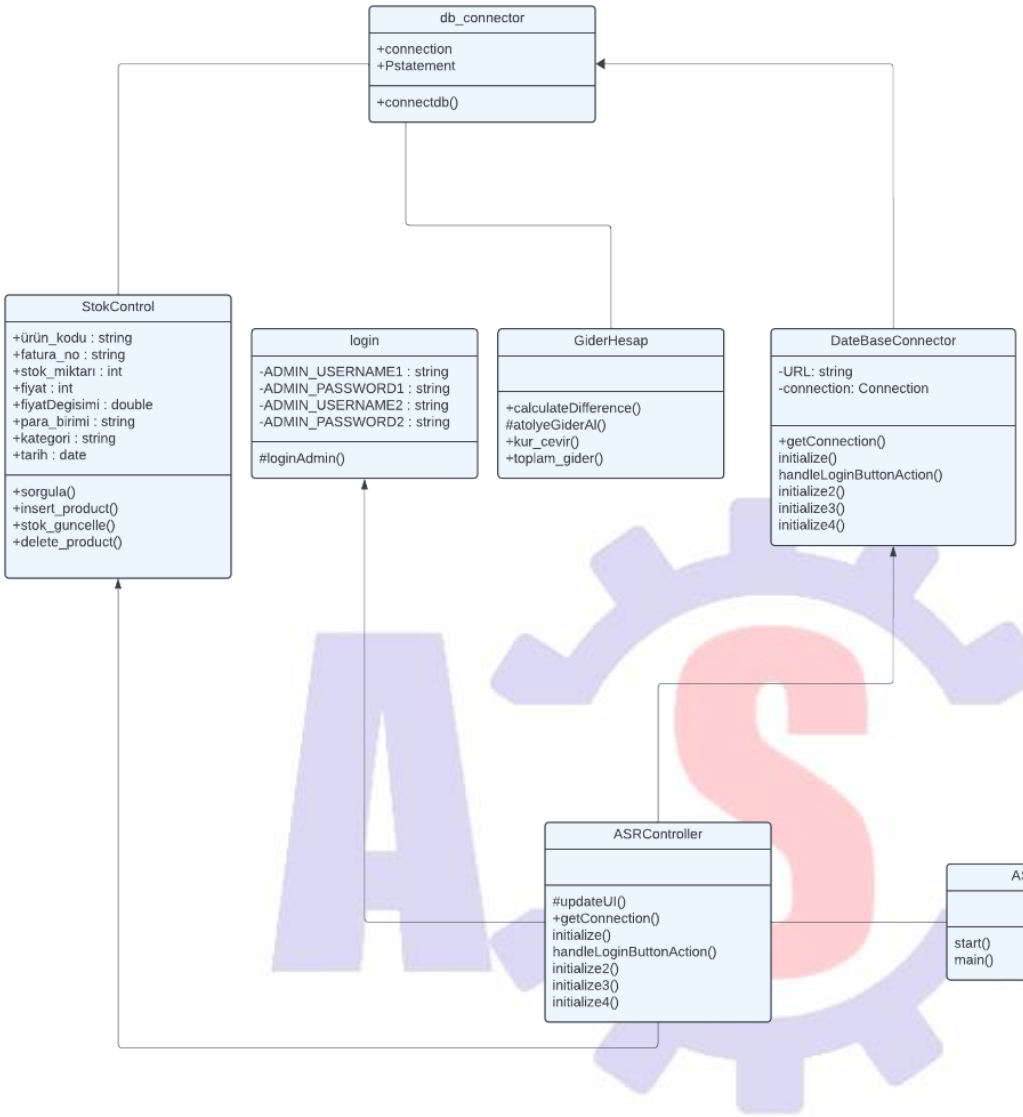
8. Proje Kodu Dokümantasyonu:

8.1. Diyagramlar:

8.1.1. Use Case Diyagramı:



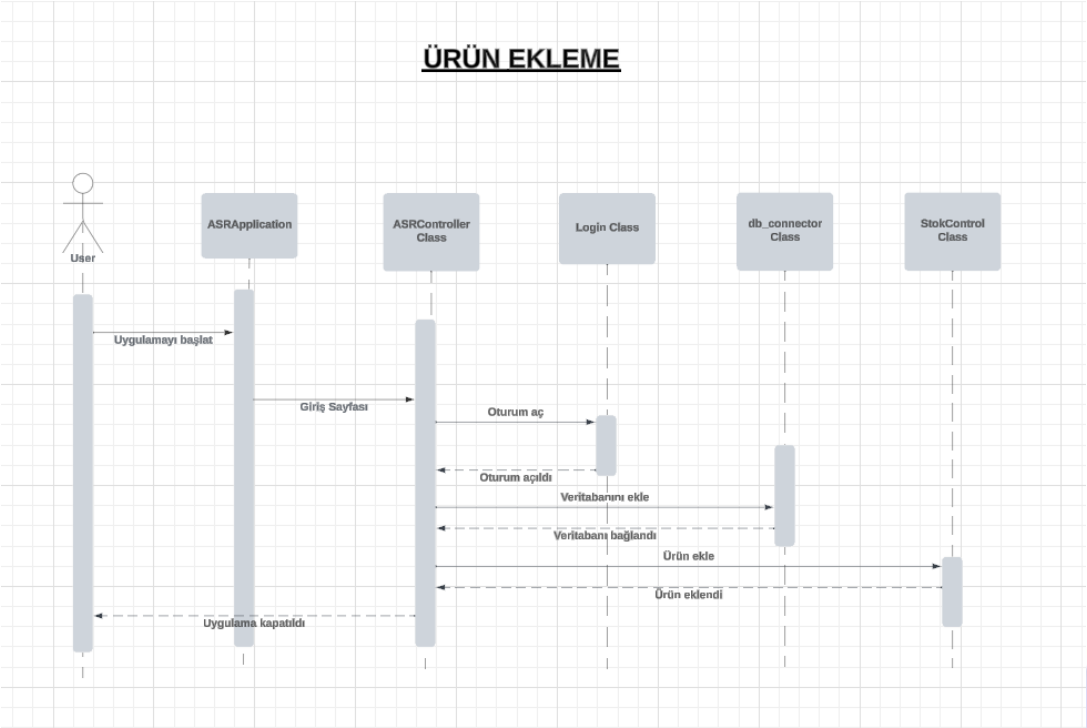
8.1.2. Sınıf Diyagramı:



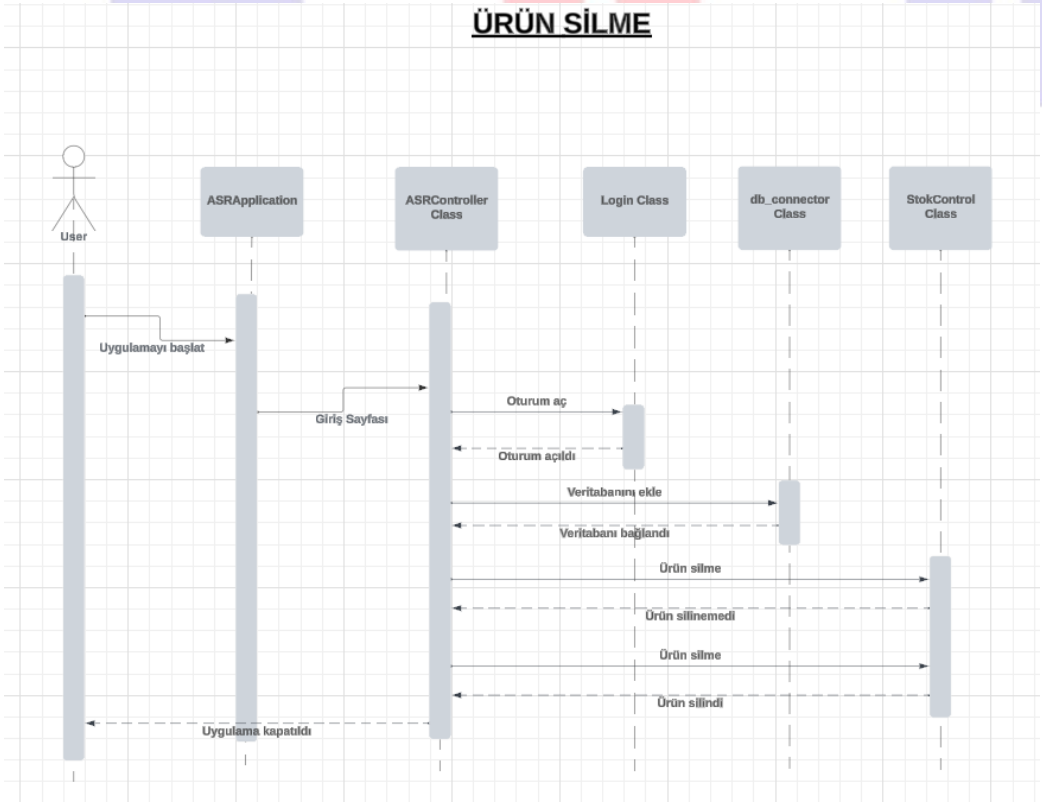
class diagram

8.1.3. Sequence Diyagramları:

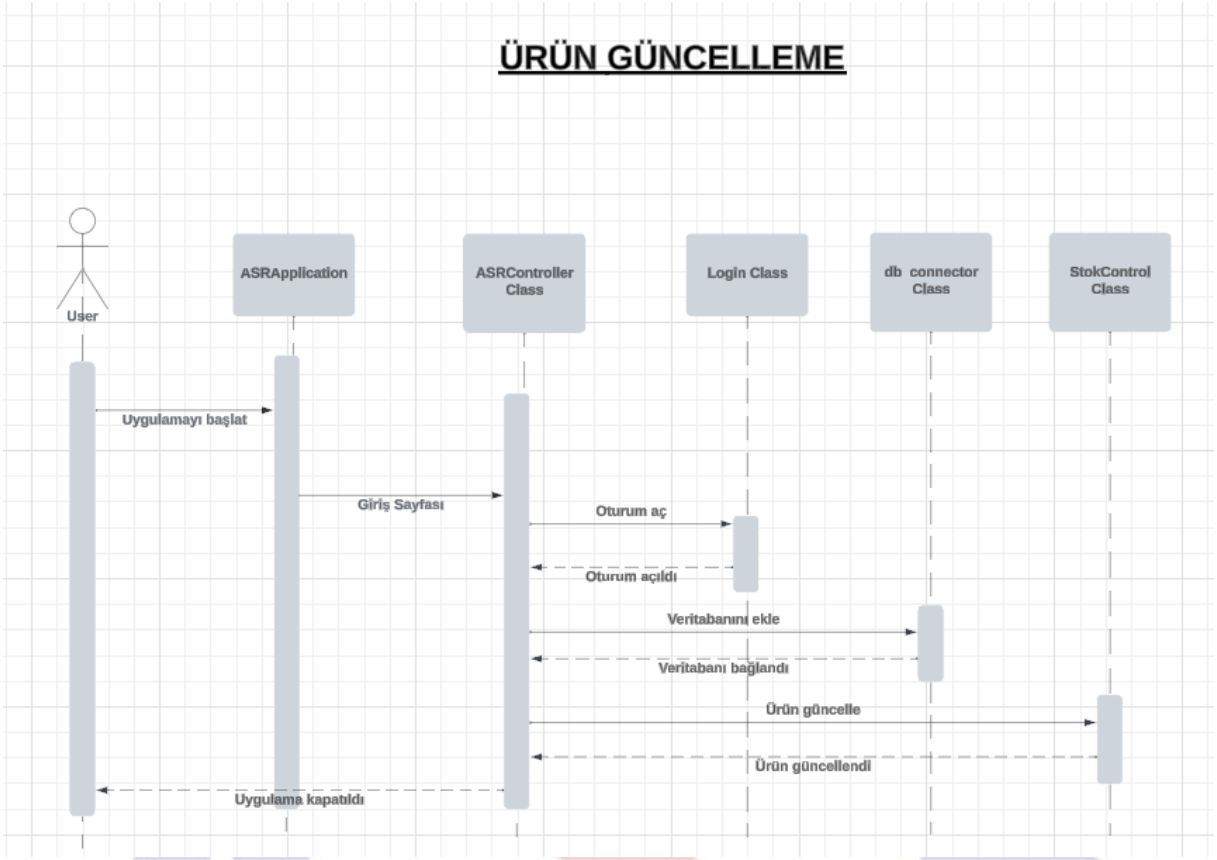
8.1.3.1.Stoğa ürün ekleme:



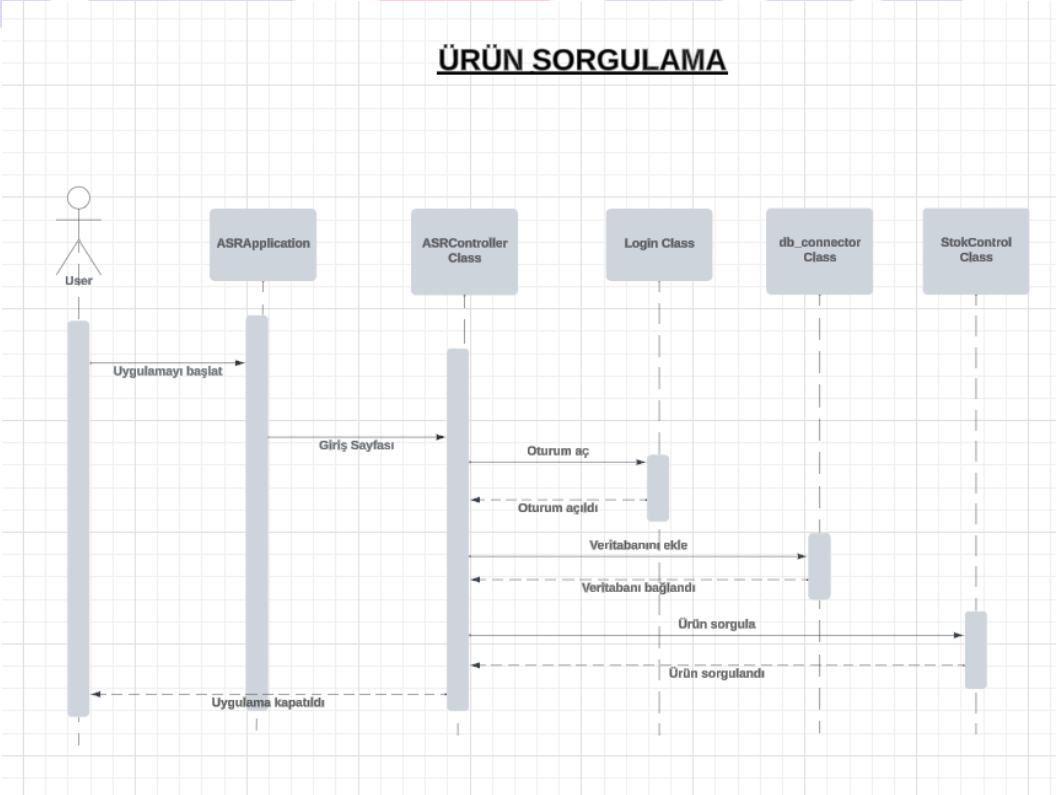
8.1.3.2.Ürün Silme:



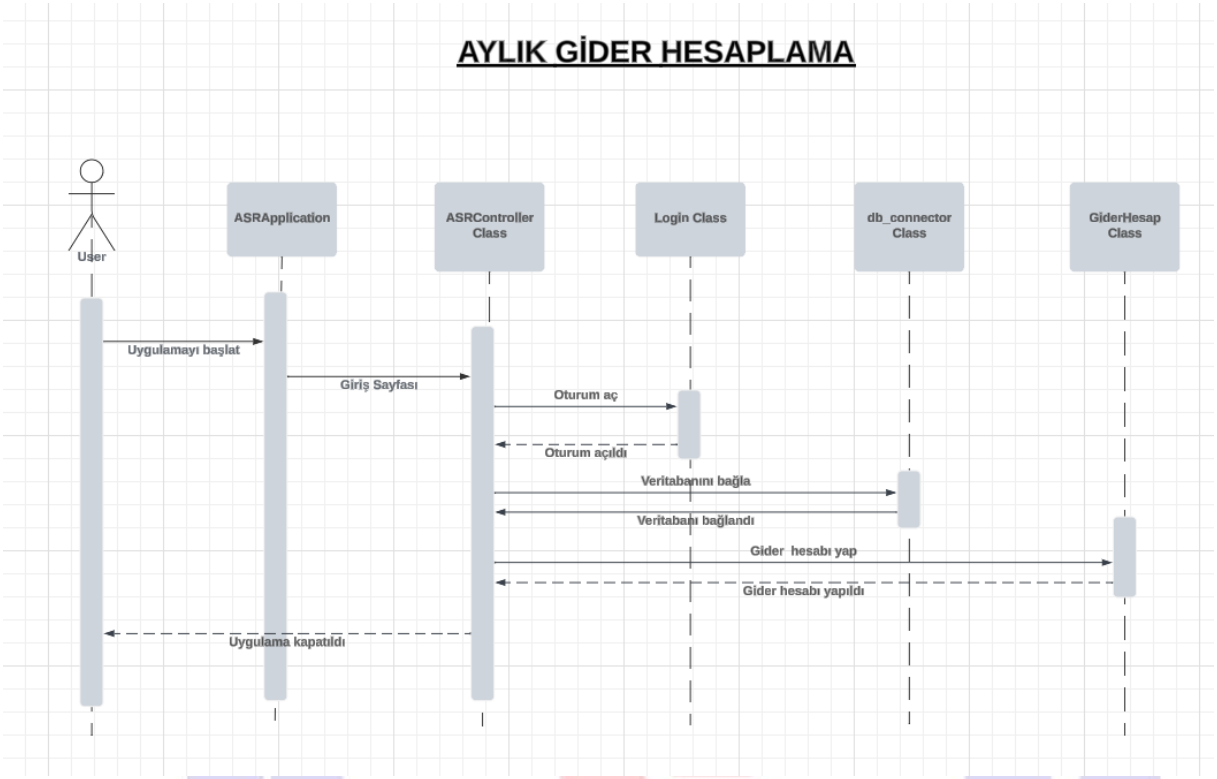
8.1.3.3. Ürün Güncelleme:



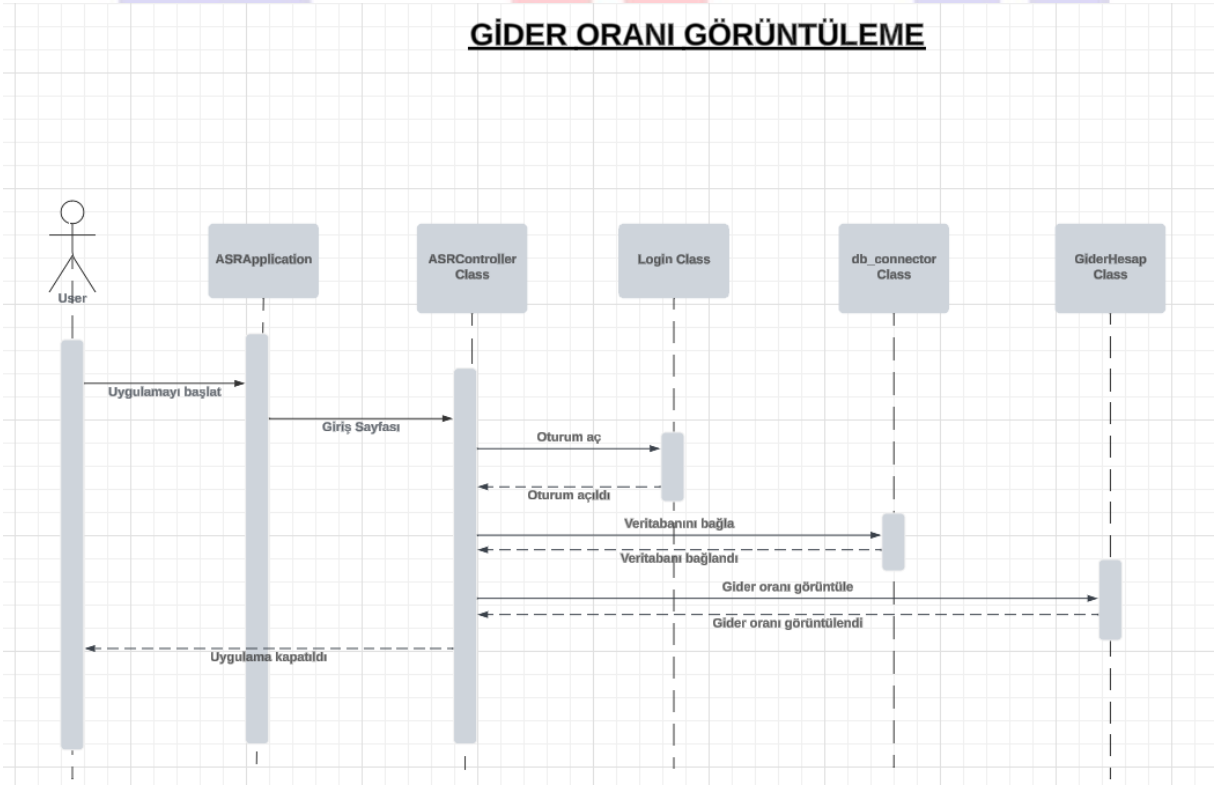
8.1.3.4. Ürün Sorgulama:



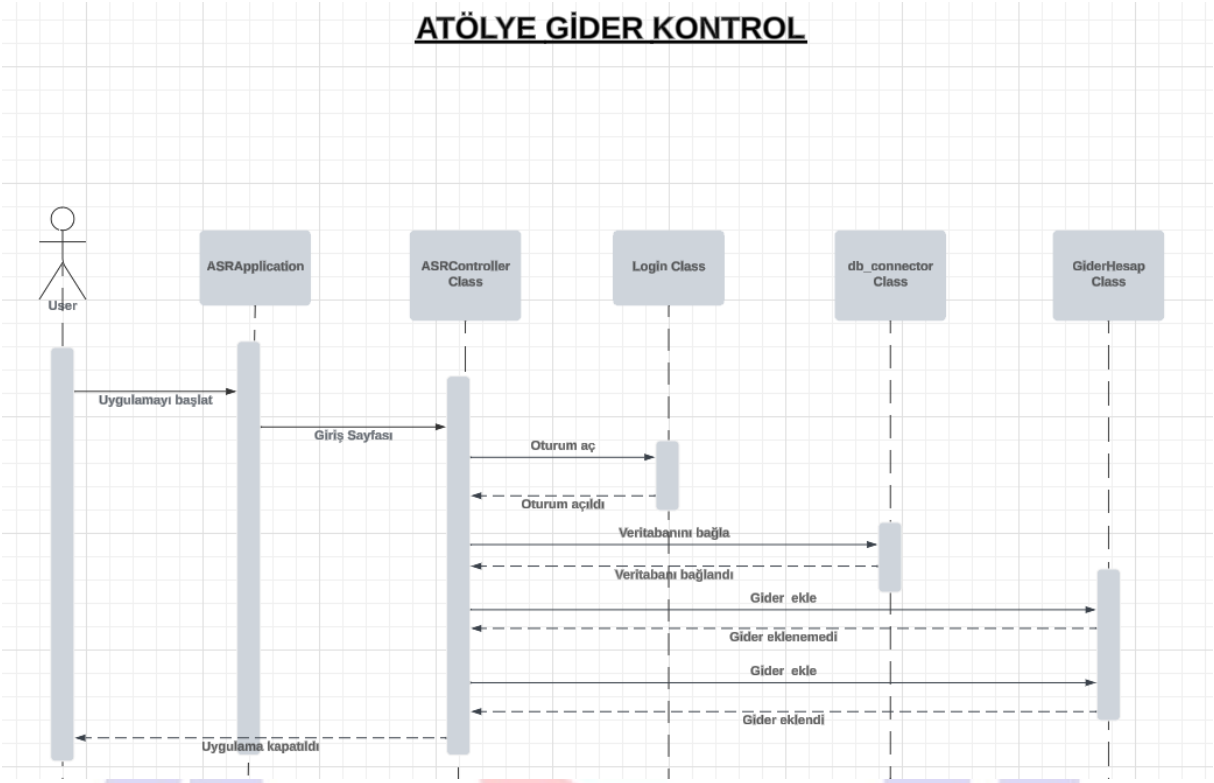
8.1.3.5. Aylık Gider Hesaplama:



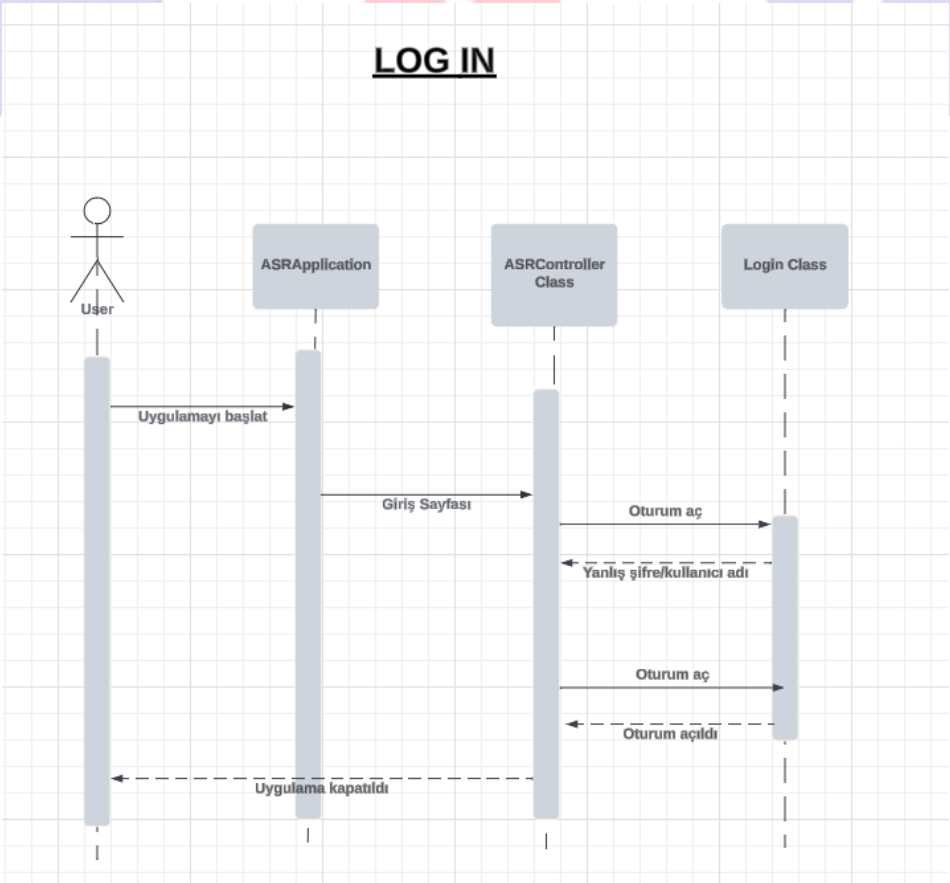
8.1.3.6. Gider Değişim Oranı Görüntüleme:



8.1.3.7. Atölye Gider Kontrol:



8.1.3.8. Uygulamaya Giriş Yapma :



AsrApplication {} sınıfı:

Bu sınıfta arayüzün genel tasarımı projeye entegre edilmiştir. Arka plan tasarımı ayarlanmıştır.

```
ImageView backgroundImageView = new ImageView(
backgroundImageView.setFitWidth(1400);
backgroundImageView.setFitHeight(800);
```

AsrController {} sınıfı:

Bu sınıfta controller yapılar oluşturulmuştur. Buttonlar, textfieldlar vb. burada tanımlanmıştır.

```
@FXML//home tabının arka planı olarak kullanıldı
private ImageView imageView;


@FXML
protected Button button1;// login butonu

@FXML
protected TextField username;//login ekranindaki username textfield i

@FXML
protected PasswordField password;//login ekranindaki password textfield i
```

Db_Connector {} sınıfı:

Bu sınıfta database ile bağlantı kurulmuştur. Projenin backend kısmında fonksiyonlar ile database da tutulan tablolar arası işlemler bağlantı sayesinde gerçekleştirilir.

```
public Connection connectdb() throws SQLException {

    connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql", "yaren", "yaren");
    return  connection;
```

GiderHesap {} sınıfı:

Gider hesaplama fonksiyonun işlemlerini gerçekleştirmek için özel oluşturulmuş sınıf.

```
ublic class GiderHesap {

   public double calculateDifference(String urun_kodu) {
       double difference = 0.0;
       try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql", "yaren", "yaren")) {

           String sql = "SELECT TotalPrice, dbill FROM stock_tracking.bills WHERE PRODUCT_NUM = ? ORDER BY dbill";
           PreparedStatement preparedStatement = conn.prepareStatement(sql);
           preparedStatement.setString(1,urun_kodu);
           ResultSet resultSet = preparedStatement.executeQuery();
```

Login {} sınıfı:

Giriş ekranı fonksiyon ve kodlarını içeren sınıf. Şifre kullanıcı kontrolleri gerçekleştirilir.

```
protected boolean loginAdmin(String username, String password) {
    return ((ADMIN_USERNAME1.equals(username) && ADMIN_PASSWORD1.equals(password))||(ADMIN_USERNAME2.equals(username) && ADMIN_PASSWORD2.equals(password)));
}
```

StokControl {} sınıfı:

Stok işlemlerinin büyük bölümünün gerçekleştiği sınıftır.

Projede kullanılan temel fonksiyonlar:

- İNSERT_PRODUCT ()
- STOK_GUNCELLE ()
- DELETE_PRODUCT ()
- SORGULA ()

Projede kullanılan temel sınıflar:

- Login {}
- Db_connector {}
- Main {}
- Stok_İslemleri {}

İnsert_product Fonksiyonu:

Bu fonksiyon veri tabanımızdaki tablomuzda gereksinimlerimizden biri olan ürün eklemek için oluşturuldu.

```
public void insert_product(String ürün_kodu,int categoryid,int stock_miktarı,
                           String stoktipi,String billno,int quantity, int Totalprice,
                           String currency) throws SQLException {
```

Bir ürünün eklenmesi için bu fonksiyonda 8 adet parametre kullanılmıştır. Bunlar başlıca:

1. ürün_kodu
2. categoryid
3. stock_miktarı
4. stoktipi
5. Billno
6. Quantity
7. Totalprice
8. Currency

Bu 8 veri insert_product fonksiyonunda kullanıcıdan alınıp tablolara eklenecek verilerdir.

Bu fonksiyonda 2 farklı tabloya ürün ekleneceği için 2 adet db_connector sınıfından nesne çağırıldı. Ürünler products ve bills tablosuna ekleneceği için ilgili tabloyla ilgili parametreler kullanarak SQL kodlarıyla ekleme işlemi yapıldı.

```
String sql = "INSERT INTO STOCK_TRACKING.PRODUCTS(product_code, categoryID, stock, stock_type)VALUES (?, ?, ?, ?);";
String sql1 = "INSERT INTO STOCK_TRACKING.BILLS(bill_no, quantity, TotalPrice, Currency, product_num, dbill)VALUES (?, ?, ?, ?,?,?);";

PreparedStatement preparedStatement = dbConnector.connectdb().prepareStatement(sql);
PreparedStatement preparedStatement1 = dbConnector1.connectdb().prepareStatement(sql1);
```

Stok_guncelle Fonksiyonu:

Bu fonksiyonda ürün kodu değişmez, fatura numarasında güncelleme yapılır.

```
public void stok_guncelle(String fatura_no ,int stock_miktarı, int fiyat,String currency,String ürün_kodu)
```

Bir ürünün güncellenmesi için bu fonksiyonda 5 adet parametre kullanılmıştır. Bunlar başlıca:

1. fatura_no
2. stock_miktarı
3. fiyat
4. currency
5. ürün_kodu

Bu veriler kullanıcıdan alınarak gerekli tablolarda güncellemeler yapılır. Aynı zamanda güncelleme yapılırken eski verilerin kaydedilmesi, fiyat değişimi de bu fonksiyonun içerisinde hesaplanır.

```
String sql = "UPDATE STOCK_TRACKING.BILLS SET BILL_NO = ? ," +
        " QUANTITY=? , TOTALPRICE = ? , CURRENCY = ? WHERE PRODUCT_NUM = ?;";
```

Değişiklikler tek tabloda yapılacağı için bir SQL sorgusu yeterli olmuştur.

Delete_product Fonksiyonu:

Sadece ürün kodu silinerek ürün tablodan silinmez. Ürün kodu primary key değildir.

```
public void delete_product(String deger)
```

Ürünün silinebilmesi için önce ürün sorgulama fonksiyonu ile ürün kodu girilerek ürün bulunur daha sonra bu ürünün bulunduğu satır ürünler tablosundan silinir

```
String sql = "DELETE FROM STOCK_TRACKING.PRODUCTS WHERE PRODUCT_CODE = ?;";
```

Sorgula Fonksiyonu:

```
public void sorgula(String ürün_kodu)
```

Ürün sorgulamak için kullanıcı ürün kodunu girer ve ekranda o ürünün stok durumunu fatura numarasını, fiyatını, fiyat değişim oranını, para birimi, kategorisi, son güncelleme tarihini görür. Bu fonksiyon tek bir parametre alıp diğer verileri bu parametreye göre tablolardan çekip gerekli değerleri döndürmelidir.

Bütün bu işlemler için 2 farklı SQL değişkeni tanımlanmıştır.

```
String sql = "SELECT bill_no,dbill,quantity," +
        "TotalPrice,Currency FROM STOCK_TRACKING.BILLS WHERE PRODUCT_NUM = ? ORDER BY dbill";
```

9. Test Dokümantasyonu:

9.1.Test Senaryoları ve Sonuçları

I. Stok Eklemek:

Senaryo:

- Yeni bir ürünü stoka eklemek.
- Var olan bir ürünün stok miktarını artırmak.
- Geçerli olmayan verilerle bir ürün eklemeyi denemek (örneğin, negatif stok miktarı).

Sonuç:

- Stoğa ürünler doğru bir şekilde ekleniyor.
- Var olan ürünün miktarı ürün güncelleme fonksiyonu sayesinde istenen şekilde artırılabiliyor.
- Geçerli olmayan veriler kullanıldığında kullanıcı hata mesajı alıyor ve girdiği veriler kaydedilmiyor.

II. Stok Silmek:

Senaryo:

- Bir ürünü stoktan silmek.
- Var olmayan bir ürünü silmeye çalışmak.

Sonuç:

- Silinmek istenilen ürün gerekli parametreler girilerek stoktan silinebiliyor.
- Var olmayan ürün sorgulama fonksiyonu sayesinde tespit edilip kullanıcıya bu doğrultuda gerekli hata mesajı gösteriliyor.

III. Stok Sorgulama:

Senaryo:

- Belirli bir ürünün stok durumunu sorgulamak.
- Stokta olmayan bir ürünün sorgulanmaya çalışılması.

Sonuç:

- İstenilen parametreler girildiğinde istenilen ürünle ilgili sorgulama başarılı bir şekilde yapılabiliyor.
- Sorgulanan ürün stokta yoksa kullanıcıya bu doğrultuda hata mesajı gösteriliyor**.**
-

IV. Firma Maliyet Hesaplama:

Senaryo:

- Tüm stoktaki ürünlerin aylık toplam maliyetini hesaplamak.
- Tüm stoktaki ürünlerin yıllık toplam maliyetini hesaplamak.
- Geçerli olmayan bir tarih aralığıyla firma maliyet sorgulamayı denemek.

Sonuç:

- Ürünlerin aylık ve yıllık olarak gider hesaplamaları ve ayrıca firmanın diğer giderleriyle birlikte toplam aylık ve yıllık gider hesaplamaları doğru bir şekilde yapılıp veri tabanında tutuluyor.
- Geçerli olmayan tarih aralıklarında sorgulama yapıldığında kullanıcıya bir hata mesajı gösteriliyor.

V. Veri Doğruluğu:

Senaryo:

- Girdi formunda hatalı veya eksik veri girişi yapmak ve sistem davranışını kontrol etmek.
- Veri tabanındaki bilgilerin doğru bir şekilde güncellendiğini doğrulamak.

Sonuç:

- Kullanıcı yanlış formatta veriler girdiğinde hata mesajı ile karşılaşıyor ve girdiği bilgiler veri tabanına kaydedilmiyor.
- Girilen bilgiler veri tabanına doğru bir biçimde kaydediliyor.

10. Sözlük:

AsrStok: Uygulamanın adı.
Fatura no /Fatura Numarası: Her ürünün stok güncelleme tarihinde güncellenen faturasına ait benzersiz anahtar.
Ürün Kodu: Her ürün sahip olduğu benzersiz anahtar.
Login: Uygulamaya giriş yapma
StokControl: Stok işlemleri fonksiyonlarının bulunduğu sınıf.
db_connector: veri tabanı bağlantısı sağlayan sınıf.
Username: kullanıcı adı
Password: şifre
LogOut: uygulamadan çıkış yapma
Uyarı mesajı/hata mesajı: kullanıcıya ekranda gösterilen kısa mesajlar.
Label: ekranda yazı yazan bölümler.
Kategori id: Kategori kimliği
Admin: Tam yetkili kullanıcı
product:Ürün (tablo adı)
Outgoings: Giderler(tablo adı)
Bill: Fatura(tablo adı)
Category:Kategori(tablo adı)

11. Referanslar ve Kaynaklar:

https://miro.com/tr/diagramming/er-diagram/

https://drawsql.app

https://www.w3schools.com/sql/sql_datatypes.asp

https://www.lucidchart.com/pages/