# Stok Takip Programı

2023-2024

Veri Tabanı Yönetim Sistemleri Dersi Projesi

Aziz Sarıtaş

# Stok Takip Programı Giriş Ekranı

Giriş ekranında 3 seçeneğimiz var

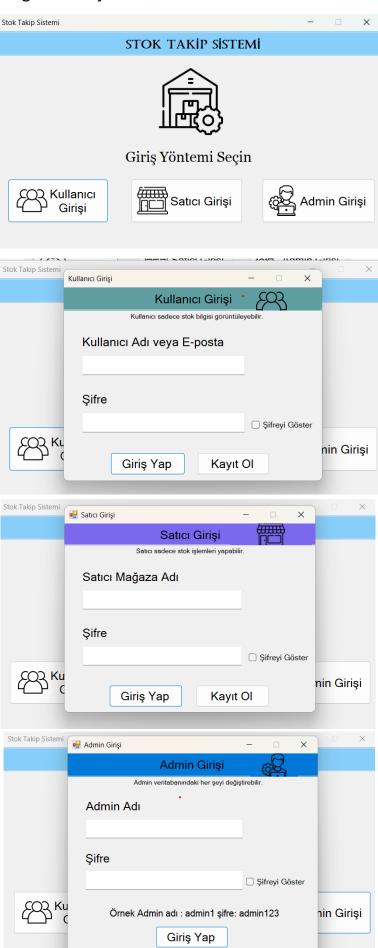
Kullanıcı Girişi: Sadece ürünleri görüntüleyebilir, ürün arayabilir veya ürün alabilir. Aldığı ürünler sipariş olarak kaydedilir.

Satıcı Girişi: Satıcı kendi ürünlerini görebilir, yeni ürün ekleyebilir, ürün silebilir, ürün arayabilir, ürün güncelleyebilir veya siparişleri görüntüleyip iptal edebilir.

Admin Girişi: Admin veri tabanındaki her şeyi değiştirebilir. Ürün, kullanıcı, satıcı ekleyebilir, güncelleyebilir, silebilir ,arayabilir.

Ayrıca ürün, kullanıcı, satıcı ve sipariş istatistiklerini görüntüleyebilir.

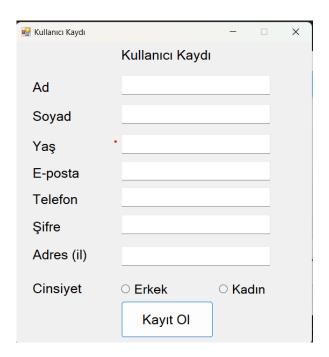
Trigger kullanılarak oluşturulmuş eklenen ve silinen ürün, kullanıcı, satıcı günlüklerine erişebilir.

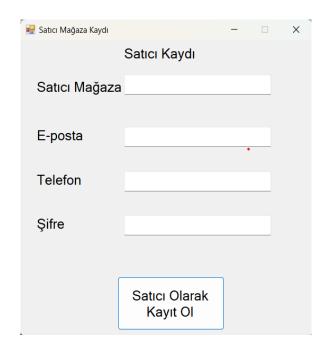


### Kullanıcı ve Satıcı Kaydı:

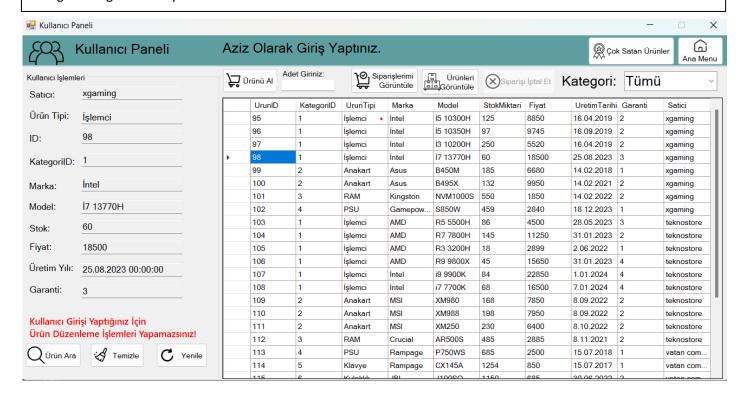
Kullanıcılar ve satıcıların sisteme giriş yapabilmesi için kaydolması gerekir.

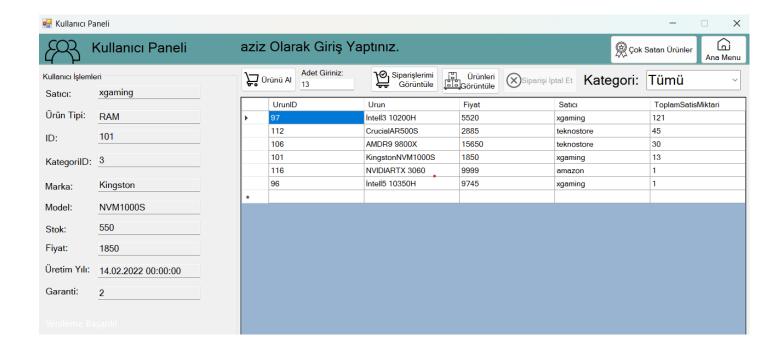
Kaydolurken yanlış değerler girilirse (yaş yerine isim yazmak gibi) program uyarı verip düzelme isteyecektir. Değerler doğru girilmediği sürece kayıt yaptırmayacaktır.





Kullanıcı Paneli. Bu kısımda kullanıcı ürünleri görebilir ve istediği üründen adet girerek Ürünü Al butonuna basarak ürün alabilir. Siparişlerimi Görüntüle Butonuna basarak aldığı ürünleri görebilir. Kategori menüsünden ürünleri kategorilere göre listeleyebilir.





Çok satan Ürünler Butonu: Bu kısımda *VIEW* kullanılarak en çok satan ürünler listelenmektedir. Yazılan view'in C# VE SQL kodları:

```
private void button7_Click(object sender, EventArgs e)
{
    //BU KISIMDA VİEW KULLANILARAK EN COK SATAN ÜRÜNLER GÖSTERİLİYOR
    butonviewDurum = true;
    baglanti.Open();
    SqlCommand sqlKomutu = new SqlCommand("SELECT * FROM EnCokSatanUrunler", baglanti); //encoksatanurunler viewi çalıştırılır
    SqlDataAdapter veriOkuyucu = new SqlDataAdapter(sqlKomutu);
    DataTable veriTablosu = new DataTable();
    veriOkuyucu.Fill(veriTablosu);
    dataGridView1.DataSource = veriTablosu; //en çok satan ürünler datagridview e yazdırılır
    sqlKomutu.ExecuteNonQuery();
    baglanti.Close();
}
```

```
SELECT TOP 10 -- "ur" id e göre group by yaptık ve "ur" fiyat satıcı adet görüntüledik UrunID,

MAX(Urun) AS Urun,

MIN(Fiyat) as Fiyat,

MAX(Satici) as Satıcı,

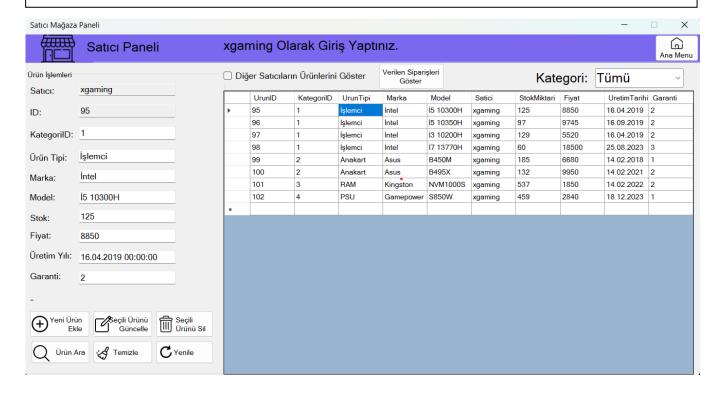
SUM(Adet) AS ToplamSatisMiktari

FROM Siparisler

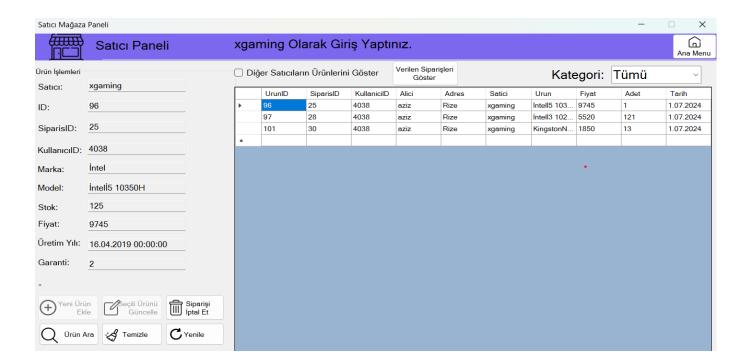
GROUP BY UrunID

ORDER BY ToplamSatisMiktari DESC;
```

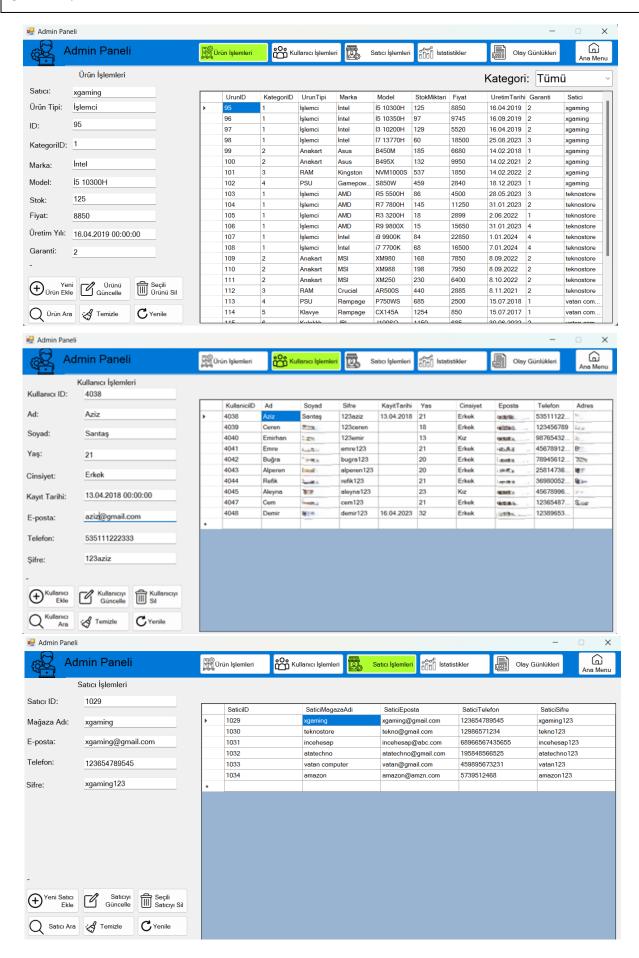
Satıcı Paneli: Satıcı burada kendi ürünlerini görüntüleyebilir ve düzenleyebilir. Yeni ürün ekleyebilir veya ürün silebilir. Satıcı kendisi dışındaki satıcıların ürünlerini de -Diğer Satıcıların Ürünlerini Göster- checkbox'ını seçerek görebilir fakat onların ürünlerini düzenleyemez.



Satıcı ayrıca Verilen Siparişleri Göster Butonuna tıklayarak kendisinden ürün alan kullanıcıların ürünlerini görebilir. Ve Siparişi İptal Et butonuna tıklayarak siparişleri iptal edebilir.



Admin Paneli: Admin burada ürün, kullanıcı, satıcı ekleyebilir, silebilir veya güncelleyebilir. Ayrıca Ürün, kullanıcı, satıcı ve sipariş İstatistiklerini görebilir. Triggerlar ile oluşturulan olay günlüklerini de görüntüleyebilir.

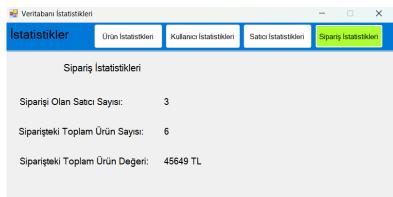


İstatistik Kısmı: Burada Ürün, Kullanıcı, Satıcı ve Sipariş İstatistikleri görüntülenebilir.



Sipariş İstatistikleri Kısmında **FONKSİYON** kullanıldı.

Fonksiyonların SQL Kodları:



```
|CREATE FUNCTION SiparisiOlanSaticiSayisi() --aktif siparisi olan satıcıları döndüren fonksiyon
RETURNS INT
BEGIN
    DECLARE @SiparistekiSaticilar INT --
    FROM Siparisler
    IF @SiparistekiSaticilar IS NULL --satici sayısı null ise 0 atar
    BEGIN
       SET @SiparistekiSaticilar = 0
    RETURN @SiparistekiSaticilar --satıcıların değerini döndürür
CREATE FUNCTION SiparisSayisi() --toplam siparis sayisini döndüren fonksiyon
RETURNS INT
BEGIN
    DECLARE @SiparisSayisi INT --int değişken atama
    SELECT @SiparisSayisi = Count(*) --siparişler tablosundaki öğelerin sayısını döndürür
    FROM Siparisler
    IF @SiparisSayisi IS NULL --değer null ise 0 atar
    BEGIN
       SET @SiparisSayisi = 0
    RETURN @SiparisSayisi --değişkeni döndürür
```

Olay Günlüğü: Bu kısımda <u>TRIGGER</u> kullanılarak eklenen ve silinen ürünler, kullanıcılar ve satıcılar görüntülenmektedir.



#### Kullanılan TRIGGER'ların SQL ve C# Kodları:

```
ALTER TRIGGER trg_Urunler_Insert --Urun tablosuna ürün eklendğinde devreye giren trigger
ON [dbo].[Urunler]
                                                   --ürün tablosunda çalışır
FOR INSERT
AS
BEGIN
         DECLARE @id INT
                                                      --değişkenler tanımlanır
         DECLARE @uruntipi NVARCHAR(50)
         DECLARE @marka NVARCHAR(50)
         DECLARE @model NVARCHAR(50)
         DECLARE @stokmiktari INT
         DECLARE @fiyat INT
        DECLARE @satici NVARCHAR(50)
        SELECT @id = UrunID,
                                                      --değişkenşer tablodaki değerlerle eşleştirilir
         @uruntipi = UrunTipi,
         @marka = Marka.
         @model = Model,
         @stokmiktari = StokMiktari.
         @fivat = Fivat.
         @satici = Satici
        FROM inserted --ekleme yapıldığı için geçici inserted kullanılır
         --parametrelere göre OlaylarUrun tablosuna ekleme yapılır
          \begin{tabular}{ll} INSERT INTO [dbo].[OlaylarUrun] VALUES ('ID: '+CAST(@id AS NVARCHAR(10)) + ' ' + @marka + ' ' + @model + ' ' + @uruntiping (and the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the context of the c
               ' Stok Miktarı: ' + CAST(@stokmiktarı AS NVARCHAR(10)) + ' Fiyat: ' + CAST(@fiyat AS NVARCHAR(10))
         + ' TL. Satıcı ' + @satici +' Tarafından '+ Cast(GETDATE() as nvarchar(25)) +' Tarihinde Eklendi.')
FND
∃ALTER TRIGGER trg_Urunler_Delete --ürün tablosundan ürün silindiğinde devreye giren trigger
  ON [dbo].[Urunler]
  FOR DELETE
 AS
BEGIN
         DECLARE @id INT
         DECLARE @uruntipi NVARCHAR(50)
         DECLARE @marka NVARCHAR(50)
         DECLARE @model NVARCHAR(50)
         DECLARE @stokmiktari INT
         DECLARE @fivat INT
         DECLARE @satici NVARCHAR(50)
         --değişkenler tanımlanır
         SELECT @id = UrunID, @uruntipi = UrunTipi, @marka = Marka, @model = Model, @stokmiktari = StokMiktari, @fiyat = Fiyat, @satici = Satici
         FROM deleted --silinme işlemi yapıldığı için geçici deleted tablosu kullanıldı
          --OlaylarUrun tablosuna silinen ürünler eklenir
         INSERT INTO [dbo].[OlaylarUrun] VALUES ('ID: '+CAST(@id AS NVARCHAR(10)) + ' ' + @marka + ' ' + @model + ' ' + @uruntipi
                Stok Miktarı:
                                             ' + CAST(@stokmiktari AS NVARCHAR(10)) + ' Fiyat: ' + CAST(@fiyat AS NVARCHAR(10))
                TL. Satici '+ @satici +' Tarafından '+ Cast(GETDATE() as nvarchar(25)) +' Tarihinde Silindi.')
 END
```

```
CREATE TRIGGER trg Kullanicilar Insert --kullanıcılar tablosuna kullanıcı eklendiğinde devreye giren trigger
ON Kullanicilar
FOR THSERT
BEGIN
    DECLARE @id INT
    DECLARE @ad NVARCHAR(50)
    DECLARE @soyad NVARCHAR(50)
    DECLARE @cinsiyet NVARCHAR(50)
    DECLARE @yas INT
    DECLARE @eposta NVARCHAR(80)
    SELECT @id = KullaniciID, @ad = Ad, @soyad = Soyad, @cinsiyet = Cinsiyet, @yas = Yas, @eposta = Eposta FROM inserted
    INSERT INTO OlaylarKullanici VALUES ('ID: '+CAST(@id AS NVARCHAR(10)) + ' ' + @ad + ' ' + @soyad + ' ' + @cinsiyet
     + CAST(@yas AS NVARCHAR(10)) + 'E-Posta: ' + @eposta+ ' Kullanıcısı '+ Cast(GETDATE() as nvarchar(25)) + 'Tarihinde Eklendi.')
END
ALTER TRIGGER trg_Kullanicilar_Delete --kullanıcılar tablosundan kullanıcı silindiğinde devreye giren trigger
ON Kullanicilar
FOR DELETE
ΔS
BEGIN
    DECLARE @id INT
    DECLARE @ad NVARCHAR(50)
    DECLARE @soyad NVARCHAR(50)
    DECLARE @cinsiyet NVARCHAR(50)
    DECLARE @yas INT
    DECLARE @eposta NVARCHAR(80)
    SELECT @id = KullaniciID, @ad = Ad, @soyad = Soyad, @cinsiyet = Cinsiyet, @yas = Yas, @eposta = Eposta FROM deleted
    INSERT INTO OlaylarKullanici VALUES ('ID: '+CAST(@id AS NVARCHAR(10)) + ' ' + @ad + ' ' + @soyad + ' ' + @cinsiyet
    + CAST(@yas AS NVARCHAR(10)) + 'E-Posta: ' + @eposta+ ' Kullanıcısı '+ Cast(GETDATE() as nvarchar(25)) + 'Tarihinde Silindi.')
ALTER TRIGGER trg_Satici_Insert --satıcı tablosuna satıcı eklendiğinde devreye giren trigger
ON Saticilar
FOR THISFRT
AS
BEGIN
    DECLARE @id INT
    DECLARE @saticiad NVARCHAR(50)
    DECLARE @saticieposta NVARCHAR(80)
    DECLARE @saticitelefon VARCHAR(50)
    SELECT @id = SaticiID, @saticiad = SaticiMagazaAdi, @saticieposta = SaticiEposta, @saticitelefon = SaticiTelefon FROM inserted
    INSERT INTO OlaylarSatici VALUES ('ID: '+CAST(@id AS NVARCHAR(10)) + ' ' + @saticiad + ' E-Posta: ' + @saticieposta + ' Telefon: '
     + CAST(@saticitelefon AS NVARCHAR(10)) +' Saticisi '+ Cast(GETDATE() as nvarchar(25)) +' Tarihinde Eklendi.')
FND
CREATE TRIGGER trg_Satici_Delete --saticilar tablosundan satici silindiğinde devreye giren trigger
ON Saticilar
FOR DELETE
AS
≒BEGIN
    DECLARE @id INT
    DECLARE @saticiad NVARCHAR(50)
    DECLARE @saticieposta NVARCHAR(80)
    DECLARE @saticitelefon VARCHAR(50)
    SELECT @id = SaticiID, @saticiad = SaticiMagazaAdi, @saticieposta = SaticiEposta, @saticitelefon = SaticiTelefon FROM deleted
    INSERT INTO OlaylarSatici VALUES ('ID: '+CAST(@id AS NVARCHAR(10)) + ' ' + @saticiad + ' E-Posta: ' + @saticieposta + ' Telefon: '
    + CAST(@saticitelefon AS NVARCHAR(10)) +' Saticisi '+ Cast(GETDATE() as nvarchar(25)) +' Tarihinde Silindi.')
FND
          rivate void button1_Click(object sender, EventArgs e)
             //eklenen ürünler
             //TRIGGER KULLANILARAK OLUŞTURULAN OLAYLAR'URUN TABLOSUNDAN EKLENEN ÜRÜNLER LİSTELENİR
             SqlCommand sqlKomutu = new SqlCommand("SELECT * FROM OlaylarUrun Where Durum LIKE '%Eklendi.'", baglanti);
             SqlDataAdapter veriOkuyucu = new SqlDataAdapter(sqlKomutu);
             DataTable veriTablosu = new DataTable();
             veri0kuyucu.Fill(veriTablosu);
             dataGridView1.DataSource = veriTablosu;
```

Kullanılan **STORED PROCEDURE**'lar: SP'lar ürün, kullanıcı, satıcı eklerken ve güncellerken kullanıldı.

```
∃CREATE PROCEDURE up_UrunEkle --Ürün eklemek içi stored procedure
    @ukid INT,
                             --ürünlerin değişken atamaları yapılır
    @utip NVARCHAR(50),
    @umarka NVARCHAR(50),
    @umodel NVARCHAR(50),
    @ustok INT,
    @ufiyat INT,
    @uuretim DATETIME,
     @ugaranti INT,
     @usatici NVARCHAR(50)
AS
BEGIN --insert into ile değişkenler Urunler tablosuna eklenir
    INSERT INTO Urunler(KategoriID, UrunTipi, Marka, Model, StokMiktari, Fiyat, UretimTarihi, Garanti, Satici)
     VALUES (@ukid, @utip, @umarka, @umodel, @ustok, @ufiyat, @uuretim, @ugaranti, @usatici)
END
_CREATE PROCEDURE up_UrunGuncelle --Ürün güncellemek için stored procedure
     @uid INT,
                                 --değişkneler atanır
     @ukid INT,
     @utip NVARCHAR(50),
     @umarka NVARCHAR(50),
     @umodel NVARCHAR(50),
     @ustok INT,
     @ufiyat INT,
     @uuretim DATETIME,
     @ugaranti INT
 AS
≐BEGIN
     UPDATE Urunler --update ile değişkenlerin yeni değerleri ile Urunler tablosunda güncelleme yapılır
         KategoriID = @ukid,
        UrunTipi = @utip,
        Marka = @umarka,
        Model = @umodel,
        StokMiktari = @ustok,
        Fiyat = @ufiyat,
        UretimTarihi = @uuretim,
         Garanti = @ugaranti
     WHERE UrunID = @uid; --ürün ide göre işlemler yapılır
 END
CREATE PROCEDURE up_KullaniciEkle --Kullanıcı eklemek için stored procedure
     @kad NVARCHAR(50),
                                      --değişkenler tanımlanır
     @ksoyad NVARCHAR(50),
     @ksifre NVARCHAR(50),
     @kkayit DATETIME,
     @kyas INT,
     @kcinsiyet NVARCHAR(10),
     @keposta NVARCHAR(50),
     @ktelefon NVARCHAR(20)
 AS
                                --insert into ile kullanıcılara eklenir
BEGIN
     INSERT INTO Kullanicilar(Ad, Soyad, Sifre, KayitTarihi, Yas, Cinsiyet, Eposta, Telefon)
     VALUES (@kad, @ksoyad, @ksifre, @kkayit, @kyas, @kcinsiyet, @keposta, @ktelefon)
 END
```

```
CREATE PROCEDURE up_KullaniciGuncelle --Kullanıcı güncellemek için stored procedure
     @kid INT,
     @kad NVARCHAR(50),
     @ksoyad NVARCHAR(50),
     @ksifre NVARCHAR(50),
     @kkayit DATETIME,
     @kyas INT,
     @kcinsiyet NVARCHAR(10),
     @keposta NVARCHAR(50),
     @ktelefon NVARCHAR(20)
 AS
⊟BEGIN
     UPDATE Kullanicilar --update ile kullanicilar tablosu güncellenir
     SET
         Ad = @kad,
         Soyad = @ksoyad,
         Sifre = @ksifre,
         KayitTarihi = @kkayit,
         Yas = @kyas,
         Cinsiyet = @kcinsiyet,
         Eposta = @keposta,
         Telefon = @ktelefon
     WHERE KullaniciID = @kid; --işlemler kullanıcı ide göre yapılır
 END
CREATE PROCEDURE up_SaticiEkle --Satıcı eklemek için stored procedure
     @sad NVARCHAR(50),
     @seposta NVARCHAR(50),
     @stelefon NVARCHAR(20),
     @sifre NVARCHAR(50)
 AS
BEGIN
                     --saticilar tablosuna ekleme yapılır
     INSERT INTO Saticilar(SaticiMagazaAdi, SaticiEposta, SaticiTelefon, SaticiSifre)
     VALUES (@sad, @seposta, @stelefon, @sifre)
 END
GREATE PROCEDURE up_SaticiGuncelle --Satıcı Güncellemek için stored procedure
     @sid INT,
     @sad NVARCHAR(50),
     @seposta NVARCHAR(50),
     @stelefon NVARCHAR(20),
     @sifre NVARCHAR(50)
 AS
BEGIN
     UPDATE Saticilar --update ile satıcılar tablosu güncellenir
     SET
         SaticiMagazaAdi = @sad,
         SaticiEposta = @seposta,
         SaticiTelefon = @stelefon,
         SaticiSifre = @sifre
     WHERE SaticiID = @sid;
 END
```

#### STORED PROCEDURE C# Kodları

```
SqlCommand sqlEklemeKomutu = new SqlCommand("up_UrunEkle",baglanti); //ürün ekleme stored procedure
sqlEklemeKomutu.CommandType = CommandType.StoredProcedure;
 //textbox değerleri özel karakter boşluk vs. değerlere karşı kontrol edilerek ekleme yapılır
if (!(string.IsNullOrEmpty(textBox9.Text)) && (textBox2.Text.All(char.IsDigit)) && (textBox5.Text.All(char.IsDigit))
     && (textBox6.Text.All(char.IsDigit))
      && (textBox8.Text.All(char.IsDigit))&&(DateTime.TryParse(textBox7.Text,out DateTime result)))
      baglanti.Open();
      //storede procedure'larda tanımlanan parametrelere göre ürünler eklenir
      sqlEklemeKomutu.Parameters.AddWithValue("@ukid", textBox2.Text);
     sqlEklemeKomutu.Parameters.AddWithValue("@utip", textBox10.Text);
sqlEklemeKomutu.Parameters.AddWithValue("@umarka", textBox3.Text);
sqlEklemeKomutu.Parameters.AddWithValue("@umarka", textBox4.Text);
     sqlEklemeKomutu.Parameters.AddWithValue("@ustok", Convert.ToInt32(textBox5.Text));
sqlEklemeKomutu.Parameters.AddWithValue("@ufiyat", Convert.ToInt32(textBox6.Text));
sqlEklemeKomutu.Parameters.AddWithValue("@uretim", Convert.ToDateTime(textBox7.Text));
      sqlEklemeKomutu.Parameters.AddWithValue("@ugaranti", Convert.ToInt32(textBox8.Text));
      sqlEklemeKomutu.Parameters.AddWithValue("@usatici", textBox9.Text);
      sqlEklemeKomutu.ExecuteNonQuery();
      label13.Text = "Ürün Ekleme Başarılı! Yenile Butonuna Basınız.";
//Ürün güncelle stored procedure ile ürün güncelleme yapılır
SqlCommand sqlEklemeKomutu = new SqlCommand("up_UrunGuncelle", baglanti); //stored procedure çağrılır
sqlEklemeKomutu.CommandType = CommandType.StoredProcedure;
 //textbox değerleri kontrol edilir
if (!(string.IsNullOrEmpty(textBox9.Text)) && (textBox2.Text.All(char.IsDigit))&& (textBox5.Text.All(char.IsDigit))
     & (textBox6.Text.All(char.IsDigit))
     && (textBox8.Text.All(char.IsDigit)) && (DateTime.TryParse(textBox7.Text, out DateTime result)))
     baglanti.Open();
     sqlEklemeKomutu.Parameters.AddWithValue("@uid", textBox1.Text);
     sqlEklemeKomutu.Parameters.AddWithValue("@ukid", textBox2.Text);
     sqlEklemeKomutu.Parameters.AddWithValue("@utip", textBox10.Text);
sqlEklemeKomutu.Parameters.AddWithValue("@umarka", textBox3.Text);
sqlEklemeKomutu.Parameters.AddWithValue("@umodel", textBox4.Text);
     sqlEklemeKomutu.Parameters.AddWithValue("@ustok", Convert.ToInt32(textBox5.Text));
sqlEklemeKomutu.Parameters.AddWithValue("@ufiyat", Convert.ToInt32(textBox6.Text));
sqlEklemeKomutu.Parameters.AddWithValue("@uuretim", Convert.ToDateTime(textBox7.Text));
sqlEklemeKomutu.Parameters.AddWithValue("@ugaranti", Convert.ToInt32(textBox8.Text));
     sqlEklemeKomutu.ExecuteNonQuery();
     label13.Text = "Güncelleme Başarılı! Yenile Butonuna Basınız.";
 //stored procedure ile kullanıcı eklem
SqlCommand sqlEklemeKomutu = new SqlCommand("up_KullaniciEkle", baglanti); //kullanıcı ekle stored procedure çalıştırılır
sqlEklemeKomutu.CommandType = CommandType.StoredProcedure;
//textbox değerleriin boş dolu olup olmadığı kontrol edilir
bool bosDurumu = (bosMuDoluMu(textBox10.Text, textBox11.Text, textBox17.Text, textBox14.Text, textBox12.Text,
     textBox13.Text, textBox15.Text,textBox16.Text));
if (bosDurumu && (textBox12.Text.All(char.IsDigit)) && (DateTime.TryParse(textBox14.Text, out DateTime result))
     &&(textBox16.Text.All(char.IsDigit)))
     baglanti.Open();
     //parameterelere göre kullanıcı tablosuna kullanıcı eklenir
     sqlEklemeKomutu.Parameters.AddWithValue("@kad", textBox10.Text.Trim());
     sqlEklemeKomutu.Parameters.AddWithValue("@ksoyad", textBox11.Text.Trim());
sqlEklemeKomutu.Parameters.AddWithValue("@ksifre", textBox17.Text.Trim());
sqlEklemeKomutu.Parameters.AddWithValue("@ksifre", textBox17.Text.Trim());
sqlEklemeKomutu.Parameters.AddWithValue("@kyayit", Convert.ToDateTime(textBox14.Text));
sqlEklemeKomutu.Parameters.AddWithValue("@kyayit", Convert.ToDataTime(textBox12.Text));
     sqlEklemeKomutu.Parameters.AddWithValue("@kcinsiyet", textBox13.Text.Trim());
sqlEklemeKomutu.Parameters.AddWithValue("@keposta", textBox15.Text.Trim());
sqlEklemeKomutu.Parameters.AddWithValue("@keposta", textBox16.Text.Trim());
     sqlEklemeKomutu.ExecuteNonQuery();
     label24.Text = "Kullanıcı Ekleme Başarılı! Yenile Butonuna Basınız.";
```

```
//stored procedure ile kullanıcı güncelleme
SqlCommand sqlEklemeKomutu = new SqlCommand("up_KullaniciGuncelle", baglanti);//kullanici güncelle stored procedure
sqlEklemeKomutu.CommandType = CommandType.StoredProcedure;
bool bosDurumu = (bosMuDoluMu(textBox10.Text, textBox11.Text, textBox17.Text, textBox14.Text, textBox12.Text,
     textBox13.Text, textBox15.Text, textBox16.Text));
if (bosDurumu && (textBox12.Text.All(char.IsDigit)) && (DateTime.TryParse(textBox14.Text, out DateTime result))
    && (textBox16.Text.All(char.IsDigit)))
    baglanti.Open();
     //parametrelere göre kullanıcı tablosundaki kullanıcılar güncellenir
     sqlEklemeKomutu.Parameters.AddWithValue("@kid", textBox9.Text.Trim());
    sqlEklemeKomutu.Parameters.AddWithValue("@kad", textBox10.Text.Trim());
    sqlEklemeKomutu.Parameters.AddWithValue("@ksoyad", textBox11.Text.Trim());
sqlEklemeKomutu.Parameters.AddWithValue("@ksifre", textBox17.Text.Trim());
sqlEklemeKomutu.Parameters.AddWithValue("@kkayit", Convert.ToDateTime(textBox14.Text));
    sqlEklemeKomutu.Parameters.AddWithValue("@kyas", Convert.ToInt32(textBox12.Text));
sqlEklemeKomutu.Parameters.AddWithValue("@kcinsiyet", textBox13.Text.Trim());
    sqlEklemeKomutu.Parameters.AddWithValue("@keposta", textBox15.Text.Trim()); sqlEklemeKomutu.Parameters.AddWithValue("@ktelefon", textBox16.Text.Trim());
     sqlEklemeKomutu.ExecuteNonQuery();
     label24.Text = "Güncelleme Başarılı! Yenile Butonuna Basınız.";
//stored procedure ile satıcı eklenir
```

```
//stored procedure ile satici eklenir
SqlCommand sqlEklemeKomutu = new SqlCommand("up_SaticiEkle", baglanti); //satici ekle stored procedure
sqlEklemeKomutu.CommandType = CommandType.StoredProcedure;

//textboxx değerleri kontrol edilir
bool bosDurumu = (bosMuDoluMu(textBox25.Text, textBox24.Text, textBox23.Text, textBox22.Text));

if (bosDurumu && (textBox23.Text.All(char.IsDigit)))
{
   baglanti.Open();

   //parametrelere göre saticilar tablosuna saticilar eklenir

   sqlEklemeKomutu.Parameters.AddWithValue("@sad", textBox25.Text);
   sqlEklemeKomutu.Parameters.AddWithValue("@seposta", textBox24.Text);
   sqlEklemeKomutu.Parameters.AddWithValue("@stelefon", textBox23.Text);
   sqlEklemeKomutu.Parameters.AddWithValue("@sifre", textBox22.Text);

   sqlEklemeKomutu.ExecuteNonQuery();
   label25.Text = "Satici Ekleme Başarılı! Yenile Butonuna Basınız.";
```

```
//stored procedure ile satici güncelleme
SqlCommand sqlEklemeKomutu = new SqlCommand("up_SaticiGuncelle", baglanti);//satici güncelle stored procedure
sqlEklemeKomutu.CommandType = CommandType.StoredProcedure;

//textboxlara girilen değerlerin kotnrolü yapılır
bool bosDurumu = (bosMuDoluMu(textBox25.Text, textBox24.Text, textBox23.Text, textBox22.Text));

if (bosDurumu && (textBox23.Text.All(char.IsDigit)))
{
   baglanti.Open();

   //parametrelere göre satıcılar listesindeki satıcıalr güncellenir
   sqlEklemeKomutu.Parameters.AddWithValue("@sid", textBox26.Text);
   sqlEklemeKomutu.Parameters.AddWithValue("@sad", textBox25.Text);
   sqlEklemeKomutu.Parameters.AddWithValue("@seposta", textBox24.Text);
   sqlEklemeKomutu.Parameters.AddWithValue("@stelefon", textBox23.Text);
   sqlEklemeKomutu.Parameters.AddWithValue("@sifre", textBox22.Text);
   sqlEklemeKomutu.ExecuteNonQuery();
   label25.Text = "Güncelleme Başarılı! Yenile Butonuna Basınız.";
```

## DATABASE DİYAGRAMI

Diyagramda Kategoriler tablosu ile ürünler tablosu birbirine KategorilD değeri üzerinden bağlı.

Siparisler tablosu ile ürünler tablosu birbirine UrunID değeri üzerinden bağlı.

Satıcılar tablosu ile kullanıcılar tablosu birbirine KullanıcıID değeri üzerinden bağlı.

