**T.C. Düzce Üniversitesi**

**Mühendislik Fakültesi**

**Bilgisayar Mühendisliği (İ.Ö.)**

**VTYS Proje Ödevi**

**Öğrencilerin Adı Soyadı : Hasan ŞAHİN (161002011)**

**Ödev Tarihi : 27.12.2018**

İÇİNDEKİLER

1.GİRİŞ 3

2.PROJE 2

2.1.ER Diagramı 2

2.2.Database Diagramı 3

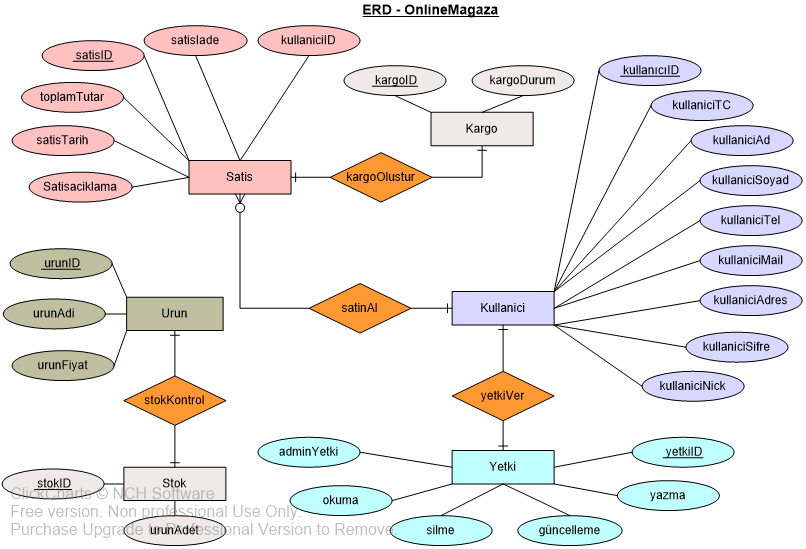
2.3.Database Oluşturma Kodları3

2.4. Triggers ve Procedures Kodları5

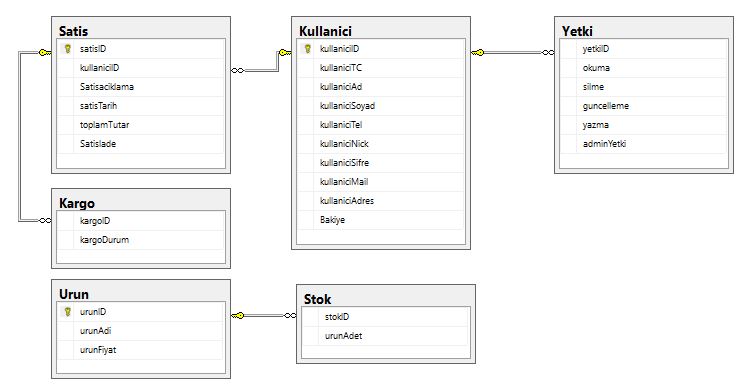
3.Projenin Çalışma Aşamaları ve İlgili Kodları 8

1. **Giriş**

Proje konusu Online Alışveriş Mağazası için veri tabanı tasarlamak ve uygulamak. Bunu yaparken de proje kapsamında istenen maddeleri yerine getirmek. Proje ara yüzü C# ile tasarlanmıştır. Veri tabanı için SQL Server 2014, Rapor kısmı için FastRaport eklentisi kullanılmıştır.

1. **Proje**
   1. **ER Diagramı**

Resim 2.1 – ER Diagramı

* 1. **Database Diagramı**

Resim 2.2 – Database Diagram

* 1. **Database Oluşturma Kodları**

|  |  |
| --- | --- |
| İlk olarak Database’ i oluşturuyoruz ve oluşturulan Database aktif hale getiriyoruz. | CREATE DATABASE onlinemagaza  use onlinemagaza |

6 adet tablo tasarlanmıştır. Bunlar sırasıyla Urun, Stok, Kullanici, Yetki, Kargo, Satis ve tablolarıdır. Sırasıyla tabloların içeriklerini inceleyecek olursak.

|  |  |
| --- | --- |
| **Urun;** Urun bilgilerinin olduğu tablodur. Tablonun sırasıyla sütunları urunID, urunAdi ve urunFiyat’dır. Bu bilgiler satış yaparken kullanılmaktadır. | CREATE TABLE Urun (  urunID int identity(1,1) primary key not null,  urunAdi varchar(20) not null,  urunFiyat money  ) |
| **Stok;** Ürünlerin stokta olup olmadığının kontrolü için eklenmiş bir tablodur. Urun tablosuna bağlı bir tablodur. Tablonun sırasıyla sütunları stokID ve urunAdet’dir. Urun tablosundaki değişlik olduğunda stok tablosuna da otomatik olarak değişiklikler yansıtılmaktadır. | CREATE TABLE Stok(  stokID int foreign key references Urun(urunID) on update cascade on delete cascade not null,  urunAdet int  ) |
| **Kullanici;** Kullanıcı bilgilerinin saklandığı tablodur. Tablonun sırasıyla sütunları kullaniciID, kullaniciTC, kullaniciAd, kullaniciSoyad, kullaniciTel, kullaniciNick, kullaniciSifre, kullaniciMail, kullaniciAdres ve bakiye’dir. Kullanıcı kayıt olurken bu bilgileri sisteme verir ve bu bilgiler kayıt olur. Sonrasında yapacağı işlemler için bu bilgileri kullanması gerekmektedir. Kullanıcı ID kısmına yetkilendirme özellikleri verilmektedir. | CREATE TABLE Kullanici(  kullaniciID int identity(1,1) primary key not null,  kullaniciTC varchar(11) UNIQUE,  kullaniciAd varchar(20),  kullaniciSoyad varchar(20),  kullaniciTel varchar(11) UNIQUE,  kullaniciNick varchar(30),  kullaniciSifre varchar(30),  kullaniciMail varchar(30) UNIQUE,  kullaniciAdres varchar(60),  Bakiye money default(0)  ) |
| **Yetki;** Kullanıcılara verilen ve admin yetkilendirmenin bulunduğu tablodur. Tablonun sırasıyla sütunları YetkiID, yazma, güncelleme, silme, okuma ve adminYetki’dir. AdminYetki sahip kişi yada kişiler diğer kullanicilara yetki verebilir ve yetkilerini alabilirler. Kullanici tablosu ile Yetki tablosu FK bağlantıyla bağlanmıştır. | CREATE TABLE Yetki(  yetkiID int foreign key references Kullanici(kullaniciID) on update cascade on delete cascade not null,  okuma bit default(0),  silme bit default(0),  guncelleme bit default(0),  yazma bit default(0),  adminYetki bit default(0),  ) |
| **Satis;** Yapılan satışların takiplerini ve gerekli bilgileri tutan tablodur. Tablonun sırasıyla sütunları satisID, kullaniciID, Satisaciklama, satisTarih, toplamTutar ve satisIade’dir. Satis tablosu ile Kullanici tablosu FK bağlantıyla bağlanmıştır. Satın alma yapan müşterinin bilgilerine erişim sağlanır | CREATE TABLE Satis(  satisID int identity(1,1) primary key not null,  kullaniciID int foreign key references Kullanici(kullaniciID) on update cascade on delete cascade not null,  Satisaciklama varchar(120),  satisTarih datetime default(getdate()),  toplamTutar money,  SatisIade varchar(200)  ) |
| **Kargo;** Satışı gerçekleşen ürün yada ürünlerin kullanıcıya kargo edilmesi için gerekli bilgilerin saklandığı tablodur. Tablonun sırasıyla sütunları kargoID ve kargoDurum’dur. Kargo tablosu ile Satis tablosu FK bağlantıyla bağlanmıştır. | CREATE TABLE Kargo(  kargoID int foreign key references Satis(satisID) on update cascade on delete cascade not null,  kargoDurum varchar(40)  ) |
| **Silinmiş Ürünler;** Trigger ile silinen ürünleri içerisinde saklayan tabolumuz. | create table SilinmisUrunVerileri(  urunID int not null,  urunAD varchar(50),  urunFiyat money,  urunStok int  ) |

**Sonradan eklenmiş olan birkaç Constraint.**

|  |  |
| --- | --- |
| **Kullanici** Tablosunun Telefon numarası girişi esnasında 11 haneli olması ve 0 ile başlamasını Check eden bir **Constraint** tanımladık.  **Kullanici** Tablosunda Mail girişi esnasında @ kullanılan bir Check eden bir **Constraint** tanımladık.  **Kullanici** Tablosunda TC girişi esnasında girilen TC’nin 11 haneli olmasını ve son hanesinin çift rakam olmasını Check eden bir **Constraint** tanımladık  **Satis** Tablosunda Satış yapıldığı esnada mevcut saat dilimine eşit olmasını Check eden bir **Constraint** tanımladık.  **Satis** Tablosunda Satış yapıldığı esnada toplam tutar sıfırdan büyük bir değer olmasını Check eden bir **Constraint** tanımladık.  **Stok** Tablosunda Urun adetlerinin sıfırın altına düşmemesini Check eden bir **Constraint** tanımladık.  **Urun** Tablosunda Urun fiyatının sıfırdan büyük olması gerektiğini Check eden bir **Constraint** tanımladık. | ALTER TABLE Kullanici ADD Constraint ck\_kullaniTel check (kullaniciTel like '0[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')  ALTER TABLE Kullanici ADD Constraint ck\_kullaniciMail check (kullaniciMail like ('%@%'))  ALTER TABLE Kullanici ADD Constraint ck\_kullaniTc Check(kullaniciTC LIKE '[^0][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][02468]')  ALTER TABLE Satis ADD Constraint ck\_satisTarih check(satisTarih =getdate())  ALTER TABLE Satis ADD Constraint ck\_toplamTutar check(toplamTutar > 0)  ALTER TABLE Stok ADD Constraint ck\_urunAdet check(urunAdet >= 0)  ALTER TABLE Urun ADD Constraint ck\_urunFiyat check(urunFiyat > 0) |

* 1. **Triggers Procedures ve İndex Kodları**

|  |  |
| --- | --- |
| **InsertYetkiKul Prosedür** Database üzerinde herhangi bir kullanıcı eklenmesi durumunda eklenmiş olan kullanıcının bilgilerini alıp o kullanıcının verilerinin kayıt edilmesini ve ilgili tablolara da veri girişi yapılmasını sağlamaktadır. Örneğin yeni bir kullanıcı kayıt olduğunda otomatik olarak bakiyesi sıfır olarak başlamaktadır ve yetkilendirme tablosunda hiçbir yetkisi olmadan kaydı tamamlanmaktadır. | CREATE procedure InsertYetkiKul  @kultc char(11),  @kulad varchar(15),  @kulsoyad varchar(15),  @kultel char(11),  @kulnick varchar(10),  @kulsifre varchar(10),  @kulmail varchar(50)  AS  declare @deger int  declare @string varchar(20) ='NULL'  declare @mony money=0.00  insert into Kullanici values(@kultc,@kulad,@kulsoyad, @kultel,@kulnick,@kulsifre,@kulmail,  @string,@mony)  select @deger=[kullaniciID] from Kullanici order by [kullaniciID] asc  GO |
| **Urunislemleri Prosedür;** Ürün hakkında güncelleme yapıldığında ilgili tablolara ulaşıp yapılan değişiklikleri oralarda da güncelleyen bir işlem yapmaktadır | create procedure urunislemleri  @urnid int,@urunfiyat money,@urunadet int  as  update urun set urunFiyat=@urunfiyat where urunID=@urnid  update stok set urunAdet=@urunadet where stokID=@urnid  go |
| **UrunEkle Prosedür;** Urun ekleme yapıldığında eklenen ürünün Urun’ler tablosuna ve Stok tablosuna eklenmesi işlemini yapmaktadır. | create procedure UrunEkle  @urunAd varchar(15),@stok int,@fiyat money  as  declare @id tinyint  insert into Urun values(@urunAd,@fiyat)  select @id=urunID from Urun order by urunID asc  insert into stok values(@id,@stok)  go |
| **Splitstring Fonksiyonu;** Ekstra olarak tasarlanmış bir Prosedür ve ilgili prosedürün tetiklediği bir fonksiyon tasarlanmıştır.  Buradaki fonksiyonun amacı gönderilen parametreleri parçalamak ve anlamlı hale getirmektir çalışma mantığı şu şekildedir. Sayı, virgül, sayı, virgül (1,2,3,2) ve bu şekilde devam etmektedir. İlk sayı urunID’ yi ikinci sayı ise urunAdet’ i temsil etmektedir. Bu şekilde indexleri tek olanlar urunID’ ler çift olanlar ise urunAdetlerini temsil etmektedir. Buna göre satın alınan ürünlerin ID lerine ve adetlerine erişme şansımız bulunmaktadır. Ekstra tablo  **Uruniade Prosedür;**İade edilmek istenen ürünün ne olduğunu ve kaç adet olduğunu fonksiyon yardımıyla bulan prosedürümüz stokta güncellemesini yapmaktadır. Bu durumda iade edilen ürün geri stokta gözükmektedir. | CREATE FUNCTION dbo.splitstring ( @stringToSplit VARCHAR(MAX) )  RETURNS  @returnList TABLE ([Name] [nvarchar] (500))  AS  BEGIN  DECLARE @name NVARCHAR(255)  DECLARE @pos INT  WHILE CHARINDEX(',', @stringToSplit) > 0  BEGIN  SELECT @pos = CHARINDEX(',', @stringToSplit)  SELECT @name = SUBSTRING(@stringToSplit, 1, @pos-1)INSERT INTO @returnList  SELECT @name  SELECT @stringToSplit = SUBSTRING(@stringToSplit, @pos+1, LEN(@stringToSplit)-@pos)  END  INSERT INTO @returnList  SELECT @stringToSplit  RETURN  END  Create proc uruniade  @degisken varchar(255),@satisID int  as  DECLARE @MyCursor CURSOR;DECLARE @urunID int;  DECLARE @urunAdet int;  BEGIN SET @MyCursor = CURSOR FOR  SELECT \* FROM dbo.splitstring(@degisken)  OPEN @MyCursor  WHILE @@FETCH\_STATUS = 0 BEGIN  print(@urunID) FETCH NEXT FROM @MyCursor  INTO @urunID FETCH NEXT FROM @MyCursor  INTO @urunAdet  update Stok set urunAdet=urunAdet+@urunAdet where stokID=@urunID  END;  update kargo set kargoDurum='Kargo iptal Edilmiştir.' where kargoID=@satisID  CLOSE @MyCursor ;DEALLOCATE @MyCursor;END; |
| **Urunsil Triggers;** Urun tablosundan urun silme işlemi gerçekleşmesi durumu için yazılmış bir trigger. Silindikten sonrada verileri deleted tablosunda saklar. | Create trigger [dbo].[Urunsil]  on [dbo].[Urun]  instead of delete  as  declare @urunStok int; declare @urnid int  select @urnid=urunID from deleted  select @urunStok= urunadet from Stok  delete from Stok where stokID=@urnid  declare @urunAd varchar(50)  declare @urunFiyat money  select @urunAd= urunAdi from deleted  select @urunFiyat = urunFiyat from deleted  insert into [Silinmis Urun Verileri] values(@urnid,@urunAd,@urunFiyat,@urunStok)  insert into [SilinmisUrunVerileri] values(@urnid,@urunAd, @urunFiyat,@urunStok)  Delete from Urun where urunID=@urnid  Delete from Stok where stokID=@urnid  GO |
| **Kulsil Triggers;** Kullanici tablosundan kullanıcı silinmesi gerçekleşmesi durumu için yazılmış bir trigger. Kullanıcı silindiğinde verileri deleted tablosunda saklar. | Create TRIGGER kulsil  ON kullanici  instead of delete  as  declare @id int  select @id=kullaniciID from deleted  delete from yetki where yetkiID=@id  go |
| **Kullaniciac Triggers;** Yeni kullanıcı oluşturulduğunda kullanıcının ID’sini inserted tablosunda saklayan ve yetki tablosuna insert yapan bir trigger. | CREATE trigger kullaniciac  on kullanici  after insert  as  declare @id int  select @id=KullaniciID from inserted  insert into Yetki values(@id,0,0,0,0,0)  GO |
| **İndex;** Arama yapılması esnasında daha hızlı sonuç almamıza yardım sağlayan index kodları. | CREATE CLUSTERED INDEX İndex2 ON kullanici(kullaniciAd);  CREATE NONCLUSTERED INDEX İndex1 ON Kargo(kargoID);  CREATE UNIQUE INDEX İndex2 ON Satis (satisID); |

1. **Projenin Çalışma Aşamaları ve İlgili Kodları**



Resim 3.1 – Giriş Ekranı

**Giriş Ekranında Gerçekleşen İlgili Kodlar**

**SQL Bağlantı Kodları**

SqlConnection conn2 = new SqlConnection("Data Source=(local);Initial Catalog=onlinemagaza;Integrated Security=True;Persist Security Info=True;User ID=hasan;Encrypt=True;TrustServerCertificate=True;Context Connection=False");

SqlConnection conn3 = new SqlConnection("Data Source=(local);Initial Catalog=onlinemagaza;Integrated Security=True;Persist Security Info=True;User ID=hasan;Encrypt=True;TrustServerCertificate=True;Context Connection=False");

**Açıklama:** Database ile bağlantı kurup kontrollü erişim sağlıyoruz.

**SQL Sorgu Fonksiyonları**

**Griddoldur() Fonksiyonu**

public void griddoldur(string kosul = "", int s = 0)

{SqlCommand kmt;

try{conn.Close(); dt.Clear(); conn.Open();

if (kosul == ""){kmt = new SqlCommand("Select s.satisID,s.Satisaciklama, CONVERT (VARCHAR(8), s.satisTarih, 4),s.toplamTutar,k.kargoDurum,s.SatisIade from Satis s join Kargo k on s.satisID=k.kargoID where s.kullaniciID=" + kullaniciid + "", conn);

}

else if (kosul != "" && s == 1) { kmt = new SqlCommand("Select s.satisID, s.Satisaciklama,CONVERT(VARCHAR(8), s.satisTarih, 4),s.toplamTutar,k.kargoDurum, s.SatisIade from Satis s join Kargo k on s.satisID=k.kargoID where s.satisID=" + kosul + " and s.kullaniciID=" + kullaniciid + "", conn);

}

else{ kmt = new SqlCommand("Select s.satisID,s.Satisaciklama,CONVERT(VARCHAR(8), s.satisTarih, 4),s.toplamTutar,k.kargoDurum,s.SatisIade from Satis s join Kargo k on s.satisID=k.kargoID where s.satisTarih='" + kosul + "' and s.kullaniciID=" + kullaniciid + "", conn);

}

SqlDataAdapter da = new SqlDataAdapter(kmt); da.Fill(dt);

dataGridView1.DataSource = dt; dataGridView1.Columns[5].Visible = false; conn.Close();

System.Windows.Forms.DataGridViewCellStyle dataGridViewCellStyle1 = new System.Windows.Forms.DataGridViewCellStyle();

dataGridViewCellStyle1.ForeColor = Color.Black;

dataGridView1.DefaultCellStyle = dataGridViewCellStyle1;

}

**Giriş Yap SQL Balantı Kodları.**

conn2.Open();

SqlCommand kmt33 = new SqlCommand("select kullaniciID from Kullanici where kullaniciNick='" + textBox8.Text.Trim() + "' and kullaniciSifre='" + textBox9.Text.Trim() + "'", conn2);

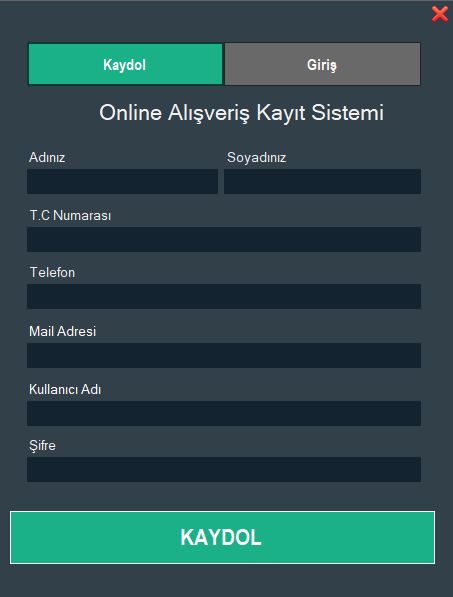
SqlDataReader dr = kmt33.ExecuteReader();

if (dr.Read()){int id = Convert.ToInt16(dr["kullaniciID"].ToString()); conn2.Close();

AnaSayfa ana = new AnaSayfa();ana.kullaniciid = id; ana.Show(); this.Hide();}

else{conn2.Close(); MessageBox.Show("Kullanıcı Adı veya Şifre Hatalı!", "UYARI!", MessageBoxButtons.OK, MessageBoxIcon.Stop); }

**Açıklama:** Kullanıcının Nick ve Şifresi veri tabanında ilgili bilgiler ile eşleşiyor mu kontrolü yapmaktadır. Eşleşme başarılı ise kullanıcıID ile birlikte ana sayfaya yönlendirilmektedir.



Resim 3.2 – Kaydol Ekranı

**Kaydol Ekranında Gerçekleşen İlgili Kodlar**

**Kaydol SQL Bağlantı Kodları**

conn2.Open();

SqlCommand kmt33 = new SqlCommand("select kullaniciNick from Kullanici where kullaniciNick='" + textBox6.Text.Trim() + "'", conn2);

SqlDataReader dr = kmt33.ExecuteReader();

if (dr.Read()){MessageBox.Show("Kullanıcı Mevcut!", "Dikkat", MessageBoxButtons.OK, MessageBoxIcon.Warning); }

**Açıklama:** İlk olarak kullanıcı adı mevcut mu kontrolü yapılıyor. Mevcut ise kullanıcı uyarı mesajı ile karşılaşıyor.

conn3.Open();

string insertYetki = "exec InsertYetkiKul '"+textBox3.Text+"','"+textBox2.Text+"', '"+textBox4.Text+"','"+textBox1.Text+"','"+textBox6.Text+"','"+textBox7.Text+ "','" + textBox5.Text + "'";

SqlCommand kmt55 = new SqlCommand(insertYetki,conn3);

kmt55.ExecuteNonQuery();

conn3.Close();

**Açıklama:** Programda çalışan ilk prosedür SQL bağlantısı komutu. Burada InsertYetkiKul prosedürü çalıştırılıyor. İlgili parametreler gönderiliyor ve prosedür aktif hale getirilerek programda yapması gerek ilgili doldurmaları ve değer atamalarını yapıyor.



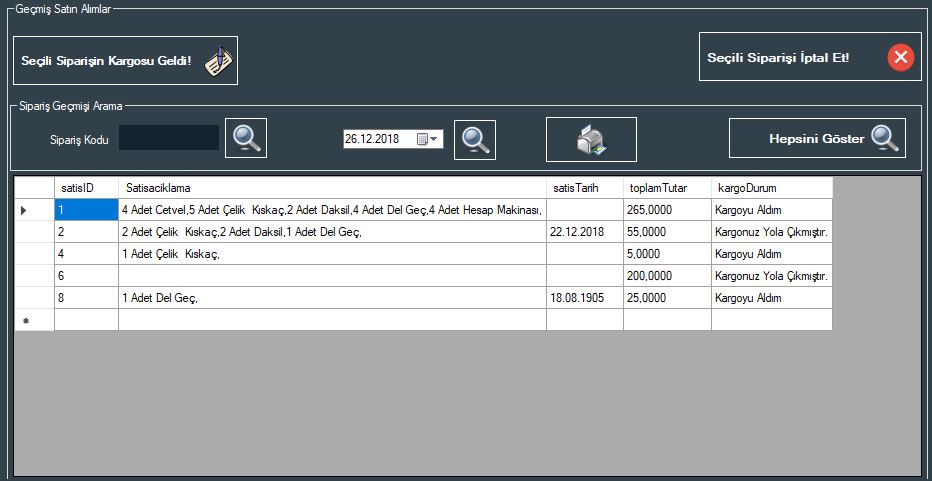
Resim 3.3 – Müşteri Mağaza Paneli

**Müşteri Mağaza Paneli Ekranında Gerçekleşen İlgili Kodlar**

|  |  |
| --- | --- |
| Hesap işlemleri butonuna basıldığında gerçekleşen olaylar.  **Otomatik veriler doldurulur.**  conn.Close();  conn.Open();  SqlCommand kmt2 = new SqlCommand( "Select \* from Kullanici where kullaniciID=kullaniciid ", conn);  SqlDataReader dr2 = kmt2.ExecuteReader();  if (dr2.Read()){label86.Text = dr2["Bakiye"].ToString();  textBox1.Text = dr2["kullaniciAd"].ToString();  textBox2.Text = dr2["kullaniciSoyad"].ToString();  textBox3.Text = dr2["kullaniciSifre"].ToString();  textBox6.Text = dr2["kullaniciMail"].ToString();  textBox4.Text = dr2["kullaniciAdres"].ToString();}  conn.Close();  **Güncelleme yapılmak istenirse.**  conn.Close();  conn.Open();  SqlCommand kmt2 = new SqlCommand( "update Kullanici set kullaniciAd='" + textBox1.Text + "',kullaniciSoyad='" + textBox2.Text + "',kullaniciSifre='" + textBox3.Text + "',kullaniciMail='" + textBox6.Text + "',kullaniciAdres='" + textBox4.Text + "' where kullaniciID=" + kullaniciid + "", conn);  kmt2.ExecuteNonQuery();  conn.Close(); MessageBox.Show("Başarıyla Güncelleştirildi"); | C:\Users\hamdi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\hesapislemleri.jpg  Resim 3.4 – Hesap İşlemleri Butonu  Resim 3.5 – Hesabım Paneli |
| Bakiye eklemek için butona basıldığında gerçekleşen olaylar. Bakiye **Ekle butonuna basıldığında.**  conn.Open();  SqlCommand kmt2 = new SqlCommand( "update Kullanici set Bakiye=Bakiye+ "+Convert.ToInt16(txtBakiye.Text)+" where kullaniciID=" + kullaniciid + "", conn);  kmt2.ExecuteNonQuery();  conn.Close();  textBox12.Text = "";  textBox13.Text = "";  textBox14.Text = "";  textBox15.Text = "";  textBox16.Text = "";  doldur(); | C:\Users\hamdi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\bakiyeButon.jpg  Resim 3.6 – Bakiye  Resim 3.7 – Bakiye Ekle Paneli |



Resim 3.8 – Satın Alma Geçmişi



Resim 3.9 – Geçmiş Satın Alımlar Paneli



Resim 3.9.1 – Sipariş Geçmişi Arama

Satın alma geçmişi butonuna basıldığında gerçekleşen olaylar. Ve geçmiş aramalara basıldığında gerçekleşen olaylar.

conn.Close(); dt.Clear(); conn.Open();

if (kosul == ""){kmt = new SqlCommand("Select s.satisID,s.Satisaciklama,s.satisTarih,

s.toplamTutar,k.kargoDurum,s.SatisIade from Satis s join Kargo k on s.satisID= k.kargoID where s.kullaniciID="+kullaniciid+"", conn); }

else if (kosul != "" && s == 1){kmt = new SqlCommand("Select s.satisID,s.Satisaciklama,s.satisTarih,s.toplamTutar,k.kargoDurum,s.SatisIade from Satis s join Kargo k on s.satisID=k.kargoID where s.satisID=" + kosul + " and s.kullaniciID=" + kullaniciid + "", conn); }

else{kmt =new SqlCommand("Select s.satisID,s.Satisaciklama,s.satisTarih,s.toplamTutar, k.kargoDurum,s.SatisIade from Satis s join Kargo k on s.satisID=k.kargoID where s.satisTarih=" + kosul + " and s.kullaniciID=" + kullaniciid + "", conn); }SqlDataAdapter da = new SqlDataAdapter (kmt);da.Fill(dt);

dataGridView1.DataSource = dt; dataGridView1.Columns[5].Visible=false;

conn.Close();

**Açıklama:** Geçmişte arama yapılırken mevcut kodların bulunduğu yer bi fonksiyondur. Fonksiyona gönderilen değerler üzerinden gerekli SQL satırları çalışmaktadır. Örneğin sipariş koduna göre arama yapıyorsa sipariş kodu ile ilgili programcının belirlediği kod gönderilir onun üzerinden ilgili SQL kodu ilgili sonuçları ekrana verir.

C:\Users\hamdi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\seciliKargoGeldi.jpg

Resim 3.9.2 – Seçili Siparişin Kargosu Geldi

DialogResult dialogResult = MessageBox.Show("Kargonuzun Geldiğini Onaylamak İstediğinize Eminmisiniz?", "Kargom Geldi!", MessageBoxButtons.YesNo);

if (dialogResult == DialogResult.Yes) {

conn.Open();

SqlCommand kmt2 = new SqlCommand("update Kargo set kargoDurum='Kargoyu Aldım' where kargoID=" + satisid + "", conn);

kmt2.ExecuteNonQuery();

griddoldur();

conn.Close(); }

**Açıklama:** Siparişi gelen müşteri geldiğine dair onaylama işlemi yaptığında gerçekleşen olaylar.

C:\Users\hamdi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\seciliKargoIptal.jpg

Resim 3.9.3 – Seçili Siparişi İptal Et

if (satisid != 0){

DialogResult dialogResult = MessageBox.Show("Satın Almayı İptal Etmek İstediğinize Eminmisiniz", "Satın Alma İptal", MessageBoxButtons.YesNo);

if (dialogResult == DialogResult.Yes) {

conn.Close();

conn.Open();

SqlCommand kmt2 = new SqlCommand("exec uruniade '"+iadeidler+"','"+satisid+"' ", conn); // prosedüre gidicek idler(iadeidler) değişkeninde.

kmt2.ExecuteNonQuery();

conn.Close();

button6.PerformClick();

magazaolustur();

MessageBox.Show("islem tamam");}

**Açıklama:** Seçili sipariş iptal edildiğinde uruniade prosedürünü aktif hale getiriyoruz ve ilgili kod satırlarını çalıştırarak gerekli işlemleri yapıyoruz.



Resim 3.10 Satın Al

int satisid=1; string aciklama=""; string idler="";

if (Convert.ToDouble(label86.Text)>=Convert.ToDouble(Toplamtutar.Text)){

for (int i = 0; i < urunid.Count; i++){

if (adet[i] > 0) {aciklama += adet[i].ToString() + " Adet " + urunAdi[i] + ",";

idler += urunid[i] + "," + adet[i] + ",";

conn.Open();

SqlCommand kmt6 = new SqlCommand("update Stok set urunAdet=urunAdet-" + adet[i] + " where stokID=(select urunID from Urun where urunAdi='" + urunAdi[i] + "')", conn);

kmt6.ExecuteNonQuery();

conn.Close();

//-------------------------- acıklama ve id kısmı------------------------} }

conn.Close();

conn.Open();

SqlCommand kmt2 = new SqlCommand("select top 1 satisID from Satis order by satisID desc", conn);

SqlDataReader dr2 = kmt2.ExecuteReader();

if (dr2.Read()){satisid = Convert.ToInt16(dr2["satisID"]) + 1; }

conn.Close();

//-------------------------- kargo oluşturuldu------------------------

DateTime dt = new DateTime();dt = DateTime.Now;

conn.Open();

SqlCommand kmt4 = new SqlCommand("insert into Satis values(" + kullaniciid + ",'" + aciklama + "'," + dt.Month + "+" + dt.Day + "+" + dt.Year + "," + Convert.ToInt16(Toplamtutar.Text) + ",'" + idler + "')", conn);

kmt4.ExecuteNonQuery();

conn.Close();

conn.Close();conn.Open();

SqlCommand kmt3 = new SqlCommand("insert into Kargo values(" + satisid + ",'Kargonuz Yola Çıkmıştır.')", conn);

kmt3.ExecuteNonQuery();

conn.Close();

//-------------------------- sipariş oluşturuldu------------------------

conn.Open();

SqlCommand kmt5 = new SqlCommand("update Kullanici set Bakiye=Bakiye-" + Convert.ToDouble(Toplamtutar.Text) + " where kullaniciID=" + kullaniciid + "", conn);

kmt5.ExecuteNonQuery();

conn.Close();

//-------------------------- bakiye güncellendi------------------------

label86.Text = (Convert.ToDouble(label86.Text)-Convert.ToDouble(Toplamtutar.Text)).ToString();

Toplamtutar.Text = "0";griddoldur();magazaolustur();

MessageBox.Show("Satın Alım Başarılı.Ürününüz Yolda!");

}

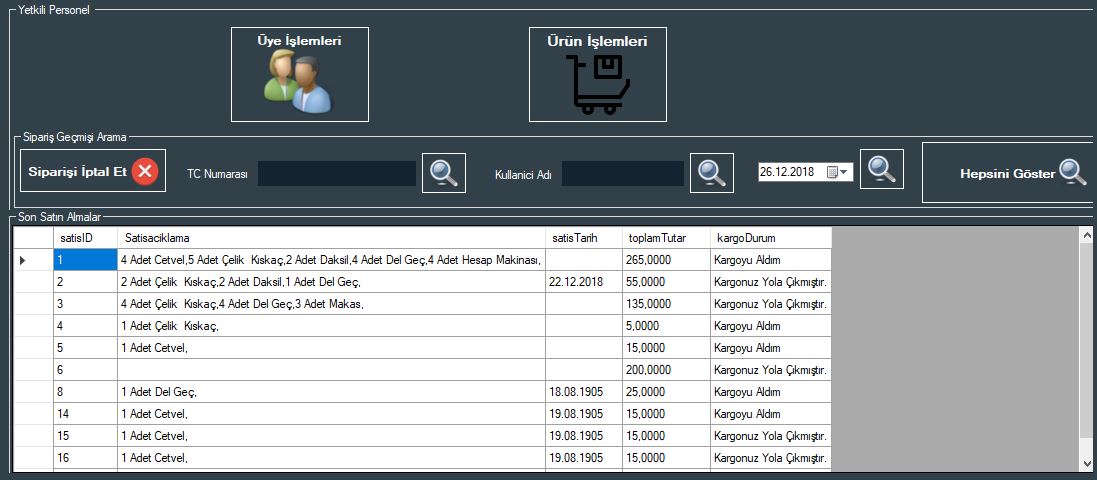
**Açıklama**: Önce bakiye kontrolü yaptıktan sonra Stok güncellenir, Kargo oluşturulur, Sipariş oluşturulur ve son bakiye güncellenir.



Resim 3.11 Yetkili Girişi Yapılırsa Giriş Sayfası



Resim 3.11.1 Yetkili Panel



Resim 3.11.1.1

|  |  |
| --- | --- |
| **SQL bağlantı kodları**  public SqlConnection conn = new SqlConnection("Data Source=(local);Initial Catalog=onlinemagaza;Integrated Security=True;Persist Security Info=True;User ID=hasan;Encrypt=True;TrustServerCertificate=True;Context Connection=False");  public SqlDataAdapter da;  public DataTable dt = new DataTable(); public int  i,id, urunid, kullaniciid;public string nick;  **Güncelleme SQL kodları**  conn.Close();  conn.Open();  SqlCommand kmt = new SqlCommand("update Yetki set okuma='"+checkBox1.Checked+"',silme='"+ checkBox3.Checked+"',yazma='"+checkBox4.Checked+"',guncelleme='"+checkBox2.Checked+"',adminyetki='"+checkBox5.Checked+"' where yetkiID="+id+"", conn); kmt.ExecuteNonQuery();  **Hesabı Sil SQL kodları**  conn.Close();  conn.Open();  SqlCommand kmt = new SqlCommand("delete from Kullanici where id=" + id + "", conn);  kmt.ExecuteNonQuery();  **Açıklama:** Kullanıcılara kullanıcı adı veya telefon numarası ile bilgilerine erişip. Gerekli yetkilendirmeleri Check etme yardımıyla kullanıcıları yetkilendirebiliriz. İlgili kullanıcının hesabını silebilir. | Resim 3.11.1.1.1 Uye İslemleri  Resim 3.11.1.1.2 Uye İslemleri Paneli |
| C:\Users\hamdi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\urunISlem.jpg  Resim 3.11.1.2.1 Ürün İşlemleri  C:\Users\hamdi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\yetkiliPanelUrunIslem.jpg | |

Resim 3.11.1.2.2 Urun Ekleme Paneli

|  |  |
| --- | --- |
| **Resim Seç:** Dosya açma tanımlayıp ilgili resmi seçeriz. | OpenFileDialog file =new OpenFileDialog();  file.ShowDialog();  pictureBox1.Image=mage.FromFile(file.FileName); |
| **Ürün Ekle:** İlgili parametreleri prosedüre gönderiyoruz. Urunekle prosedürü gerekli işlemleri program içerisinde yerine getirmektedir. Yeni eklenen ürün programın mağaza sayfasına da eklenmektedir. | string xsltFile = Application.StartupPath + @"\\" + textBox5.Text + ".jpg"; pictureBox1.Image.Save(xsltFile,ImageFormat.Jpeg); conn.Open();  string veri = "exec urunekle '" + textBox5.Text + "','" + textBox3.Text + "','" + textBox2.Text + "'"; SqlCommand kmt2 = new SqlCommand(veri, conn); kmt2.ExecuteNonQuery(); conn.Close();  MessageBox.Show("Ürün başarıyla eklendi."); |
| **Toplu Veri Ekleme:** Toplu veri eklemek için C#’ın kendi kütüphanelerini kullandık. Readcell özelliği ile cell de bulunan verileri return ederek kullanmamızı sağlıyor. Bu aldığımız verileri de toplu olarak stok update yapmamızı salıyor. | OpenFileDialog dosya2 = new OpenFileDialog();  dosya2.Filter = "Excell Dosyası | \*.xlsx; | Tüm Dosyalar | \*.\* "; dosya2.Title = "Toplu Stok İşlemi - Excell Dosyası Seçin!";  dosya2.ShowDialog();  string Dosyayolu = dosya2.FileName;  excell excel = new excell(@"" + Dosyayolu + "", 1);  for (int i = 1; i < 100; i++){  try{string urunadi = excel.ReadCell(i, 0);  int adet = Convert.ToInt16(excel.ReadCell(i, 1));  conn.Open();  SqlCommand kmt2 = new SqlCommand("update Stok set urunAdet=urunAdet+" + adet + " where stokID =(Select urunID from Urun where urunAdi='" + urunadi + "')", conn);  kmt2.ExecuteNonQuery(); conn.Close();  }catch (Exception) { button8.PerformClick();break; }} |
| **Veri İmport Etme:** Önce bir excel nesnesi oluşturuluyor. Object missing ile Workbook veriliyor. Daha sonra datagridviewde ki verileri teker teker Rows[].Cells[] şeklinde çekerek excel’e aktarıyor. | Microsoft.Office.Interop.Excel.Application exel1 = new Microsoft.Office.Interop.Excel.Application();  exel1.Visible = true;  object missing = Type.Missing;  Workbook wb = exel1.Workbooks.Add(missing);  Worksheet ws = (Worksheet)wb.Sheets[1];  int baslamakolonu = 1; int baslamasatiri = 1;  for (int j = 0; j < dataGridView1.Columns.Count; j++){  Range myRange = (Range)ws.Cells[baslamasatiri, baslamakolonu + j];  myRange.Value2 = dataGridView1.Columns[j].HeaderText;  }baslamasatiri++;  for (int i = 0; i < dataGridView1.Rows.Count; i++)  {for (int j = 0; j < dataGridView1.Columns.Count; j++)  {Range myRange = (Range)ws.Cells[baslamasatiri + i, baslamakolonu + j];  myRange.Value2 = dataGridView1[j, i].Value == null ? "" : dataGridView1[j, i].Value; myRange.Select();}} |
| **Hepsini Göster:** Bütün ürünlerin görüntülenmesine ve stok kontrolünün yapılmasına imkan sağlayan bir fonksiyon tasarlanmıştır. | public void doldur(string kosul="")  {SqlCommand kmt;  try{conn.Close(); dt.Clear(); conn.Open();  if(kosul==""){kmt = new SqlCommand("Select u.urunID,u.urunAdi,u.urunFiyat,s.urunAdet from Urun u join Stok s on u.urunID=s.stokID order by s.urunAdet asc", conn);  }  else{kmt = new SqlCommand("Select u.urunID,  u.urunAdi ,u.urunFiyat,s.urunAdet from Urun u join Stok s on u.urunID=s.stokID where u.urunAdi=  '"+kosul+"' order by s.urunAdet asc", conn);  }  SqlDataAdapter da = new SqlDataAdapter(kmt);  da.Fill(dataset, "dataset"); da.Fill(dt);  dataGridView1.DataSource = dt; conn.Close();  dataGridView1.Columns[0].Visible = false;  System.Windows.Forms.DataGridViewCellStyle dataGridViewCellStyle1 = new System.Windows.Forms.DataGridViewCellStyle();  dataGridViewCellStyle1.ForeColor = Color.Black;  dataGridView1.DefaultCellStyle = dataGridViewCellStyle1; }} |
| **Ürün Güncelle:** urunislemleri prosedürü sayesinde ilgili güncellemeler gerçekleşmektedir. Parametreler programdan ilgili yerlerden çekilmektedir. | conn.Open(); string veri = "exec urunislemleri '"+urnid+"','"+textBox1.Text+ "','" + textBox4.Text + "'";  SqlCommand kmt2 = new SqlCommand(veri, conn);  kmt2.ExecuteNonQuery(); conn.Close();  ana.magazaolustur();  MessageBox.Show("Ürün Başarıyla güncelleştirildi"); |
| **Sil:** Silinmesini istediğimiz ürünün kaldırılması ile ilgili SQL kodları | conn.Open();  SqlCommand kmt2 = new SqlCommand("delete from Urun where urunID=" + urunid + "", conn);  kmt2.ExecuteNonQuery();  conn.Close();  MessageBox.Show("Ürün Başarıyla Silindi"); |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |



Resim 3.11.2 Yedek Al Yedek Dön Paneli

|  |  |
| --- | --- |
| **Yedek Al:** Butona basıldığında .bak uzantılı olan backup dosyamızı dosya konumunu belirttiğimiz yere kayıt ediyoruz.  **Yedekten Dön:** Butona basıldığında .bak uzantılı olan backup dosyamızı dosya konumundan seçiyor tekrardan SQL’e import ediyoruz.  Sonrasında ilk olarak eski veri tabanımızı kapatıyoruz sonrasında backup edeceğimiz dosyamızı import ediyoruz. | SaveFileDialog save = new SaveFileDialog();  save.Filter = "Sql Backup Dosyası|\*.bak";  save.OverwritePrompt = true;  save.ShowDialog(); conn.Open();  SqlCommand backupKod = new SqlCommand( "BACKUP DATABASE [onlinemagaza] TO DISK = '" + save.FileName + "' WITH INIT ", conn); backupKod.ExecuteNonQuery(); conn.Close();  OpenFileDialog dosya2=new OpenFileDialog();  dosya2.Title = "Toplu Yedek İşlemi - Backup Dosyası Seçin!"; dosya2.ShowDialog();  string Dosyayolu = dosya2.FileName;  conn.Open();  SqlCommand backupDon = new SqlCommand("USE [master] ALTER DATABASE[onlinemagaza] SET offline with rollback immediate RESTORE DATABASE [onlinemagaza] FROM DISK = N'" + Dosyayolu + "' WITH REPLACE alter database [onlinemagaza] set online with rollback immediate ", conn); backupDon.ExecuteNonQuery();  conn.Close(); |