DBCONTEXT İŞLEMLERİ

* EntityFramework C# Windows Form Projesinde SqlServer ile Database bağlantısı için kullanılan bir class oluşturuldu. Bu işlem Projenin Ağ üzerindeki bir Database yada Localde bulunan database için gerçekleştirilebilir ve İnsert, Update, Delete işlemlerini Database içerisinde StoredProcedure yazmadan yapmaya imkan sağlamakta.
* Public class olarak AppDbContext : DbContext oluşturulup EntityFrameworkte veritabanı işlemlerini gerçekleştirmek için kullanılacak bir sınıf oluşturuldu.
* Form üzerinde İnsert işlemi yapmak için oluşturulacak planda öncelikle İnsert işlemini databasede yapacağım tablo için bir class oluşturuyorum bu classta public değişkenler bulunmakta ve oluşturulan bu class ın EntityFramework olabilmesi için class ı public şekilde tanımlıyorum
* İnsert işleminin bu class üzerinden geçmesi değişkenlerin veri tiplerinin uyuşması ve doğrulanması için DataContext içerisine DbSet oluşturuyorum bu DbSetin oluşma şekli
  + Public DbSet<clsPrinter> Printer {get; set;}
* clsPrinter dan sonra gelen Printer Kodu Name i Databasedeki Tablonuzun adı ile aynı olmalıdır.
* Şu an DbContext içerisinde eksik olan kısım ise Database bağlantımız bu bağlantıyı şu şekilde yapıyoruz
  + Protected override void OnConfiguring ( DbContextOptionsBuilder optionBuilder) { optionsBuilder.UserSqlServer(“Veritabanı Bağlantı Texti “) }
* İnsert işlemini gerçekleştirdiğimizde Class ve Tablo DbContext içerisinde tanımlı olduğu için işlem DbContext içerisinden geçmektedir bu sebeple Veritabanı bağlantısını Program kendi gerçekleştirmekte.

İnsert işleminden önce kayıt yaparken içerisinden veri alacağımız Combobox’ı doldurmaya

* Formun .cs kısmında ayrıca bir private void oluşturup combobox doldurma işlemini burda yapıp Form\_Load olayını eklicez.
* Void içerisinde Database işlemleri için using oluşturuyoruz
  + Using ( var context = new AppDbContext())
  + {
    - Combobox1.DisplayMember = “Text olarak gürünücek Kolon”;
    - Combobox1.ValueMember = “Arka planda çalışacak olak ID”;
    - Comobox1.DataSource = context .Stok.ToList();
  + }
* Yukarıdaki using işleminin içerisinde DbContext classımızı tanımladık sonrasında Combobox için Arkaplan ve Arayüzde görünmesi gereken kolonları tanımladık Data Soruce kısmında ise DbContext içerisinde tanımladığımız Tablo içerisindeki satırlara yönelik classlar oluşturup bağladığımız DbContextimizin ismini yazıyoruz ( Database Tablosu ile aynı name de olaması gereken kısım)
* Son olarak içerisinde using ile işlem yaptığımız Private voidimizi Form\_Load olayına ekliyoruz ve Combobx’ımıza veri çekmiş oluyoruz.

Şimdi Insert işlemine başlayabiliriz.

Insert işlemimizde 2 textboxtan ve 1 de comboxtan veri çekip İnsert edicez. Bu işlem için tabiki öncelikle Buton Click olayına using ile yukarıdaki gibi DbContext sınıfımızı tanımlıyoruz.

Using içerisine sonrak işlem olarak Kayıt yapacağımız ve DbContext ile bağladığımız Database tablomuzdaki Kolonlarla aynı olan değişkenleri barındıran classımızı tanımlıyoruz.

Using ( var context = new AppDbContext)

{

clsPrinter p = new clsPrinter

{

Marka = txtMarka.Text ,

Model = txtModel.Text,

TonerID = ConvertToInt32(cmbToner.SelectedValue)

}

}

context.Printer.Add(p);

context.SaveChanges();

Combobox işleminde neden using içerisinde veri çekilicek tablonun classı olduğu halde bu classı tanımlamadık diyebilirsiniz. Class içerisinde Combobox işlemi yapılammakta Insert, Update veya Delete işlemi değil veri çekme işlemi olduğu için.

Dikkat etmeniz gereken nokta using içerisindeki class içerisinde işlem yaparken ( , ) kullanmanız gerekmektedir. Class içerisinden çıkıp using içerisindeki Add işlemi ise bu kod satırından önceki yapılan işlemleri database tablosuna kaydetmenizi sağlar. Sonraki SaveChanges ise yaptığınız bu işlemlerin bağlantısını kurup arakdaki prosedürleri doğrular ve kaydetmenizi onaylar.