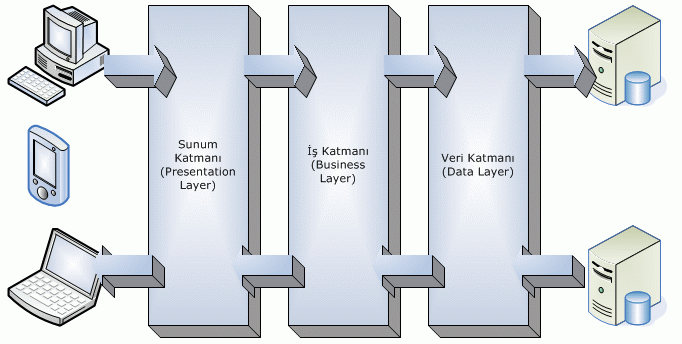
**KATMANLI MİMARİ NEDİR?**

Günümüzde yapılan ve geliştirilen projelere sürekli kullanıcı ve yöneticilerin istekleri dur durak bilmeden devam eder iken yeni versiyonlar çıkıyor ve bunlar zaman içerisinde yapılan projeleri gün geçtikçe daha karmaşık hale getiriyor. Projenin karmaşıklığı arttığı taktirde projeye yeni katılan birisi için veya sunum yaparken anlatımı ve anlaşılırlığı azalmaktadır. Bu olayı çözmek için yapılan denemeler ve çözüm yolları sonunda katmanlı mimari ortaya çıkmış ve günümüzde birden farklı platform ve dil için kullanılmaktadır. Katmanlı mimari yaptığımız projeyi düzene koyarak geliştirmemizi sağlayan, kod karmaşıklığını gözle görülebilir bir ölçüde azaltan ve yapılan hataları daha kolay bir şekilde bulmamıza yarar.

Katmanlı mimari ile beraber veriye erişme, verinin nasıl kullanılacağını, nasıl değişimler yapılabileceğini ve kullanıcıya nasıl bir gösterim sunulabileceğini çok iyi bir şekilde yönetmemize yardımcı oluyor. Katmanlı mimari sayesinde bu yapıyı parçalara ayırarak bu işlemlerin daha iyi yönetilebilmesini sağlıyoruz.

Temelde 3 katman olmak üzere 4 katman veya daha fazla katman üzerinden çalışabileceğiniz bir ortam yaratabilirsiniz. Bu katmanlar Data Layer, Business Layer ve Presentation Layer olmak üzere 3 temel katmandan oluşmaktadır. Bu katmanları aşağıdaki görsel ile anlatıp daha detaylı bir şekilde göz atalım.



**Data Access Layer**

Genel DAL olarak kısaltılmaktadır. DAL katmanında genellikle veri silme, güncelleme, ekleme ve veri çekme işlemleri yapılan katmandır. Kısacası bu katmanda sadece veritabanı ile ilgili işlemler yapılır.

**Business Layer**

Business Layer katmanını BL olarak kısaltılıyor. Bu katmanın aslında asıl amacı kodların direkt olarak Data Access Layer’a erişmesini engellemektir. Bu katmanda ayrıca hangi kimlerin erişmesi gerektiğini ve Validation kontrollerinizi yapıp Presentation ve Data Access Layer’ları arasında bir geçiş katmanı olarak kullanılabilir.

**Presentation Layer**

Presentation Layer diğer adı ile sunum katmanıdır. Bu katman direkt olarak kullanıcının gördüğü ve kullandığı katmandır. Sunum katmanı form ekranları, web siteleri veya konsol ekranları olabilir. Bu katmanda kullanıcın girdiği veriler BL katmanına oradan da Access Data Katmanına iletilir. Kullanıcı bir bilgi görmek ister ise database’den istenen bilgiler Datagrid, Listview veya ComboBox gibi değer tutan nesneler içerisine aktarılır. Ve böylelikle kullanıcı istediği bilgiyi ekranda görebilir.

**Entities**

Yukarıda anlattığımız katmanlar olması gerekenler ve temel katmanlardır. Siz istediğiniz kadar katmanları proje ve programlarınıza göre arttırabilirsiniz. Entity katmanı ise genellikle DTO olarak da adlandırdığımız Data Transfer Object kısmıdır. Genellikle database tablolarımızla aynı değerleri tutan class’lar oluşturulur ve bu değerlerin update, delete ve insert gibi komutları karşılamak amaçlı prop’lar oluşturulur(Get- Set). Ayrıca duruma göre bu class’ları kullanarak code first gibi farklı yollarla da database işlerinizi halledebilirsiniz.

***Proje Anlatımı Başlangıcı***

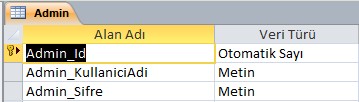
Programa geçmeden önce oluşturduğumuz database’in tablolarını göstererek başlamak daha iyi olacaktır. Programımız 4 katmandan oluşacak ve Entity katmanı birnevi direkt olarak database’deki bağlantılarımızla ilişkili olacaktır.

***Database Oluşturma***

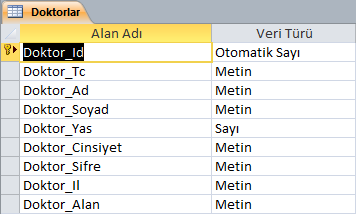
Hastane Otomasyonu programımızda 6 farklı tablo bulunmaktadır.



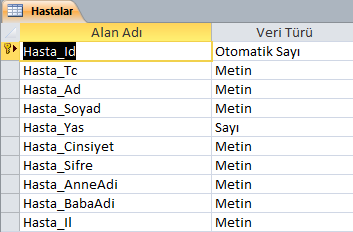
1. **Admin Tablosu**



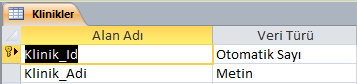
1. **Doktor Tablosu**



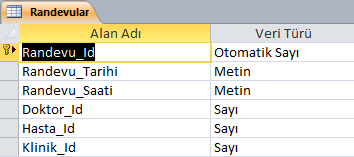
1. **Hasta Tablosu**



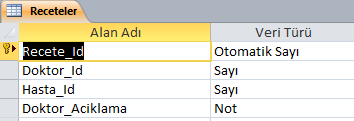
1. **Klinik Tablosu**



1. **Randevu Tablosu**

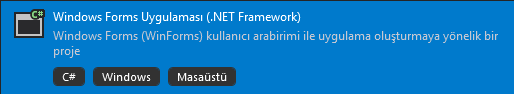


1. **Reçete Tablosu**



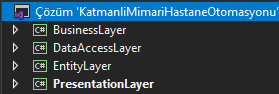
***Projeyi Açma***

Hastane Otomasyonu programımız için database’i oluşturduktan sonra visual studio üzerinden projemizi açıyoruz. (Windows Form - .Net Framework -- > 4.7.2)



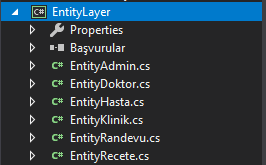
***Katmanları Oluşturma***

Projemizi açtıktan sonra sunumunda başında anlattığım ve kullandığım 4 adet katmanımızı oluşturuyoruz.



Öncelikle database’imizi oluşturduk. Ondan sonra ise programı açıp kullanacağımız katmanlarımızı(Class Library) oluşturduk. Şimdi ise database tablolarımızla bağlantılı olacak şekilde öncelikle EntityLayer katmanımızı oluşturmaya başlayalım. EntityLayer katmanında her tablonun içinde bulunan verileri programa entegre etme amacı ile bir nesne oluşturma tabanını varsayarak class’larımızı ve prop’larımızı(get-set) oluşturalım.

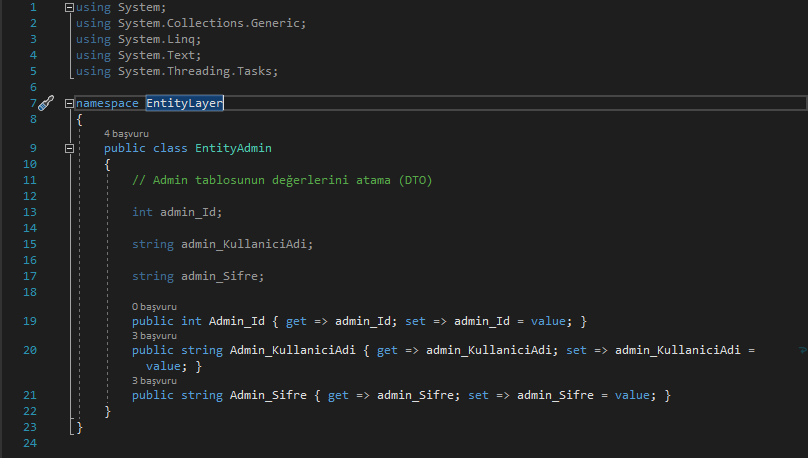
EntityLayer katmanıda database ile eş değer olduğundan 6 farklı class tanımlayacağız.



***Entity Layer Katmanını Oluşturma***

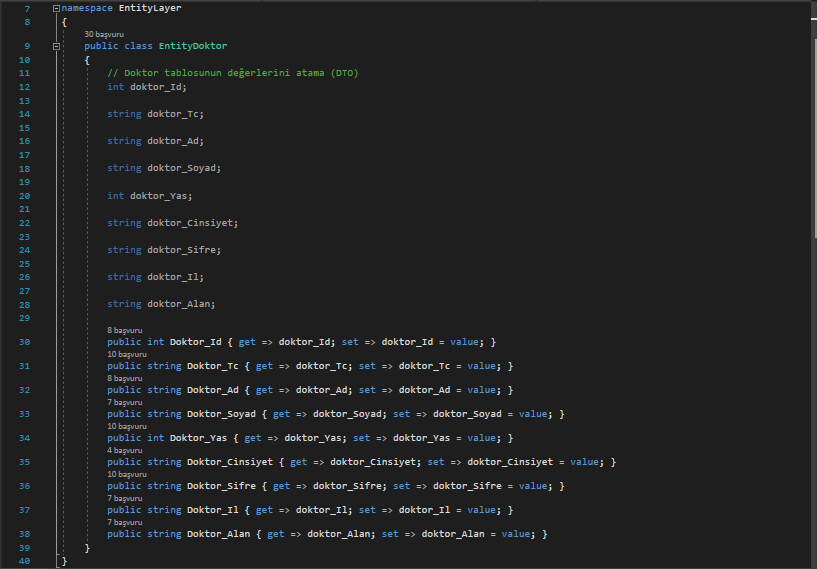
1. **Entity Admin**

Bu class’ımızda database’de admin tablosunda bulunan değerleri burada da tanımladık. (DTO)



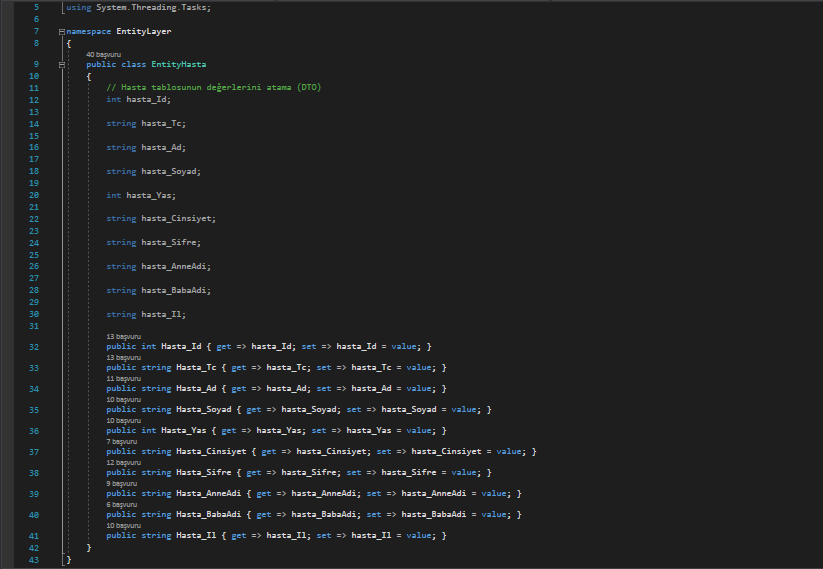
1. **Entity Doktor**

Bu class’ımızda database’de doktor tablosunda bulunan değerleri burada da tanımladık. (DTO)



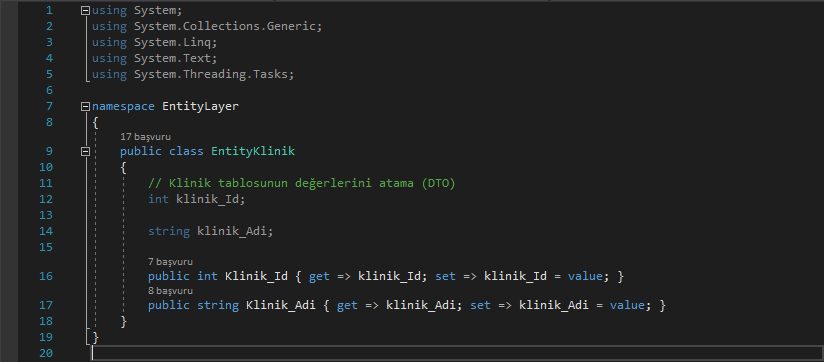
1. **Entity Hasta**

Bu class’ımızda database’de hasta tablosunda bulunan değerleri burada da tanımladık. (DTO)



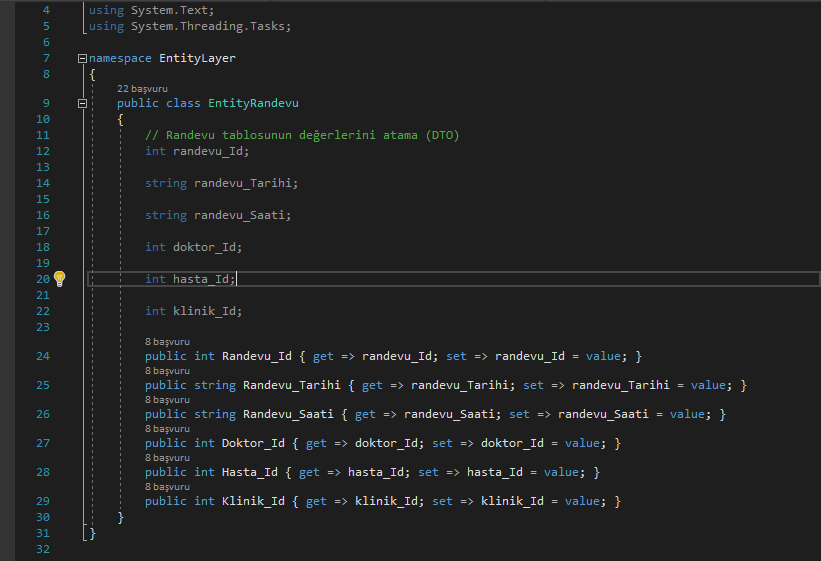
1. **Entity Klinik**

Bu class’ımızda database’de klinik tablosunda bulunan değerleri burada da tanımladık. (DTO)



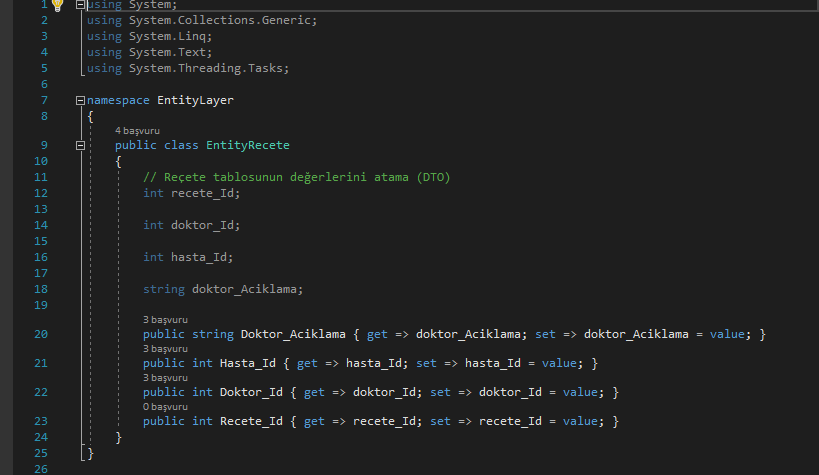
1. **Entity Randevu**

Bu class’ımızda database’de randevu tablosunda bulunan değerleri burada da tanımladık. (DTO)



1. **Entity Reçete**

Bu class’ımızda database’de reçete tablosunda bulunan değerleri burada da tanımladık. (DTO)



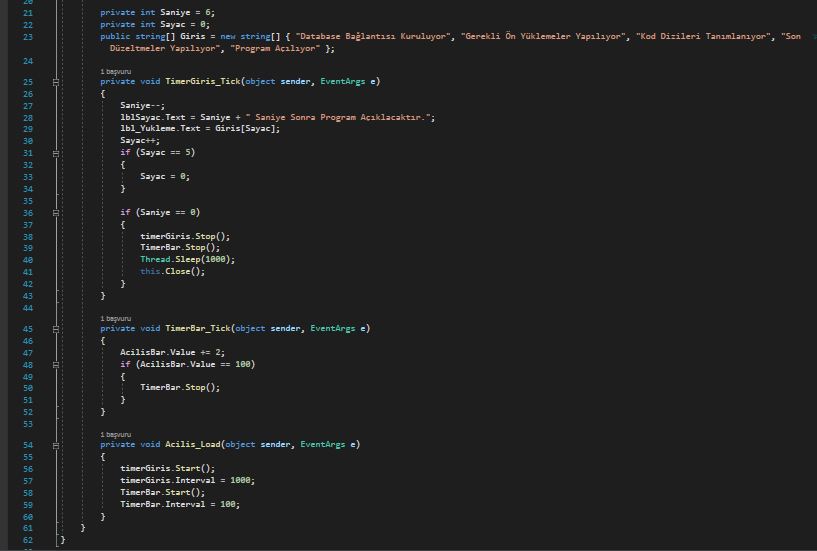
***Presentation(Sunum) Katmanı***

1. **Açılış Ekranı**

Program açıldığında Ana ekran yani giriş ekranı gelmeden açılış ekranı gelmektedir.

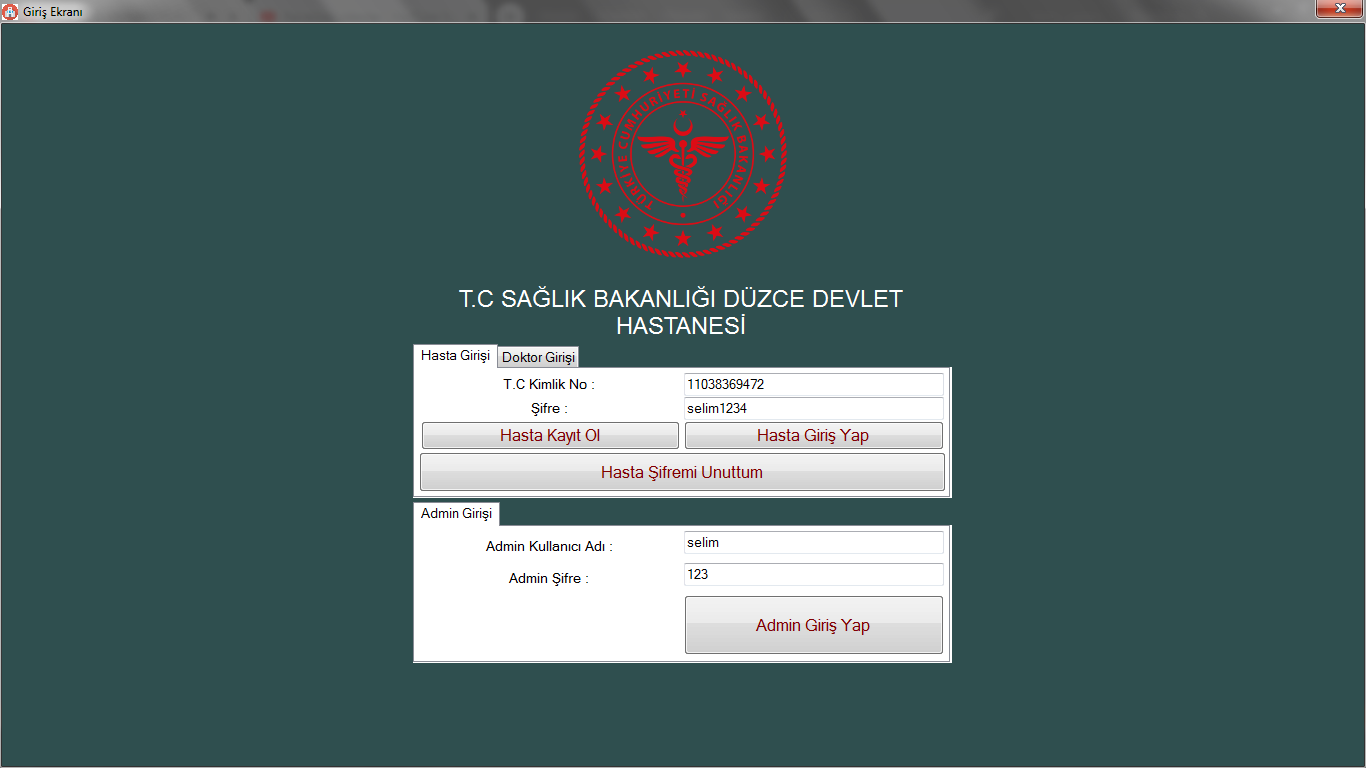


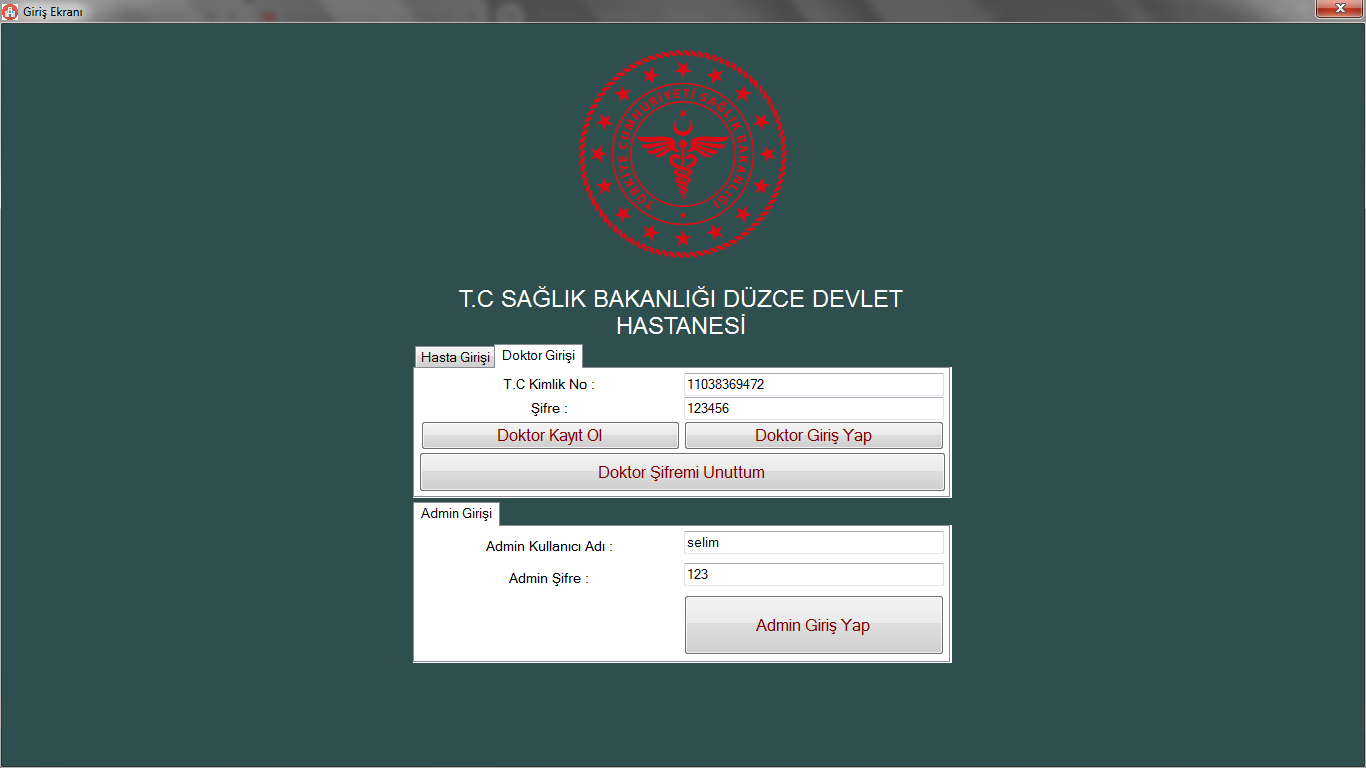
Giris adında string dizi bulunmaktadır. 5 saniye süre boyunca dizideki yazılar geçiyor ve yazılar geçerken bir loading bar doluyor.



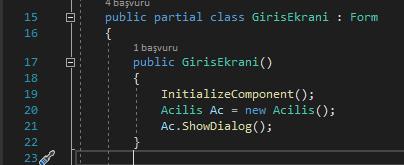
1. **Giriş Ekranı**

Açılış ekranından sonra giriş ekranı bizi karşılamaktadır. Giriş ekranında bizi Hasta,Doktor ve Admin için bazı kısımlar mevcuttur. Hasta ve doktor için Kayıt Olma, Giriş yapma ve Şifremi Unuttum kısımlar mevcut iken admin için ise sadece Giriş Yap kısmı bulunmaktadır.

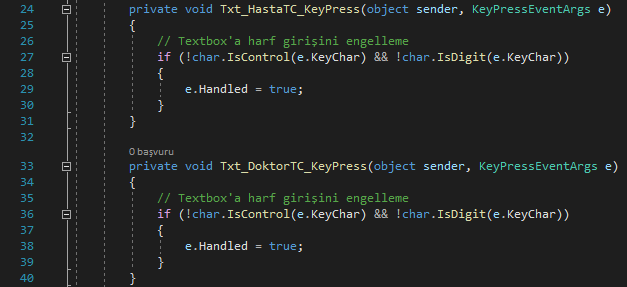




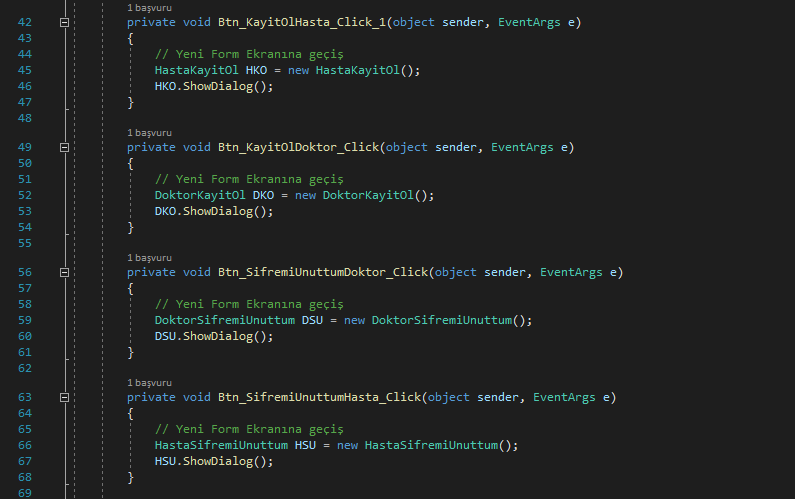
Giriş ekranında ilk olarak Açılış formu açılıyor. Oda yukarıda belirttiğimiz ilk form oluyor.



Giriş ekranında Doktor ve Hasta girişi için bulunan TC Giriş kısımlarında harf girişini engelleme.



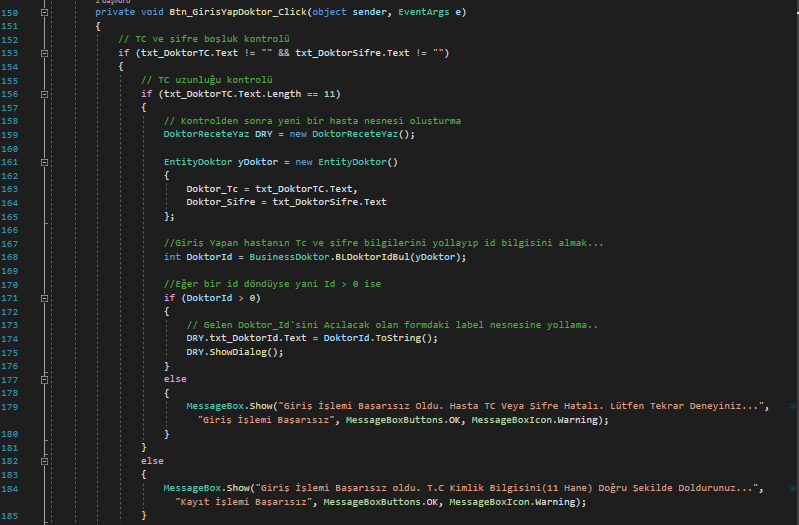
Giriş ekranında Doktor ve Hasta için kayıt ol ve şifremi unuttum formlarını açmak için ShowDialog() kullanılmıştır.



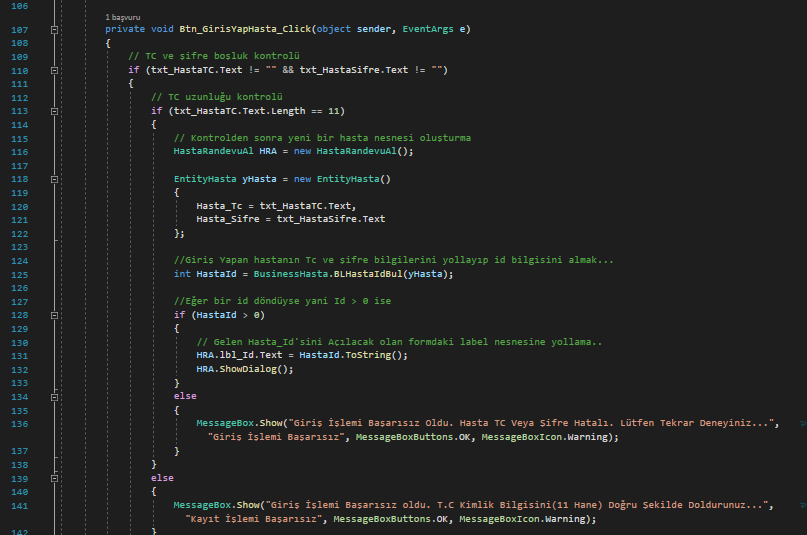
Admin Giriş Button\_Click kodları. Öncelikle Kullanıcı adı ve Şifre boşluk kontrolü yapılmıştır. Eğer boş değiller ise yeni Admin nesne oluşturulup içeriği doldurulmuştur. Sonrasında Business katmanındaki AdminKontrolEt fonksiyonuna yeni oluşturduğumuz nesne gönderilmiştir. Gelen sonuç bool tipindedir, true ise Admin Kontrol Paneli Form’u açılacaktır. False ise kullanıcıya giriş yapılamadı hatası verilecektir.



Doktor Giriş Button\_Click kodları. Öncelikle Tc ve Şifre boşluk kontrolü yapılmıştır. Eğer boş değiller ise TC’nin 11 hane kontrolü yapılmıştır. Eğer 11 hane ve şifre doğru ise yeni Doktor nesne oluşturulup içeriği doldurulmuştur. Sonrasında Business katmanındaki DoktorIdBul fonksiyonuna yeni oluşturduğumuz nesne gönderilmiştir. Gelen sonuç int tipindedir, Eğer gelen sayı 0’dan büyük ise Doktor Reçete Yaz Form’u açılacaktır. 0 ise kullanıcıya giriş yapılamadı hatası verilecektir.



Hasta Giriş Button\_Click kodları. Öncelikle Tc ve Şifre boşluk kontrolü yapılmıştır. Eğer boş değiller ise TC’nin 11 hane kontrolü yapılmıştır. Eğer 11 hane ve şifre doğru ise yeni Hasta nesne oluşturulup içeriği doldurulmuştur. Sonrasında Business katmanındaki HastaIdBul fonksiyonuna yeni oluşturduğumuz nesne gönderilmiştir. Gelen sonuç int tipindedir, Eğer gelen sayı 0’dan büyük ise Hasta Randevu al Form’u açılacaktır. 0 ise kullanıcıya giriş yapılamadı hatası verilecektir.

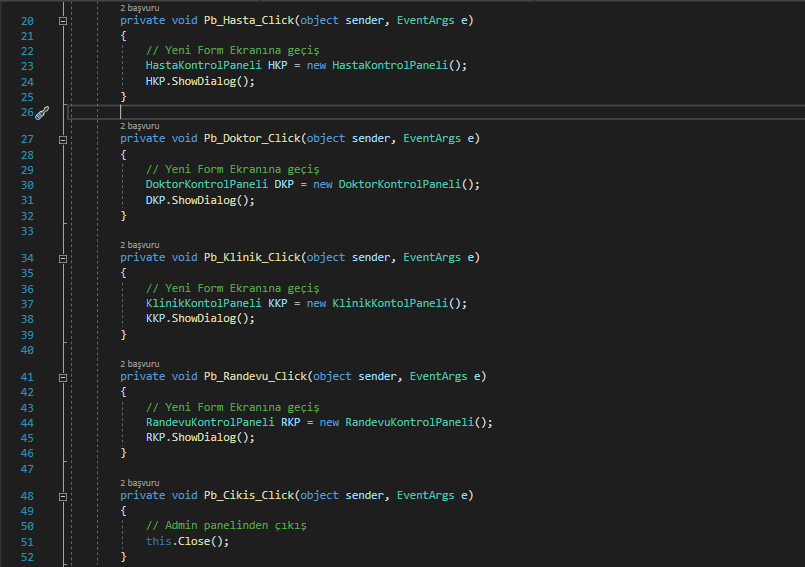


1. **Admin Kontrol Paneli**

Giriş ekranında Admin kullanıcı ve şifre doğru girildiğinde Admin Kontrol Paneli açılmaktadır. Admin panelinden Hasta, Doktor, Klinik ve Randevu bilgilerini listeleyebilir, silebilir ve güncelleyebilir.

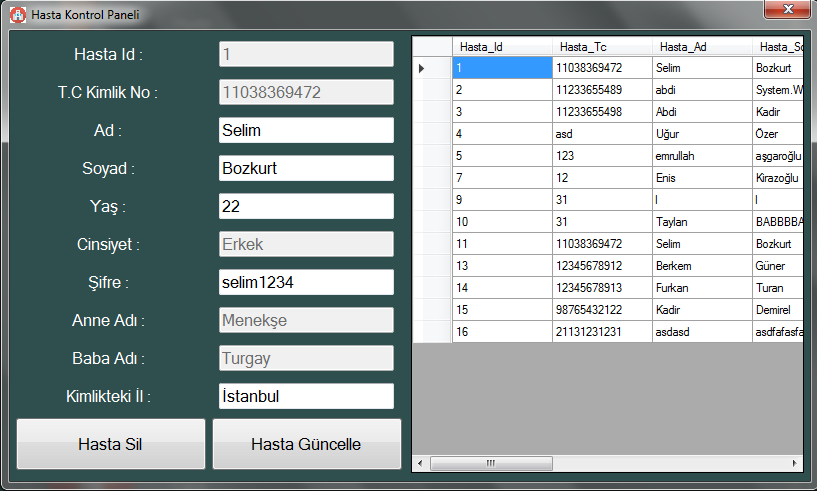


Admin Kontrol Panelinin kodları ise sadece form açmaktadır. Solda bulunan 4 bölüm için Formlar açılır iken Çıkış butonu ise sadece Admin formunu kapatmaktadır.

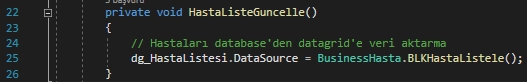


1. **Hasta Kontrol Paneli**

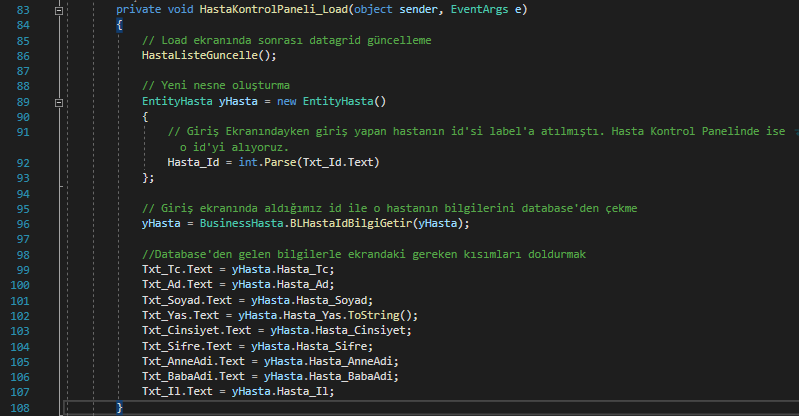
Admin girişi sağlantıktan sonra Hastalar sağdaki DataGrid üzerinde listeleniyor. DataGrid’ten seçilen hastanın bilgileri sol taraftaki textbox’lara bilgiler doluyor. Değiştirilmesin izin verilmeyen değerlerin enable özellikleri false olarak ayarlanmıştır. Admin bu ekranda değişikler yaptıktan sonra hasta bilgilerini güncelleyebilir veya seçilen hastayı silebilir.



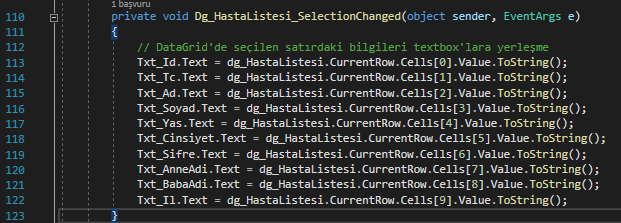
Hasta Kontrol Panelinin Load’ında sağda bulunan datagrid’e hasta bilgilerini çekmek için bir fonksiyon oluşuturuyoruz. Bu fonksiyon Load,Güncelleme sonrası ve silme işlemi sonrasında kullanıyoruz.



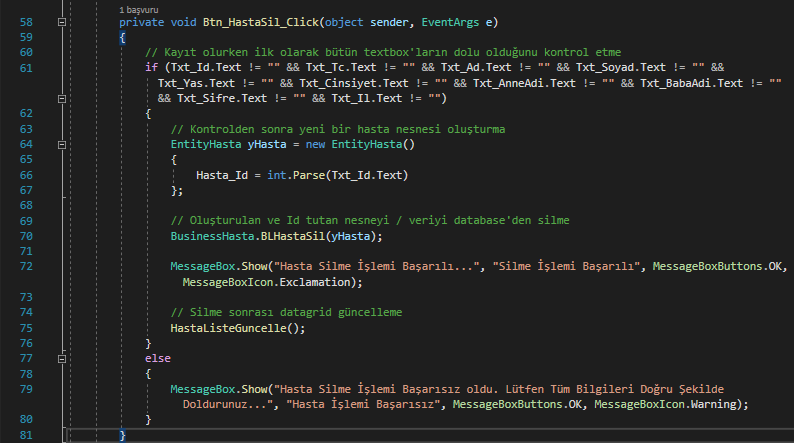
Form Load ekranında DataGrid’e gelen ilk kişinin bilgilerini ilk olarak seçilmiş gibi gösterip bilgilerini sol kısımdaki textboxlara aktarılmaktadır. Seçilen hastanın bilgilerine Id’sinden ulaşılmaktadır.



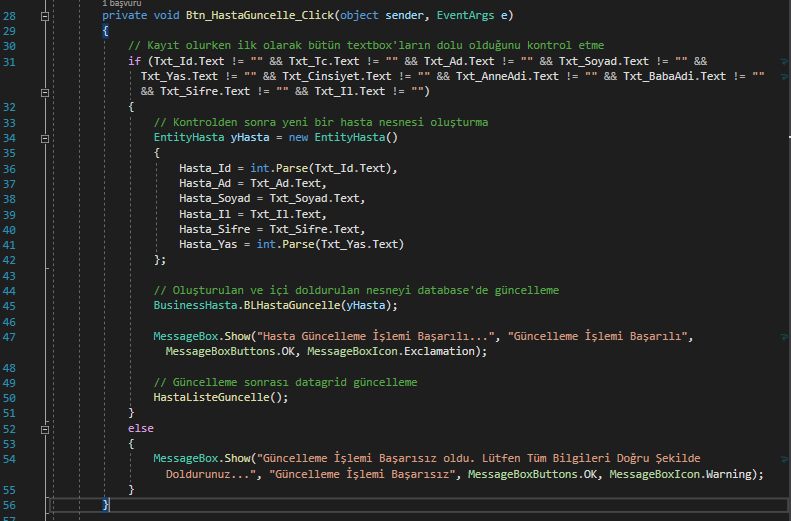
DataGrid’e gelen hastaları seçtiğimiz zaman seçilen hastanın bilgilerini gerekli textbox alanlarına atama.



Hasta Sil datagrid üzerinden seçilen hastanın bilgilerin soldaki textboxlara geldiğini görmüştük. Hastayı silmeye çalıştığımız zaman öncelikle tüm bilgilerin doldu olduğunu kontrol edip yeni bir hasta nesnesi oluşturuyoruz ve o nesneye sadece hastanın id değerini aktarıyoruz. Id bilgisine sahip hasta nesnesini business katmanına hastasil fonksiyonuna parametre olarak yollayıp. Hastanın silinmesini sağlıyoruz. Silindikten sonra ise datagrid’i güncelleyecek olan HastaListeGuncelle fonksiyonunu çağırıyoruz.



Hasta Güncelle datagrid üzerinden seçilen hastanın bilgilerin soldaki textboxlara geldiğini görmüştük. Hastayı güncellemeye çalıştığımız zaman öncelikle tüm bilgilerin doldu olduğunu kontrol edip yeni bir hasta nesnesi oluşturuyoruz ve o nesneye hastanın değişecek olan değerlerini aktarıyoruz. İçi bilgi dolu olan hasta nesnesini business katmanına hastaguncelle fonksiyonuna parametre olarak yollayıp. Hastanın güncellenmesini sağlıyoruz. Silindikten sonra ise datagrid’i güncelleyecek olan HastaListeGuncelle fonksiyonunu çağırıyoruz.

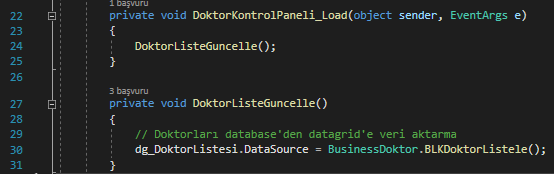


1. **Doktor Kontrol Paneli**

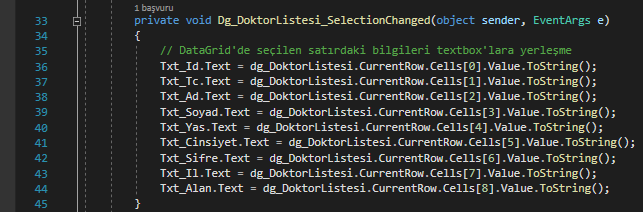
Admin girişi sağlantıktan sonra doktorlar sağdaki DataGrid üzerinde listeleniyor. DataGrid’ten seçilen doktorun bilgileri sol taraftaki textbox’lara bilgiler doluyor. Değiştirilmesin izin verilmeyen değerlerin enable özellikleri false olarak ayarlanmıştır. Admin bu ekranda değişikler yaptıktan sonra doktor bilgilerini güncelleyebilir veya seçilen doktoru silebilir.



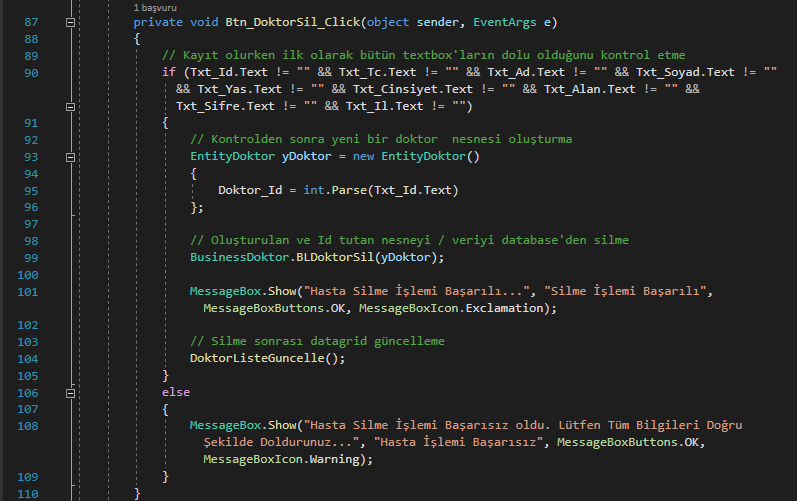
Doktor Kontrol Panelinin Load’ında sağda bulunan datagrid’e doktor bilgilerini çekmek için bir fonksiyon oluşuturuyoruz. Bu fonksiyon Load,Güncelleme sonrası ve silme işlemi sonrasında kullanıyoruz.



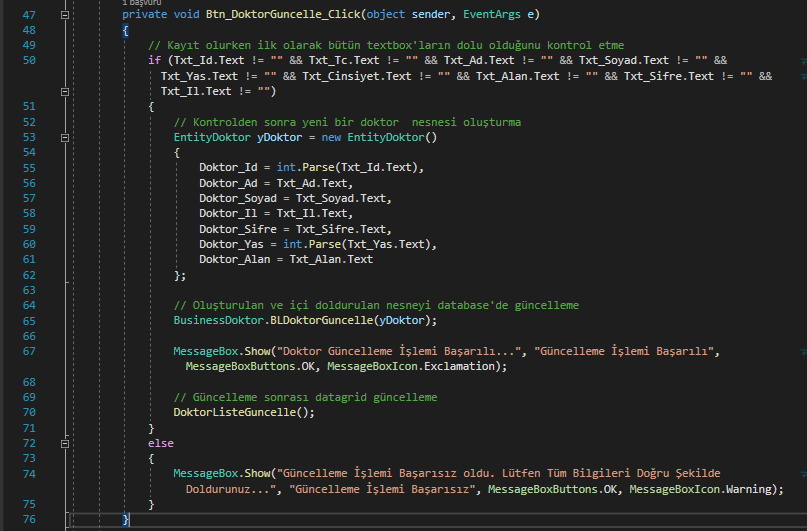
DataGrid’e gelen doktorları seçtiğimiz zaman seçilen doktorun bilgilerini gerekli textbox alanlarına atama.



Doktor Sil datagrid üzerinden seçilen doktorun bilgilerin soldaki textboxlara geldiğini görmüştük. Doktoru silmeye çalıştığımız zaman öncelikle tüm bilgilerin doldu olduğunu kontrol edip yeni bir doktor nesnesi oluşturuyoruz ve o nesneye sadece doktorun id değerini aktarıyoruz. Id bilgisine sahip doktor nesnesini business katmanına doktorsil fonksiyonuna parametre olarak yollayıp. Doktorun silinmesini sağlıyoruz. Silindikten sonra ise datagrid’i güncelleyecek olan DoktorListeGuncelle fonksiyonunu çağırıyoruz.

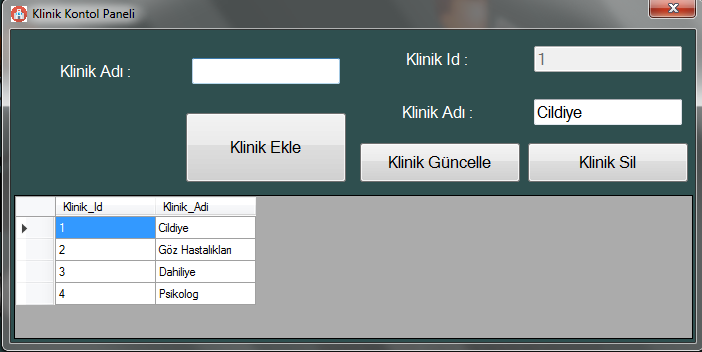


Doktor Güncelle datagrid üzerinden seçilen doktorun bilgilerin soldaki textboxlara geldiğini görmüştük. Doktoru güncellemeye çalıştığımız zaman öncelikle tüm bilgilerin doldu olduğunu kontrol edip yeni bir doktor nesnesi oluşturuyoruz ve o nesneye doktorun değişecek olan değerlerini aktarıyoruz. İçi bilgi dolu olan doktor nesnesini business katmanına doktorguncelle fonksiyonuna parametre olarak yollayıp. Doktorun güncellenmesini sağlıyoruz. Silindikten sonra ise datagrid’i güncelleyecek olan DoktorListeGuncelle fonksiyonunu çağırıyoruz.

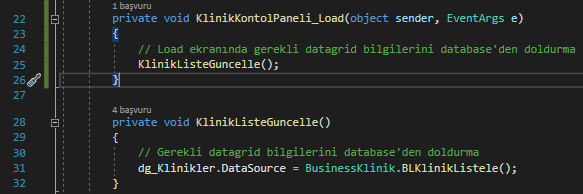


1. **Klinik Kontrol Paneli**

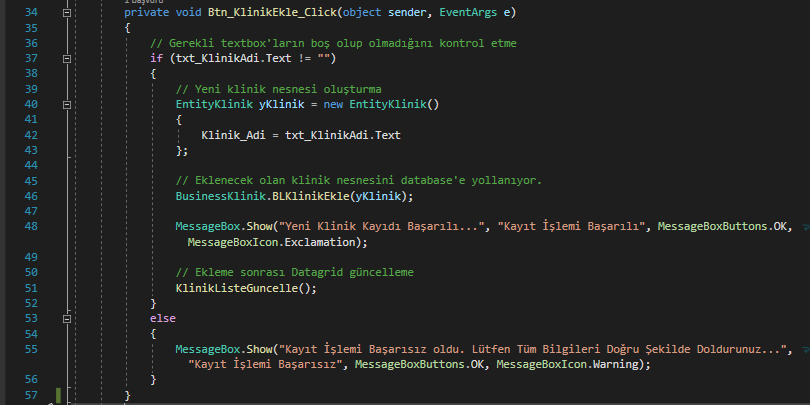
Admin girişi sağlantıktan sonra Klinikler aşağıdaki DataGrid üzerinde listeleniyor. DataGrid’ten seçilen kliniğin bilgileri sağ üst taraftaki textbox’lara bilgiler doluyor. Değiştirilmesine izin verilmeyen değerlerin enable özellikleri false olarak ayarlanmıştır. Admin bu ekranda değişikler yaptıktan sonra klinik ekleyebilir ve klinik bilgilerini güncelleyebilir veya seçilen kliniği silebilir.



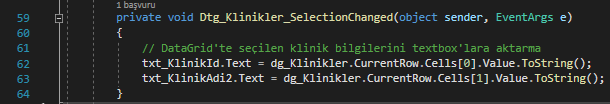
Klinik Kontrol Panelinin Load’ında aşağıda bulunan datagrid’e klinik bilgilerini çekmek için bir fonksiyon oluşuturuyoruz. Bu fonksiyon Load,Ekleme sonrası,Güncelleme sonrası ve silme işlemi sonrasında kullanıyoruz.



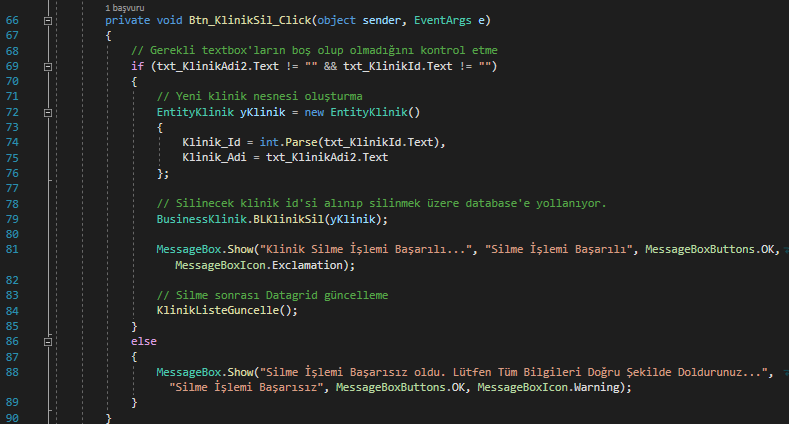
Admin Klinik Kontrol Panelin’de Hasta ve Doktorlara nazaran Klinik ekleyebilir. Bu kısımda ise sadece Kliniğin adı girilmesi yeterlidir. Kod kısmında ise ilk olarak Klinik adı girilecek olan textbox’ın boşluğu kontrol ediliyor. Dolu ise yeni bir klinik nesnesi oluşturulup Business Layet’daki KlinikEkle oluşturulan klinik nesnesi yollanıyor. Nesne yollandıktan sonra Datagrid üzerindeki bilgiyi güncellemek için KlinikListeGuncelle fonksiyonu çağırılıyor.



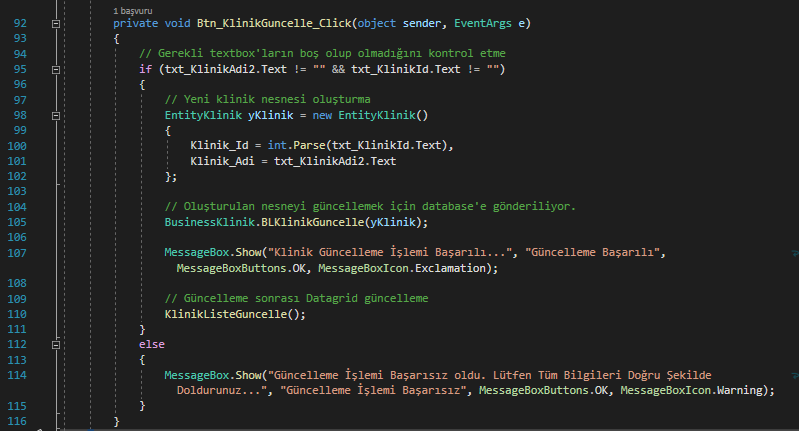
DataGrid’e gelen klinikleri seçtiğimiz zaman seçilen kliniğin bilgilerini gerekli textbox alanlarına atama.



Klinik Sil datagrid üzerinden seçilen kliniğin bilgilerin üstteki textboxlara geldiğini görmüştük. Kliniği silmeye çalıştığımız zaman öncelikle tüm bilgilerin dolu olduğunu kontrol edip yeni bir klinik nesnesi oluşturuyoruz ve o nesneye sadece kliniğin id değerini aktarıyoruz. Id bilgisine sahip klinik nesnesini business katmanına kliniksil fonksiyonuna parametre olarak yollayıp. Kliniği silinmesini sağlıyoruz. Silindikten sonra ise datagrid’i güncelleyecek olan KlinikListeGuncelle fonksiyonunu çağırıyoruz.

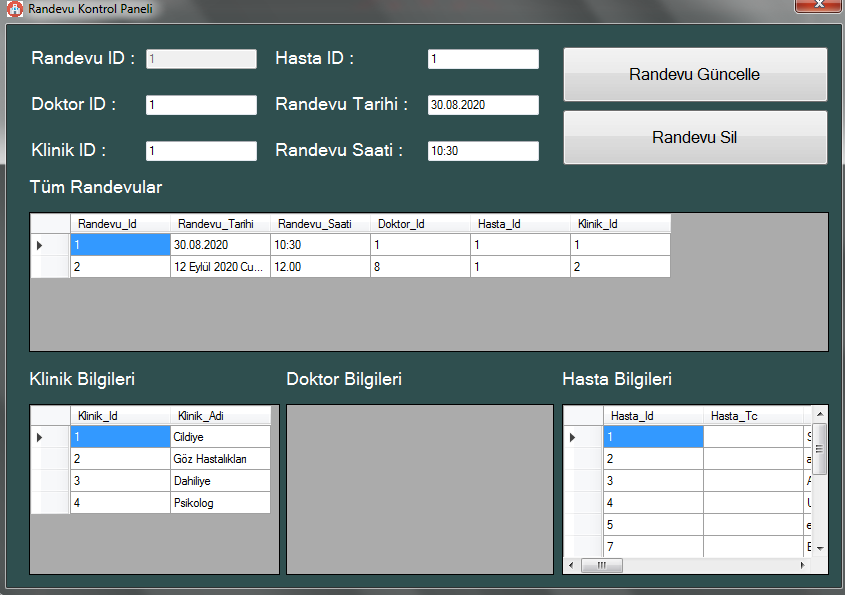


Klinik Güncelle datagrid üzerinden seçilen kliniği bilgilerin üstteki textboxlara geldiğini görmüştük. Kliniği güncellemeye çalıştığımız zaman öncelikle tüm bilgilerin dolu olduğunu kontrol edip yeni bir klinik nesnesi oluşturuyoruz ve o nesneye kliniğin değişecek olan değerlerini aktarıyoruz. İçi bilgi dolu olan klinik nesnesini business katmanına klinikguncelle fonksiyonuna parametre olarak yollayıp. Kliniğin güncellenmesini sağlıyoruz. Silindikten sonra ise datagrid’i güncelleyecek olan KlinikListeGuncelle fonksiyonunu çağırıyoruz.

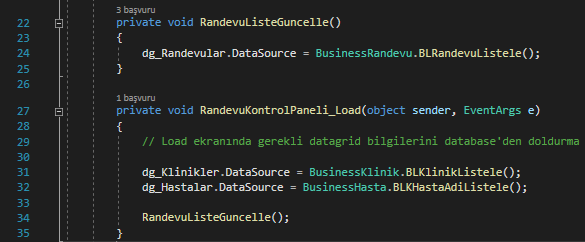


1. **Randevu Kontrol Paneli**

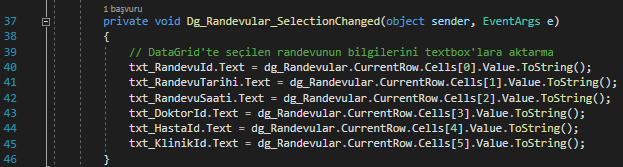
Admin girişi sağlantıktan sonra Randevular aşağıdaki DataGrid üzerinde listeleniyor. DataGrid’ten seçilen ranvunun bilgileri üst taraftaki textbox’lara bilgiler doluyor. Değiştirilmesine izin verilmeyen değerlerin enable özellikleri false olarak ayarlanmıştır. Admin bu ekranda değişikler yaptıktan sonra randevu bilgilerini güncelleyebilir veya seçilen randevuyu silebilir.



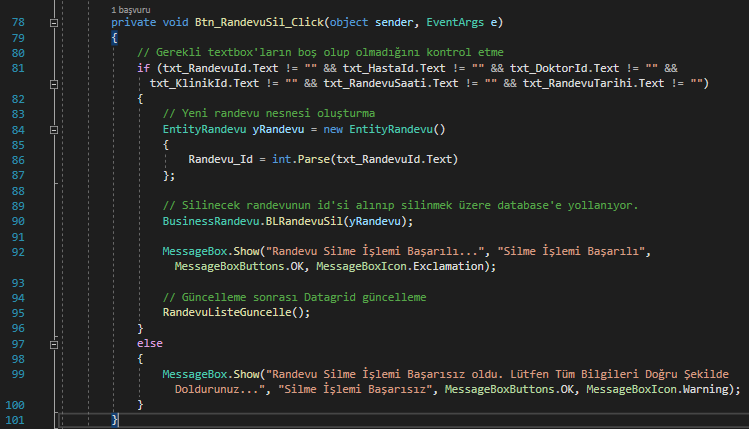
Randevu Kontrol Panelinin Load’ında aşağıda bulunan datagrid’e randevu bilgilerini çekmek için bir fonksiyon oluşuturuyoruz. Bu fonksiyon Load,Güncelleme sonrası ve silme işlemi sonrasında kullanıyoruz.



DataGrid’e gelen randevuları seçtiğimiz zaman seçilen randevunun bilgilerini gerekli textbox alanlarına atama.



Randevu Sil datagrid üzerinden seçilen randevunun bilgilerin üstteki textboxlara geldiğini görmüştük. Randevu silmeye çalıştığımız zaman öncelikle tüm bilgilerin dolu olduğunu kontrol edip yeni bir randevu nesnesi oluşturuyoruz ve o nesneye sadece randevunun id değerini aktarıyoruz. Id bilgisine sahip klinik randevu business katmanına randevusil fonksiyonuna parametre olarak yollayıp. Randevunun silinmesini sağlıyoruz. Silindikten sonra ise datagrid’i güncelleyecek olan RandevuListeGuncelle fonksiyonunu çağırıyoruz.



Randevu Güncelle datagrid üzerinden seçilen randevu bilgilerin üstteki textboxlara geldiğini görmüştük. Randevuyu güncellemeye çalıştığımız zaman öncelikle tüm bilgilerin dolu olduğunu kontrol edip yeni bir randevu nesnesi oluşturuyoruz ve o nesneye randevunun değişecek olan değerlerini aktarıyoruz. İçi bilgi dolu olan randevu nesnesini business katmanına randevuguncelle fonksiyonuna parametre olarak yollayıp. Randevunun güncellenmesini sağlıyoruz. Silindikten sonra ise datagrid’i güncelleyecek olan RandevuListeGuncelle fonksiyonunu çağırıyoruz.

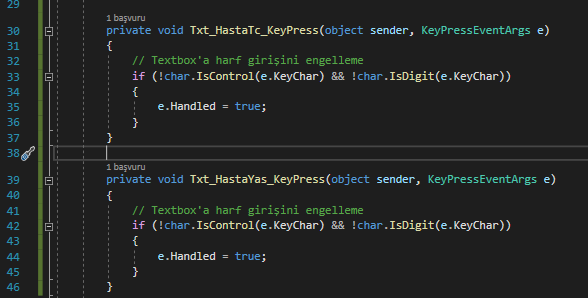


1. **Hasta Kayıt Ol**

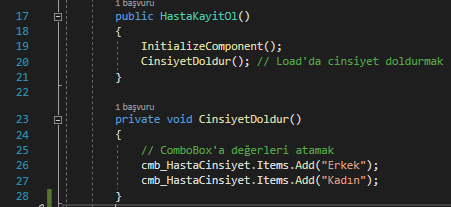
Önceki kısımlarda sadece Admin’in neler yapabileceğini anlattık. Burada ise üyelik veya admin özelliklerine sahip olmayan bir hastanın kayıt olmasını yapıyoruz. Gereken bilgiler sağlandıktan sonra hasta kayıt olabilir ve sonrasında işlemlerini yapabilir.



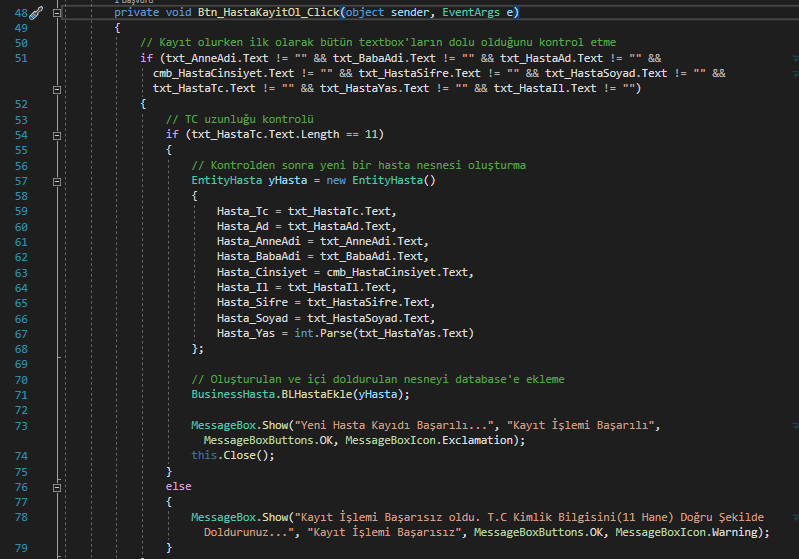
Hasta Kayıt ekranında TC Ve yaş kısımlarında harf girişini engelleme.



Burada Hasta Kayıt formu açılmadan önce cinsiyet bilgisini tutan comboBox’ın içeriğini ekleme.



Kayıt olurken öncelikle gerekli bütün bilgilerin doğru olduğunu kontrol ediyoruz. Eğer doğru ve Tc kimlik numarası uzunluğu 11 ise yeni bir hasta nesnesi oluşturuyoruz. Oluşturduğumuz nesneyi kullanıcın girdiği bilgileri ile dolduruyoruz. İçini bilgilerle doldurduğumuz nesneyi Business Layer üzerinden DataAccess Layer’daki BLHastaEkle fonksiyonuna parametre olarak yolluyoruz. Kayıt işlemi olduktan sonra Kayıt Ol formunu kapatıyoruz.

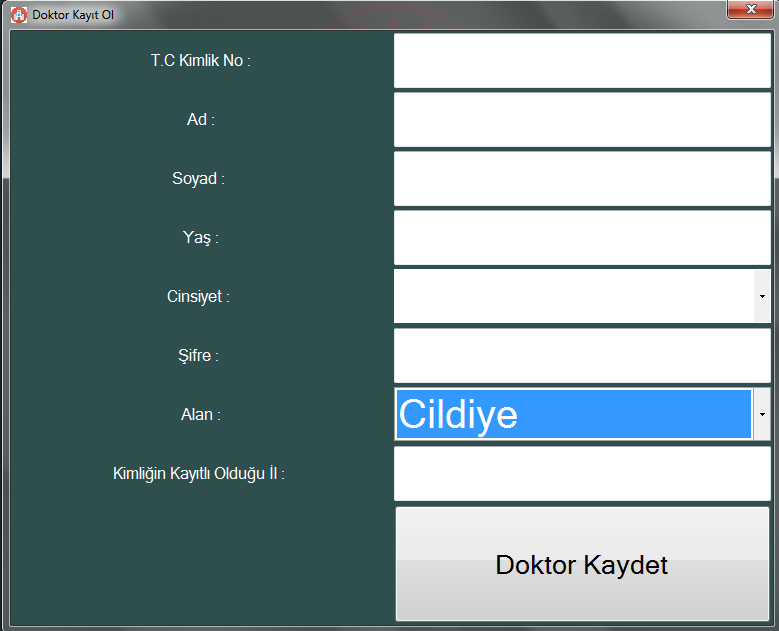


Boş olan bilgi kısımları eksik ise kullanıcıyı uyarmak amaçlı foreach ile bir uyarı kısmı yaptık.

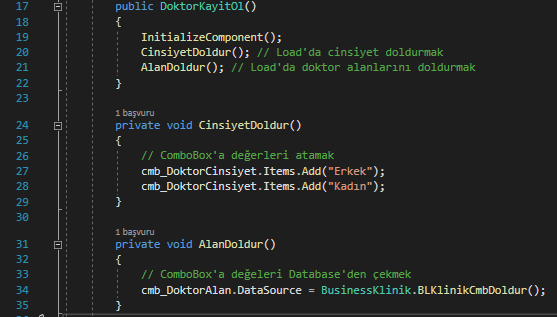


1. **Doktor Kayıt Ol**

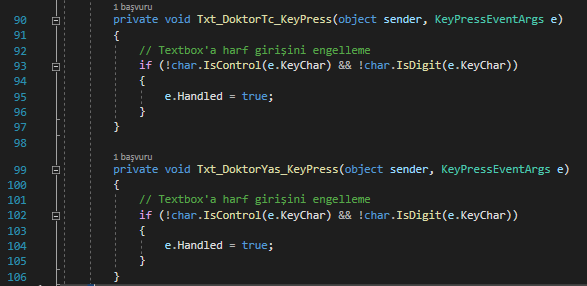
Bu Form’da üyelik veya admin özelliklerine sahip olmayan bir doktorun kayıt olmasını yapıyoruz. Gereken bilgiler sağlandıktan sonra doktor kayıt olabilir ve sonrasında işlemlerini yapabilir.



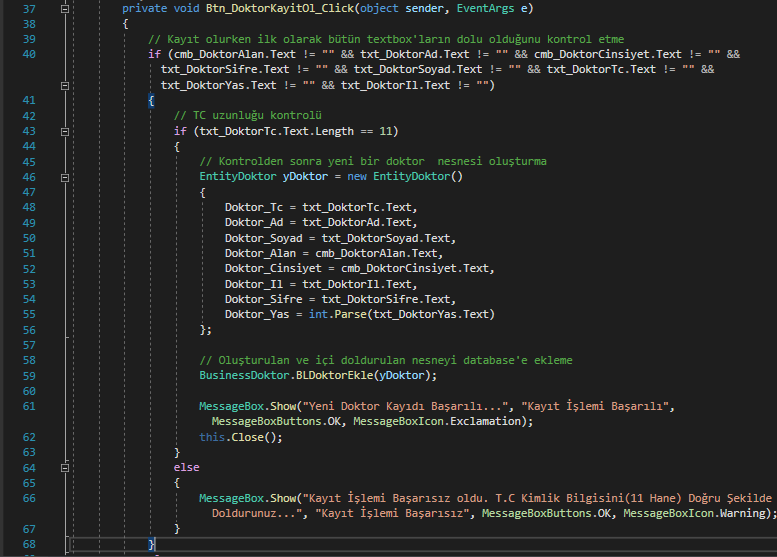
Burada Hasta Kayıt formu açılmadan önce cinsiyet bilgisini ve Doktorların alan bilgisini tutan comboBox’ların içeriğini ekleme.



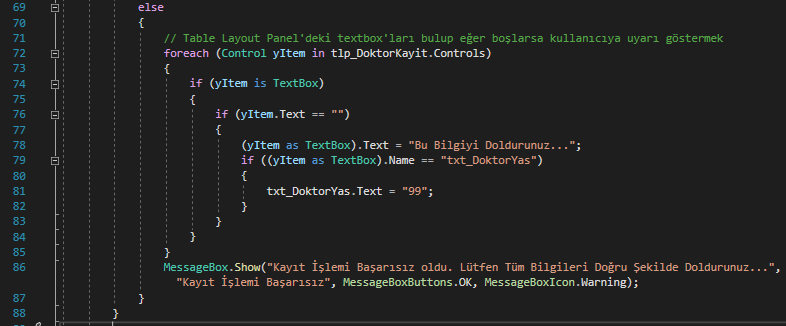
Doktor Kayıt ekranında TC Ve yaş kısımlarında harf girişini engelleme.



Kayıt olurken öncelikle gerekli bütün bilgilerin doğru olduğunu kontrol ediyoruz. Eğer doğru ve Tc kimlik numarası uzunluğu 11 ise yeni bir doktor nesnesi oluşturuyoruz. Oluşturduğumuz nesneyi kullanıcın girdiği bilgileri ile dolduruyoruz. İçini bilgilerle doldurduğumuz nesneyi Business Layer üzerinden DataAccess Layer’daki BLDoktorEkle fonksiyonuna parametre olarak yolluyoruz. Kayıt işlemi olduktan sonra Kayıt Ol formunu kapatıyoruz.

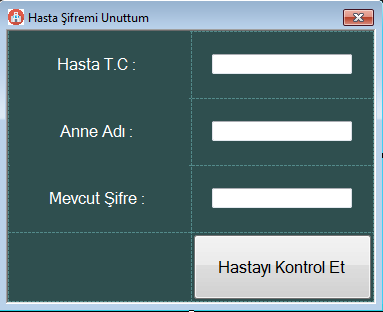


Boş olan bilgi kısımları eksik ise kullanıcıyı uyarmak amaçlı foreach ile bir uyarı kısmı yaptık.

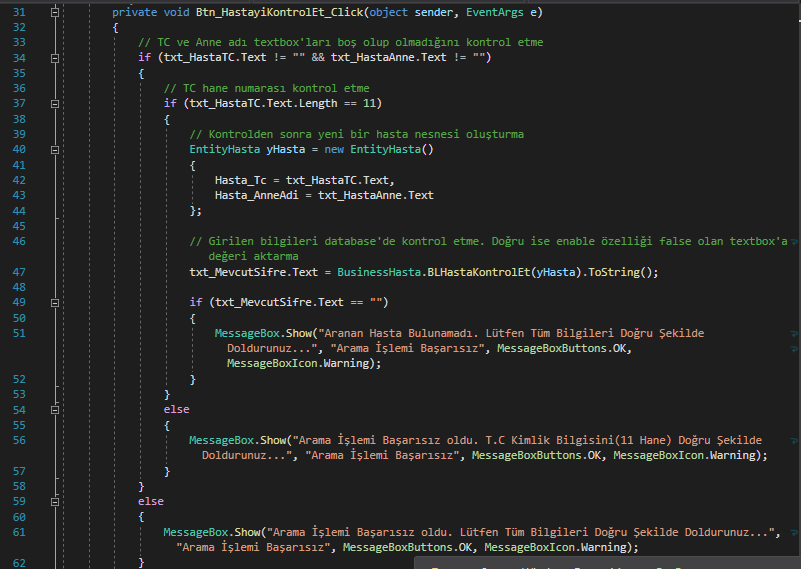


1. **Hasta Şifremi Unuttum**

Daha öncesinde hasta kayıdı yapılmış fakat şifresini unutan kişiler için yapılmış bir Form’dur. Şifresini unutan hastalar şifresini ve anne adını girip hali hazırda kullandıkları fakat unuttukları şifrelerini mevcut şifre kısmından görebilirler.

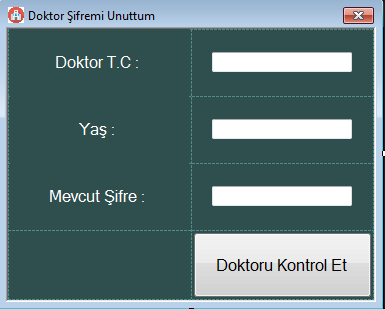
****

Öncelikle giriş değerlerinin boş olup olmadığını ve TC’nin 11 hane olduğunu kontrol ediliyor. Sonrasında doğru ise yeni bir hasta nesnesi oluşturup değerlerle dolduruyoruz. Aldığımız verileri Business Layer üzerinden DAL katmanındaki HastaKontrolEt Fonskiyonuna parametre olarak yolluyoruz. Eğer bilgiler doğrulanır ise o hastanın mevcut şifresini geri enable özelliği false olan textbox’a gönderiliyor.

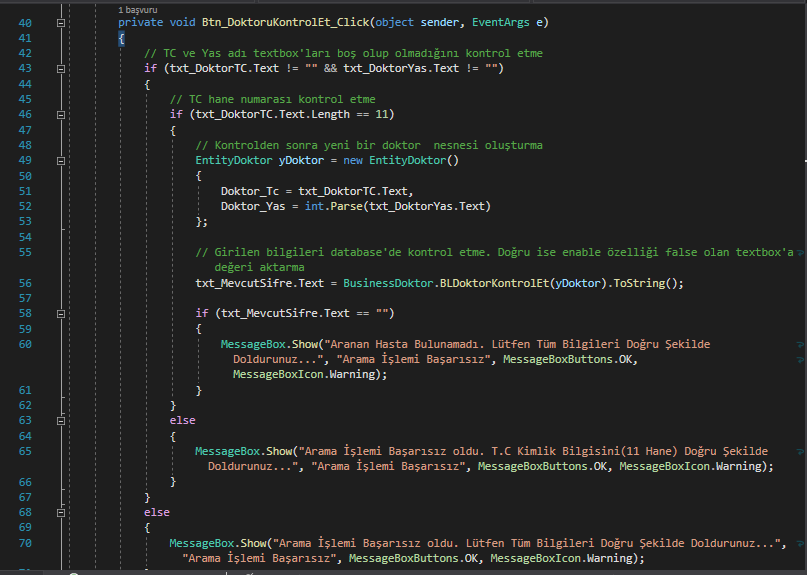
****

1. **Doktor Şifremi Unuttum**

Daha öncesinde doktor kayıdı yapılmış fakat şifresini unutan kişiler için yapılmış bir Form’dur. Şifresini unutan doktorlar şifresini ve yaşını girip hali hazırda kullandıkları fakat unuttukları şifrelerini mevcut şifre kısmından görebilirler.

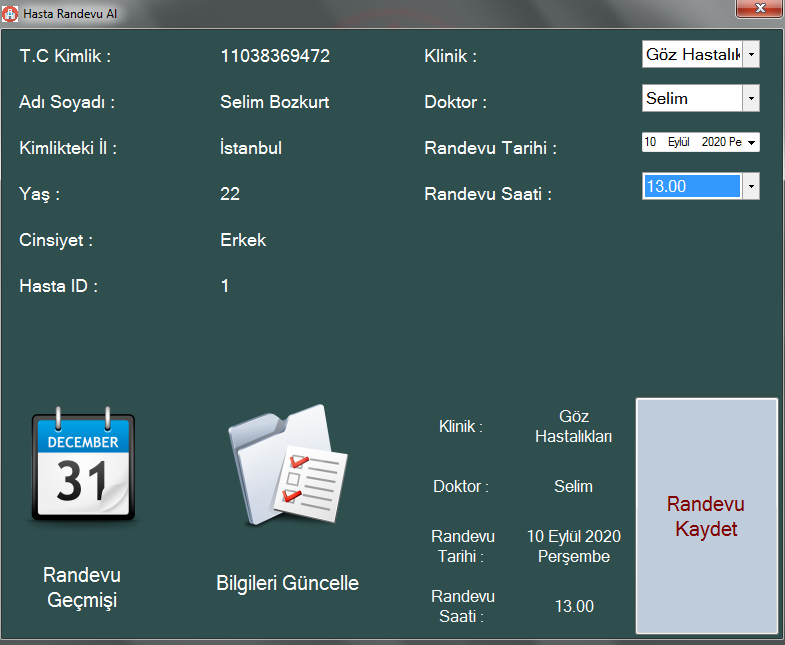


Öncelikle giriş değerlerinin boş olup olmadığını ve TC’nin 11 hane olduğunu kontrol ediliyor. Sonrasında doğru ise yeni bir doktor nesnesi oluşturup değerlerle dolduruyoruz. Aldığımız verileri Business Layer üzerinden DAL katmanındaki DoktorKontrolEt Fonskiyonuna parametre olarak yolluyoruz. Eğer bilgiler doğrulanır ise o doktorun mevcut şifresini geri enable özelliği false olan textbox’a gönderiliyor.

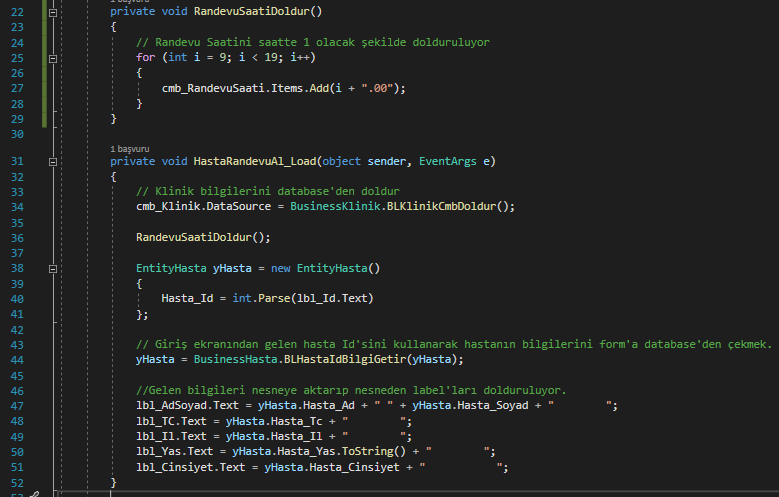


1. **Hasta Randevu Al**

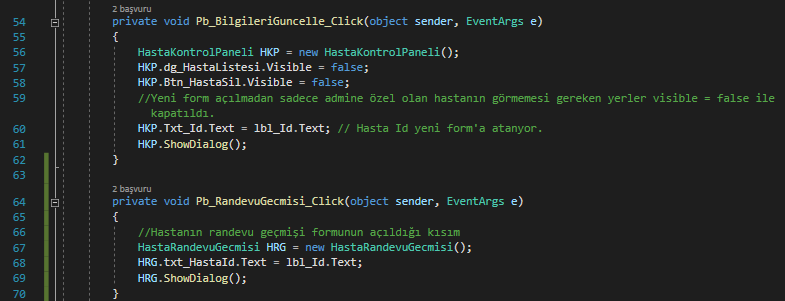
Hasta girişini yaptıktan sonra sol üstte girişi yapan hastanın bilgileri gelir. Sağ üst kısımda ise klinikleri seçtikçe o kliniğin doktorları altındaki kısıma gelir. Sol altta hasta randevu geçmişine ulaşabilir ve kendi bilgilerini güncelleyebilir. Sağ alttaki kısımda ise sağ üstte seçtiği bilgiler mevcuttur. Son kez doğruluğunu teğit edip randevusunu alabilir.



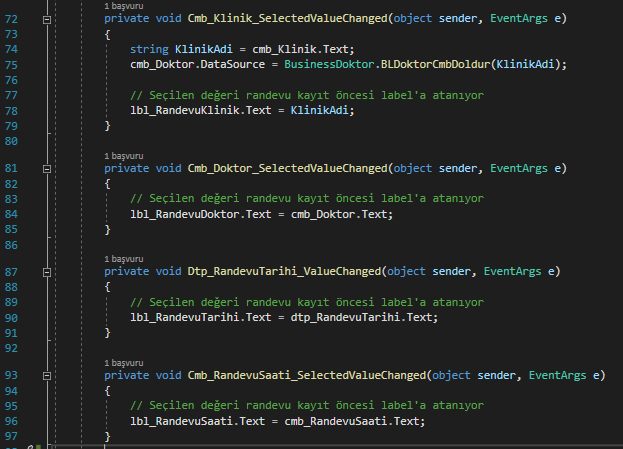
Hasta Randevu Ol formu açılırken gerekli comboBox’ları doldurma ve giriş yaparken Hasta Id bilgisini bu Form’a yollamıştık. Yolladığımız o Id’yi kullanarak gerekli hasta bilgilerini database’den çektik ve gerekli yerleri doldurduk.



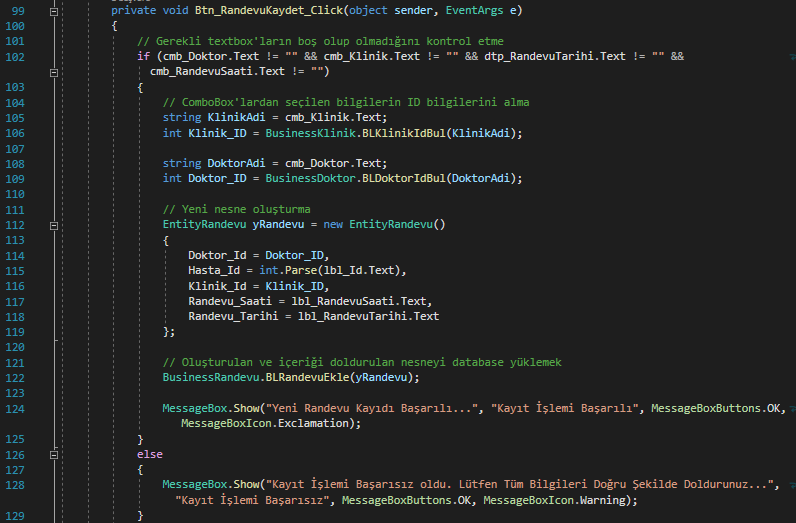
Hastanın bilgisini güncellyebileceği ve randevu geçmişine bakabileceği Form’ların açılış kodları.



Öncelikle klinik seçildiğinde o kliniğin doktorlarını diğer comboBox’ta güncelliyoruz. Sonrasında ise sağ altta bulunan ve seçtiğiniz bilgileri teğit edebileceğiniz kısıma seçtiğiniz değerleri aktarıyoruz.

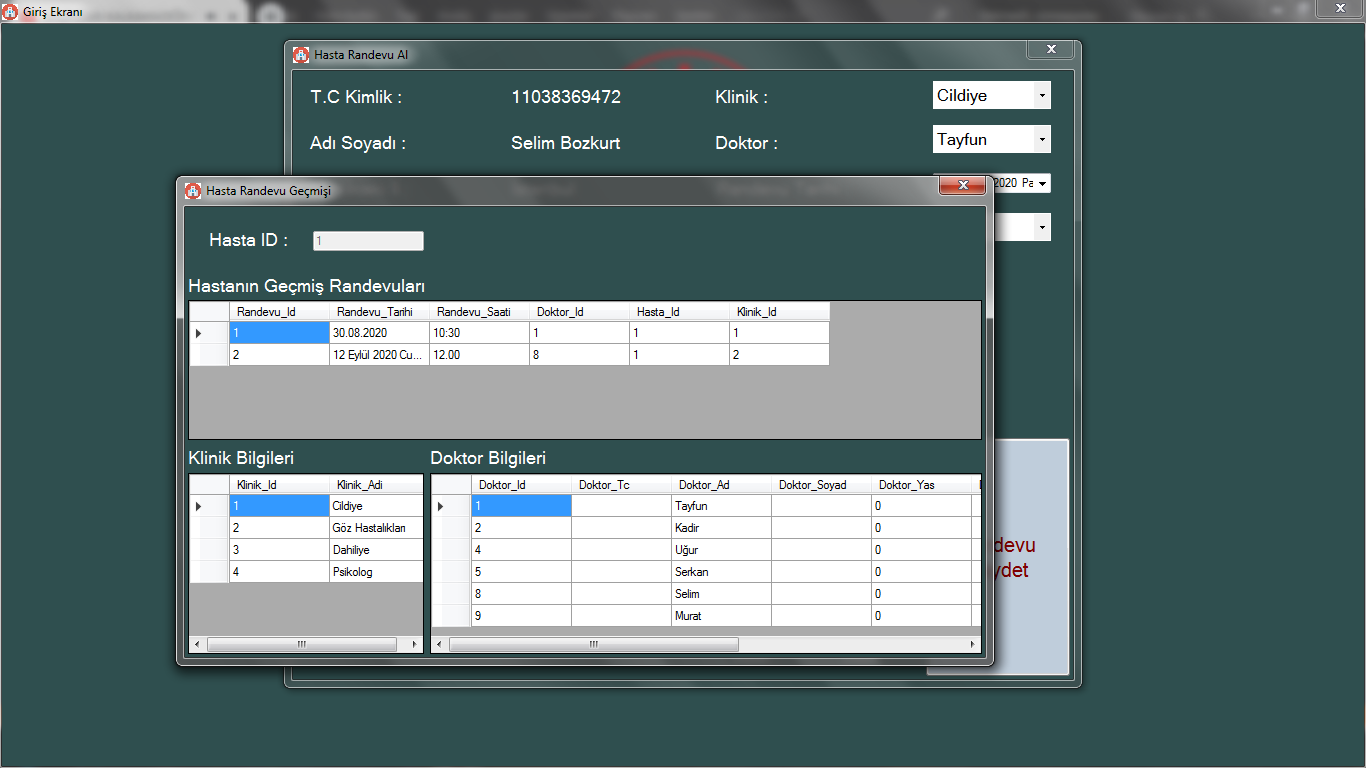


Hasta Randevu Al Formunda herşeyi hallettikten sonra ve gerekli değerleri seçtikten sonra randevu kaydederken ilk olarak değerlerin boş olup olmadığını kontrol ediyoruz. Boş değil ise gerekli bilgileri yeni oluşturduğumuz randevu nesnesine atayıp. DAL katmanındaki RandevuEkle fonksiyonuna parametre olarak yolluyoruz.



1. **Hasta Randevu Geçmişi**

Hasta daha öncesinde aldığı randevularını görebileceği form ekranıdır.

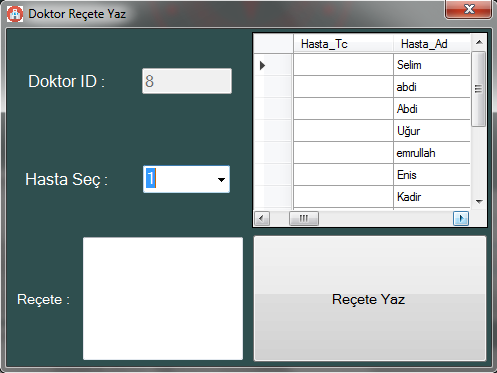


Bu formda sadece listelemeler olduğu için database’den gerekli tabloları datagrid’e aktarıyoruz.

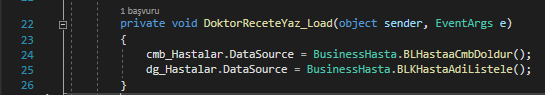


1. **Doktor Reçete Yaz**

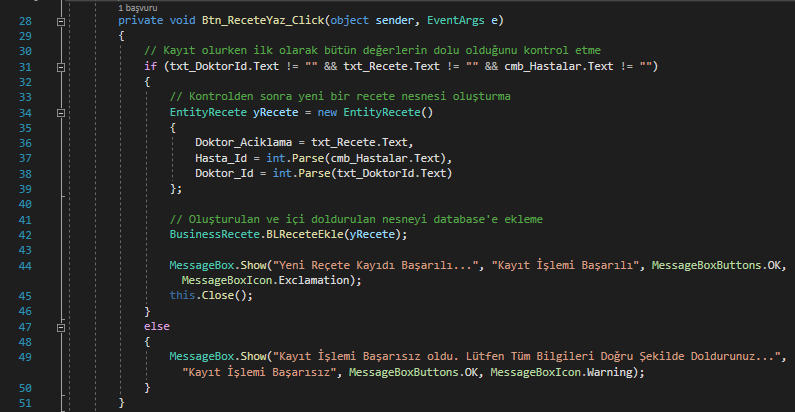
Doktor girişini yaptıktan sonra hastasını seçtikten sonra hastası için reçete yazabilir. Hastayı Id’den seçtiğimizden sağdaki datagrid’den hangi id hangi hastaya denk geliyor görebilir.



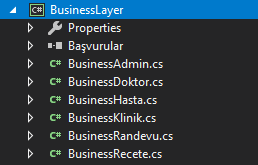
Load kısmında gerekli alanları database’den doldurma.



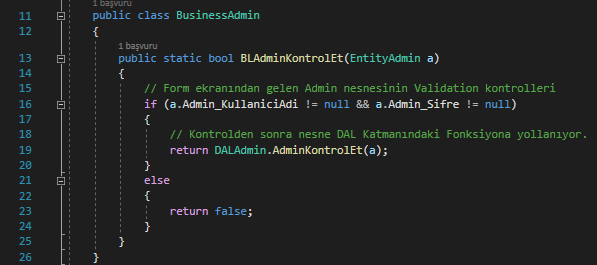
Gerekli alanlar doldurulduktan sonra yeni reçete nesnesi oluşturulur. Nesnenin içeriği doldurulduktan sonra DAL katmanındaki ReçeteEkle Fonksiyonuna reçete nesnesi yollanıyor.



***Business(İş) Katmanı Oluşturma***

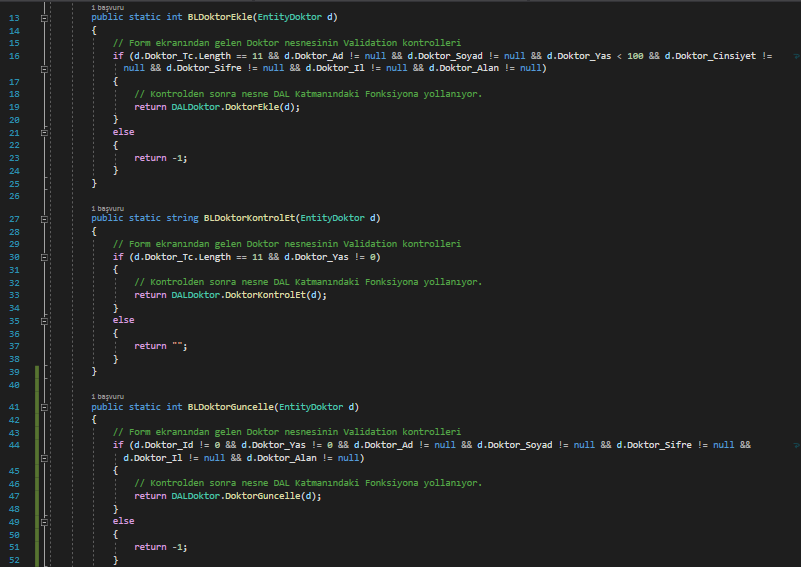


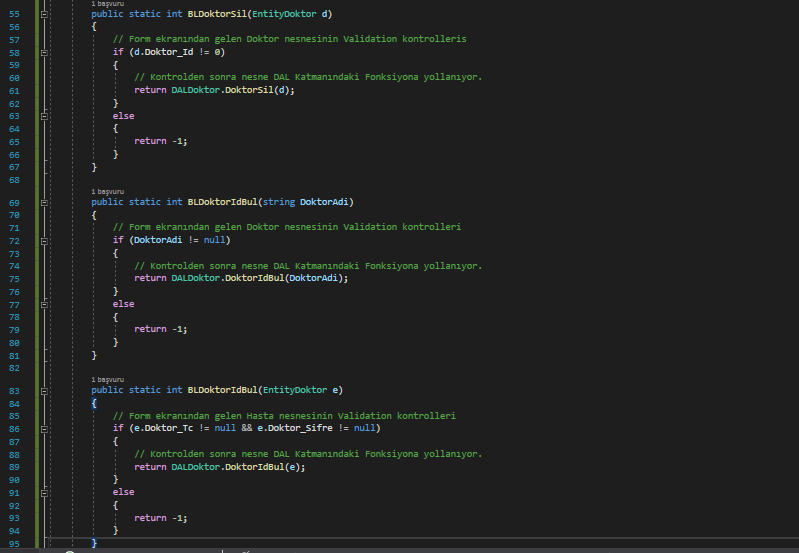
1. **Business Admin**

Sunum katmanından gelen nesneye a adını veriyoruz. Ve a nesnesi ile gelen değerlerin Validation kontrollerini yapıyoruz ve kontrollerden sonra gelen nesneyi DAL Katmanındaki fonksiyonlara yolluyoruz.****

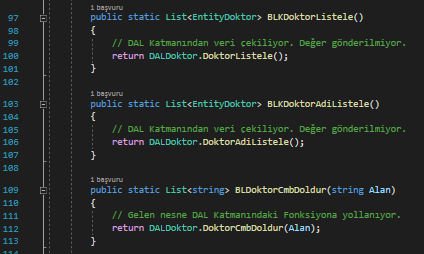
1. **Business Doktor**

Sunum katmanından gelen nesneye d adını veriyoruz. Ve d nesnesi ile gelen değerlerin Validation kontrollerini yapıyoruz ve kontrollerden sonra gelen nesneyi DAL Katmanındaki fonksiyonlara yolluyoruz.

****

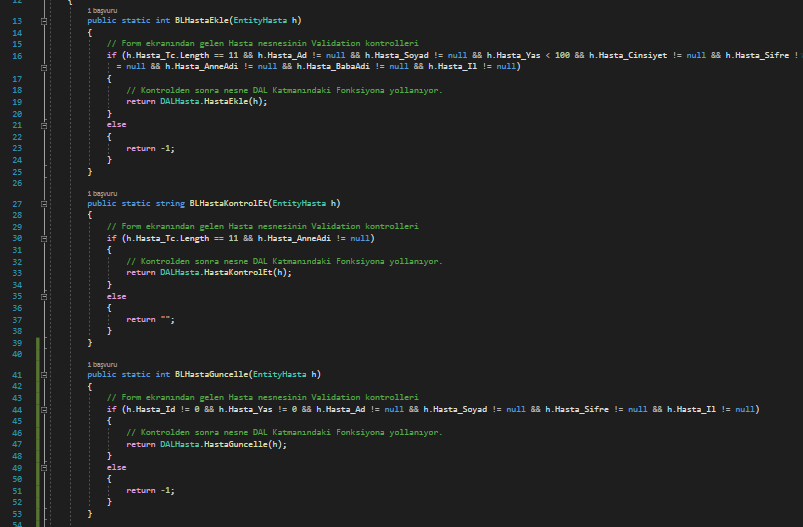
****

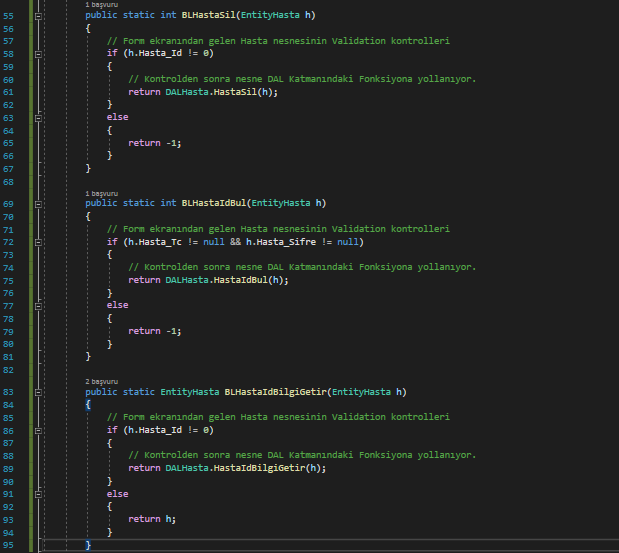
Üstteki kısımda sadece nesne gönderip database’de işlem yapılmasını beklerken bu kodlarda ise bazılarında parametre yollayıp bazılarında ise yollamadan database’den veri almak için ara katman olarak kullanıyoruz.

****

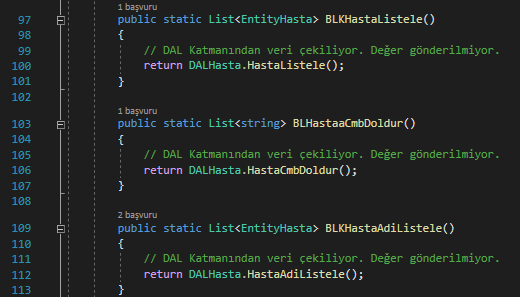
1. **Business Hasta**

Sunum katmanından gelen nesneye a adını veriyoruz. Ve a nesnesi ile gelen değerlerin Validation kontrollerini yapıyoruz ve kontrollerden sonra gelen nesneyi DAL Katmanındaki fonksiyonlara yolluyoruz.

****

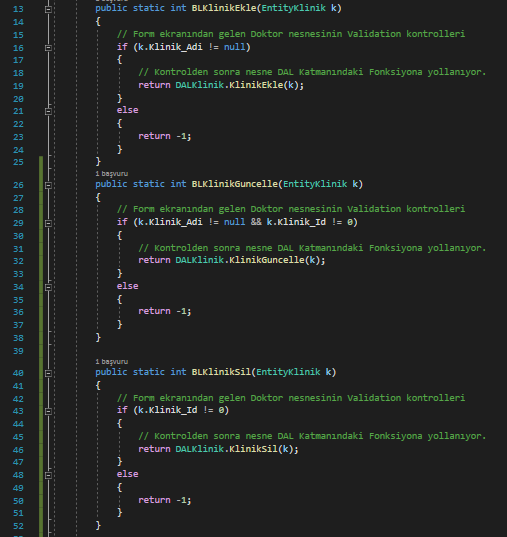
****

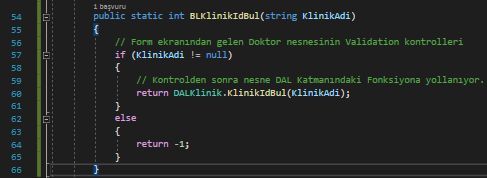
Üstteki kısımda sadece nesne gönderip database’de işlem yapılmasını beklerken bu kodlarda ise bazılarında parametre yollayıp bazılarında ise yollamadan database’den veri almak için ara katman olarak kullanıyoruz.

****

1. **Business Klinik**

Sunum katmanından gelen nesneye a adını veriyoruz. Ve a nesnesi ile gelen değerlerin Validation kontrollerini yapıyoruz ve kontrollerden sonra gelen nesneyi DAL Katmanındaki fonksiyonlara yolluyoruz.

****

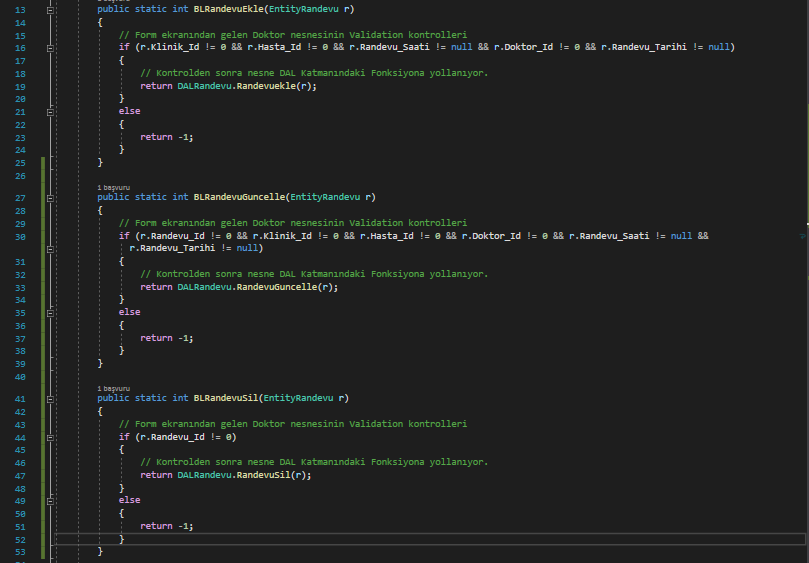
****

Üstteki kısımda sadece nesne gönderip database’de işlem yapılmasını beklerken bu kodlarda ise bazılarında parametre yollayıp bazılarında ise yollamadan database’den veri almak için ara katman olarak kullanıyoruz.

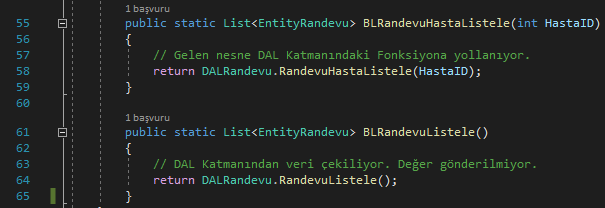
****

1. **Business Randevu**

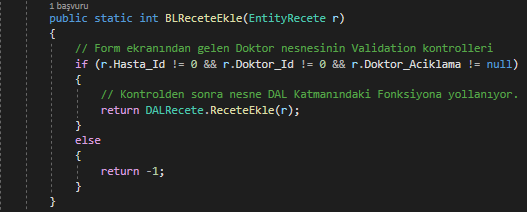
Sunum katmanından gelen nesneye r adını veriyoruz. r nesnesi ile gelen değerlerin Validation kontrollerini yapıyoruz ve kontrollerden sonra gelen nesneyi DAL Katmanındaki fonksiyonlara yolluyoruz.

****

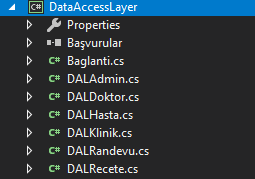
Üstteki kısımda sadece nesne gönderip database’de işlem yapılmasını beklerken bu kodlarda ise bazılarında parametre yollayıp bazılarında ise yollamadan database’den veri almak için ara katman olarak kullanıyoruz.

****

1. **Business Reçete**

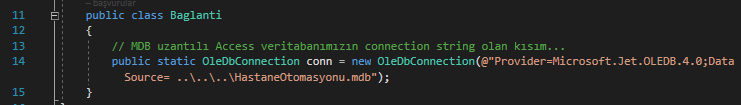
Sunum katmanından gelen nesneye r adını veriyoruz. Ve r nesnesi ile gelen değerlerin Validation kontrollerini yapıyoruz ve kontrollerden sonra gelen nesneyi DAL Katmanındaki fonksiyonlara yolluyoruz.****

***Data Access Katmanı Oluşturma***



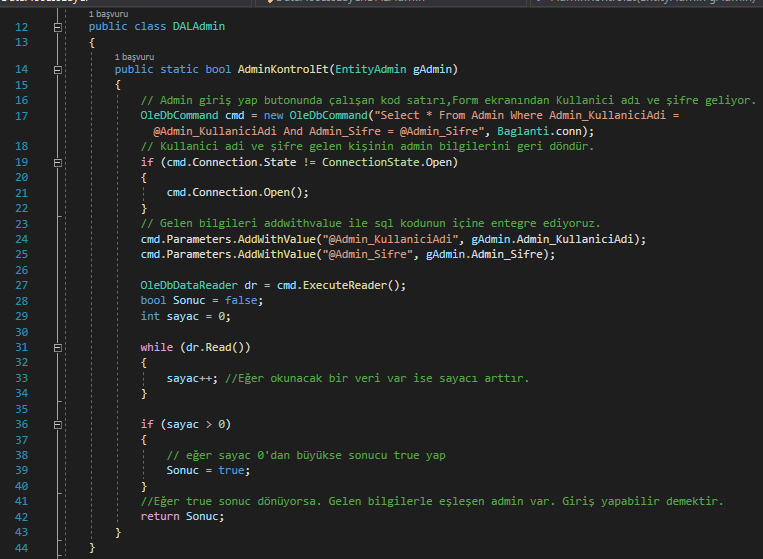
1. **Bağlantı**

Programımızın Database bağlantısı yapan connection stringimizin bulunduğu class’ımızdır. Bu projede Access bağlantısı kullanılmıştır.



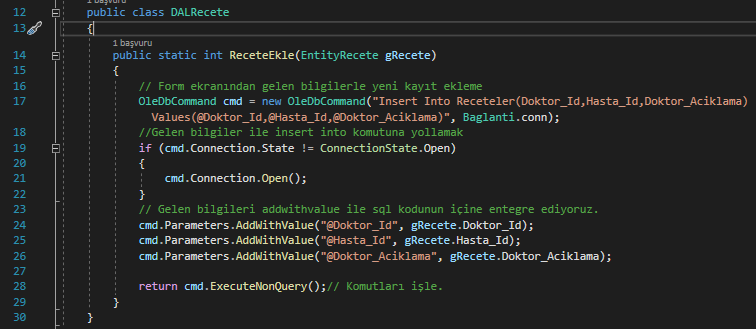
1. **Data Access Admin**

Business Layer katmanından gelen Admin nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Eğer o kullanıcı adı ve şifreye sahip bir admin var ise dr.Read dosyası çalışıyor. Bu çalışması sayacı arttıyor. Eğer sayaç 0’dan büyük ise sonucumuz true oluyor ve bool şeklinde true’yu geri döndürüyoruz ve kullanıcı giriş yapabiliyor.



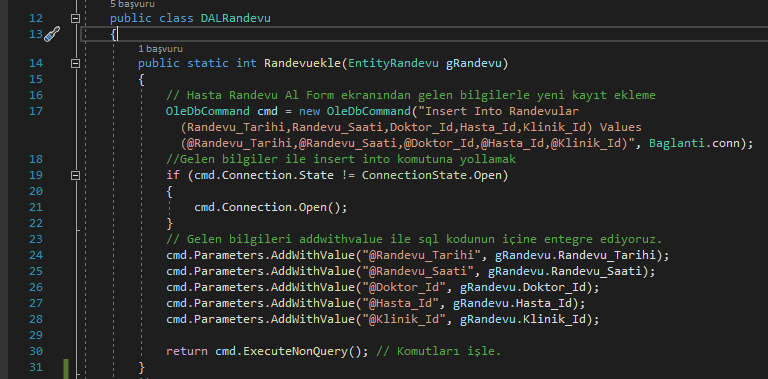
1. **Data Access Reçete**

Business Layer katmanından gelen reçete nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz Insert olduğunundan ekleme işlemi yapılmış oluyor.

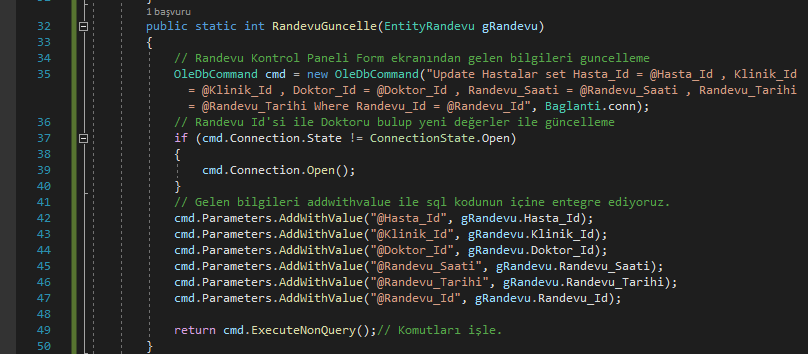


1. **Data Access Randevu**

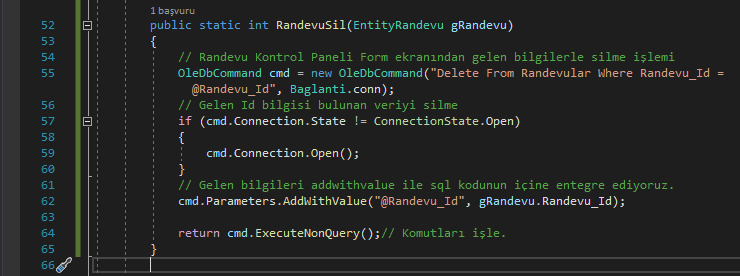
Business Layer katmanından gelen randevu nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz Insert olduğunundan ekleme işlemi yapılmış oluyor.



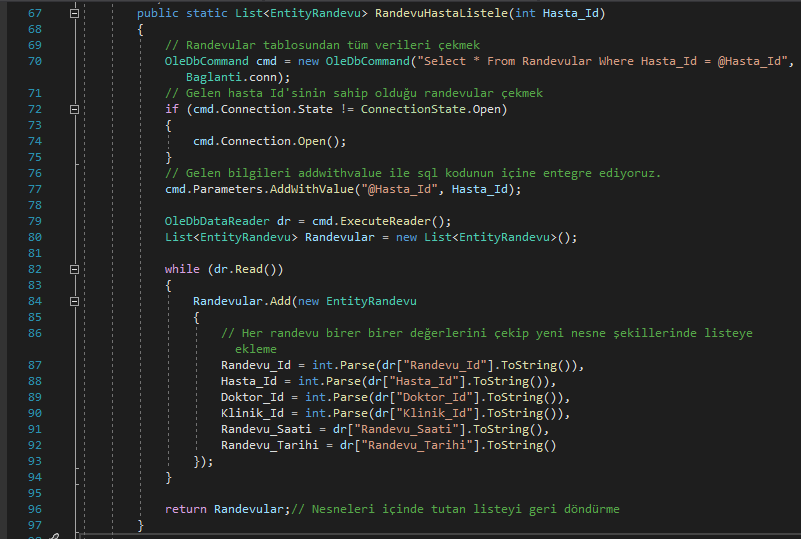
Business Layer katmanından gelen randevu nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz update olduğunundan güncelleme işlemi yapılmış oluyor. Güncellemeyi ise gelen nesnenin Id’sini kullanarak randevuyu bulup değerleri değiştiriyoruz.



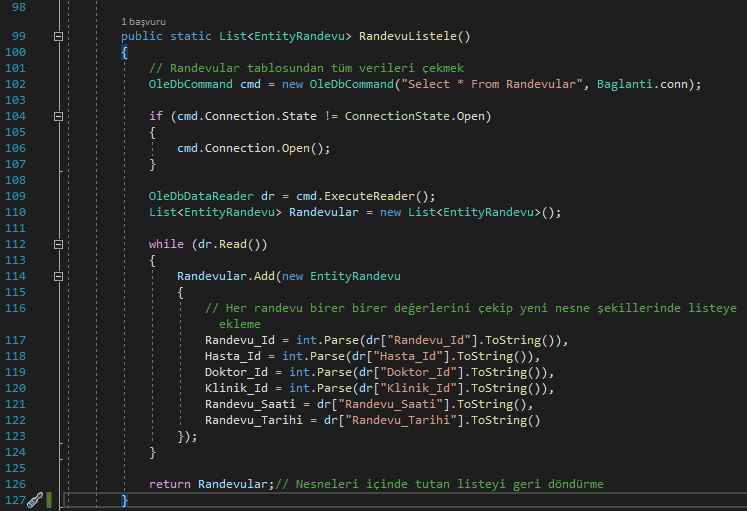
Business Layer katmanından gelen randevu nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz delete olduğunundan silme işlemi yapılmış oluyor. Silme işlemi ise gelen nesnenin Id’sini kullanarak randevuyu bulup gelen randevuyu siliyoruz.



Business Layer katmanından gelen bir hastaId’miz var. Gelen int değerini parametre olarak sql sorgumuza ekliyoruz. Gelen Id’yi sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise gelen Id’yi kullanarak randevuyu bulup gelen randevunun bilgilerini geri yolluyoruz.

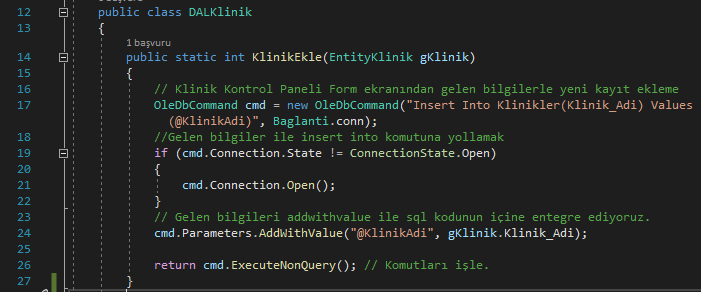


Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise bütün randevuları bulup gelen randevunun bilgilerini geri yolluyoruz.

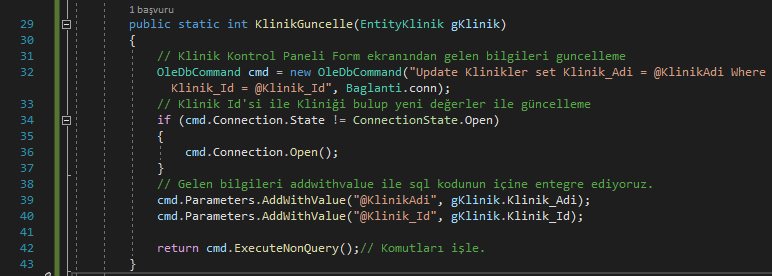


1. **Data Access Klinik**

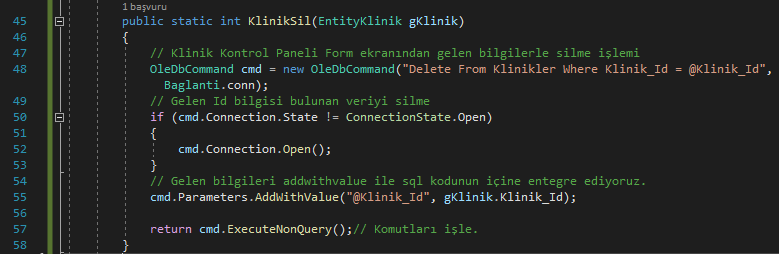
Business Layer katmanından gelen klinik nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz Insert olduğunundan ekleme işlemi yapılmış oluyor.



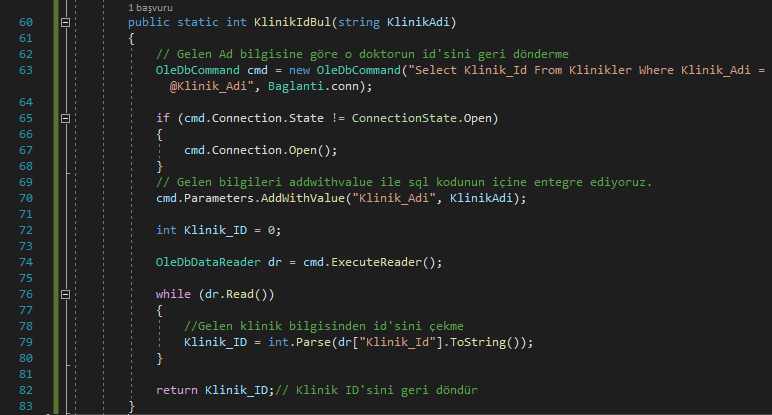
Business Layer katmanından gelen klinik nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz update olduğunundan güncelleme işlemi yapılmış oluyor. Güncellemeyi ise gelen nesnenin Id’sini kullanarak randevuyu bulup değerleri değiştiriyoruz.



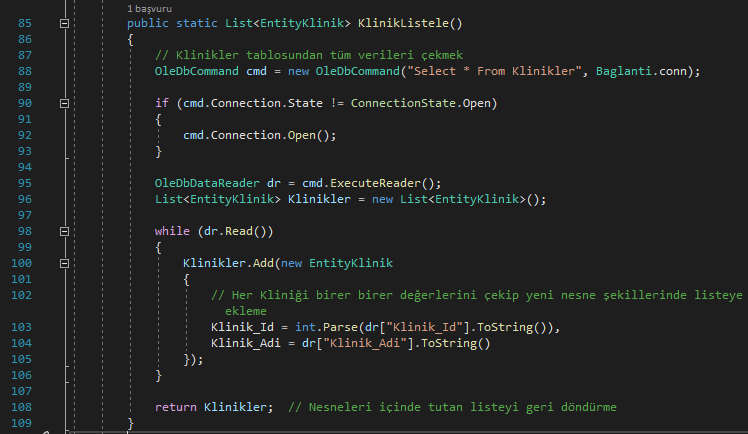
Business Layer katmanından gelen klinik nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz delete olduğunundan silme işlemi yapılmış oluyor. Silme işlemi ise gelen nesnenin Id’sini kullanarak randevuyu bulup gelen kliniği siliyoruz.



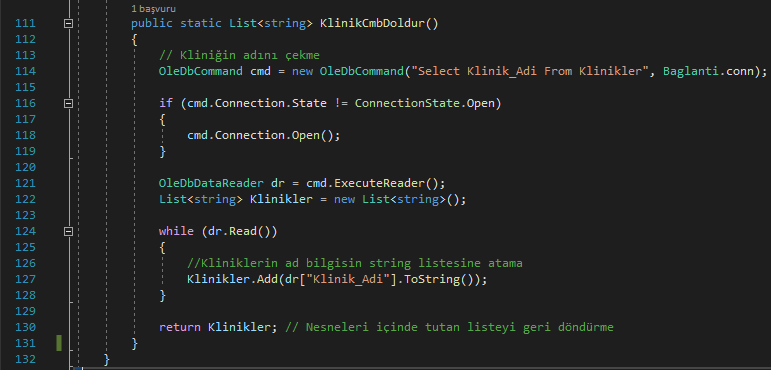
Business Layer katmanından gelen bir Klinik Adımız var. Gelen string değerini parametre olarak sql sorgumuza ekliyoruz. Gelen Klinik Adını sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise gelen Klinik adını kullanarak randevuyu bulup gelen kliniğin Id bilgisini geri yolluyoruz.



Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise bütün klinikleri bulup gelen kliniklerin bilgilerini geri yolluyoruz.

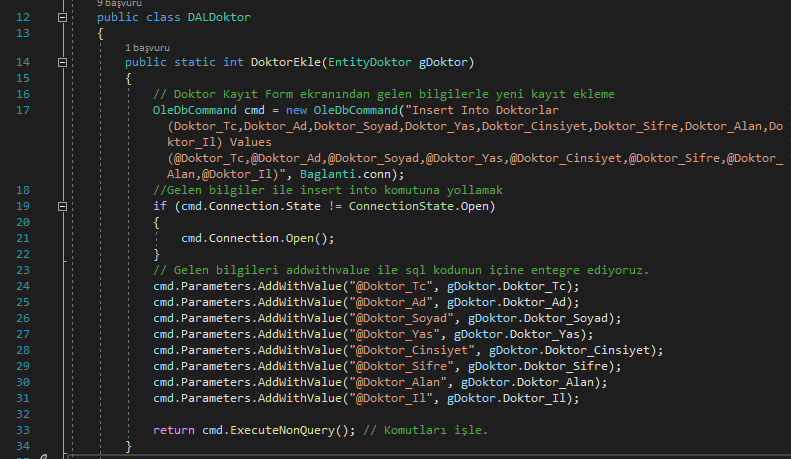


Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise bütün klinikleri bulup gelen kliniklerin sadece Klinik Adı bilgisini geri yolluyoruz ve comboBox içeriğini dolduruyoruz.

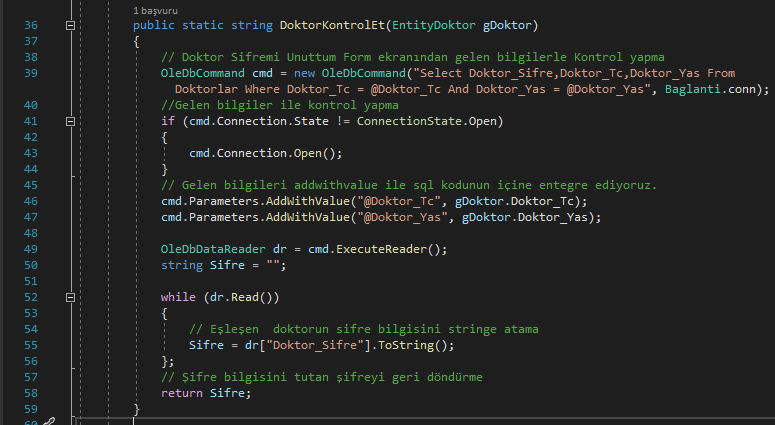


1. **Data Access Doktor**

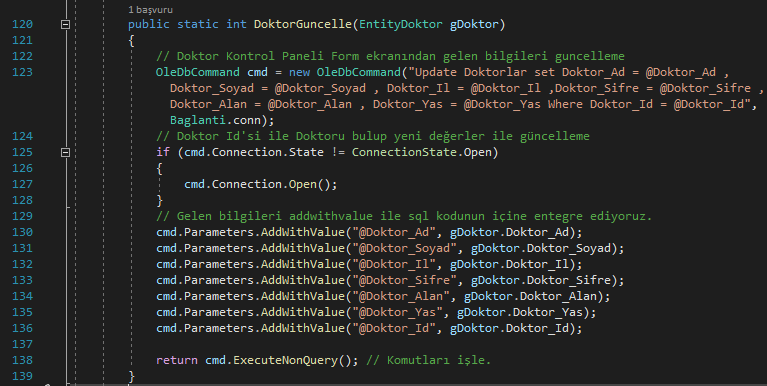
Business Layer katmanından gelen doktor nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz Insert olduğunundan ekleme işlemi yapılmış oluyor.



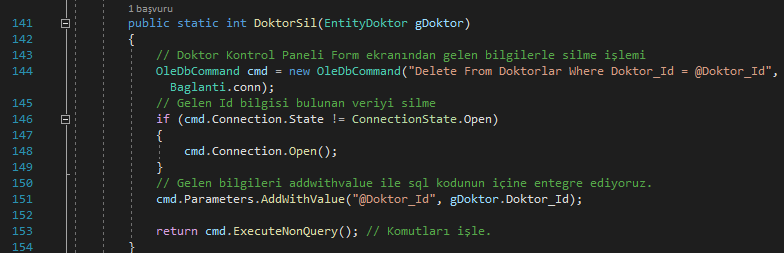
Business Layer katmanından gelen doktor nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz Select olduğunundan veri çekme işlemi yapılmış oluyor.TC ve Yaş bilgisi doğru ise o doktorun şifre bilgisini geri döndürüyor.



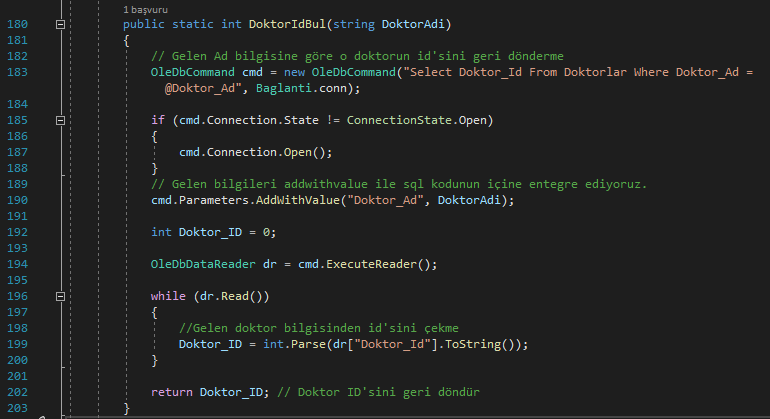
Business Layer katmanından gelen doktor nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz update olduğunundan güncelleme işlemi yapılmış oluyor. Güncellemeyi ise gelen nesnenin Id’sini kullanarak doktoru bulup değerleri değiştiriyoruz.



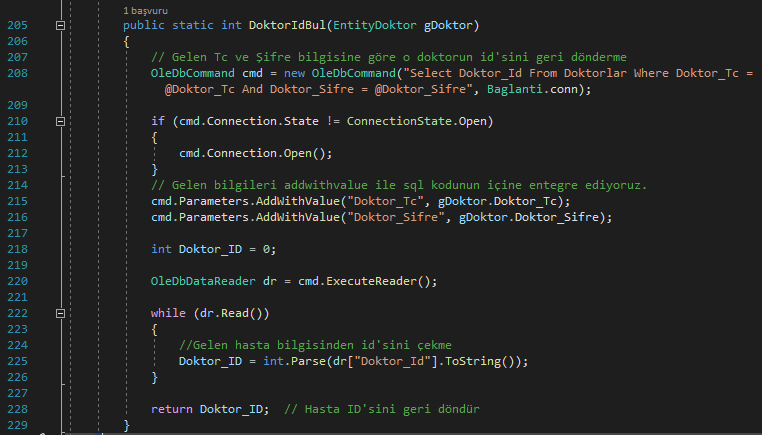
Business Layer katmanından gelen doktor nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz delete olduğunundan silme işlemi yapılmış oluyor. Silme işlemi ise gelen nesnenin Id’sini kullanarak doktoru bulup gelen doktoru siliyoruz.



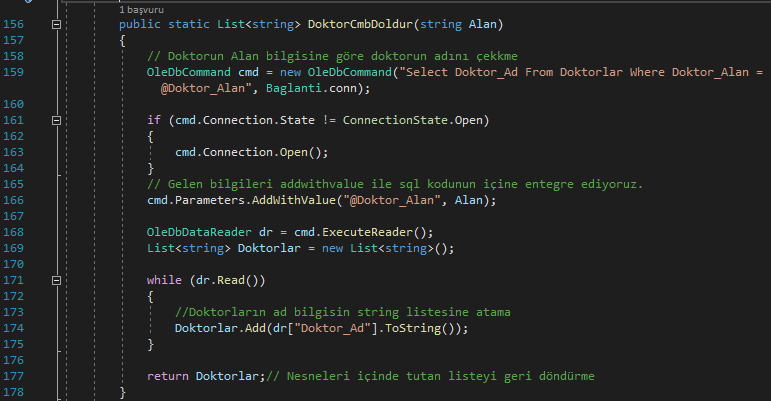
Business Layer katmanından gelen bir Doktor Adımız var. Gelen string değerini parametre olarak sql sorgumuza ekliyoruz. Gelen Doktor Adını sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise gelen doktor adını kullanarak doktoru bulup gelen doktorun Id bilgisini geri yolluyoruz.



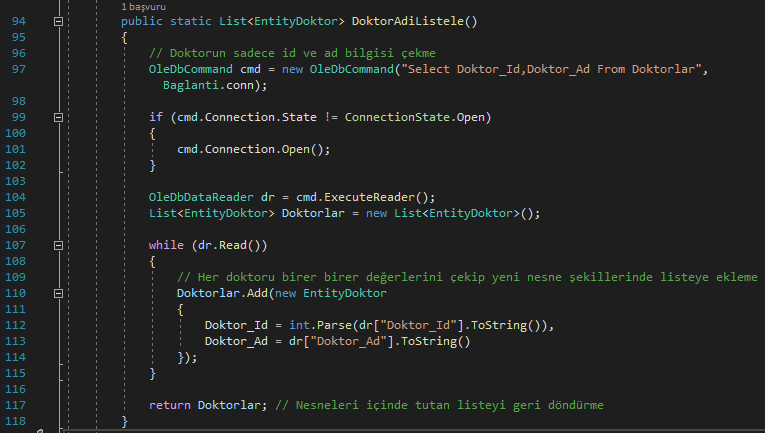
Business Layer katmanından gelen bir Doktor nesnemiz var. Gelen nesnemizin değerlerini parametre olarak sql sorgumuza ekliyoruz. Gelen Doktor Tc Ve Şifresini sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise gelen doktorun Tc’sini ve şifresini kullanarak doktoru bulup gelen doktorun Id bilgisini geri yolluyoruz.



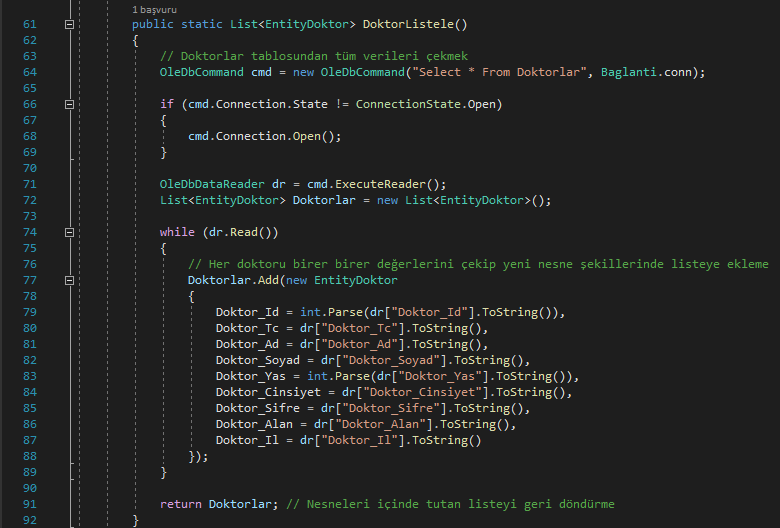
Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise bütün doktorları bulup gelen doktorların sadece Doktor Adı bilgisini geri yolluyoruz ve comboBox içeriğini dolduruyoruz.



Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise bütün doktorları bulup gelen doktorların sadece ad bilgisini geri yolluyoruz.

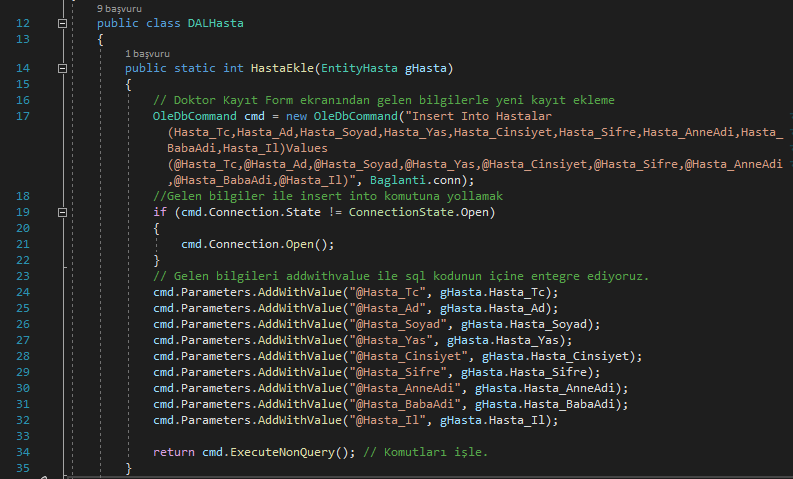


Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise bütün doktorları bulup gelen doktorların bilgilerini geri yolluyoruz.

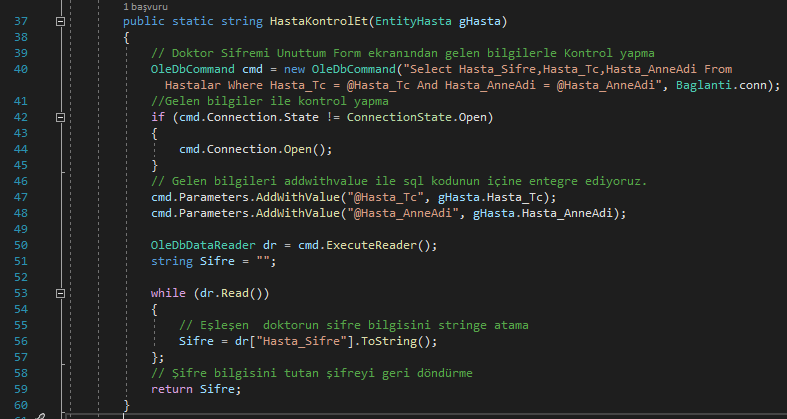


1. **Data Access Hasta**

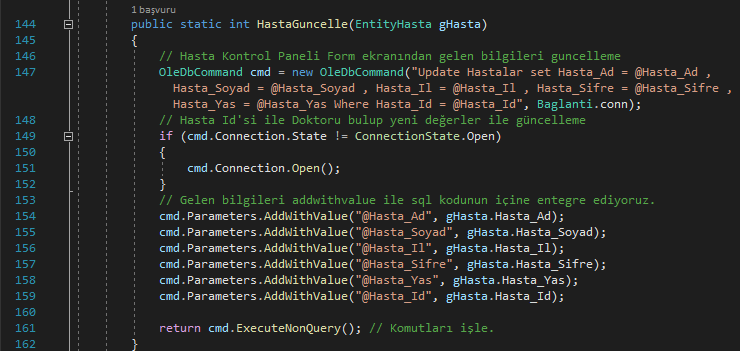
Business Layer katmanından gelen hasta nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz Insert olduğunundan ekleme işlemi yapılmış oluyor.



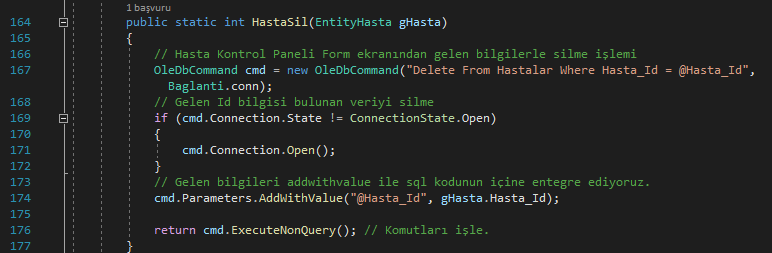
Business Layer katmanından gelen hasta nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz Select olduğunundan veri çekme işlemi yapılmış oluyor. TC ve Anne Adı bilgisi doğru ise o doktorun şifre bilgisini geri döndürüyor.



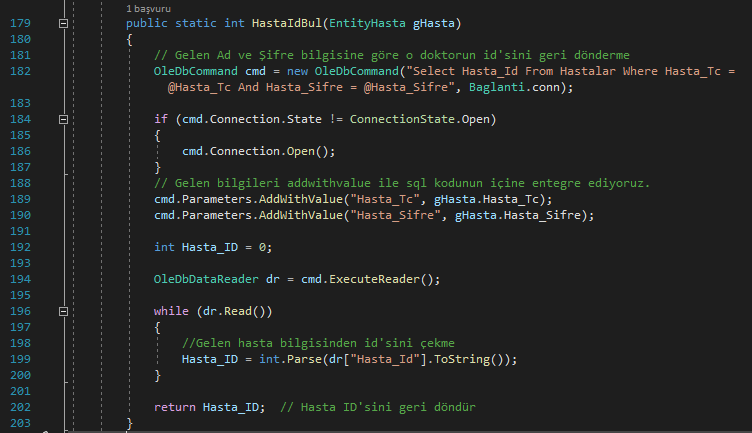
Business Layer katmanından gelen hasta nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz update olduğunundan güncelleme işlemi yapılmış oluyor. Güncellemeyi ise gelen nesnenin Id’sini kullanarak hastayı bulup değerleri değiştiriyoruz.



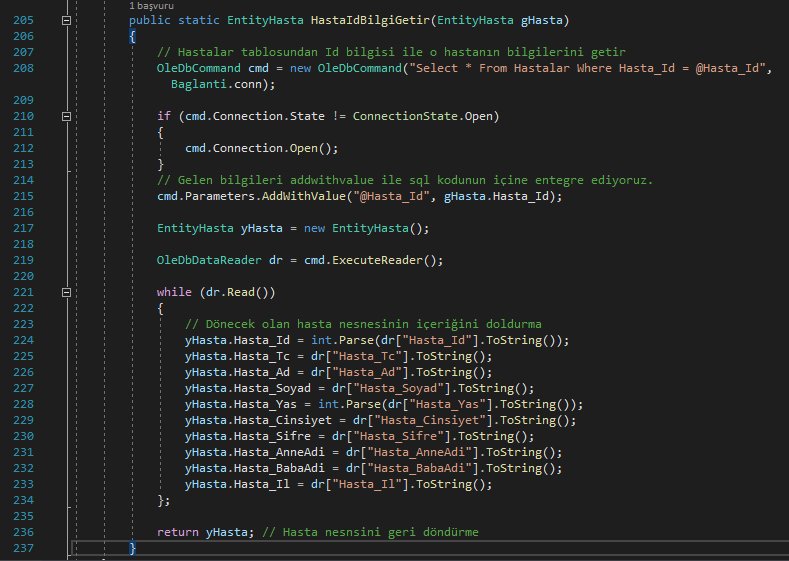
Business Layer katmanından gelen hasta nesnesinin içinde gelen bilgilerini parametre olarak sql sorgumuza ekliyoruz. Gelen bilgileri sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz delete olduğunundan silme işlemi yapılmış oluyor. Silme işlemi ise gelen nesnenin Id’sini kullanarak doktoru bulup gelen hastayı siliyoruz.



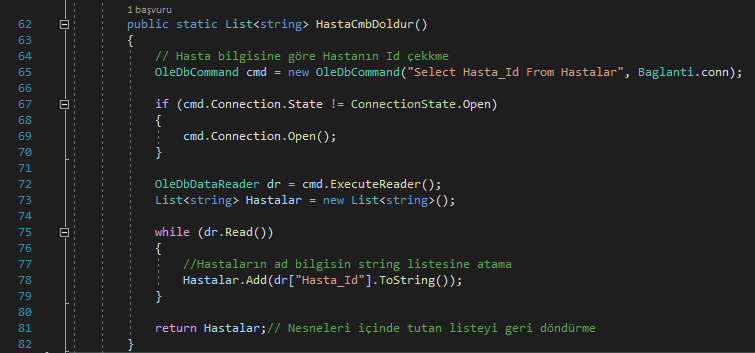
Business Layer katmanından gelen bir hasta nesnemiz var. Gelen nesnemizin değerlerini parametre olarak sql sorgumuza ekliyoruz. Gelen hasta Tc Ve Şifresini sql sorgusuna ekledikten sonra sql komutumuzu çalıştırıyoruz. Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise gelen hastanın Tc’sini ve şifresini kullanarak hastayı bulup gelen hastanın Id bilgisini geri yolluyoruz.



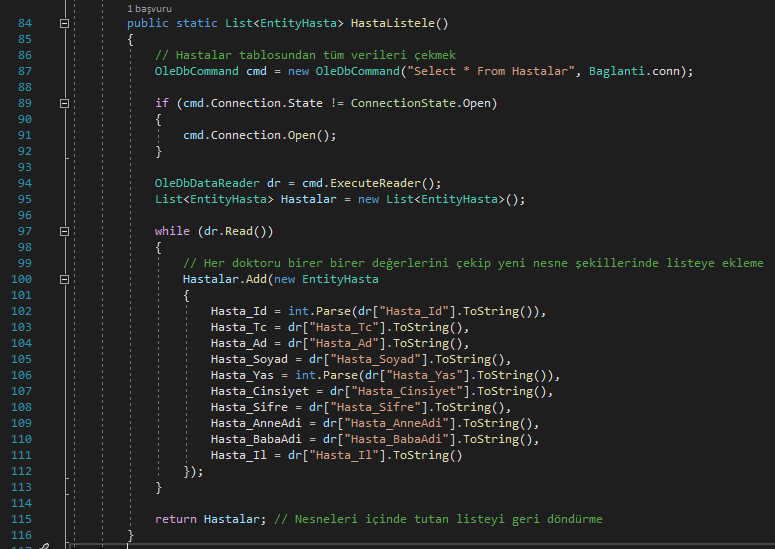
Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise Business Layer’dan gelen hasta id’sini kullanarak bulunan hastanın bütün bilgilerini geri yolluyoruz.



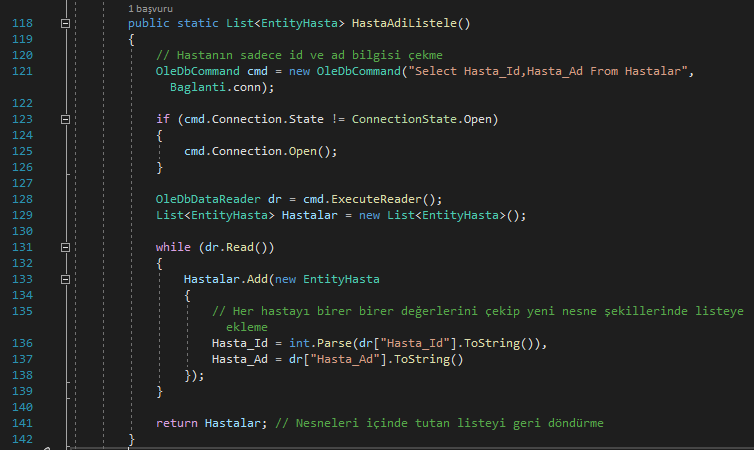
Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise bütün hastaları bulup gelen hastaların sadece Hasta Id bilgisini geri yolluyoruz ve comboBox içeriğini dolduruyoruz.



Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise bütün hastaları bulup gelen hastaların bilgilerini geri yolluyoruz.



Buradaki komutumuz select olduğunundan okuma işlemi yapılmış oluyor. Veri çekme işlemi ise bütün hastaları bulup gelen hastaların sadece ad bilgisini geri yolluyoruz.



Katmanlı mimari nedir? Konusundan anlatmaya başlayarak bir database nasıl oluşturulur? Katmanlı mimari projesi nasıl açılır? Hangi katmanları kullandık? Ne programı yaptık? Gibi sorulara karşılık verebilecek şekilde bir katmanlı mimari projesi yaptık. Bu proje Hastane Otomasyonuydu. Database’de 6 farklı tablo programda ise 4 farklı katman kullandık. Size yardımcı olabilecek bir sunum hazırlamaya çalıştım. Şimdiden kolay gelsin.