



**T.C.
DÜZCE ÜNİVERSİTESİ
İŞLETME FAKÜLTESİ**

YÖNETİM BİLİŞİM SİSTEMLERİ LİSANS PROGRAMI

NESNE TABANLI PROGRAMLAMA PROJE RAPORU

Hazırlayanlar

**222211203 Aslı TEKE
222211074 Büşra DİLŞEN
212211015 Elif Hilal KARAKAŞ
222211076 Sümeyra KIZILCAOĞLU
222211073 Ömer Faruk ARSLAN**

Github linki: <https://github.com/O-F-A/HastaneProjesi.git>

**Ders Sorumlusu
Dr. Öğr. Üyesi Ali Akaytay**

Hastane Projesi Dokümantasyonu

1. Projenin Gereksinimleri, Amacı ve Kullanımı

Projenin Amacı:

Hastane Projesi, hastaların kayıt işlemleri, doktor randevuları, duyuru yönetimi ve kullanıcıların sağlık bilgilerine erişimlerini sağlayan bir sağlık yönetim sistemidir. Bu proje, hastaların hastaneye kaydını, randevu oluşturulmasını ve duyuruların yönetilmesini dijital ortamda yapmayı amaçlar.

Proje Gereksinimleri:

- **Hastaların Kayıt İşlemi:** Hastaların ad, soyad, TC kimlik numarası, telefon numarası, şifre ve cinsiyet gibi bilgileri sisteme kaydedilmelidir.
- **Randevu Sistemi:** Hastalar, doktorlar ve branşlar seçilerek randevu oluşturulabilmelidir.
- **Duyuru Sistemi:** Sekreterler hastalar için duyurular ekleyip yönetebilmelidir.
- **Kullanıcı Giriş Sistemi:** Sistemde kullanıcılar (hastalar, sekreterler) kendi bilgileriyle giriş yapabilmelidir.

Kullanım:

1. **Hasta Kaydı:** Hastalar, hasta kayıt formunu doldurarak sisteme kaydolabilir.
2. **Randevu Alma:** Hasta, sekreter aracılığıyla veya sistemdeki arayüzle randevu alabilir.
3. **Duyuru Takibi:** Sekreter, hastalar için duyuru ekleyebilir ve hastalar bu duyurulara erişebilir.

2. Kurulum ve Çalıştırma Talimatları

Kurulum Talimatları:

1. Gerekli Yazılımların Kurulumu:

- Visual Studio IDE'sini indirin ve kurulum işlemini tamamlayın. (İndirme linki: [Visual Studio](#))

- SQL Server Management Studio (SSMS) yazılımını bilgisayarınıza yükleyin. ([SSMS İndirme Linki](#))
2. **Veritabanının Oluşturulması:**
- SQL Server üzerinden **HastaneProje** adlı bir veritabanı oluşturun.
 - Veritabanında aşağıdaki tabloları oluşturun:
- Tbl_Hastalar:** Hasta bilgilerini saklar.
Tbl_Randevular: Randevu detaylarını içerir.
Tbl_Duyurular: Sekreterlerin eklediği duyuruları saklar.
- Her tablonun gerekli alanlarını ve ilişkilerini düzenleyin.
3. **Proje Dosyalarının İndirilmesi ve Açılması:**
- Proje ZIP dosyasını bilgisayarınıza indirin ve bir klasöre çıkarın.
 - Visual Studio'da "Open Project" seçeneğiyle proje dosyasını açın.

Çalıştırma Talimatları:

1. Proje, Visual Studio üzerinden başlatılır.
2. Sağ üst köşedeki "Start" butonuna tıklanır.
3. Uygulama başarıyla çalıştırıldıktan sonra giriş ekranı kullanıcıya gösterilir.

3.Sistem Gereksinimleri

Hastane Projesi'nin sorunsuz bir şekilde çalışabilmesi için hem donanım hem de yazılım açısından bazı gereksinimlerin karşılanması gerekmektedir. Sistem, en az Windows 7 veya üstü bir işletim sistemi gerektirirken, 4 GB RAM ve 1 GHz hızında bir işlemci ile stabil bir şekilde çalışabilir. Yazılım tarafında ise Visual Studio 2019 ve SQL Server 2014 veya üzeri sürümlerin kurulu olması gerekir.

Gereksinim Türü	Gereksinimler
Donanım Gereksinimleri	
İşletim Sistemi	Windows 7 veya daha yeni bir sürüm
RAM	Minimum 4GB
İşlemci	1GHz veya daha hızlı işlemci
Depolama Alanı	500mb boş disk alanı
Yazılım Gereksinimleri	
IDE	Microsoft Visual Studio2019 veya daha yeni sürüm
Veritabanı Yönetimi	SQL Server 2014 veya daha yeni sürüm
Framework	.NET Framework 4.7 veya yüksek sürüm

4. Sınıf ve Metotların Tanımı

Proje, nesne tabanlı programlama prensipleriyle geliştirilmiş ve birden fazla sınıf içerir. Hasta sınıfı, ad, soyad, TC kimlik numarası gibi bilgileri depolarken, randevu sınıfı tarih, saat, branş ve doktor bilgilerini içerir. Sekreter sınıfı ise duyuru ve randevu yönetimi işlevlerini üstlenir. Örneğin, Hasta.Kaydet() metodu, hastanın bilgilerini veritabanına kaydederken, Sekreter.DuyuruEkle() metodu yeni bir duyuru ekler.

Sınıf Adı	Özellikler	Metodlar
Hasta	Ad, Soyad, TcKimlikNo, TelefonNo, Sifre, Cinsiyet	Kaydet(): Hasta bilgilerini veritabanına kaydeder.
Randevu	Tarih, Saat, Brans, Doktor	Kaydet(): Randevu bilgisini veritabanına kaydeder.
Sekreter	AdSoyad, TcKimlikNo	RandevuEkle(Randevu randevu): Yeni bir randevu ekler. DuyuruEkle(string duyuruMetni): Hastalar için duyuru ekler.
Doktor	AdSoyad, Branş	MuayeneEt(Hasta hasta): Muayene işlemini başlatır.

5.Kod Yorumlaması

Inheritance (Miras Alma)

Inheritance, nesne tabanlı programlamanın temel stratejilerinden biridir. Bu ilke, bir sınıfın başka bir sınıfın özelliklerini ve bunların ayrılmasını sağlar. C# dilinde miras alma, kodun daha düzenli, tekrarsız ve genişletilebilir bir yapıya sahip olabilmesi olanaklarını tanıma olanağı sağlar. Üst sınıf (temel sınıf) özellikleri ve yöntemleri, alt sınıflar tarafından miras yoluyla temin edilebilir.

Projemizde Miras Almanın Kullanımı

Bu projenin miras alma ilkesi, verilerin saklanması ve kullanılabilirliğinin sağlanması amacıyla sağlanmıştır. Kodda yer alan degiskenler sınıfı, temel bir sınıf olarak tanımlanmış ve diğer sınıflara miras verilmiştir. Bu tasarım, başka bir sınıfı kullanarak degiskenler sınıfta farklılaşan seçenekleri erişim sağlamak için tercih edilmiştir.

```
degiskenler.cs
HastaneProjesi
HastaneProjesi.degiskenler
tckimlikno

1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using System.Windows.Forms;
7
8 namespace HastaneProjesi
9 {
10     // degiskenler sınıfı, hastane projesinde kişisel bilgileri saklamak için kullanılan bir sınıftır
11     public class degiskenler
12     {
13         // TC kimlik numarasını saklayan değişken
14         public string tckimlikno;
15
16         // Kişinin ad ve soyad bilgisini saklayan değişken
17         public string adsoyad;
18     }
19 }
20
21
```

Kodda, **degiskenler** sınıfı şu özellikleri içermektedir:

- **TC Kimlik Numarası** (tckimlikno)
- **Ad Soyad** (adsoyad)

Bu sınıf, **bildirim** adı verilen başka bir sınıfa miras olarak verilmiştir. Bu sayede **bildirim** sınıfını kullanarak doğrudan degiskenler sınıfı tanımlı değişkenlere erişim aralığı. Böyle bir yapı, kodun tekrarını önler ve kullanımı daha basit hale getirir. Bu tür bir yapı, kod tekrarını ve hastanedeki kişilerin aralarındaki ortak veri yönetimini hem genişletilebilirliği hem de genişletilebilirliği sağlar.

Polimorfizm (Çok Biçimlilik)

Polimorfizm (Çok Biçimlilik), nesnel programlamanın (OOP) ilkelerinin temellerinden biridir ve aynı isimli bir fonksiyonun, farklı türev sınıflarında farklı davranışlar sergilemesine olanak tanır. Bu ilke, yazılımın geliştirme sürecinde esneklik sağlayarak kodun genişletilebilirliğini artırır.

Polimorfizm Türleri

1. **Statik Polimorfizm:** Derleme zamanında (derleme zamanı) kaydedilir.
2. **Dinamik Polimorfizm:** Çalışma zamanında (runtime) kaydedilir. Bu, genellikle **override** ile sağlanan ve sanal yöntemler (sanal yöntemler) ile sona erer.

Projemizde Polimorfizm Kullanımı

Bu projede polimorfizm, **bildirim** adı verilen temel sınıfta tanımlanmış olarak adlandırılan **msjcagir** sanal (sanal) yöntem aracılığıyla sağlanır. Bu metot, türetilen sınıflarda override anahtarlanabilir, özelleştirilmiş ve kendi sınıfında kendi özelliklerine uygun davranışlar korunmuştur.

```
bildirim.cs
HastaneProjesi
HastaneProjesi.ozelBildirimSekreter
msjcagir(string X, string Y)

1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using System.Windows.Forms;
7
8 namespace HastaneProjesi
9 {
10     // Temel sınıf
11     public class bildirim : degiskenler
12     {
13         // Bu metod sanal (virtual) olarak tanımlandı, böylece türetilmiş sınıflar tarafından override edilebilir.
14         public virtual void msjcagir(string X, string Y)
15         {
16             MessageBox.Show("Sayın Hasta " + X + " TC: " + Y + " Hoşgeldiniz \nSisteminize giriş yaptınız. \n GEÇİŞ OLSUN", "BİR MESAJIN VAR", MessageBoxButtons.OK, MessageBoxIcon.Information);
17         }
18     }
19
20     // Türetilmiş sınıf: Sekreter için özel mesaj
21     public class ozelBildirimSekreter : bildirim
22     {
23         // override anahtar kelimesi ile temel sınıftaki metod özelleştirilmiştir
24         public override void msjcagir(string X, string Y)
25         {
26             MessageBox.Show("Sayın Sekreter " + X + " TC: " + Y + " Hoşgeldiniz \nSisteminize giriş yaptınız. \n HOLAY GELSİN", "BİR MESAJIN VAR", MessageBoxButtons.OK, MessageBoxIcon.Information);
27         }
28     }
29
30     // Türetilmiş sınıf: Doktor için özel mesaj
31     public class ozelBildirimDoktor : bildirim
32     {
33         // override anahtar kelimesi ile temel sınıftaki metod özelleştirilmiştir
34         public override void msjcagir(string X, string Y)
35         {
36             MessageBox.Show("Sayın Doktor " + X + " TC: " + Y + " Hoşgeldiniz \nSisteminize giriş yaptınız. \n HOLAY GELSİN", "BİR MESAJIN VAR", MessageBoxButtons.OK, MessageBoxIcon.Information);
37         }
38     }
39 }
40
41
```

Koddaki bileşenler şu özellikleri içermektedir;

1. Temel Sınıf (bildirim):

- Msjcagir Yöntem, temel sınıfta genel bir davranışı sergileyecek şekilde gösterilebilir.
- Bu yöntem, bir kişinin mesaj göstermek amacıyla kullanılmaktadır.

2. Türetilmiş Sınıflar (ozelBildirimSekreter ve ozelBildirimDoktor):

- Her iki sınıf, temel sınıftan miras alır.
- Msjcagir Metodun override anahtarlama yöntemi her sınıfta özelleştirilmiştir
- Sekreter ve doktor için özel mesajlar yer alıyor.

Abstract (Soyutlama)

Soyutlama, nesne ayrıntıları programlamanın temel prensiplerinden biridir. Yazılımın karmaşıklığı ve kodun yeniden kullanılabilirliğini artırmak için kullanılan bu ilke, gereksiz ayrıntılardan kaçınarak yalnızca önemli odaklanmayı sağlar. Bu sağlamanın somut bir örneği, soyut sınıflar (**soyut sınıflar**) ve soyut metotlar (**soyut yöntemler**) kullanımımızdır.

Soyut Sınıfların Özellikleri

1. Genel Yapıyı Tanımlar:

- Soyut sınıflar, bir sınıfın sahip olması gereken temel özellikler ve metotları içerir.
- Ancak soyut sınıflardaki soyut metotların içeriği doldurulmaz. Bu yöntemler, türetilen sınıflarda tanınmak zorundadır.

2. Tamamlanmamış Metotlar:

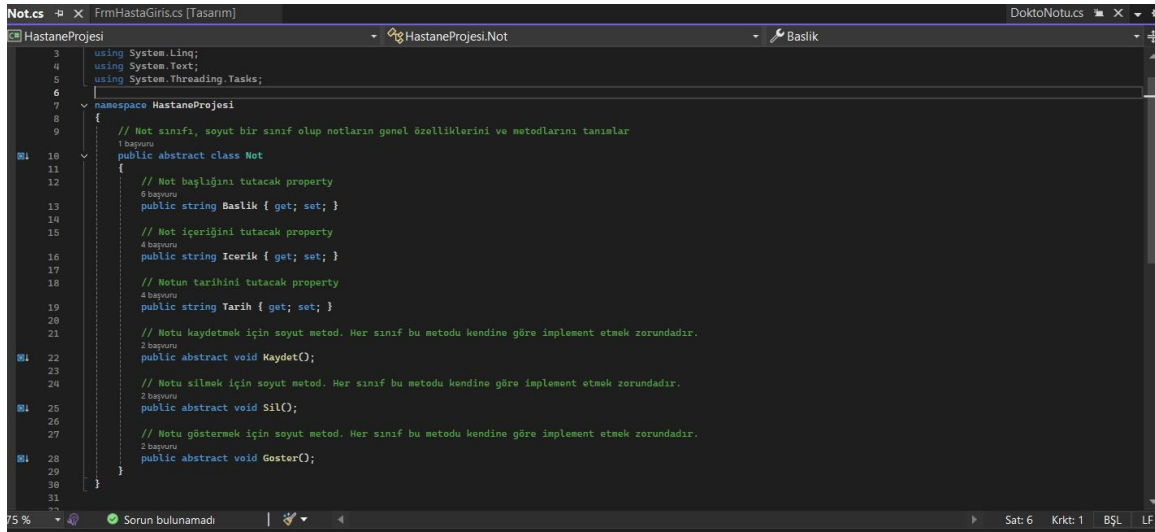
- a. Soyut metotlar, yalnızca imza ile baskı ve türemiş sınıflar tarafından **çoğaltılarak** çoğaltılır.

3. Türetim Şartı:

- a. Soyut sınıflar, doğrudan bir nesne oluşturmak için üretilemez. Ancak türetilmiş bir sınıf üzerinden kullanılır.

Projede Soyut Sınıf Kullanımı

Proje kapsamında, **Not** genel olarak bir soyut sınıf oluşturulmuş ve aşağıdaki gibi genel özellikler ile yöntemler mevcuttur:



```
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace HastaneProjesi
8 {
9     // Not sınıfı, soyut bir sınıf olup notların genel özelliklerini ve metodlarını tanımlar
10     public abstract class Not
11     {
12         // Not başlığını tutacak property
13         public string Baslik { get; set; }
14
15         // Not içeriğini tutacak property
16         public string Icerik { get; set; }
17
18         // Notun tarihini tutacak property
19         public string Tarih { get; set; }
20
21         // Notu kaydetmek için soyut metod. Her sınıf bu metodu kendine göre implement etmek zorundadır.
22         public abstract void Kaydet();
23
24         // Notu silmek için soyut metod. Her sınıf bu metodu kendine göre implement etmek zorundadır.
25         public abstract void Sil();
26
27         // Notu göstermek için soyut metod. Her sınıf bu metodu kendine göre implement etmek zorundadır.
28         public abstract void Goster();
29     }
30 }
31
```

Soyutlama Sınıfı Kullanımı: DoktorNotSınıf

Not sınıftan türetilen **DoktorNot** sınıfı, soyut metotların kendine özgü olanları var. Örneğin, bir kayıt notları için kaydetme, silme veya görüntülemeyi bu sınıfta belirler.

DoktorNot Sınıfının İşlevleri:

1. **Kaydetme:** Doktor notlarının belirli bir formata uygun şekilde kaydedilmesini sağlar.
2. **Silme:** Doktorların sistemden ayrılmaları için gerekli işlemleri içerir.
3. **Görüntüleme:** Doktor notlarını bir kullanıcı veya bir dosyaya gösterir.

Aynı zamanda bu kodda kapsülleme örneği de bulunmaktadır.

Kapsülleme, nesnel programlamanın temel ilkelerinden biri olup, kayıtlı ve bu erişime olanak sağlayan metotların bir sınıf içinde bir araya getirilmesini sağlar. Bu ilke, veriyi elde etmek ve pazarlar arası bağımlılıkları azaltmak için kullanılır.

```
// Not başlığını tutacak property
6 başvuru
public string Baslik { get; set; }

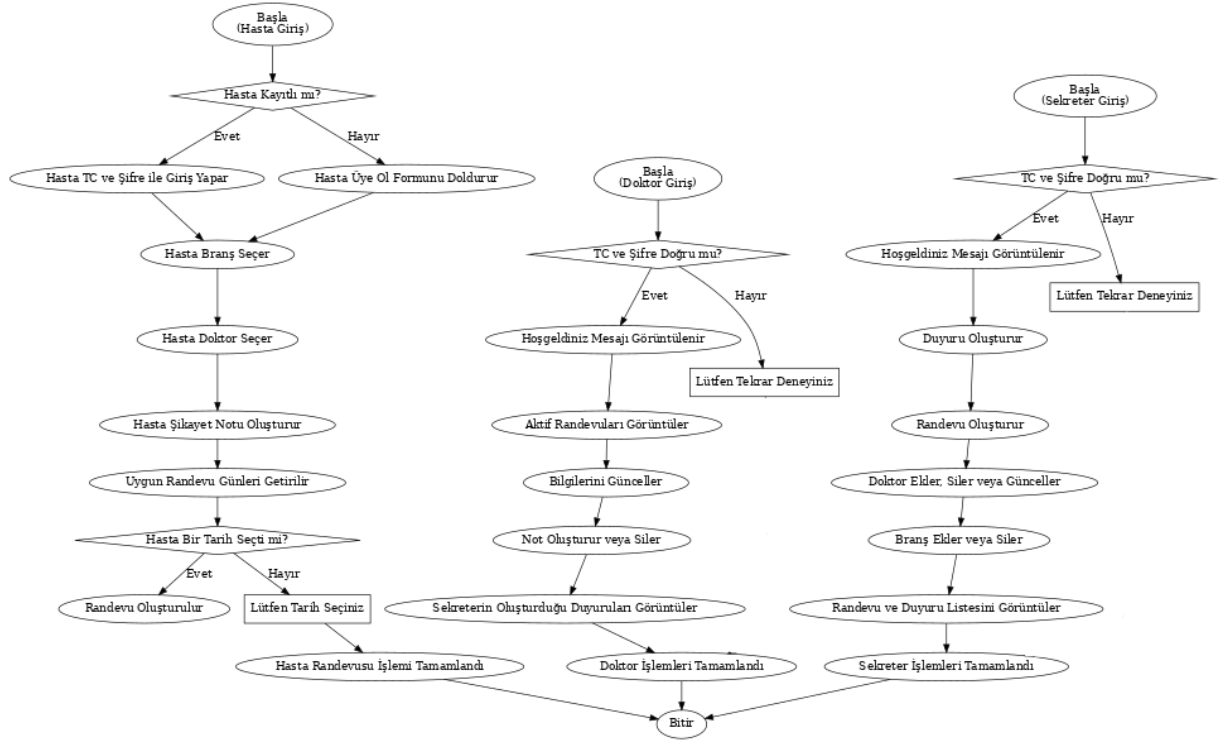
// Not içeriğini tutacak property
4 başvuru
public string Icerik { get; set; }

// Notun tarihini tutacak property
4 başvuru
public string Tarih { get; set; }
```

6.Akış Diyagramları

Akış Diyagramı Nedir?

Akış diyagramlarının amacı, bir süreci, sistemi veya işlemi adım adım görselleştirerek anlaşılır bir şekilde sunmaktır. Bu diyagramlar, bir işlemin nasıl ilerlediğini, karar noktalarını, alternatif yolları ve sürecin tamamlanması için izlenmesi gereken adımları gösterir. Akış diyagramları, karmaşık süreçlerin daha kolay analiz edilmesine, planlanmasına ve iyileştirilmesine yardımcı olur. Ayrıca ekipler arasında iletişimi güçlendirir, süreçlerdeki potansiyel hataları tespit etmeyi kolaylaştırır ve iş akışını optimize etmek için temel bir araç olarak kullanılır. Hem teknik hem de teknik olmayan kullanıcılar için sürecin görsel bir özetini sunar.



7.Kullanıcı Arayüzü ve Navigasyon

Hastane Projesi kullanıcı dostu bir grafik arayüz ile hastalar, doktorlar ve sekreterler için kolay erişim sağlar. Ana menü her kullanıcı grubunun ihtiyaçlarına göre özelleştirilmiş alt bölümlere ayrılmıştır.

Bölüm	Açıklama	Ana İşlevler
Ana Giriş Ekranı	Kullanıcılar, TC kimlik numaralarını ve şifrelerini girerek sisteme giriş yapabilir.	Kayıt Ol: Yeni kullanıcı kaydı. Şifremi Unuttum: Şifre kurtarma.
Hasta Paneli	Hastalar, giriş yaptıktan sonra sistemin özelliklerine erişim sağlayabilir.	Randevu Al: Branş ve doktor seçerek randevu oluşturma. Duyurular: Güncel hastane duyurularını görüntüleme.
Sekreter Paneli	Sekreterler, sistemde randevu ve duyuru yönetimi yapabilir.	Randevu Yönetimi: Tarih, saat ve doktor bilgilerini düzenleme. Duyuru Yönetimi: Yeni duyurular ekleme ve mevcut duyuruları düzenleme.
Navigasyon Özellikleri	Kullanıcıların sistemde kolaylıkla gezinebileceği araçları içerir.	Menü Çubuğu: Kullanıcıların hızlı geçiş yapmasını sağlar. Ana Menü Bağlantısı: Her sayfada ana menüye dönme seçeneği.

8.Dış Kütüphaneler ve Bağımlılıklar

System.Data.SqlClient:

- SQL Server ile bağlantı kurmak ve veri alışverişi yapmak için kullanılır. Veritabanı işlemleri için kritik öneme sahiptir.

System.Windows.Forms:

- Grafik kullanıcı arayüzü (GUI) tasarımı için kullanılır. Formlar, butonlar, textbox gibi görsel bileşenlerin oluşturulmasını sağlar.

.NET Framework:

- Projenin temel çalışma ortamını sağlar. Gelişmiş işlevsellik ve performans için .NET Framework 4.7 veya daha üstü önerilir.

9. Gelecekteki Geliştirme Önerileri

Projede gelecekte uygulanabilecek geliştirmeler arasında, hastalara yönelik raporlama ve analiz modüllerinin eklenmesi öne çıkmaktadır. Ayrıca, SMS tabanlı hatırlatma sistemi, hastalara yaklaşan randevuları hakkında bilgi verebilir. Mobil uyumluluk ve çoklu dil desteği gibi özellikler, kullanıcı deneyimini artırabilir. Doktorların hasta geçmişine erişebileceği ve muayene notları tutabileceği bir modül eklenmesi de sistemin fonksiyonelliğini artıracaktır. Bu öneriler, projeyi daha kullanıcı dostu ve kapsamlı bir sağlık yönetim sistemine dönüştürme potansiyeline sahiptir.

