

DÜZCE ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

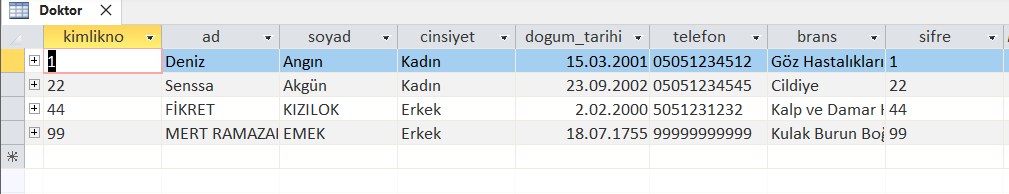
BİLGİSAYAR MÜHENDİSLİĞİ

GÖRSEL PROGRAMLAMA DERSİ

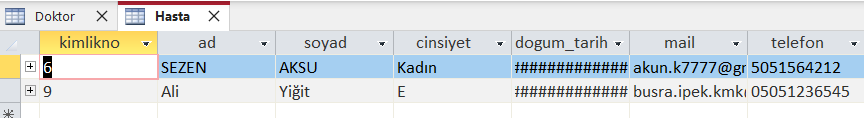
İbrahim GÜLLÜOĞLU:

AMAÇ: Bu programda doktor-hasta randevu ve görüşme kayıt işlemlerinin yapılması temel ihtiyaçtır. Bu programda sekreter, doktor ve hasta bulunmaktadır. Programın amacı randevunun olabildiğinde hızlı bir şekilde alınabilmesi ve bu randevunun mevcut sistemler içerisinde izlenebilmesini amaçlamaktadır. Birden çok bölümden randevu almayı sağlarken bu randevuların güncellenebilmesi veya silinebilmesi de bir o kadar kolaylaşmaktadır.

DATABASE



Microsoft Access üzerinde yapılan Database fazlasıyla ufak olmakla birlikte çok fazla işlem gerektiren bir Database değildir. Veri tabanı içerisinde veriler ilk halinden alınmıştır Tam hali ise sonraki sayfalarda raporlanacaktır.

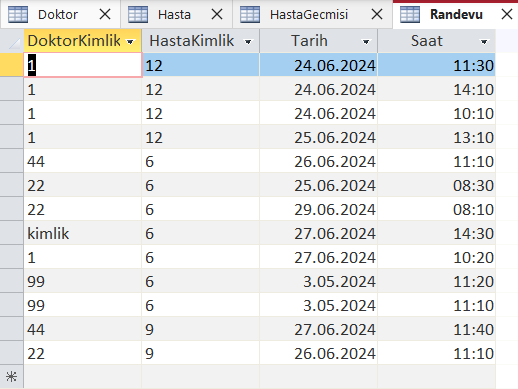
Bu kısımda Doktor tablosu gözükmektedir. Burada yazan veriler arayüz üzerinden uygulama yoluyla girilmiş olup el ile girme yapılmamıştır. Bu mevcut tablo içerisinde ad, soyad, cinsiyet, dogum\_tarihi, telefon, branş, şifresi açıkça görülmektedir. 

Bu tabloda hastalar gösterilmektedir. Bu hastalar daha sonra çoğulcu olarak güncellenecektir. Eğer hasta bildirim almak isterse mail ’de gönderilebilmektedir veya telefonla arama yöntemi de kullanılabilir.

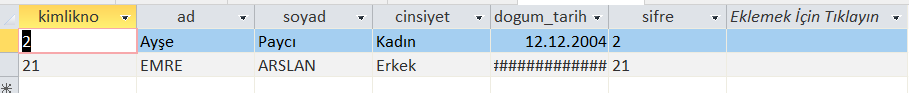
metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yukarıda görülen tabloda açıkça HastaGecmisi tablosudur. Bu tablo içerisinde hasta hakkında bilgiler verilmektedir. Deneme amaçlı yapılan örneklerde tanı gorus ve tahlil boş bırakılmıştır. Mevcut durumda bu kısımların düzgün olduğu görülmektedir. Aynı zamanda boş geçilemez yapılacaktır.

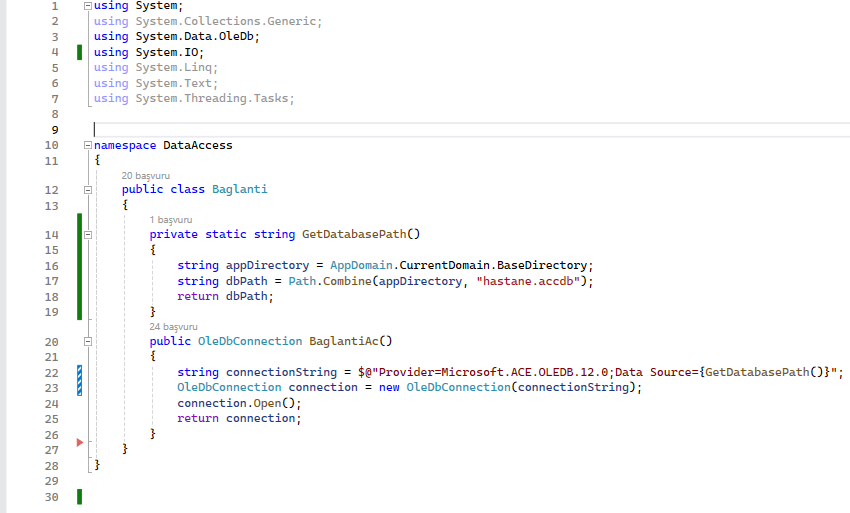


Yukarıda verilen randevu tablosu gösterilmektedir. Randevular uygulama içerisinde daha düzgün bir şekilde gözükmekle beraber eski randevularda gözükmektedir. Aynı zaman diliminde randevu seçilememektedir. Bu kullanıcı eliyle özellikle engellenmiştir. Aynı gün içerisinde birden fazla randevu seçilebilmektedir. Bu randevu sistemi MHRS’de(Merkezi Hasta Randevu Sistemi) bulunduğu için birden çok randevu seçme hakkı kullanıcıdan alınmamıştır.



Tam yukarıda sekretere ait bilgiler verilmiştir. Bu bilgiler kullanılarak sekreter ataması ve eklemesi yapılabilmektedir. Sekreter sistemin tamamına hâkim olmakla beraber doktordan daha fazla yetkiyle donatılmıştır.

BUSİNESS VE ACCESS CONNECTİON



Baglanti Sınıfı: Baglanti adında bir sınıf tanımlanmıştır.

GetDatabasePath Metodu: Bu metot, veri tabanı dosyasının yolunu belirler. AppDomain.CurrentDomain.BaseDirectory yöntemi, uygulamanın çalıştığı dizini alır. Ardından, Path.Combine yöntemiyle bu dizine ve veri tabanı dosyasının adına (hastane.accdb) göre tam dosya yolunu oluşturur ve döndürür.

BaglantiAc Metodu: Bu metot, veri tabanına bir bağlantı açar. İlk olarak, bağlantı dizgesi connectionString oluşturulur. Bu dize, Microsoft Access veri tabanına erişmek için gerekli olan bağlantı bilgilerini içerir. Bağlantı dizesinde Provider=Microsoft.ACE.OLEDB.12.0 ile Microsoft Access veri sağlayıcısının kullanılacağı belirtilir. Data Source ile veri tabanının dosya yoluna, GetDatabasePath metodu aracılığıyla ulaşılır. Sonra, OleDbConnection sınıfından bir connection örneği oluşturulur ve bu bağlantı açılır (connection.Open()). Bağlantı nesnesi ardından çağrıldığı yere döndürülür.

Bu kod, bir Microsoft Access veritabanına bağlanmak için kullanılan basit bir sınıfı tanımlar. Bu sınıf, veritabanı işlemlerini gerçekleştirmek için kullanılabilir, örneğin veri eklemek, silmek, güncellemek veya sorgulamak gibi işlemler. Ancak, bu sınıf veritabanı bağlantısını açar ancak kapatmaz, bu nedenle gerçek uygulamada bağlantının düzgün şekilde kapatılması önemlidir, genellikle using blokları veya Dispose yöntemi kullanılarak yapılır.

metin, ekran görüntüsü, doküman, belge, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduNamespace ve Sınıf Tanımı:

Kod, "Business" adında bir namespace içine yerleştirilmiştir.

DataGrid adında bir sınıf tanımlanmıştır.

Üye Değişkenler:

baglanti, doktor, hasta ve sekreter adında dört özel üye değişken tanımlanmıştır. Bu değişkenler, DataAccess isim alanındaki sınıfların nesnelerini içerir. Bu sınıflar, veritabanı bağlantısını ve veri erişim işlemlerini gerçekleştirir. Burada yapılan kod detayı sadece Business için yapılmış olup Doktor için gerekli bilgileri aktarmakta ve işlemektedir. Bu bilgiler ve metotlar sırasıyla başlıklar altında aşağıda anlatılmaktadır.

Metotlar:

filldatagriddoktor(): "Doktor" tablosundan tüm verileri çeker ve bir DataTable nesnesinde döndürür.

filldatagridhasta(): "Hasta" tablosundan tüm verileri çeker ve bir DataTable nesnesinde döndürür.

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldufilldatagridsekreter(): "Sekreter" tablosundan tüm verileri çeker ve bir DataTable nesnesinde döndürür.

filldatagridhastagecmis(string kimlik): Belirli bir hasta kimliği (kimlik) ile ilişkilendirilmiş geçmiş verileri ("HastaGecmisi" tablosundan) çeker ve bir DataTable nesnesinde döndürür.

filldatagridhastarandevu(string DoktorKimlik , DateTime Tarih): Belirli bir doktor kimliği (DoktorKimlik) ve tarih (Tarih) ile ilişkilendirilmiş randevu verilerini ("Randevu" tablosundan) çeker ve bir DataTable nesnesinde döndürür.

Veritabanı Bağlantısı ve Sorgular:

Her bir "fill" metodu, veritabanı bağlantısını açar (baglanti.BaglantiAc()), ilgili sorguyu yürütür ve sonuçları bir DataTable içine doldurur.

Metotlar, bir OleDbConnection nesnesi kullanarak veritabanına bağlanır ve bir OleDbCommand nesnesiyle SQL sorgularını yürütür. Veritabanı bağlantısı using bloğu içinde tanımlanmıştır, böylece işlem tamamlandıktan sonra otomatik olarak kapatılır.

Parametreli sorgular, SQL enjeksiyon saldırılarını önlemek için kullanıcı girdilerini güvenli bir şekilde işler. Bu sorgular @kimlik ve @DoktorKimlik, @Tarih gibi parametrelerle filtrelenir.

Hata İşleme:

Veritabanı işlemlerinde hata oluşursa, catch bloğu içindeki kod çalışır. Ancak, bu durumda sadece konsola "Hata" yazdırılır. Kullanıcıya veya sistem yöneticisine daha anlamlı bir hata mesajı sağlanmamıştır. Bu nedenle, hata yönetimi daha kapsamlı hale getirilmelidir, böylece hata durumlarında kullanıcıya uygun geri bildirim sağlanabilir.

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu kodlar, bir C# uygulamasında kullanılmak üzere bir sınıf içeriyor. Bu sınıf, bir grafik oluşturmak için gereken verileri veri tabanından almayı amaçlıyor. Bu kodlar içerisinde Doktorların Hasta Sayısına göre ZedGraph halleri Business’a verilmiştir. Burada istenilen şeyler daha sonra ZedGraph anlatıldığı zaman daha detaylı anlatılacak olup şu an sadece kod kısımları anlatılmaktadır. Burada istenilen veriler Access üzerinden çekilmekle beraber Presentation üzerinde arayüze işlenecektir. Kullanıcı pek fazla iş yapmyacaktır. Bu kodlar içerisinde SQL sorgusu fazlasıyla önemli olmakla beraber bu soru içerisinde gruplama daha doğrusu saydırma işlemi yapılmaktadır. İşte kodların yaptığı işlemlerin daha genel bir açıklaması:

Namespace'lerin Eklenmesi: Kodun başında, gerekli kütüphaneler using ifadeleriyle ekleniyor. Bu ifadeler, kodun belirli kısımlarında kullanılacak olan hazır işlevleri ve sınıfları içerir.

DoktorHastaSayısı\_ZedGraph Sınıfı: Bu sınıf, grafik oluşturma işlemleri için tasarlanmış. DoktorHastaSayısı\_ZedGraph adında bir sınıf tanımlanıyor. Bu sınıf, grafik oluşturmak için gerekli verileri sağlayacak metotları içerir.

Veritabanı Bağlantısı Oluşturulması: DoktorHastaSayısı\_ZedGraph sınıfı içinde, Baglanti adında bir sınıfın örneği (baglanti) oluşturuluyor. Bu, veritabanı bağlantısını sağlayacak bir nesne.

LoadGraphData Metodu: Bu metot, veritabanından veri alarak bir grafik oluşturur. İlk olarak, veritabanı bağlantısı açılır. Ardından, bir SQL sorgusu kullanılarak doktorların adlarını ve randevularındaki hasta sayılarını içeren veriler alınır. Bu veriler, bir PointPairList nesnesine eklenir ve son olarak bu nesne geri döndürülür. Kodun genel amacı, veri tabanından alınan doktorların adı ve randevudaki hasta sayısı gibi verileri kullanarak bir grafik oluşturmaktır. Bu şekilde, doktorların hasta sayılarına ilişkin bir görselleştirme elde edilir.

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu C# kodu bir tıbbi uygulamada veritabanına doktor, hasta ve sekreter bilgileri eklemek için kullanılan bir sınıfı içerdiği açıkça görülmektedir.

using ifadeleri: Bu ifadeler, kodun bağımlılıklarını yönetmek için kullanılır. Bu kodda, System, System.Collections.Generic, System.Linq, System.Text, System.Threading.Tasks, DataAccess ve System.Data.OleDb adlı alanlar kullanılmıştır.

Namespace Business: Bu kod parçası, Business adında bir namespace içine yerleştirilmiştir. Bu, kodun iş mantığını gruplamak için yapılmıştır.

public class Ekle: Bu sınıf, veritabanına yeni kayıtlar eklemek için gerekli metotları içerir. DoktorEkle metodu: Yeni bir doktor kaydı eklemek için kullanılır. Metot, doktorun adı, soyadı, cinsiyeti, kimlik numarası, doğum tarihi, branşı, telefon numarası ve şifresini parametre olarak alır. Bu bilgileri alır ve bunları bir SQL sorgusu aracılığıyla Doktor tablosuna ekler. HastaEkle metodu: Yeni bir hasta kaydı eklemek için kullanılır. Metot, hastanın adı, soyadı, cinsiyeti, kimlik numarası, doğum tarihi, e-posta adresi ve telefon numarasını parametre olarak alır. Bu bilgileri alır ve bunları bir SQL sorgusu aracılığıyla Hasta tablosuna ekler. SekreterEkle metodu: Yeni bir sekreter kaydı eklemek için kullanılır. Metot, sekreterin adı, soyadı, cinsiyeti, kimlik numarası, doğum tarihi ve şifresini parametre olarak alır. Bu bilgileri alır ve bunları bir SQL sorgusu aracılığıyla Sekreter tablosuna ekler.

DataAccess.Baglanti nesnesi: Ekle sınıfının bir özelliği olan baglanti, veritabanı bağlantısını yönetmek için kullanılan Baglanti sınıfından bir örnektir.

try-catch blokları: Her metot içinde veritabanı işlemleri try-catch bloğu içinde gerçekleştirilmiştir. Bu, herhangi bir hata durumunda istisnaların uygun şekilde ele alınmasını sağlar.

OleDbConnection ve OleDbCommand kullanımı: Veritabanı bağlantısı ve sorguları oluşturmak için OleDbConnection ve OleDbCommand sınıfları kullanılmıştır. Bu sınıflar, veritabanı işlemlerini gerçekleştirmek için kullanılan standart .NET Framework sınıflarıdır.

Kodun temel amacı, farklı tiplerdeki kullanıcıları (doktor, hasta, sekreter) veri tabanına eklemek için gereken işlevselliği sağlamaktır. Bu işlevsellik, veri tabanı bağlantısı üzerinden SQL sorguları kullanılarak gerçekleştirilir.

metin, ekran görüntüsü, yazılım, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu kod business layer sınıfı olan Giris sınıfını tanımlar. Bu sınıf, veri tabanı işlemlerini gerçekleştirir ve kullanıcı giriş işlemlerini kontrol eder.metin, ekran görüntüsü, paralel, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu using ifadeleri, gerekli namespace'leri ekler. Bu namespace'ler, veri tabanı işlemleri için gerekli olan sınıfları içerir.

Giris sınıfı, public erişim belirleyicisiyle tanımlanır. Bu sınıf, kullanıcı giriş işlemlerini gerçekleştiren metotları içerir.

DataAccess namespace'i altında bulunan Baglanti sınıfından bir örnek (baglanti) oluşturulur. Bu örnek, veri tabanı bağlantısını yönetir.

GirisYap metodu, kullanıcı girişini kontrol eder. Bu metot, kullanıcı adı (username) ve şifre (password) parametrelerini alır. Ayrıca, kullanıcının adını (ad), kimlik numarasını (kimlik) ve giriş türünü (giris) referans olarak alır.

Metodun içinde iki adet SQL sorgusu (query ve query2) tanımlanır. Birinci sorgu, "Doktor" tablosundan doktorların, ikinci sorgu ise "Sekreter" tablosundan sekreterlerin bilgilerini çeker.

OleDbConnection sınıfı kullanılarak veri tabanına bağlantı oluşturulur ve BaglantiAc metodunu kullanarak bağlantı açılır.

OleDbCommand nesneleri oluşturulur ve ilgili sorgular parametrelerle birlikte bu komutlara atanır.

try-catch-finally bloğu içinde, veri tabanı sorguları çalıştırılır ve sonuçlar OleDbDataReader nesneleriyle okunur.

reader ve reader2 nesneleri üzerinden sorgu sonuçları kontrol edilir. Eğer bir sonuç bulunursa, ilgili tabloya ait veriler okunarak ilgili sınıfın nesnesi oluşturulur (Doktor veya Sekreter). Ardından, bu bilgiler referans parametreler aracılığıyla dışarı aktarılır (ad, kimlik ve giriş).

Eğer kullanıcı bulunamazsa, giriş değişkeni 0 olarak ayarlanır.

catch bloğu içinde olası hata durumları ele alınır ve bir hata durumunda Exception sınıfı kullanılarak bir istisna (exception) fırlatılır.

finally bloğu içinde bağlantı kapatılır.

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturulduBu sınıf, bir veri tabanında mevcut olan doktor, hasta ve sekreter bilgilerini güncellemek için kullanılır. İlgili veri tabanı işlemleri için DataAccess adında bir namespace içerisindeki Baglanti sınıfını kullanır.

İçeride üç adet metot bulunmaktadır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturulduDoktorGuncelle: Bu metot, doktor bilgilerini güncellemek için kullanılır. Parametre olarak eski kimlik numarası, yeni kimlik numarası, ad, soyad, cinsiyet, doğum tarihi, telefon, branş ve şifre alır. Veri tabanında Doktor tablosunda ilgili doktorun kimlik numarasına göre güncelleme yapar.

HastaGuncelle: Bu metot, hasta bilgilerini güncellemek için kullanılır. Parametre olarak eski kimlik numarası, yeni kimlik numarası, ad, soyad, cinsiyet, doğum tarihi, mail ve telefon alır. Veri tabanında Hasta tablosunda ilgili hastanın kimlik numarasına göre güncelleme yapar.

SekreterGuncelle: Bu metot, sekreter bilgilerini güncellemek için kullanılır. Parametre olarak eski kimlik numarası, ad, soyad, cinsiyet, kimlik numarası, doğum tarihi ve şifre alır. Veri tabanında Sekreter tablosunda ilgili sekreterin kimlik numarasına göre güncelleme yapar.

Her metot, öncelikle bir OleDbConnection nesnesi oluşturarak veri tabanına bağlanır. Ardından güncelleme işlemini gerçekleştirmek üzere OleDbCommand nesnesi oluşturur ve SQL UPDATE sorgusunu çalıştırır. Parametreler kullanılarak SQL sorgusundaki yer tutucular değerlerle doldurulur. Eğer bir hata oluşursa, catch bloğu içinde hata yönetimi yapılır. Son olarak, bağlantı kapatılır.

metin, ekran görüntüsü, yazılım, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu sınıf, bir ZedGraph grafik nesnesi oluşturarak veri tabanından gelen verilere dayalı olarak bir grafik oluşturmayı amaçlar İlgili sınıfın adı HatalıkBölümleri\_Zed\_Graph ve Business namespace'i altında yer alır. İçerisinde DataAccess namespace'i altında bulunan Baglanti sınıfından bir örnek alır.

Sınıf içerisinde yalnızca bir metot bulunur:

LoadGraphData: Bu metot, veri tabanından doktorların kimlik numaralarına göre hasta sayılarını alır. Bu verileri PointPairList türündeki bir nesneye yükler ve geri döndürür. Bu liste, ZedGraph kütüphanesinin bir parçasıdır ve bir grafik için x-y koordinat çiftlerini saklar. Veri tabanından gelen her bir kayıt, doktor kimlik numarasını ve hasta sayısını içerir. Bu veriler, bir döngü kullanılarak okunur ve PointPairList'e eklenir. Her bir çift, grafikte bir noktayı temsil eder, x değeri indis olarak, y değeri ise hasta sayısı olarak atanır.

Doktorların hasta sayılarına göre bir grafik oluşturmaktır. Bu grafik, ZedGraph kütüphanesi kullanılarak oluşturulur ve istatistiksel analiz veya görselleştirme gereksinimlerini karşılamak üzere kullanılabilir.

Yukarıda verilen ZedGraph kod blokları daha sonra detaylı olarak arayüzde anlatılacaktır.

ZedGraphları kullanabilmek için NuGet paketlerini açarak indirmek gerekmektedir. İnmediği sürece araç kutusundan bir parça olarak kullanılamaz. Aynı zamanda kullanımı çok kolay olup kullanıcıya direkt veri vermektedir.

Grafik ve istatistik kullanarak bu verileri gösterdiğini tekrar belirtmekte fayda vardır.

metin, ekran görüntüsü, paralel, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduC# dilinde bir e-posta gönderme işlemini gerçekleştiren bir sınıfı içerir. Aşağıdaki bileşenlerin kullanıldığını görebilirsiniz:

Namespace'lerin İçeri Aktarılması: Kodda kullanılan farklı namespace'leri (using ifadeleriyle) içeri aktarır. Bu, farklı sınıf ve özelliklere erişimi sağlar.

Mail Sınıfı: Mail adında bir sınıf tanımlanır. Bu sınıf, e-posta gönderme işlemlerini gerçekleştirir.

Bağlantı ve Posta Nesneleri: DataAccess isim alanındaki Baglanti ve Posta sınıflarından nesneler oluşturulur. Bunlar, veri tabanı bağlantısı ve e-posta yapılandırma bilgileri gibi verilere erişim sağlar.

SendMail Metodu: E-posta gönderme işlemini gerçekleştiren bir metodudur. Bu metot, parametre olarak bir adres listesi (List<string>), bir kimlik numarası (string kimlik), bir başlık (string baslik) ve bir konu (string konu) alır. Veritabanı Bağlantısı Kurma: Baglanti sınıfının BaglantiAc metodunu kullanarak bir OleDb bağlantısı kurulur. Veri tabanından E-posta Adreslerini Alma: Veri tabanından belirli bir kimlik numarasına (kimlik) göre eşleşen e-posta adreslerini almak için bir sorgu çalıştırılır. Bu e-posta adresleri Hasta sınıfından nesneler oluşturularak listeye eklenir. SMTP Ayarlarının Yapılandırılması: SMTP sunucusuna erişmek için gerekli olan bilgiler Posta nesnesinden alınır ve SmtpClient nesnesine atanırdı. E-posta Gönderme: Listeden her bir e-posta adresine bir e-posta gönderilir. Gönderim işlemi SmtpClient kullanılarak gerçekleştirilirse. Hata Yönetimi ve Bağlantı Kapatma: Herhangi bir hata durumunda bir istisna yakalanır ve konsola hata mesajı yazdırılır. Bağlantı kapatma işlemi finally bloğunda yapılır, böylece bağlantı her durumda güvenle kapatılır. Bir veri tabanından e-posta adreslerini almak ve bu adreslere belirli bir başlık ve içerikle e-posta göndermek için kullanılacaktır.

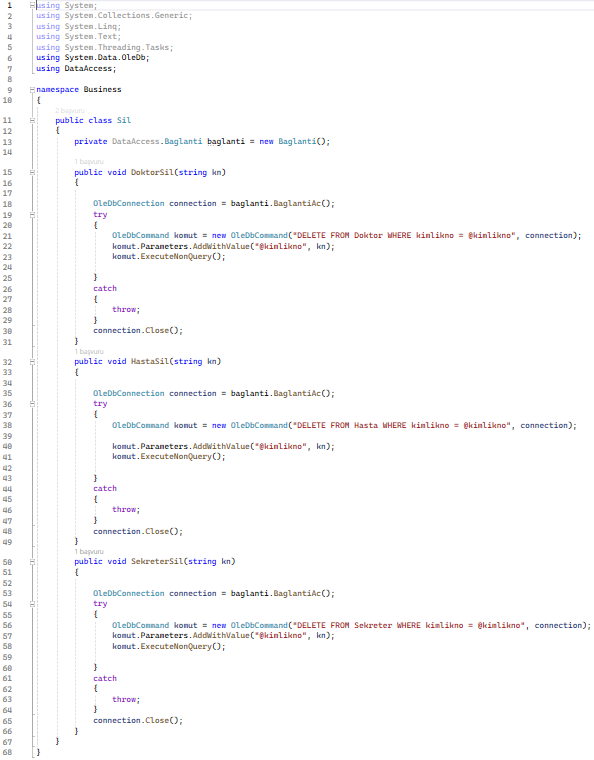
metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yukarıda gösterilen kod bloklarında sağlık uygulamasında randevu oluşturma ve randevuyla ilgili işlemleri gerçekleştirmek için kullanılabilir. Randevu sınıfı, randevu oluşturma ve randevu işlemlerini gerçekleştirmek için kullanılır. Bu sınıfın üyeleri şunlardırlar. RandevuOlustur metodu, bir hastanın belirli bir doktordan randevu almasını sağlar. Parametreler arasında hastanın kimlik numarası (hastakimlik), doktorun kimlik numarası (doktorkimlik), randevu saati (saat) ve randevu tarihi (tarih) bulunur. Bu bilgiler veri tabanına eklenir. RandevuAktifligi metodu, belirli bir doktorun belirli bir tarihte ve saatte randevusu olup olmadığını kontrol eder. Parametreler arasında doktorun kimlik numarası (dkimlik), randevu tarihi (randevutarihi) ve randevu saati (randevusaati) bulunur. Dolu parametresi, randevunun mevcut olup olmadığını belirtmek için kullanılır. RandevuKaydet metodu, bir hastanın doktorla yaptığı görüşme hakkında bilgileri kaydeder. Parametreler arasında hastanın kimlik numarası (kimlikno), teşhis (tanı), görüşler (gorus), yapılan testler (tahlil) ve doktorun kimlik numarası (doktor\_kimlik) bulunur.

Veri tabanı işlemleri için OleDb kullanır ve DataAccess.Baglanti sınıfından bir örnek oluşturarak veri tabanı bağlantısını sağlar. Veri tabanında Randevu ve HastaGecmisi tabloları kullanılır.

Bu sağlık uygulamasında kullanıcıların (doktor, hasta, sekreter gibi) veri tabanından silinmesini sağlayan işlevleri içerir. Genellikle bir uygulamanın kullanıcı yönetimi modülünde kullanılırlar. İşlevler, her bir kullanıcı türü için ayrı ayrı oluşturulmuştur ve her biri ilgili tablodan kullanıcıyı kimlik numarasına göre siler.

Sil Sınıfı: Bu sınıf, kullanıcıların silinmesiyle ilgili işlevleri barındırır.

DoktorSil Metodu: Bu metot, Doktor tablosundan bir doktorun silinmesini gerçekleştirir. Parametre olarak doktorun kimlik numarası alır

HastaSil Metodu: Bu metot, Hasta tablosundan bir hastanın silinmesini gerçekleştirir. Parametre olarak hasta kimlik numarası alır

SekreterSil Metodu: Bu metot, Sekreter tablosundan bir sekreterin silinmesini gerçekleştirir. Parametre olarak sekreter kimlik numarası alır

Her bir işlev, veri tabanı bağlantısı oluşturarak veri tabanına erişir. Veri tabanı bağlantısı OleDbConnection kullanılarak sağlanır. Daha sonra, SQL DELETE ifadesi kullanılarak ilgili tablodan belirtilen kimlik numarasına sahip kullanıcı silinir. İşlem başarılı bir şekilde tamamlandıktan sonra, bağlantı kapatılır. Ancak, bir hata meydana gelirse, bu hata dışarıya fırlatılır (throw).

Mevcutta gösterilen kodlar, kullanıcıların veri tabanından silinmesi için temel işlevleri sağlar. Ancak, gerçek bir uygulamada, bu işlemlere ek güvenlik önlemleri eklenmelidir. Örneğin, kullanıcıların silinmesi için yetkilendirme kontrolü sekreter üzerinden yapılmalıdır.



metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu Kod, doktor ve hasta bilgilerini veri tabanından çekmek ve işlemek için gerekli mantığı sağlar. Bunun için Sırala adında bir sınıf bulunmaktadır.

İlk olarak, DoktorSırala metodu doktorları sıralamak için kullanılır. Bu metot, belirli bir uzmanlık alanına sahip doktorları veritabanından seçer. Uzmanlık alanı parametresi olarak alınır ve bu parametreye göre doktorlar filtrelenir. Seçilen doktorların adı ve soyadı, bir liste nesnesine eklenir. Ayrıca, bu metot, seçilen ilk doktorun kimlik numarasını da bir referans parametresi aracılığıyla döndürür.

Daha sonra, HastaSırala metodu belirli bir hasta kimliğine göre hasta bilgilerini sıralamak için kullanılır. Bu metot, veritabanından belirtilen hasta kimliğine sahip hastanın bilgilerini alır. Hasta bilgileri, hasta adı, soyadı, cinsiyeti, doğum tarihi, e-posta adresi ve telefon numarası gibi detaylarla birlikte bir liste nesnesine eklenir.

Son olarak, DoktorBilgileriSırala metodu belirli bir doktor kimliğine göre doktor bilgilerini sıralar. Bu metot, veritabanından belirtilen doktor kimliğine sahip doktorun adı, soyadı, uzmanlık alanı, cinsiyeti ve doğum tarihi gibi bilgileri alır. Bu bilgiler de bir liste nesnesine eklenir.

Bu kod, veritabanı işlemleri için OleDb kütüphanesini kullanır. Bağlantı nesnesi, veritabanına bağlanmak için kullanılır. Doktor ve Hasta sınıfları, ilgili verileri taşımak için kullanılır. Bu şekilde, doktor ve hasta bilgileri veritabanından çekilir ve işlenir, ardından kullanıcı arayüzüne veya başka bir işleme katmanına iletilir.

PRESENTATİON LAYER KODLARI VE AÇIKLAMALARI

Sol tarafta Giris1.cs form dosyasına ait kodlar verilmiştir. Burası Presentation Layer’ın il giriş ekranı olacaktır. Daha sonrasın çok daha detaylı olarak anlatılacaktır.

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldubutton1\_Click metodu, birinci düğmeye tıklandığında gerçekleşen olayları içerir. Eğer "Giris1" adında bir form açık değilse, yeni bir Giris1 formu oluşturulur ve gösterilir. Bu form, kullanıcının birinci giriş işlemlerini yapabileceği bir pencere olabilir.

Bu kod parçası, kullanıcıyı uygulamanın ana giriş noktasından farklı noktalarına yönlendirmek için kullanılır. Kullanıcı, belirli bir işlevi gerçekleştirmek için ilgili düğmelere tıklayarak farklı giriş formlarına erişebilir. Bu sayede uygulama, farklı kullanıcı etkileşimlerini yönetebilir ve işlevselliği bölümlere ayırabilir.

Burada yapılanları anlatmak gerekirse önce Business katmanı ile gerekli bağlantılar kurulmaktadır. Access’te bulunan veriye göre giriş yapmaktadır. Aralarında çok fazla fark bulunmamaktadır bu iki dosyanın. İkisinden birisi doktor arayüzüne diğeri ise sekreter arayüzüne gitmektedir.

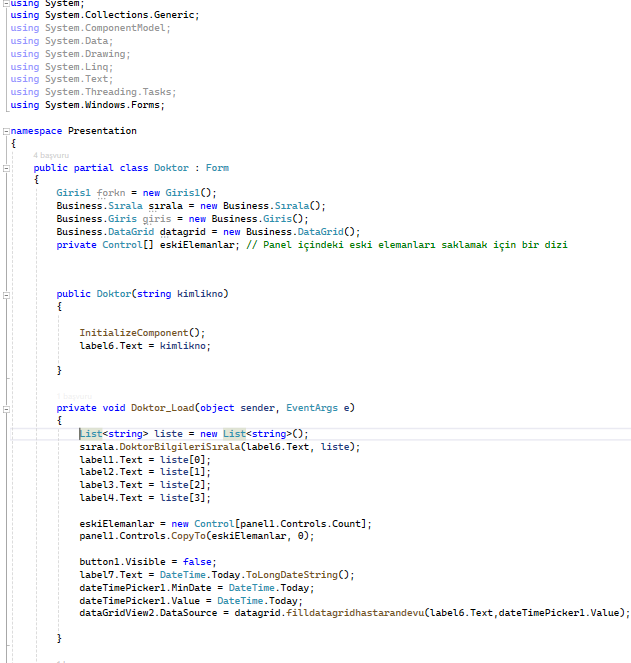
Ancak giderken progressbar kullanılarak belli değerler aldığı görülmüştür. Bu kısımlar arayüzün daha düzenli ve güzel gözükmesi için yapılmaktadır.

bir Windows Forms uygulamasının giriş formunu içerir. Kullanıcıdan ad, şifre ve kimlik numarası gibi bilgileri girmesini bekler. Kullanıcı bu bilgileri girdikten sonra, giriş bilgilerinin geçerliliği kontrol edilir ve kullanıcıya uygun mesajlar gösterilir. Bu mesajlar, kullanıcının giriş bilgilerinin doğruluğuna veya hatalarına bağlı olarak değişir.

Form yüklendiğinde herhangi bir işlem yapılmaz. Ancak, kullanıcı giriş yap butonuna tıkladığında button2\_Click metodunu tetikler. Bu metot, kullanıcının giriş bilgilerini kontrol eder. Eğer kullanıcı adı veya şifre boşsa veya boşluk içeriyorsa, kullanıcıya uyarı verilir ve işlem sonlandırılır. Dolu ise, Business katmanındaki Giris nesnesi oluşturulur ve GirisYap metoduna kullanıcı adı, şifre ve referans parametreleri olarak aktarılır.

GirisYap metodunun dönüş değerine göre farklı işlemler yapılır.

* Eğer giriş başarılıysa, bir ilerleme çubuğu tamamlanır ve "Giriş Yapılıyor" mesajı gösterilir. Ardından, ilgili doktor formu açılır.
* Eğer giriş bilgileri yanlışsa, bir ilerleme çubuğu tamamlanır ve "Yanlış Giriş Yaptınız, Tekrar Deneyin" mesajı gösterilir.
* Eğer girilen kimlik numarası hatalıysa, bir ilerleme çubuğu tamamlanır ve "Hatalı Kimlik Numarası Girdiniz" mesajı gösterilir. Kullanıcı bilgileri temizlenir.

Sonuç olarak, bu kod parçası, kullanıcının giriş bilgilerini kontrol eder ve uygun mesajlar gösterirken, doğru giriş yapıldığında ilgili işlemi gerçekleştirir.

aşağıdaki kod parçası, bir Windows Forms uygulamasının doktor panelini temsil eder. Bu panel, doktorun kendi bilgilerini görüntülemesine ve randevularıyla ilgili işlemleri gerçekleştirmesine olanak tanır.

İlk olarak, Doktor sınıfı, bir doktorun bilgilerini ve randevularını yönetmek için tasarlanmıştır. Form yüklendiğinde (Doktor\_Load metodu), doktorun bilgileri Business.Sırala sınıfı aracılığıyla DoktorBilgileriSırala metodunu kullanarak alınır ve ekranda gösterilir. Bu, doktorun adı, soyadı, doğum tarihi, uzmanlık alanı gibi bilgileri içerir. Ayrıca, bugünün tarihini gösteren bir etiket (label7) ve tarih seçici (dateTimePicker1) de bulunur. Randevuları görüntülemek için bir DataGridView (dataGridView2) oluşturulur ve doktorun kimlik numarasına göre randevular doldurulur.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Hasta randevularının görüntülenmesi için dataGridView2\_CellDoubleClick metodu tanımlanır. Kullanıcı bir hücreye çift tıkladığında, seçilen hastanın kimlik numarası alınır ve bu kimlik numarası ile bir görüşme formu (Gorusme1) oluşturulur. Oluşturulan form, mevcut ekran elemanları silinerek (panel1.Controls.Clear()) panel1 içine yerleştirilir. Bu sayede, doktor seçilen hastanın detaylarını inceleyebilir ve gerekli işlemleri gerçekleştirebilir.

"Geri" düğmesine (button1) tıklandığında, button1\_Click metoduna geçilir. Bu metot, önceki ekranın elemanlarını (eskiElemanlar) panel1 içine yerleştirir ve mevcut ekran elemanlarını silerek (panel1.Controls.Clear()) temizler. Bu işlem sayesinde, doktor tekrar randevuları görüntüleyebilir ve diğer işlemleri gerçekleştirebilir.

Son olarak, dateTimePicker1\_ValueChanged metodu, tarih seçici değeri değiştiğinde tetiklenir. Bu metot, doktorun belirli bir tarihte randevularını görüntülemesine olanak tanır. Seçilen tarihe göre randevuları yeniden doldurarak (datagrid.filldatagridhastarandevu), doktora belirli bir tarihte randevularını yönetme ve planlama imkânı sağlar. Bu kodlar doktorun kendi bilgilerini görüntülemesini ve randevularını yönetmesini sağlar, böylece verimli bir hasta takibi yapabilir ve gerekli işlemleri gerçekleştirebilir.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduBölüm\_DoktorSayısı sınıfı, doktor sayısını ve bunlara bağlı hasta sayısını bar grafiği şeklinde gösteren bir form oluşturur. Business.DoktorHastaSayısı\_ZedGraph sınıfından bir nesne oluşturulur ve bu nesne, grafik verilerini sağlar. Form yüklendiğinde (Bölüm\_DoktorSayısı\_Load metodu), DisplayGraph metodunu çağırarak grafiğin görüntülenmesi sağlanır.

DisplayGraph metodunda, ZedGraph kütüphanesi kullanılarak bir grafik oluşturulur. Grafik penceresinin pane'i (GraphPane) alınır ve zedgraphj nesnesinden gelen verilerle bir PointPairList oluşturulur. Bu veriler, doktorları ve bunlara bağlı hasta sayılarını içerir.

Daha sonra, oluşturulan PointPairList kullanılarak bir çubuk grafiği (BarItem) eklenir. Çubukların dolgusu kahverengi (System.Drawing.Color.Brown) olarak belirlenir ve her çubuğun üzerine hasta sayısı etiketleri eklenir.

X ekseninin etiketleri, doktorların isimlerinden oluşur. Bu isimler, PointPairList içindeki Tag özelliğinden alınır. X ekseni başlığı "Doktorlar" olarak ayarlanırken, Y ekseni başlığı "Hasta Sayısı" olarak belirlenir.

Son olarak, eksenlerin değişiklikleri uygulanarak (AxisChange), grafik güncellenir ve gösterilir.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bölüm\_HastaSayısı sınıfı, branşlara göre hasta sayısını bar grafiği şeklinde gösteren bir form oluşturur. Business.HatalıkBölümleri\_Zed\_Graph sınıfından bir nesne oluşturulur ve bu nesne, grafik verilerini sağlar. Form yüklendiğinde (Bölüm\_HastaSayısı\_Load metodu), DisplayGraph metodunu çağırarak grafiğin görüntülenmesi sağlanır.

DisplayGraph metodunda, ZedGraph kütüphanesi kullanılarak bir grafik oluşturulur. Grafik penceresinin pane'i (GraphPane) alınır ve zedgraphbr nesnesinden gelen verilerle bir PointPairList oluşturulur. Bu veriler, branşları ve bunlara bağlı hasta sayılarını içerir.

Daha sonra, oluşturulan PointPairList kullanılarak bir çubuk grafiği (BarItem) eklenir. Çubukların dolgusu mavi (System.Drawing.Color.Blue) olarak belirlenir ve her çubuğun üzerine hasta sayısı etiketleri eklenir.

X ekseninin etiketleri, branşların isimlerinden oluşur. Bu isimler, PointPairList içindeki Tag özelliğinden alınır. X ekseni başlığı "Branşlar" olarak ayarlanırken, Y ekseni başlığı "Hasta Sayısı" olarak belirlenir. Son olarak, eksenlerin değişiklikleri uygulanarak (AxisChange), grafik güncellenir ve gösterilir.

Yukarıda verilen iki sayfa yazılanlar ortak olarak yazılmıştır. Çünkü ZedGrapphlar detaylı olarak anlatılmıştır ve birbirlerine çok benzeyen tek parçalı yapıya sahiptirler. Bu sebepten dolayı benzer cümle kalıpları içerisinde anlatılma gereği duyulmuştur

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturulduWindows Forms uygulamasında eski hastane kayıtlarını göstermek için bir form oluşturur. EskiKayıtlar adında bir sınıf tanımlanır ve bu sınıf, belirli bir hastanın eski kayıtlarını göstermek için tasarlanır. Form yüklendiğinde (EskiKayıtlar\_Load metodu), Business.DataGrid sınıfından bir nesne oluşturulur ve bu nesne aracılığıyla belirli bir hastanın eski kayıtları dataGridView1 adlı DataGridView bileşenine doldurulur. Bunun için filldatagridhastagecmis metodunun kullanılması gerekmektedir. Bu metot, hastanın kimlik numarasına göre eski kayıtları getirir ve DataGridView'e doldurur. Formun oluşturulması sırasında, bir kimlik numarası parametresi alınır ve bu kimlik numarası sınıfın içindeki kimlikno değişkenine atanır. Bu sayede, form yüklendiğinde hangi hastanın eski kayıtlarının gösterileceği belirlenir.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yanda ve aşağıda verilen 3 sayfalık kod serisi olmaklar beraber anlatılacaktır. Sırası ile şöyledir:

Formun Oluşturulması ve Başlatılması: Randevu1 sınıfı, Form sınıfından türetilmiş bir sınıftır ve formun oluşturulması ve başlatılması için gerekli kodları içerir. Bu form, birçok Windows Forms kontrolünü içerir, bunlar arasında comboBox4, comboBox2, comboBox3, checkedListBox1, dateTimePicker1, label ve button gibi kontrol öğeleri bulunur.

Doktor ve Branş Seçimi: Kullanıcı, doktorun branşını seçmek için comboBox4 kontrolünü kullanır. comboBox4\_SelectedIndexChanged olayı, branş seçimi değiştiğinde çalışır ve seçilen branşa göre uygun doktorları getirir ve comboBox2'ye ekler.

Doktor Seçimi: Kullanıcı, doktoru comboBox2 üzerinden seçer. comboBox2\_SelectedIndexChanged olayı, doktor seçimi değiştikçe çalışır.

Tarih Seçimi: Kullanıcı, randevu tarihini dateTimePicker1 üzerinden seçer. dateTimePicker1\_ValueChanged olayı, tarih değiştiğinde çalışır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Randevu Saati Seçimi: Kullanıcı, randevu saati seçmek için comboBox3'ü kullanır. comboBox3\_SelectedIndexChanged olayı, randevu saati seçimi değiştiğinde çalışır.

Randevu Saatleri Listelenmesi: checklistsaat metodu, seçilen saat aralığına göre uygun randevu saatlerini checkedListBox1 kontrolüne ekler. checkedlistlistele metodu, seçilen tarih ve doktora göre dolu ve boş randevu saatlerini kontrol eder ve dolu olanları listeden kaldırır.

Randevu Oluşturma: Kullanıcı, randevu oluşturmak için button10 (muhtemelen "Ekle" butonu) butonuna basar. button10\_Click olayı, randevu oluşturulduğunda çalışır ve kullanıcının girdiği bilgilere dayanarak randevuyu oluşturur. Hasta Bilgilerinin Görüntülenmesi: Kullanıcı, hasta bilgilerini görmek için button1 (muhtemelen "Görüntüle" butonu) butonuna basar. button1\_Click olayı, hasta bilgilerini görüntülediğinde çalışır.

Kontrol Fonksiyonları: KontrolEt metodu, gerekli alanların doldurulup doldurulmadığını kontrol eder ve gerekli durumlarda işlem yapılmasını sağlar.

metin, ekran görüntüsü, doküman, belge, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduYukarıda anlatılan kodlar kullanıcıların doktor, branş, tarih ve saat seçmelerine izin vererek randevu oluşturmalarını sağlar. Ayrıca, seçilen tarih ve doktora göre uygun randevu saatlerini listeler ve dolu olanları filtreler.

Sekreter, doktorları, hastaları ve diğer sekreterleri sisteme kaydetme yeteneğine sahiptir.

Form, üç farklı üye türü için giriş alanları içerir: doktorlar, hastalar ve diğer sekreterler. Kullanıcı, hangi türde bir üye eklemek istediğini seçmek için bir comboBox kullanır. Seçilen üye türüne göre ilgili giriş alanları görünür hale gelir ve kullanıcı gerekli bilgileri girer.

"Üye Ekle" butonuna tıklandığında, kullanıcının girdiği bilgilere dayanarak yeni bir üye oluşturulur. Örneğin, doktor ekleme işleminde, kullanıcının girdiği ad, soyad, cinsiyet, kimlik numarası, doğum tarihi, branş, telefon numarası ve şifre bilgileri kullanılarak bir doktor hesabı oluşturulur. Aynı şekilde, hasta veya sekreter ekleme işlemleri de benzer şekilde gerçekleşir.

metin, ekran görüntüsü, doküman, belge, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduMenü çubuğundaki seçenekler aracılığıyla, sekreter farklı işlevlere erişebilir. Örneğin, "Randevu Oluşturma" seçeneği, randevu oluşturmak için ayrı bir formun açılmasını sağlar. Benzer şekilde, "Silme/Güncelleme", "Doktor/Hasta Sayısı", "Görüşme" gibi seçenekler de farklı işlevleri gerçekleştirmek üzere ilgili formları başlatır.

Kullanıcı, comboBox üzerinden seçtiği üye türüne göre giriş alanlarını değiştirebilir. Örneğin, doktor ekleme modunda, doktor bilgilerini girmek için ilgili giriş alanları aktif hale gelirken, hasta veya sekreter ekleme modunda ilgili giriş alanları aktif hale gelir ve diğerleri pasif hale gelir.

Sağlık kuruluşunda çalışan bir sekreterin işlerini daha organize bir şekilde yönetmesine yardımcı olur. Doktor, hasta ve sekreter bilgilerinin sistemde tutulması ve yönetilmesi, randevuların oluşturulması ve diğer işlevlerin yerine getirilmesi gibi temel görevler bu uygulama üzerinden kolayca gerçekleştirilebilir.

metin, ekran görüntüsü, doküman, belge, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, doküman, belge, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduKullanıcıların doktorlar, hastalar ve sekreterler gibi sağlık sistemi kullanıcılarını yönetmelerini sağlayan bir arayüz sunuyor.

Formun oluşturulması bölümünde, SilmeGuncelleme adında bir form sınıfı tanımlanır. Bu form, doktorlar, hastalar ve sekreterler gibi sağlık sistemi kullanıcılarının listelendiği bir DataGridView kontrolü ve bu kullanıcıları silme ve güncelleme işlemlerini gerçekleştirmek için butonlar içerir.

Veri tabanından verilerin yüklenmesi bölümünde, form yüklendiğinde veya kullanıcı belirli bir menü öğesine tıkladığında, ilgili veriler veri tabanından alınarak DataGridView kontrolüne doldurulur. Bu işlem, SilmeGuncelleme\_Load fonksiyonunda gerçekleştirilir ve menü öğelerine tıklama olaylarında çağrılır.

Kullanıcı etkileşimi bölümünde, kullanıcılar doktorlar, hastalar ve sekreterler gibi sağlık sistemi kullanıcıları arasında gezinirken veya bu kullanıcıları silme veya güncelleme işlemleri yaparken etkileşime geçer. Menü öğelerine tıklama olayları, belirli bir kullanıcı türü için verilerin yüklenmesini tetikler. Silme işlemi için kullanıcı seçtiği öğeyi seçer ve silme butonuna tıklar. Güncelleme işlemi ise kullanıcı belirli bir öğeyi düzenleme modundayken gerçekleştirilir.

DataGridView hücreleri düzenlendiğinde, kullanıcı verileri güncellemek istediğinde dataGridView1\_CellEndEdit ve dataGridView1\_CellBeginEdit olayları tetiklenir. Bu olaylar, düzenleme işlemlerini işlemek ve güncelleme işlemlerini gerçekleştirmek için kullanılır. Sonuç olarak, bu kod parçaları bir Windows Forms uygulamasının temel işlevselliğini tanımlar ve kullanıcıların sağlık sistemi kullanıcılarını görüntülemelerini, silmelerini ve güncellemelerini sağlar. Bu, bir sağlık yönetim sistemi veya benzeri bir uygulama geliştirirken kullanılabilecek temel bir yapıdır.

Aşağıda Gorusme1 isimli forma ait içerik kodları verilmiştir.

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

public partial class Gorusme1 : Form: Gorusme1 adında bir sınıf tanımlanır. Bu sınıf Form sınıfından türetilir ve bir Windows Forms formunu temsil eder. Mail, randevu ve sırala adında Business isim alanında tanımlanmış nesneler tanımlanır. Bunlar, belirli iş mantıklarını gerçekleştirmek için kullanılan sınıfları örneklendirir. \_brans, \_ad, \_cinsiyet, ve \_kimlik adında dört tane private değişken tanımlanır. Bu değişkenler, sınıfın içinde kullanılacak ve dışarıdan gelen bilgileri saklamak için kullanılır. Gorusme1 sınıfının bir kurucu metodu bulunur. Bu kurucu metot, sınıfın örneklerini oluştururken çağrılır ve gelen parametreleri sınıfın private değişkenlerine atar.

button4\_Click, button3\_Click, button1\_Click, Gorusme1\_Load, ve button2\_Click adında beş tane olay işleyici metodu bulunur. Bu metotlar, form üzerindeki butonların tıklanma veya form yüklenme gibi olayları işler.

Her bir butonun tıklanma durumunda yapması gereken işler farklıdır:

button4\_Click: Eski kayıtları göstermek için başka bir form açar.

button3\_Click: Bir mail göndermek için gerekli bilgileri kullanarak Mail sınıfını kullanır.

button1\_Click: Randevu kaydını Randevu sınıfını kullanarak kaydeder.

Gorusme1\_Load: Form yüklendiğinde hasta bilgilerini doldurmak için Sırala sınıfını kullanır.

button2\_Click: Formdaki bilgileri alarak bir PDF dosyası oluşturur ve kaydeder.

Bu kodlar, bir hasta-doktor görüşmesi için bir arayüz sağlar ve bu görüşmenin kaydedilmesi, mail ile gönderilmesi, ve PDF olarak kaydedilmesi gibi işlemleri gerçekleştirir. button2\_Click metodunda, PDF dosyası oluşturma işlemi gerçekleştirilir. Bu metod, bir SaveFileDialog (Kaydetme Dosyası İletişim Kutusu) açar ve kullanıcıya bir PDF dosyası kaydetme konumu seçme şansı verir. Eğer kullanıcı bir konum seçerse, using bloğu içinde bir Document nesnesi oluşturulur ve PDF dosyası oluşturulmak üzere açılır. Document nesnesine başlık, doktor bilgileri, hasta bilgileri, tanı, görüş ve tahlil sonuçları gibi metinler eklenir. Dosya oluşturulduktan sonra, bir mesaj kutusuyla kullanıcıya başarılı bir şekilde PDF oluşturulduğu ve kaydedildiği bilgisi verilir. Eğer bir hata oluşursa, kullanıcıya bir hata mesajı gösterilir.

Aşağıda Gorusme1 isimli formun içindeki PDF aracının çalışma mantığı gösterilecektir. button2\_Click altında yazılan kodlar tek tek anlatılacaktır.

SaveFileDialog'in Açılması: Kullanıcı bir PDF dosyası kaydetmek istediğinde, SaveFileDialog ile bir dosya iletişim kutusu açılır. Kullanıcı bu iletişim kutusu aracılığıyla bir dosya adı ve konumu seçer.

Document Nesnesinin Oluşturulması: Eğer kullanıcı bir kayıt yeri seçerse, bir Document nesnesi oluşturulur. Bu Document nesnesi, iTextSharp kütüphanesi kullanılarak PDF dosyasının içeriğini temsil eder.

PDF Dosyasının Açılması: Document nesnesi, PdfWriter ile birlikte bir dosya akışına bağlanarak PDF dosyasının oluşturulması ve yazılabilir hale getirilmesi sağlanır. Bu adımdan sonra PDF dosyası oluşturulmaya hazır hale gelir.

Başlık Ekleme: PDF dosyasına, "Görüşme Kaydı" gibi bir başlık eklenir. Bu başlık, doktor ve hasta bilgileri ile birlikte görüşme detaylarının altında yer alır.

Doktor Bilgilerinin Eklenmesi: Doktorun adı, cinsiyeti ve branşı gibi bilgiler PDF dosyasına eklenir. Bu bilgiler, görüşme detaylarının başında yer alır ve doktor hakkında temel bilgileri içerir.

Hasta Bilgilerinin Eklenmesi: Hasta adı, cinsiyeti, kimlik numarası, doğum tarihi ve telefon numarası gibi bilgiler de PDF dosyasına eklenir. Bu bilgiler, hasta hakkında önemli kimlik bilgilerini içerir ve doktorun görüşme sırasında referans alabileceği bilgilerdir.

Tanı, Görüş ve Tahlil Sonuçlarının Eklenmesi: Görüşme sırasında elde edilen tanı, görüş ve tahlil sonuçları gibi bilgiler, PDF dosyasının en altına eklenir. Bu bilgiler, doktorun hastanın durumu hakkında tuttuğu notları içerir ve sonuç olarak doktorun teşhisini ve önerilerini içerir.

Dosyanın Kapatılması ve Kaydedilmesi: PDF dosyasının içeriği tamamlandıktan sonra dosya kapatılır ve kullanıcının seçtiği konuma kaydedilir. Eğer dosya kaydedilirken bir hata oluşursa, kullanıcıya bir hata mesajı gösterilir ve işlem durdurulur.

Kullanıcıya Bilgilendirme Mesajı: PDF dosyası başarıyla oluşturulduktan ve kaydedildikten sonra kullanıcıya bilgilendirme mesajı gösterilir. Bu mesaj, dosyanın başarıyla oluşturulduğunu ve kullanıcının belirlediği konuma kaydedildiğini belirtir.

Hata Durumunda Kullanıcıya Bilgilendirme: Eğer PDF oluşturma işlemi sırasında bir hata oluşursa, kullanıcıya hata mesajı gösterilir. Bu mesaj, oluşan hatanın detaylarını içerir ve kullanıcıya hatanın nedenini anlaması için yardımcı olur.

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

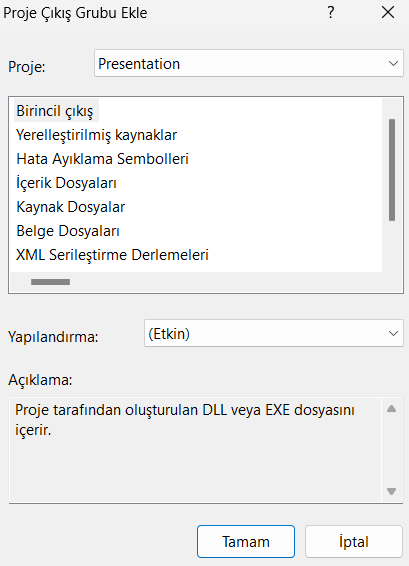
Açıklama otomatik olarak oluşturuldu

**ARAYÜZ VE PROGRAMIN ÇALIŞMA MANTIĞI**

1. **Setup Haline Getirilmesi**

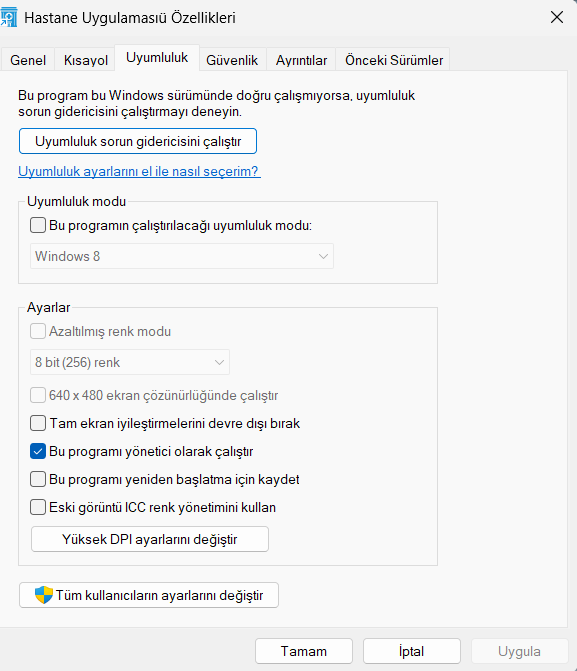
metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Program setup haline getirilirken sağ taraftaki her yerin boş olduğu gözlemlenmektedir. Çalışan bir setup olduğu için tekrardan oluşturulma amacı güdülmemiştir. Application Folder’a farenin sağ tuşu ile tıklanılır ve Add>>Proje Çıkışı seçilerek yandaki ekran hizmete gelir. Bu ekrana tamam dendikten sonra yukarıda gösterilen ekran görüntüsünde olduğu gibi her şey kullanıcının önüne serilir. Daha sonra veri tabanı dosyasını seçmek için Add>>Dosya seçilir ve gerekli dizinden veri tabanı da seçilerek devam edilir.

Ek olarak yukarıda görüldüğü üzere bir icon bulunmaktadır. Kısayola bir icon eklendikten sonra kısayol dosyası User’s Desktop yazan yere atılır. Burada amaç görsel olarak kısayolun güzel gözükmesi ve programı açmak için ana dizine gitmemektir.

1. **Programın Çalıştırılması**

Öncelikli olarak programın kısayoluna sağ tıklıyor ve Özellikler seçeneğini seçiyoruz.(Windows 11 için anlatım yapılmıştır.) 

Yukarıdaki ekran Genel başlığı altında gelecektir. Burada aşağıda görülen ‘Bu programı yönetici olarak çalıştır’ tiki seçilir. Bu tik seçildikten sonra sırasıyla Uygula ve Tamam butonuna basılarak program çalıştırma kısmına geçilir başlanır. Programın yönetici olarak çalıştırılmasının tek amacı veri tabanına erişim sağlanmak istenmesidir. Eğer yönetici olarak çalıştırılmazsa veri tabanı salt okunur olarak açılır.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduKısayola bastıktan sonra ekrana gelen yönetici iznine evet denir ve devam edilir.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

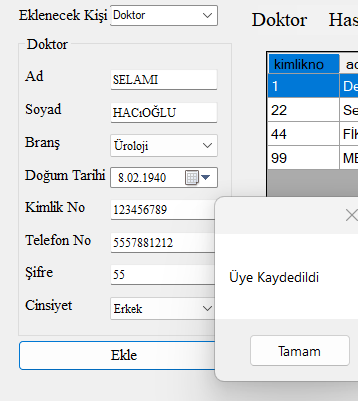
Açıklama otomatik olarak oluşturuldu Bundan sonra solda görülen giriş ekranı gelecektir. Bu ekran üzerinden hem sekreter hem de doktor girişi yapılabilmektedir. Bu ekranı kullanarak istenilen giriş yapılabilmektedir. Biz başlangıç olarak anlatacağımız için önce sekreter şifresini girerek sekreter girişi yapıyoruz.

Eğer hatalı bir giriş yapılır ise sağda görülen uyarı ekranını verecek ve yeşil yükleme barı ilerlemeyecektir.

metin, ekran görüntüsü, yazılım, ekran, görüntüleme içeren bir resim

Açıklama otomatik olarak oluşturuldu

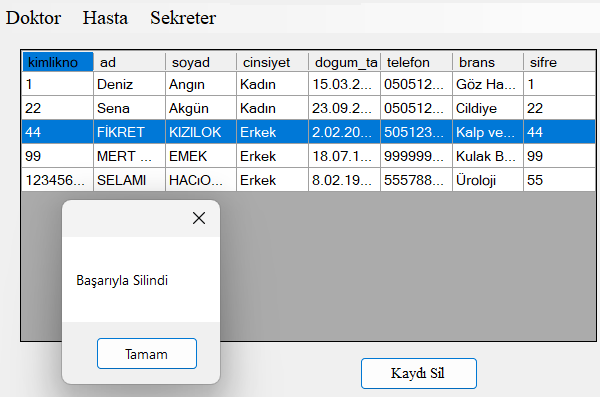
metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduKarşımıza çıkan bu ekran üzerinde birden çok işlem kolaylıkla yapılabilmektedir. Sekreter kullanıcının kolay kullanımı amaçlanmıştır. Şimdi yeni bir doktor ve yeni bir hasta ekleyerek devam edeceğiz

metin, ekran görüntüsü, yazılım, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu Sağ tarafta üroloji alanında yeni bir doktor eklenmiştir. Aşağıya ise Ayşegül isimli bir hasta eklenmiştir. Şimdi yapılmak istenen şey mevcut doktora mevcut hastanın randevusunu almaktır. Bu kısımlar yapıldıktan grafikler aynı akranda bulunmasından dolayı gösterilecektir. Ek olarak doğru mail girilirse hastaya mail gönderilebilmektedir. Eğer mail doğru değil ise mail’in boşa gittiğini söylenebilir. Ek olarak sekreterde eklenebilmektedir ancak benzer ekrana sahip olmalarından dolayı gösterilmemiştir.

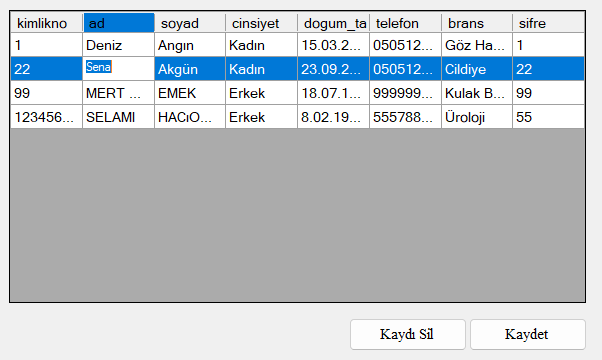
Bu kısımda silme güncellemenin nasıl yapıldığı anlatıalcaktır. Daha önceki sayfalarda gösterilen doktorlardan birisi silinmek isteniyor.



Yukarıda doktorlardan birisi silinmeye çalışılmıştır.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduDoktorun silindiği açıkça gözükmektedir. Aynı zamanda güncelleme için sol altta Kaydet butonu bir anda görünmektedir. Bunun dışında orada bulunan butona basılamaz. Yukarıdaki görüntüde görünmeyen buton birazdan gösterilecektir.



metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduSol tarafta doktorun adı değiştirilerek güncellenmek istenmiştir. İsim yazıldıktan sonra yukarıda bahsettiğimiz kaydet butonuna tıklanılır ve isim güncellenir. Daha sonra bu buton tekrardan kaybolur. Aşağıdaki görselde ismin güncellendiği gözlenebilmektedir.

metin, ekran görüntüsü, ekran, görüntüleme, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yukarıdan Doktor/Hasta sayısı isimli başlığa basılarak ekrana Doktor isimlerine göre hasta sayısı gösterilmektedir. Her doktordan doktora değişebilmekte olup yeni eklenen doktorun sıfır hastası olduğu da gösterilmektedir.

metin, ekran görüntüsü, ekran, görüntüleme, sayı, numara içeren bir resim

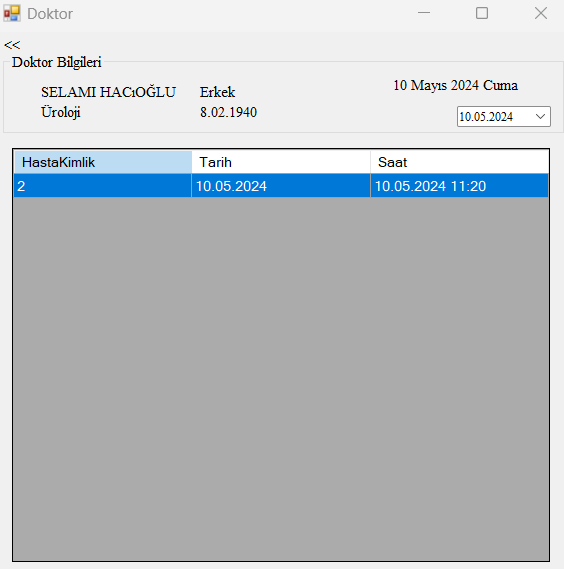
Açıklama otomatik olarak oluşturuldu

Hemen yukarıda ise Bölüm/Hasta Sayısı altında branşlara göre alınan hasta randevuların grafiği göstermektedir. Göz hastalıklarını çok fazla hastanın seçtiği gözlemlenmiştir.

metin, ekran görüntüsü, ekran, görüntüleme, yazı tipi içeren bir resim

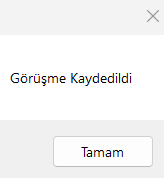
Açıklama otomatik olarak oluşturuldu

Sol üstte bulunan Randevu Oluşturma başlığına tıklanarak yeni bir sayfa açılır. Bu sayfada yeni eklenen doktorumuz ile yeni eklenen hasta için bir randevu oluşturulur. Şunu belirtmek gerekir ki eğer yanlış hasta kimlik numarası girilirse hata vermektedir. Aynı zamanda hasta ismi yazmadığı takdirde de randevu verilmektedir. Yanlış girilen kimlik numaraları ile randevu alınmasının önüne geçilmeye çalışılmıştır. Benzer bir şekilde bir saat seçildiği takdirde o saate herhangi bir hasta randevu alamamaktadır.

Doktorun şifresini girerek kendi eklediğimiz doktorun kendi kullandığı ekranına giriş yapıyoruz. Burada doktor randevularına göre tarih seçebilmektedir. İster 10 Mayıs isterse de 10 Haziran tarihlerinde randevu olup olmadığına bakabilmektedir. Randevu hakkında tüm bilgiler doktor ekranına görülmektedir. Şimdi hastanın kimlik numarasının bulunduğu satıra çift tıklama yaparak diğer ekranın aşağıya inmesi sağlıyoruz.

metin, ekran görüntüsü, ekran, görüntüleme, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

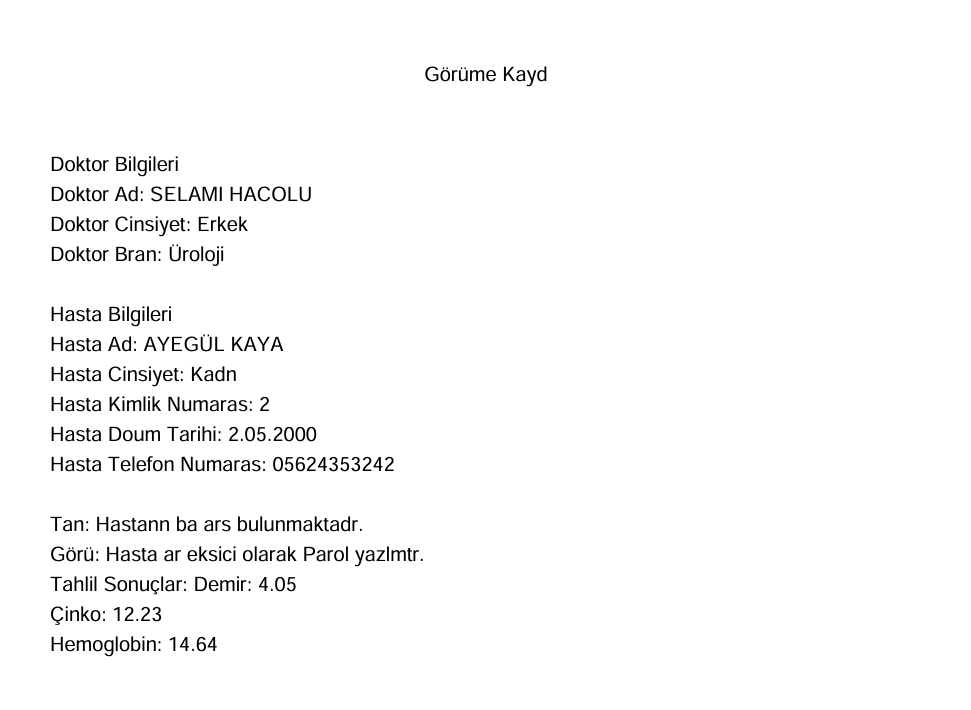
Yukarıda görüldüğü Hasta ve Doktor bilgileriyle beraber randevu görüşmesinin ekranı gelmektedir. Buradan geri dönebilmek için Tüm Randevular butonuna basılabilir veya işleme devam edilir. Görüşmeyi Kaydet butonuna basılırsa görüşme kaydedilir. Aynı zamanda metinlerin içi normalde boş olmakla beraber doktor tarafından doldurulmuştur.

Şimdi Mail Gönder tuşuna basılarak mail gönderilir. Eğer mail doğru girilirse aşağıda verilen ekran görüntüsünde olduğu gibi mail gelmektedir. Yanlış veya var olmayan bir mail girilirse asla gelmeyecektir.

metin, ekran görüntüsü, yazı tipi, iş kartı içeren bir resim

Açıklama otomatik olarak oluşturuldu

PDF butonuna basılır ve kaydetmesi bekleniri, konumu kullanıcı kendisi seçer ve hasta bilgilerini istediği yere raporlayarak kaydeder.



Yukarıda verilen ekran görüntüsünde gösterildiği gibi tüm bilgileri PDF içerisine kaydedebilmektedir. Ancak PDF dosyası Türkçe karakterlerden kaynaklı olarak böyle gözükmektedir.

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

En son ise Eski Kayıtlar butonuna basılarak hasta için verilen değerlendirme sistem üzerinden doktor tarafından incelenebilmektedir. Uygulama hakkında verilen son kısımda anlatılmış ve uygulama kapatılarak bitirilmiştir.