

Hasta Takip ve Yönetim Sistemi

1. Çağla Gök
Bilgisayar Mühendisliği
Kocaeli Üniversitesi
caglagok369@gmail.com

2. Yaren Aybala Koşar
Bilgisayar Mühendisliği
Kocaeli Üniversitesi
yarenaybala2004@hotmail.com

Özetçe—Bu proje, bir hastane yönetim sistemi olarak tasarlanmış bir web uygulamasıdır. Bu web uygulamasında, hastaların kayıt oluşturabileceği, doktorlara randevu alabileceği, tıbbi raporları saklayabileceği ve genel olarak sağlıkla ilgili işlemleri yönetebileceği bir platform sunulmaktadır.

I. GİRİŞ

Projemizi Visual Studio (ADO.NET), MsSQL, C#, HTML, JavaScript ve SQL gibi araçlar ve programlama dilleri kullanarak geliştirdik. Veri setini faker kütüphanesi ile üretip json formatında sakladık. Bu json dosyasını veri tabanına aktararak web uygulamamızda veri tabanından çekerek kullandık. Bu proje bir veri tabanı projesi olmakla birlikte web uygulamasıdır. Projede verilen istelere göre hasta, doktor ve yönetici kendi sayfasına giriş yaparak kendi izinleri doğrultusunda gerekli işlemleri gerçekleştirebilmektedir.



II. KULLANIM

A. Veri Seti

Faker kütüphanesiyle ürettiğimiz veri setinde 500 adet hasta, 100 adet doktor, 1 adet yönetici, 4976 adet tıbbi rapor ve 10000 adet randevu bilgisi vardır.

B. Hasta Girişi

Bir hasta kendi ID ve şifresiyle giriş yapabilmektedir. Giriş yaptığı sayfada (Hasta Dashboard) hastanın randevu bilgileri ve tıbbi rapor bilgileri görüntülenmektedir.

Randevu Tarihi	Randevu Saati	Doktor Adı
16/01/2024	11:00:00	Dr. Mehmet Yılmaz
16/01/2024	14:00:00	Dr. Ayşe Kaya
16/01/2024	17:00:00	Dr. Ali Demir
16/01/2024	19:00:00	Dr. Zeynep Çelik
16/01/2024	21:00:00	Dr. Mustafa Arslan
16/01/2024	23:00:00	Dr. Elif Şahin
16/01/2024	01:00:00	Dr. Fatma Özkan
16/01/2024	03:00:00	Dr. Cemal Kaya

Hasta kendi randevularını düzenleyip silebilmektedir. Tıbbi raporlarının detaylarını görüntüleyip

silebilmektedir. Tıbbi rapor görüntüle butonuna bastıktan sonra rapor indirebilir, raporu düzenleyebilir, rapor içeriğini görüntüleyebilmektedir. Hasta kendi ana sayfasında yeni randevu oluşturabilir ve yeni tıbbi rapor ekleyebilir. Ayrıca kendi bilgilerinde değişiklik yapıldığında bildirimlerini görebilmektedir.

C. Hasta Kayıt

Henüz kaydı olmayan bir hasta “Hasta Kayıt” butonuyla kendini kaydedebilmektedir.

D. Doktor Girişi

Doktor kendi ID ve şifresiyle giriş yapabilmektedir. Giriş yaptığı sayfada (Doktor Dashboard) alınan randevular görüntülenmektedir. Herhangi bir randevunun detaylarını görüntülenebilmektedir. Detaylarda hasta bilgileri görüntülenmektedir.

Randevu Tarihi	Randevu Saati	Hasta Adı	Hasta Soyadı	İşlemler
16/01/2024	10:00:00	Emirhan	Smith	Detayları Gözet
16/01/2024	12:00:00	Chloe	Bee	Detayları Gözet
16/01/2024	14:00:00	Chloe	Bar	Detayları Gözet
16/01/2024	16:00:00	Bar	Taylor	Detayları Gözet
16/01/2024	18:00:00	Bar	Taylor	Detayları Gözet
16/01/2024	20:00:00	Chloe	Wise	Detayları Gözet
16/01/2024	22:00:00	Bar	Wise	Detayları Gözet
16/01/2024	24:00:00	Taylor	Sullivan	Detayları Gözet
16/01/2024	26:00:00	Bar	May	Detayları Gözet
16/01/2024	28:00:00	Bar	Hunt	Detayları Gözet

E. Yönetici Girişi

Yönetici kendi ID ve şifresiyle giriş yapabilmektedir. Giriş yaptığı sayfada (Yönetici Dashboard) hastaların ve doktorların tamamı görüntülenebilmektedir. Burada herhangi bir hastanın detaylarına bakabilir, düzenleyebilir ve hastayı sistemden silebilir. Hastaya tıbbi rapor ekleyebilir, raporu silebilir, randevu ekleyebilir, hali hazırda olan randevuları güncelleyip silebilir.

Hasta ID	Adı	Soyadı	İşlemler
1	James	Croley	Detayları Gözet Sil
2	Kristen	Smith	Detayları Gözet Sil
3	Lucas	Donaherty	Detayları Gözet Sil
4	Christy	Bar	Detayları Gözet Sil
5	Raymond	Wilde	Detayları Gözet Sil
6	Randy	Murphy	Detayları Gözet Sil
7	Kenneth	Coleman	Detayları Gözet Sil

Doktor ID	Adı	Soyadı	Unvanları	İşlemler
1	Matthew	Barber	Podiatrist	Detayları Gözet Sil
2	Ida	Shaw	Cardiologist	Detayları Gözet Sil
3	Donna	Barber	Neurologist	Detayları Gözet Sil
4	James	Wright	Cardiologist	Detayları Gözet Sil
5	Barbara	Wright	Cardiologist	Detayları Gözet Sil
6	Robert	Clark	Neurologist	Detayları Gözet Sil

Hastaya tıbbi rapor ekleyebilmekte ve hastaya ait tıbbi raporları görüntüleyip düzenleme ve indirme yapabilmektedir.

Herhangi bir doktorun bilgisini detaylı inceleyebilir, düzenleyebilir ve doktoru silebilir. Ayrıca bu sayfada yeni hasta ve doktor ekleyebilir.

III. YÖNTEM VE İLERLEYİŞ

Projemizin veri tabanı tasarımını hastalar tablosu, doktorlar tablosu, yönetici tablosu, randevular tablosu, tıbbi raporlar tablosu ve bildirimler tablosu olacak şekilde yaptık. İsterlerde

Column Name	Data Type	Allow Nulls
HastaID	int	<input type="checkbox"/>
Ad	nvarchar(50)	<input checked="" type="checkbox"/>
Soyad	nvarchar(50)	<input checked="" type="checkbox"/>
DogumTarihi	date	<input checked="" type="checkbox"/>
Cinsiyet	nvarchar(10)	<input checked="" type="checkbox"/>
TelefonNumarasi	nvarchar(15)	<input checked="" type="checkbox"/>
Adres	nvarchar(255)	<input checked="" type="checkbox"/>
Password	nvarchar(255)	<input checked="" type="checkbox"/>

istenenlere göre her bir tablo için primary key ve foreign keyleri belirleyip tablolara SQL kodlarını

kullanarak ekledik. Tablolarımızı normalizasyon kurallarına göre oluşturduk. Projemizde bir tabloya yapılan ekleme ve çıkarma işlemleri için trigger fonksiyonları yazdık ve bu fonksiyonlarla diğer tabloları güncelledik. Tabloların içeriklerini görüntüleyebilmek için:

“SELECT * FROM *ilgili tablo*adı;” komutu kullanılır.

Arayüz tasarımında cshtml ve javascript kodları kullandık.

Home Index.cshtml

Title: Sayfa başlığı, "Giriş Ekranı" olarak ayarlanmıştır. CSS Bağlantısı: styles.css adlı bir CSS dosyası sayfaya bağlanır. Bu dosya, sayfanın görünümünü özelleştirmek için kullanılabilir. Başlık ve İşlem Talimatı: display-4 sınıfıyla birlikte büyük bir başlık görüntülenir: "Hoş Geldiniz". Ardından, "Lütfen yapmak istediğiniz işlem türünü seçiniz:" şeklinde bir talimat metni görüntülenir. Giriş Butonları: Hasta, doktor ve yönetici için giriş butonları bulunur. Her buton, ilgili controller ve action'a yönlendirme yapar. Örneğin, "Hasta Girişi" butonu, "LoginHasta" action'ı için "Hastalar" controller'ına yönlendirir. Hasta Kayıt Butonu: Yeni bir hasta kaydı oluşturmak için bir buton bulunur. Bu buton, "Create" action'ı için "Hastalar" controller'ına yönlendirme yapar. Bu HTML dosyası, kullanıcıların siteye giriş yapabileceği veya yeni bir hesap oluşturabileceği basit bir giriş ekranı oluşturur.

HomeController;

Yapıcı Metot ILogger türünden bir parametre alıyor. ILogger, ADO.NET yerleşik bir günlük kayıt mekanizmasıdır ve uygulama içinde loglama işlevselliği sağlar. Index Metodu ana sayfayı (Home/Index.cshtml) görüntülemek için kullanılır. Home/Index.cshtml, uygulamanın ana sayfasının görünümünü temsil eder. Bu metod, IActionResult türünde dönüş yapar ve Index.cshtml görünümünü render etmek için kullanılır. Privacy Metodu "Gizlilik Politikası" sayfasını (Privacy.cshtml) görüntülemek için kullanılır. Bu sayfa, kullanıcıların gizlilik politikası hakkında bilgi alabilecekleri bir sayfadır. Error Metodu hata durumlarında bir hata sayfasını (Error.cshtml) görüntülemek için kullanılır. ResponseCache özelliği, hata sayfasının önbelleğe alınmasını engeller, böylece hata sayfası her zaman dinamik olarak oluşturulur. Ayrıca, hata sayfasının RequestId'si, kullanıcının hata raporunu izlemek için kullanılabilir.

Hasta arayüzünde;

HastaInterfaceController Sınıfı; bir denetleyici sınıftır ve Controller sınıfından türetilmiştir. _connectionString, veri tabanı bağlantı dizesini tutar. Yapıcı metod, IConfiguration

parametresi alır ve bağlantı dizesini configuration nesnesinden alır. Index metodu, giriş yapmış kullanıcının (hastanın) bilgilerini görüntüler. hastaID, oturumdan (Session) alınır; eğer oturumdan alınamazsa varsayılan olarak 1 atanır. HastaDashboardViewModel, randevular, tıbbi raporlar ve bildirimlerle doldurularak View'a gönderilir. GetRandevular: Hastanın randevularını veri tabanından çeker. GetTibbiRaporlar: Hastanın tıbbi raporlarını veri tabanından çeker. GetHastaBildirimler: Hastanın bildirimlerini veri tabanından çeker. MarkAsRead metodu, belirtilen bildirimID ile bildirimi okundu olarak işaretler ve Index aksiyonuna yönlendirir. GetHastaRandevular metodu, hastanın randevularını getirir ve RandevuViewModel ile View'a gönderir.

HastalarController Sınıfı; Bu sınıf, hasta yönetimi ile ilgili işlemleri içeren bir ADO.NET denetleyicisidir. Özel Değişken, Veritabanı bağlantı dizesini tutar. Yapıcı Metot yapılandırma dosyasından veritabanı bağlantı dizesini alır ve özel değişkene atar. Bağlantı dizesi tanımlı değilse bir istisna fırlatır. Index Metodu hastaların listesini sayfalama (pagination) ile birlikte döner. Veritabanından hasta sayısını alır ve sayfalama için gerekli hesaplamaları yapar. Belirli bir sayfaya ait hastaları çekerek View'a döner. LoginHasta Metodları hastaların giriş yapmasını sağlar. Giriş bilgilerini kontrol eder ve başarılı girişlerde oturum (session) bilgilerini ayarlar. Bildirimler Metodu belirli bir hastaya ait bildirimleri veritabanından alır ve View'a döner. Details Metodu belirli bir hastanın detaylarını, randevularını ve tıbbi raporlarını getirir. Kullanıcının rolüne göre yetki kontrolü yapar. Create Metodu yeni bir hasta eklemek için formu döner ve ekleme işlemini yapar. Form gönderildiğinde, yeni hasta bilgilerini veritabanına ekler. Edit Metodu bir hastayı düzenlemek için formu döner ve düzenleme işlemini yapar. Form gönderildiğinde, hasta bilgilerini veritabanında günceller. Delete Metodu bir hastayı silmek için formu döner ve silme işlemini yapar. Form gönderildiğinde, belirtilen hastayı veritabanından siler.

Doktor arayüzünde;

DoktorInterfaceController Sınıfı; Bağlantı Dizesi veritabanı bağlantı dizesini yapılandırma dosyasından (_connectionString) alır. Index Metodu giriş yapmış doktorun ana sayfasını döner. Oturumdan doktorID alır. Eğer doktorID null ise, doktorun giriş yapması için LoginDoktor sayfasına yönlendirir. Doktor giriş yaptıysa, randevular, tıbbi raporlar ve bildirimleri alarak bunları bir DoktorDashboardViewModel modeline ekler ve View'a döner. GetDoktorBildirimler Metodu doktorun bildirimlerini getirir. Veritabanında doktorun bildirimlerini sorgular ve List<Bildirim> olarak döner. MarkAsRead Metodu bir bildirimi okundu olarak işaretler. Veritabanında belirtilen bildirimID'ye sahip bildirimi okundu (Okundu = 1) olarak günceller ve ardından Index metoduna yönlendirir. GetDoktorunRandevulari Metodu doktorun randevularını getirir. Veritabanında doktorun randevularını sorgular ve List<Randevu> olarak döner. Randevu bilgilerine ek olarak, hasta adı ve soyadı da alınır. GetDoktorunTibbiRaporlari Metodu doktorun tıbbi raporlarını getirir. Veritabanında doktorun tıbbi raporlarını sorgular ve List<TibbiRapor> olarak döner. Rapor bilgilerine ek olarak, hasta adı ve soyadı da alınır. Veritabanı Bağlantısı: Her işlemde SQL bağlantısı açılır ve gerekli sorgular çalıştırılır. Oturum Yönetimi doktorun oturum bilgileri (DoktorID) kullanılarak

ilgili veriler çekilir. DoktorDashboardViewModel modeli, doktorun randevuları, tıbbi raporları ve bildirimleri ile ilgili bilgileri taşıyarak View'a gönderilir. Bildirim İşaretleme: Bildirimler okundu olarak işaretlenebilir. Bu işlem veritabanında güncellenir ve ardından ana sayfaya yönlendirilir.

DoktorlarController Sınıfı ;Bağlantı Dizesi veritabanı bağlantı dizesini yapılandırma dosyasından (_connectionString) alır. Index Metodu tüm doktorların listesini getirir ve görüntüler. Veritabanından doktor bilgilerini çekerek bir listeye ekler ve bu listeyi View'a gönderir. LoginDoktor Metodu doktorların giriş yapmasını sağlar. GET giriş formunu döner. POST girilen bilgileri doğrular ve doğruysa doktoru oturuma kaydeder, ardından doktor arayüzüne yönlendirir. Details Metodu belirli bir doktorun detaylarını ve randevularını görüntüler. Doktor bilgilerini ve randevularını veritabanından çeker, ViewBag'e randevuları ekleyerek View'a döner. Create Metodu yeni bir doktor ekler. GET boş bir form döner. POST formdan gelen verileri alır, veritabanına ekler ve yöneticinin arayüzüne yönlendirir. Edit Metodu belirli bir doktorun bilgilerini düzenler. Düzenlenecek doktorun bilgilerini formda döner. Güncellenmiş bilgileri alır ve veritabanında günceller, ardından bir önceki sayfaya veya varsayılan sayfaya yönlendirir. Delete Metodu belirli bir doktoru siler. Silinecek doktorun bilgilerini doğrulama için formda döner. Doktoru veritabanından siler ve yöneticinin arayüzüne yönlendirir. Oturum Yönetimi: Doktor girişinde oturum verileri saklanır (DoktorID ve Role). Hata Yönetimi: Giriş işlemlerinde hata mesajları görüntülenir. ViewBag ve TempData: Randevu bilgileri ve önceki URL gibi geçici veriler için kullanılır.

Yönetici arayüzü;

YöneticilerController Sınıfı: Yapıcı metod IConfiguration aracılığıyla bağlantı dizesini alır ve _connectionString değişkenine atar. Index Metodu HTTP GET isteğiyle çağrıldığında çalışacak olan metoddur. Veritabanından yöneticileri çeker ve YöneticiDashboardViewModel nesnesine ekler. Bu nesne, yönetici panelinin ana görünümünü temsil eder ve içinde yöneticilerin yanı sıra hastalar ve doktorlar da bulunabilir. LoginYönetici Metodları kullanıcı girişi sağlayan metotlardır. Bir HTTP GET isteğiyle çağrılan LoginYönetici metodu, kullanıcı giriş sayfasını görüntüler. Bir HTTP POST isteğiyle çağrılan LoginYönetici(int YöneticiID, string password) metodu ise kullanıcının kimlik bilgilerini doğrular ve doğruysa yönetici paneline yönlendirir. GetHastalar ve GetDoktorlar Metodları veritabanından hastaları ve doktorları çekmek için kullanılan yardımcı metotlardır.

YöneticilerInterfaceController; Yapıcı metod, IConfiguration aracılığıyla bağlantı dizesini alıyor ve _connectionString değişkenine atıyor. Index Metodu yönetici arayüzünün ana sayfasını oluşturuyor. Sayfa numarası parametresi alıyor ve belirli bir sayfa için hastaları ve doktorları getiriyor. Bu, sayfa sayfalama işlevselliği sağlar. Ayrıca, toplam sayfa sayısını hesaplar ve bunu görüntülemek için bir model oluşturur. HastaEkle Metodları HTTP GET isteğiyle çağrılan HastaEkle metodu, bir hasta eklemek için bir form görüntüler. Bir HTTP POST isteğiyle çağrılan HastaEkle(Hasta hasta) metodu ise gelen hasta bilgilerini veri tabanına ekler. HastaSil Metodları HTTP GET isteğiyle çağrılan HastaSil metodu, bir hasta silme işlemi için bir form görüntüler. Bir HTTP POST isteğiyle

çağrılan HastaSil(int hastaID) metodu ise belirtilen hasta ID'sine sahip hastayı veritabanından siler. DoktorEkle ve DoktorSil Metodları: Hasta işlemleri ile benzer şekilde, doktor ekleme ve silme işlemleri için metotlar bulunmaktadır. Randevuİptal, TıbbiRaporEkle, TıbbiRaporGuncelle Metodları randevu iptali, tıbbi rapor ekleme ve tıbbi rapor güncelleme işlemlerini gerçekleştirir. Benzer şekilde, HTTP GET ve POST istekleri için ayrı ayrı metotlar bulunur. HastaGuncelle ve DoktorGuncelle Metodları Varolan bir hastanın veya doktorun bilgilerini güncellemek için kullanılan metotlardır. Bu metotlar da HTTP GET ve POST istekleri için ayrı ayrı bulunur.

Randevular arayüzü;

Yapıcı metod, IConfiguration türünden bir parametre alır. Bu parametre, uygulamanın yapılandırma bilgilerini içerir, özellikle de bağlantı dizisini (connection string). Index Metodu randevuların listesini getirir. Veritabanından tüm randevuları çeker ve bunları bir listeye ekler. Sonra ViewBag aracılığıyla bu randevuları görünüme (View) geçirir ve Index.cshtml görünümünü render etmek için kullanır. GetUzmanlikAlanlari Metodu doktorların uzmanlık alanlarını döndürür. Veritabanından tüm farklı uzmanlık alanlarını çeker ve bunları bir liste olarak JSON formatında döndürür. GetHastanelerByUzmanlik Metodu belirli bir uzmanlık alanına sahip doktorların çalıştığı hastaneleri döndürür. Veritabanından belirtilen uzmanlık alanına sahip doktorların çalıştığı hastaneleri çeker ve bunları bir liste olarak JSON formatında döndürür. GetDoktorlarByHastaneAndUzmanlik Metodu belirli bir hastanede ve belirli bir uzmanlık alanına sahip doktorları döndürür. Veritabanından belirtilen hastanede ve belirtilen uzmanlık alanına sahip doktorları çeker ve bunları bir liste olarak JSON formatında döndürür. Details Metodu belirli bir randevunun detaylarını getirir. Veritabanından belirtilen randevunun detaylarını çeker ve bunları bir Randevu nesnesi olarak görünüme geçirir. Create Metodu yeni bir randevu oluşturur. Kullanıcıdan gelen bilgileri kullanarak bir randevu kaydı oluşturur ve bu kaydı veritabanına ekler. Ayrıca, doktorun bildirimlerine yeni bir randevu ekleme işlemi yapar. Edit Metodu belirli bir randevuyu düzenlemek için kullanılır. Kullanıcı tarafından yapılan değişikliklere göre randevu kaydını günceller. Delete Metodu belirli bir randevuyu siler. Veritabanından belirtilen randevuyu siler ve işlem sonrası kullanıcıyı ilgili sayfaya yönlendirir.

Tıbbi rapor arayüzü;

TıbbiRaporlarController yapıcı metodu IConfiguration aracılığıyla bir _connection string alır. Index Metodu tüm tıbbi raporları veri tabanından çeker ve bir liste olarak görüntüler. Details Metodu belirli bir tıbbi raporun ayrıntılarını görüntüler. Create Metodu yeni bir tıbbi rapor oluşturur. Hasta veya doktor tarafından oluşturulabilir. Gerekli bilgileri almak için bir form gösterir. Create Post metodu yeni bir tıbbi rapor oluşturur ve veritabanına kaydeder. Edit metodu varolan bir tıbbi raporu düzenleme ekranını gösterir. Edit Post metodu varolan bir tıbbi raporu günceller. Delete metodu belirli bir tıbbi raporu silme işlemini gerçekleştirir. DeleteConfirmed metodu bir tıbbi raporu siler ve ardından yönlendirme işlemini gerçekleştirir. GetTıbbiRaporById Metodu belirli bir tıbbi raporu ID'ye göre veri tabanından alır. Download metodu belirli bir tıbbi raporu

indirme işlemini gerçekleştirir. Veri tabanından URL alır, bu URL'yi kullanarak dosyayı indirir ve kullanıcıya sunar.

IV. SONUÇ

Projemizde, her bir isteri tek tek ele aldık ve başarıyla tamamladık. Hastalar, doktorlar ve yöneticiler, MHRS gibi bir sistem aracılığıyla randevu alabilir ve kendi bilgilerini görüntüleyebilirler. Bu proje, HTML, SQL gibi dilleri öğrenmemize ve veri tabanı kullanımını pratiğe dökmemize olanak sağladı.

KAYNAKLAR

- [1] <https://www.cihankaraca.com/wp-content/uploads/2019/08/burunk%C4%B1r%C4%B1klari.jpg>
- [2] https://synevo.com.tr/upload/b_159201714856488.jpg
- [3] <https://cf.kizlarsoruyor.com/q10947974/91c91528-8227-4560-b5ca-fd1ec9fe7245.jpg>
- [4] <https://mavimarti.net/wp-content/uploads/2015/08/mmpi-300x200.jpg>
- [5] https://kbb-forum.net/journal/uploads/figure_KBB_503_1.jpg
- [6] https://dentamar.com.tr/_yuk/blogresim/panaromikrontgen.jpg
- [7] <https://www.webtekno.com/images/editor/default/0003/62/4cc4b6e179797dcbfdee1a70bc7becbc2eb64b4c.jpeg>
- [8] <https://www.hekimoglugoruntuleme.com/wp-content/uploads/2016/01/sinus-grafisi-fiyaatlari-2023.jpg>
- [9] https://cdn.pixabay.com/photo/2016/01/08/22/35/xray-1129430_1280.jpg
- [10] <https://betatom.com.tr/wp-content/uploads/2023/07/BEYIN1-0-1.jpg>
- [11] <https://betatom.com.tr/wp-content/uploads/2023/07/BEYIN1-2-1.jpg>
- [12] https://miro.medium.com/v2/resize:fit:1400/1*scU1iIV95BIqkRX2IDQ8g.jpeg
- [13] <https://cf.kizlarsoruyor.com/q14643393/primary-share.png?33>
- [14] <https://image.milimaj.com/i/milliyet/75/869x477/5c8e1ead45d2a06b40b2f7cc.jpg>
- [15] <https://www.emartomografi.com/wp-content/uploads/2022/11/kardiyak-mr-fiyaatlari.png>
- [16] <https://drgulhizkaratas.com/wp-content/uploads/bobrek-ultrasonu.jpg>
- [17] https://png.pngtree.com/background/20231217/original/pngtree-ultrasound-film-of-a-woman-kidney-kidney-health-ray-photo-picture-image_6861011.jpg
- [18] <https://www.webtekno.com/images/editor/default/0003/62/6a171940e05345ffa924e22537d347501cdc6687.jpeg>
- [19] [Nasıl Yapılır 13 Mvc File Upload ile Resim Yükleme - YouTube](#)

Yalancı kod

Program

Main(args)

Oluştur HostBuilder(args)

.ConfigureWebHostDefaults(webBuilder =>
webBuilder.UseStartup<Startup>());

Başlat ve çalıştır.

Startup

Configuration

Constructor(configuration)

Configuration = configuration

ConfigureServices(services)

Servisleri ekle: Controllers ve Views için
AddControllersWithViews, Dağıtılmış bellek önbellegi için
AddDistributedMemoryCache, HTTPS yönlendirmesi için
AddHttpsRedirection, Session yönetimi için AddSession

Configure(app, env)

Geliştirme modunda ise

Geliştirme hata sayfasını kullan

Değilse

Genel hata işleyicisini kullan

HSTS kullan

HTTPS yönlendirmesini etkinleştir

Statik dosyaları sun

Yönlendirme kullan

Yetkilendirme kullan

Session kullan

Endpoint'i yapılandır

YöneticilerController

Constructor(configuration)

ConnectionString'i al

Index()

Yöneticileri al:

Tüm yöneticileri veritabanından al

Hastaları ve Doktorları al

ViewModel oluştur ve göster

Index sayfasını göster

HomeController

Constructor(logger)

Logger'ı al

Index() Ana sayfayı göster

HastalarController

DoktorlarController

