2024

ISPARTA UYGULAMALI BİLİMLER ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ YAZILIM MÜHENDİSLİĞİ TASARIM RAPORU

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

Tasarım (Design)

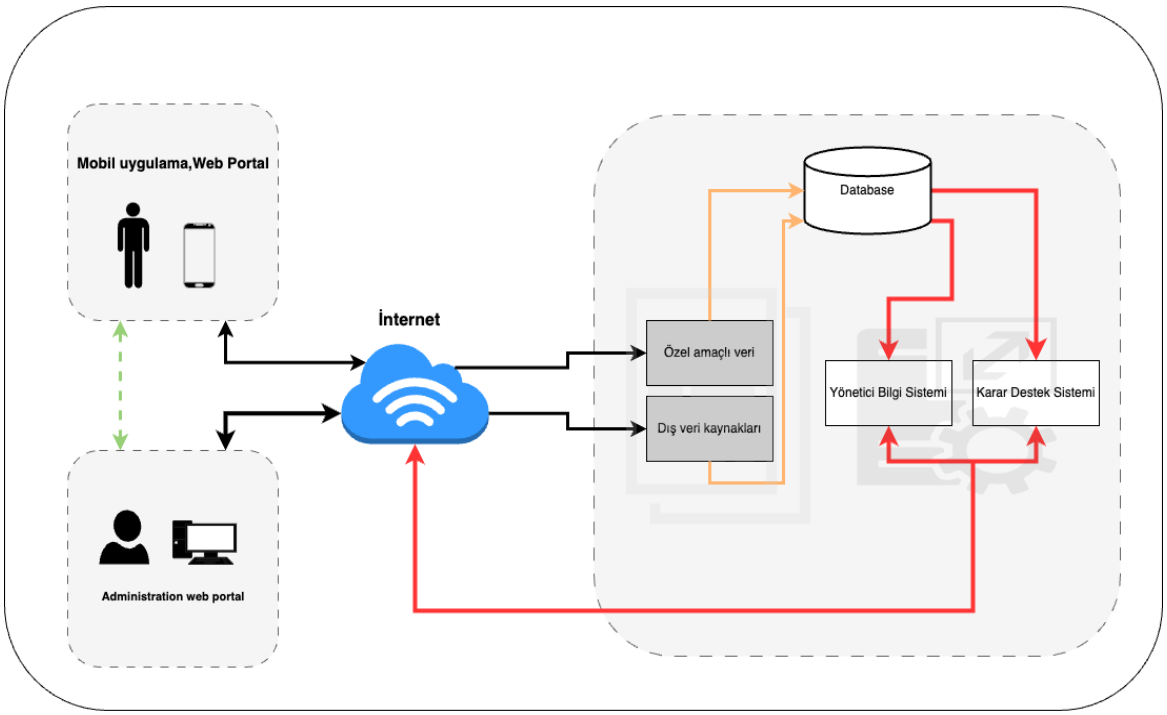
1 Sistem Tasarımı (System Design)

1a Tasarım Hedefleri (Design Goals)

1. **Kullanıcı Deneyimi Odaklı Arayüz**: Sistemin kullanıcı arayüzü, bireylerin diyetisyenlerle randevu planlaması ve iletişim kurmalarını kolaylaştıracak şekilde tasarlanmalıdır. Basit, kullanıcı dostu ve etkileşimli bir arayüz, kullanıcıların platformu rahatça kullanmalarını sağlar.
2. **Güvenlik ve Gizlilik**: Kullanıcıların kişisel sağlık bilgileri, sıkı güvenlik önlemleriyle korunmalıdır. Veriler, endüstri standardı şifreleme yöntemleri ile güvence altına alınmalı ve kullanıcıların gizliliği her zaman ön planda tutulmalıdır.
3. **Uyumluluk ve Entegrasyon**: Sistem, sağlık sektöründe yaygın olarak kullanılan diğer yazılımlarla entegre edilebilmelidir. Örneğin, elektronik sağlık kayıtları (ESK) sistemleri ile uyumlu olmalı ve veri alışverişini kolaylaştırmalıdır.
4. **Esneklik ve Ölçeklenebilirlik**: Sistem, gelecekteki genişlemelere ve değişikliklere uyum sağlayacak şekilde esnek olmalıdır. Yeni özelliklerin ve modüllerin eklenmesi, sistemdeki veri miktarının artması gibi durumlara hazır olmalıdır.
5. **Performans ve Hız**: Sistem, hızlı yanıt süreleri ve yüksek performans sağlamalıdır. Kullanıcıların randevu alma ve iletişim kurma gibi temel işlemleri hızlı bir şekilde gerçekleştirebilmelidir.
6. **Erişilebilirlik**: Sistem, engelli kullanıcılar da dahil olmak üzere herkes için erişilebilir olmalıdır. Engelli kullanıcılar için ek destekler sağlanmalı ve kullanıcı dostu bir deneyim sunulmalıdır.
7. **Veri Yönetimi ve Analitik**: Sistem, kullanıcı verilerini etkin bir şekilde yönetmeli ve analiz edebilmelidir. Bu, diyetisyenlerin kullanıcıların sağlık geçmişlerini daha iyi anlamalarına ve daha iyi hizmet sunmalarına olanak tanır.
8. **Geliştirme ve Bakım Kolaylığı**: Sistem, geliştirme sürecinde ve sonrasında kolayca yönetilebilir olmalıdır. Kodun anlaşılır olması, hata ayıklama süreçlerinin kolay olması ve sistem bakımının sorunsuz yapılabilmesi önemlidir.
9. **Mobil Uyum**: Sistem, mobil cihazlardan erişilebilir olmalıdır. Mobil uygulamalar veya web tabanlı arayüzler, kullanıcıların istedikleri zaman ve herhangi bir yerden platforma erişmelerini sağlamalıdır.
10. **Tutarlılık ve Kolaylık**: Kullanıcı arayüzü, tutarlılık prensiplerine dayalı olarak tasarlanmalıdır. Menü yapıları, düğme yerleşimleri ve diğer arayüz öğeleri, kullanıcıların platformu kolayca gezmesini sağlayacak şekilde tutarlı olmalıdır.
11. **Çoklu Dil Desteği**: Sistem, farklı dillerde kullanılabilme özelliği sunmalıdır. Kullanıcıların tercih ettikleri dili seçmelerine olanak tanıyan çoklu dil seçenekleri, platformun daha geniş bir kullanıcı kitlesine hitap etmesini sağlar.
12. **Gerçek Zamanlı Bildirimler**: Sistem, randevu hatırlatmaları, iletişim talepleri ve diğer önemli bildirimleri gerçek zamanlı olarak kullanıcılara iletmelidir. Bu, kullanıcıların randevu ve iletişim planlarını düzenli tutmalarına yardımcı olur.
13. **Diyetisyen İzleme ve Performans Değerlendirmesi**: Sistem, diyetisyenlerin performansını izlemek ve değerlendirmek için araçlar sağlamalıdır. Bu, diyetisyenlerin kullanıcılarla etkili bir şekilde iletişim kurmasını ve sağlık hedeflerine ulaşmalarını destekler.
14. **Kapsamlı Raporlama Yetenekleri**: Sistem, kullanıcıların sağlık geçmişleri, ilerlemeleri ve diğer önemli bilgiler hakkında kapsamlı raporlar oluşturabilmelidir. Bu raporlar, hem kullanıcılar hem de diyetisyenler için faydalıdır ve sağlık hedeflerine ulaşma sürecini izlemeyi kolaylaştırır.

15. **Sosyal Medya Entegrasyonu**: Sistem, kullanıcıların sağlık ve beslenme ile ilgili deneyimlerini sosyal medya platformlarında paylaşmalarını ve diğer kullanıcılarla etkileşime geçmelerini sağlayacak entegrasyonlar sunmalıdır. Bu, kullanıcıların birbirlerinden destek almasını ve motive olmalarını sağlar.
16. **Eğitim ve Kaynak Merkezi**: Sistem, kullanıcılara sağlık ve beslenme konularında bilgi ve kaynaklar sunmalıdır. Eğitim materyalleri, makaleler, video dersleri ve diğer kaynaklar, kullanıcıların daha bilinçli kararlar almasına yardımcı olur.

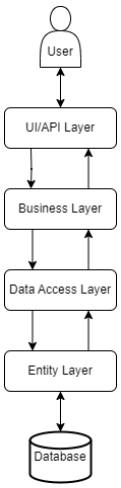
2 (Current Software Architecture)



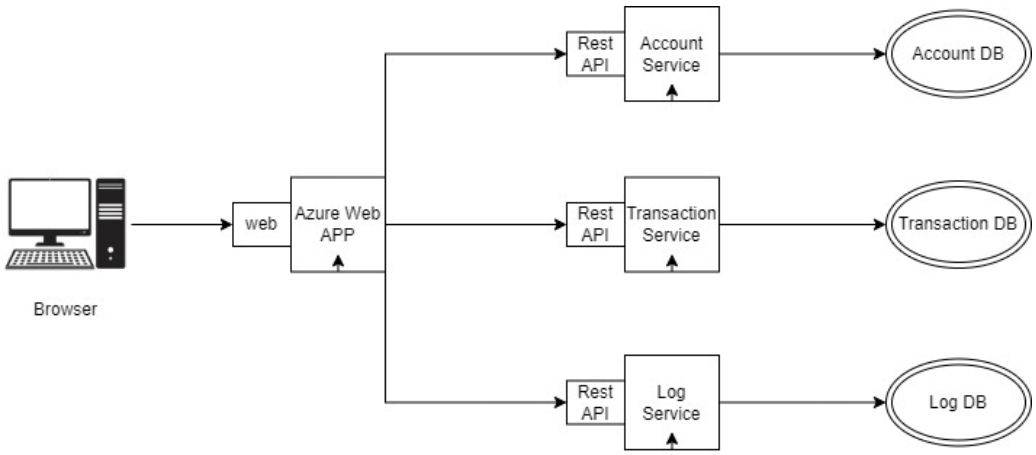
Şekil 1 Mimari sistem tasarımı

3 Önerilen Yazılım Mimarisi (Proposed Software Architecture)

3a Genel Bakış-İnceleme (Overview)

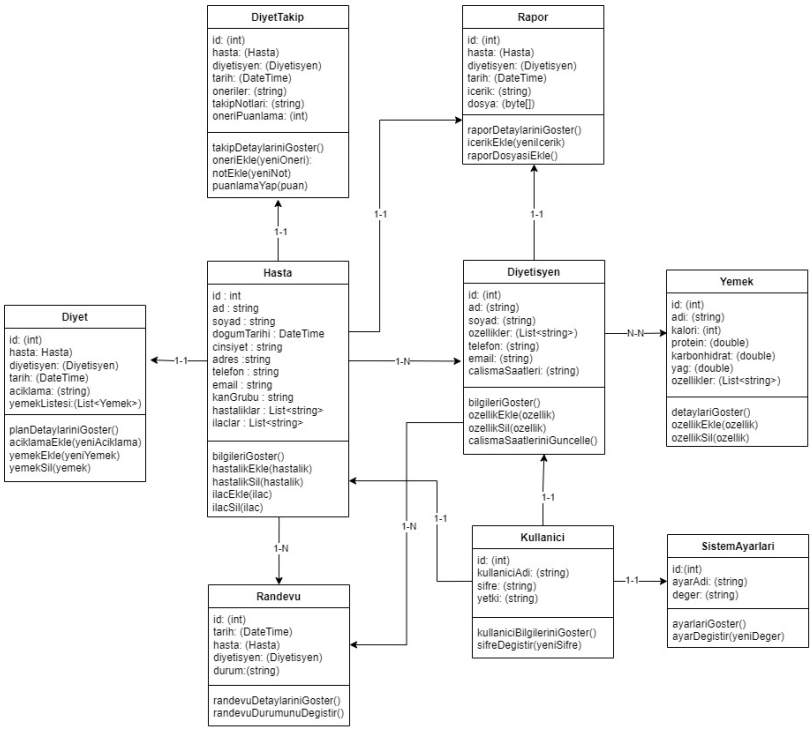


Şekil 2 Katmanlı Mimari



Şekil 3 Uygulama mimarisi diyagramı

3b Statik Model-Class Diyagramları



Şekil 4 Class Diyagramı

Hasta Sınıfı:

Değişkenler:

- id: Hasta ID'si (int)
- ad: Hasta adı (string)
- soyad: Hasta soyadı (string)
- dogumTarihi: Hasta doğum tarihi (DateTime)
- cinsiyet: Hasta cinsiyeti (string)
- adres: Hasta adresi (string)

- telefon: Hasta telefon numarası (string)
- email: Hasta e-posta adresi (string)
- kanGrubu: Hasta kan grubu (string)
- hastaliklar: Hasta geçmiş hastalıkları (List<string>)
- ilaclar: Hasta kullandığı ilaçlar (List<string>)

Methodlar:

- bilgileriGoster(): Hasta bilgilerini konsola yazdırır.
- hastalikEkle(hastalik): Hasta geçmiş hastalıklarına yeni bir hastalık ekler.
- hastalikSil(hastalik): Hasta geçmiş hastalıklarından bir hastalığı siler.
- ilacEkle(ilac): Hasta kullandığı ilaçlara yeni bir ilaç ekler.
- ilacSil(ilac): Hasta kullandığı ilaçlardan bir ilacı siler.

Randevu Sınıfı:

Değişkenler:

- id: Randevu ID'si (int)
- tarih: Randevu tarihi (DateTime)
- hasta: Randevuya katılan hasta (Hasta)
- diyetisyen: Randevuya atanmış diyetisyen (Diyetisyen)
- durum: Randevu durumu (string)

Methodlar:

- randevuDetaylariniGoster(): Randevu detaylarını konsola yazdırır.
- randevuDurumunuDegistir(yeniDurum): Randevu durumunu günceller.

Diyetisyen Sınıfı:

Değişkenler:

- id: Diyetisyen ID'si (int)
- ad: Diyetisyen adı (string)
- soyad: Diyetisyen soyadı (string)
- ozellikler: Diyetisyen uzmanlık alanları (List<string>)
- telefon: Diyetisyen telefon numarası (string)
- email: Diyetisyen e-posta adresi (string)
- calismaSaatleri: Diyetisyen çalışma saatleri (string)

Methodlar:

- bilgileriGoster(): Diyetisyen bilgilerini konsola yazdırır.
- ozellikEkle(ozellik): Diyetisyenin uzmanlık alanlarına yeni bir özellik ekler.
- ozellikSil(ozellik): Diyetisyenin uzmanlık alanlarından bir özelliği siler.
- calismaSaatleriniGuncelle(yeniSaatler): Diyetisyenin çalışma saatlerini günceller.

Diyet Planı Sınıfı:

Değişkenler:

- id: Diyet planı ID'si (int)
- hasta: Diyet planı atanmış hasta (Hasta)
- diyetisyen: Diyet planını oluşturan diyetisyen (Diyetisyen)
- tarih: Diyet planının oluşturulma tarihi (DateTime)
- aciklama: Diyet planının açıklaması (string)
- yemekListesi: Diyet planında bulunan yemeklerin listesi (List<Yemek>)

Methodlar:

- planDetaylariniGoster(): Diyet planı detaylarını konsola yazdırır.
- aciklamaEkle(yeniAciklama): Diyet planına yeni bir açıklama ekler.
- yemekEkle(yeniYemek): Diyet planına yeni bir yemek ekler.
- yemekSil(yemek): Diyet planından bir yemeği siler.

Yemek Sınıfı:

Değişkenler:

- id: Yemek ID'si (int)
- adi: Yemeğin adı (string)
- kalori: Yemeğin kalori değeri (int)
- protein: Yemeğin protein miktarı (double)
- karbonhidrat: Yemeğin karbonhidrat miktarı (double)
- yag: Yemeğin yağ miktarı (double)
- ozellikler: Yemeğin içerdiği özel malzemeler veya bilgiler (List<string>)

Methodlar:

- detaylariGoster(): Yemeğin detaylarını konsola yazdırır.
- ozellikEkle(ozellik): Yemeğe yeni bir özellik ekler.
- ozellikSil(ozellik): Yemekten bir özelliği siler.

Diyet Takip Sınıfı:

Değişkenler:

- id: Diyet takip ID'si (int)
- hasta: Diyet takibini yapan hasta (Hasta)
- diyetisyen: Diyet takibini oluşturan diyetisyen (Diyetisyen)
- tarih: Diyet takibinin yapıldığı tarih (DateTime)
- oneriler: Diyetisyenin verdiği öneriler (string)
- takipNotlari: Diyet takibi sırasında alınan notlar (string)
- oneriPuanlama: Hasta tarafından önerilerin değerlendirilmesi (int)

Methodlar:

- takipDetaylariniGoster(): Diyet takip detaylarını konsola yazdırır.
- oneriEkle(yeniOneri): Diyet takibine yeni bir öneri ekler.
- notEkle(yeniNot): Diyet takibine yeni bir not ekler.
- puanlamaYap(puan): Önerilere puan verir.

Rapor Sınıfı:

Değişkenler:

- id: Rapor ID'si (int)
- hasta: Raporu isteyen hasta (Hasta)
- diyetisyen: Raporu oluşturan diyetisyen (Diyetisyen)
- tarih: Raporun oluşturulma tarihi (DateTime)
- icerik: Raporun içeriği (string)
- dosya: Rapor dosyası (byte[])

Methodlar:

- raporDetaylariniGoster(): Rapor detaylarını konsola yazdırır.
- icerikEkle(yeniIcerik): Rapor içeriğine yeni bilgiler ekler.
- raporDosyasiEkle(dosyaYolu): Rapor dosyasını ekler.

Sistem Ayarları Sınıfı:

Değişkenler:

- id: Ayarlar ID'si (int)
- ayarAdi: Ayarın adı (string)
- deger: Ayarın değeri (string)

Methodlar:

- ayarlariGoster(): Sistem ayarlarını konsola yazdırır.
- ayarDegistir(yeniDeger): Ayarın değerini değiştirir.

Kullanıcı Sınıfı:

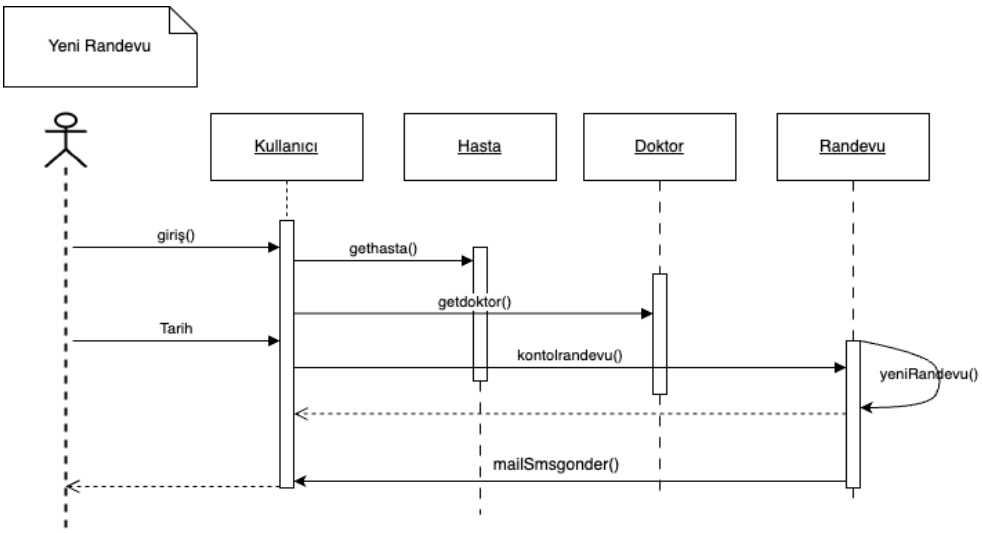
Değişkenler:

- id: Kullanıcı ID'si (int)
- kullaniciAdi: Kullanıcı adı (string)
- sifre: Kullanıcı şifresi (string)
- yetki: Kullanıcının yetkisi (string)

Methodlar:

- kullaniciBilgileriniGoster(): Kullanıcı bilgilerini konsola yazdırır.
- sifreDegistir(yeniSifre): Kullanıcının şifresini değiştirir.

3c Dinamik Model-Squence(Sıralama) Diyagramları



3d Donanım / yazılım eşlemesi (Hardware / software mapping)

Sunucu (Server)

- Kimliği: SRV001
- Amaç: Merkezi veri depolama ve işleme
- Aldığı yazılım bileşeni: Uygulama sunucusu yazılımı
- Teknik Özellikler:
    - İşlemci: Intel Xeon Gold 6242R, 2.8GHz, 24 çekirdek
    - RAM: 64GB DDR4 ECC bellek
    - Depolama: 2TB SSD
    - Ağ Arayüzleri: 2x 10 Gigabit Ethernet
    - Güç Girişi: 110-240V AC, 50-60Hz
    - Konektör: RJ45 Ethernet, USB, VGA
    - İşletim Sistemi: Windows Server 2019

Ağ Anahtarı (Switch)

- Kimliği: SW001

- Amaç: Ağ trafiğini yönlendirme ve kontrol
- Aldığı yazılım bileşeni: Ağ yönetim yazılımı
- Teknik Özellikler:
    - Port Sayısı: 48x 1 Gigabit Ethernet, 4x 10 Gigabit Ethernet
    - İşletim Sistemi: Cisco IOS
    - Güç Girişi: 100-240V AC, 50-60Hz
    - Konektör: RJ45 Ethernet, SFP/SFP+ modülleri
    - Yönetim Arabirimi: Web tabanlı GUI, SSH, Telnet

Depolama Alanı (Storage Array)

- Kimliği: STR001
- Amaç: Büyük veri depolama ve yedekleme
- Aldığı yazılım bileşeni: Veri yönetimi yazılımı
- Teknik Özellikler:
    - Kapasite: 100TB (10x 10TB SAS HDD)
    - RAID Desteği: RAID 5, RAID 6
    - Bağlantı Arayüzü: 2x 10 Gigabit Ethernet, Fibre Channel
    - Yedekleme: Otomatik yedekleme planları
    - Güç Girişi: 200-240V AC, 50-60Hz
    - Konektör: Fibre Channel, Ethernet, USB

Monitör

- Kimliği: MON001
- Amaç: Görsel arayüz sağlama
- Teknik Özellikler:
    - Boyut: 27 inç
    - Çözünürlük: 2560x1440 (2K)
    - Bağlantı: HDMI, DisplayPort

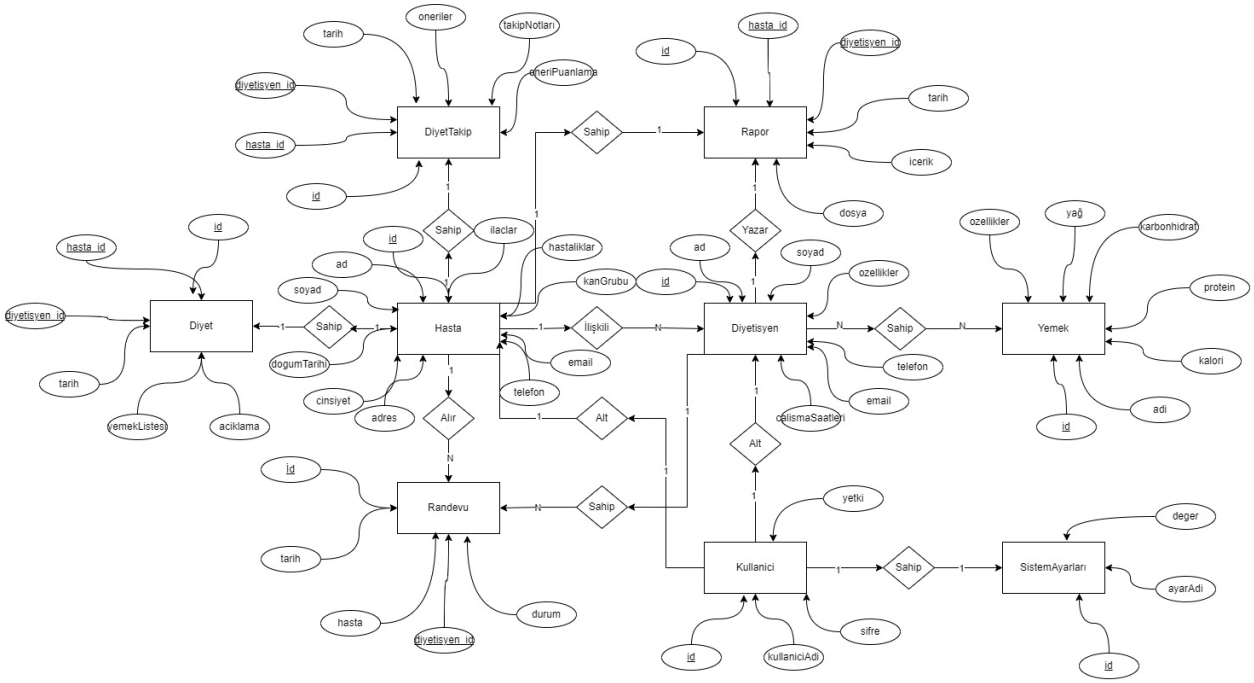
- Güç Girişi: 100-240V AC, 50-60Hz
- Konektör: HDMI, DisplayPort

Yazıcı

- Kimliği: PRN001
- Amaç: Belge ve rapor basımı
- Teknik Özellikler:
  - Tür: Renkli lazer yazıcı
  - Hız: 30 ppm (sayfa başına)
  - Bağlantı: USB, Ethernet
  - Kağıt Boyutu: A4, A5, Letter, Legal
  - Güç Girişi: 100-240V AC, 50-60Hz
  - Konektör: USB, Ethernet Bu bileşenler,

Projenin donanım altyapısını oluşturmak için bu ekipmanlar kullanılacaktır. Her bir bileşenin özellikleri, işlevleri ve teknik detayları dikkate alınarak seçilmiştir.

3e ER-Diyagramı-Veri Sözlüğü (Data Dictionary)



Şekil 2 ER Diyagramı

Tablo 1 Veri Sözlüğü Tablosu

Sözlük Elemanı	Tanım	Örnek
Veri Elemanı Numarası	Her bir veri elemanı için eşsiz sayısal tanım.	1,2,3.....
Veri Elemanı	Veri elemanını tanımlamak için kullanılan alfanümerik kısaltma.	HST_ID, DYTSY_ID
Veri Elemanı Adı	Veri alanının açık adı.	Hasta ID, Diyetisyen ID
Veri Elemanının Tanımı	Alanın anlamını, hesaplanma yöntemini ve referansları içeren detaylı açıklama.	Hasta kimlik numarası, Diyetisyen kimlik numarası
Veri Kaynağı	Veri elemanının kaynağı.	Hastane bilgi sistemi, Diyetisyen kayıt formu
Veri Sorumlusu	Veri elemanından sorumlu kişi.	Veri yöneticisi, Diyetisyen
Veri Tipi ve Uzunluğu	Veri elemanının tipi ve uzunluğu.	Sayısal (int), Karakter (varchar)
İzin Verilen Değerler ve Parametreler	Veri elemanına izin verilen değerler ve parametreler.	A, B, C, ..., 0-100
Veri Elemanının Varlık Durumu	Veri elemanının zorunlu, isteğe bağlı veya koşullu olma durumu.	Zorunlu, İsteğe Bağlı, Koşullu
Anlamsal Kural	Veri elemanının anlamını açıklayan kural.	Hasta ID, Diyetisyen ID
Veri Oluşturma	Veri elemanının oluşturulduğu tarih.	2023-01-15
Veri Güncelleme	Veri elemanının son güncelleme tarihi.	2024-05-20

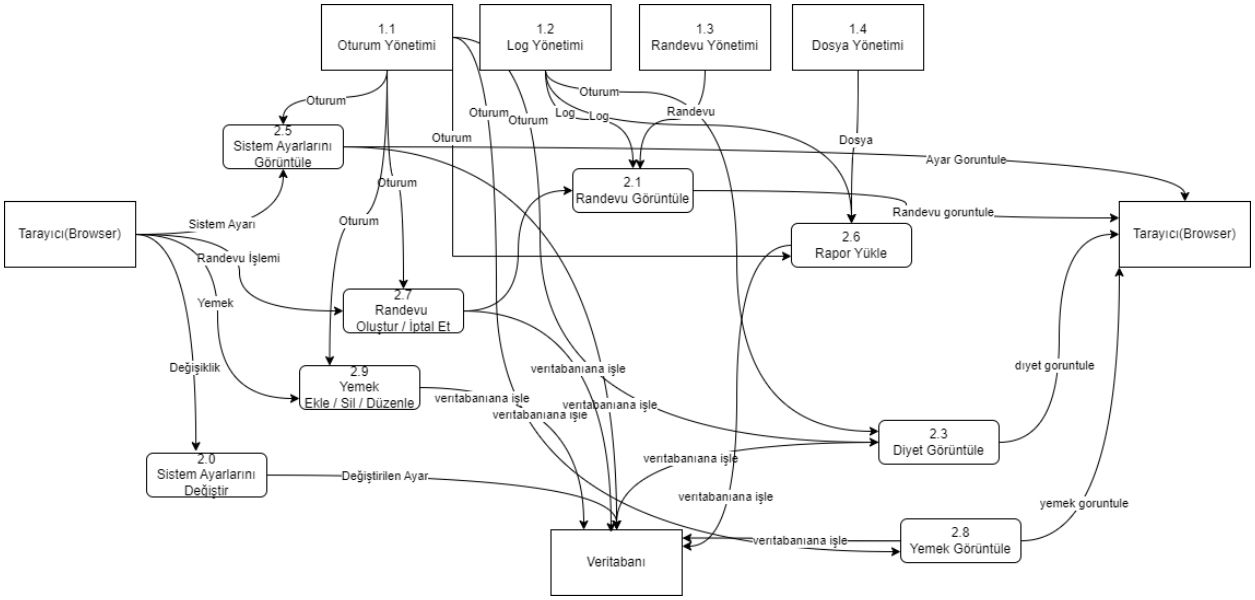
Alternatif Veri Adları	Veri elemanının alternatif adlandırmaları.	Hastane Kimlik Numarası, Diyetisyen Kimlik Numarası
Gizlilik	Veri elemanının gizlilik durumu.	Gizli, Gizli Değil
Kullanım	Veri elemanının sıklıkla kullanıldığı yerler ve raporlar.	Hasta tablosu, Diyetisyen raporu
Kabul Edilebilirlik	Veri elemanının kabul edilebilirlik düzeyi.	Kabul Edilen, Kullanımdan Kaldırılan
Toplama Metotları	Verinin nasıl toplandığı.	Manuel giriş, Otomatik alım
Çokluk Durumu	Veri elemanının tekil veya çoklu olma durumu.	Tekil, Çoğul
Diğer Hususlar		

Tablo 2 Veri Sözlüğü

Nitelik	**Açıklama**	**Örnek**
Hasta ID	Hasta kimlik numarası.	1, 2, 3
Hasta Adı	Hastanın ad bilgisi.	Ayşe, Mehmet
Hasta Soyadı	Hastanın soyad bilgisi.	Yılmaz, Demir
Doğum Tarihi	Hastanın doğum tarihi.	1990-05-15, 1985-11-20
Cinsiyet	Hastanın cinsiyeti.	Kadın, Erkek
Adres	Hastanın ikamet ettiği adres bilgisi.	İstanbul, Ankara
Telefon	Hastanın telefon numarası.	555-123-4567, 555-987-6543
E-posta	Hastanın e-posta adresi.	ayse@example.com, mehmet@example.com
Kan Grubu	Hastanın kan grubu bilgisi.	A+, B-
Geçmiş Hastalıklar	Hastanın geçmişte yaşadığı hastalıklar.	Hipertansiyon, Diyabet
Kullandığı İlaçlar	Hastanın kullandığı ilaçlar.	Metformin, İnsülin
Randevu ID	Randevu kimlik numarası.	101, 102, 103
Randevu Tarihi	Hastanın randevu aldığı tarih.	2024-05-15 10:00, 2024-05-18 14:30
Diyetisyen ID	Randevuyu oluşturan diyetisyenin kimlik numarası.	201, 202, 203
Durum	Randevu durumu.	Onaylandı, İptal Edildi
Diyet Planı ID	Diyet planı kimlik numarası.	301, 302, 303
Açıklama	Kilo kontrolü, Glisem kontrolü	Kilo kontrolü, Glisem

		kontrolü
Yemek ID	Yemek kimlik numarası.	401, 402, 403
Yemek Adı	Yemeğin adı.	Sebzeli Tavuk, Peynirli Salata
Kalori	Yemeğin kalori değeri.	300, 150
Protein	Yemeğin kalori değeri.	20, 10
Karbonhidrat	Yemeğin karbonhidrat miktarı.	30, 15
Yağ	Yemeğin yağ miktarı.	10, 5
Özellikler	Yemeğin içeriğinde bulunan özel malzemeler.	Glutensiz, Laktozsuz
Diyet Takip ID	Diyet takip kimlik numarası.	501, 502, 503
Öneriler	Diyetisyen tarafından verilen öneriler.	Daha fazla su için, Egzersiz yapın
Takip Notları	Diyet takibi sırasında alınan notlar.	Günlük egzersiz yapıldı, Su içimi düzenlendi
Puanlama	Önerilere yapılan puanlama.	4, 5
Rapor ID	Rapor kimlik numarası.	601, 602, 603
İçerik	Rapor içeriği.	Haftalık ilerleme raporu, Kan test sonuçları
Dosya	Rapor dosyası (byte array).	[binary data]
Ayarlar ID	Ayarlar kimlik numarası.	701, 702, 703
Ayar Adı	Ayarın adı.	Dil, Tema
Değer	Ayarın değeri.	Türkçe, Koyu Tema
Kullanıcı ID	Kullanıcı kimlik numarası.	801, 802, 803
Kullanıcı Adı	Kullanıcının kullanıcı adı.	admin, user
Şifre	Kullanıcının şifresi.	******
Yetki	Kullanıcının yetkisi.	Admin, Standart Kullanıcı

3f Veri Akış Şemaları (VAD)



Şekil 3 VAD

3g Kalıcı Veri yönetimi (Persistent Data management)

- Tam (Full) Yedek: Seçilen bir veri kaynağının tamamının yedeklenmesi işlemidir. Bu yedekleme yöntemi, sistemin tüm verilerini yedeklemek için kullanılır ve genellikle artımlı veya fark yedeklemeleri öncesinde gerçekleştirilir.

- Artımlı (Incremental) Yedek: Bir önceki yedeğin ardından değişen veya yeni eklenen verilerin yedeklenmesi işlemidir. Yedeklerin geri yüklenmesi sırasında önce tam yedek, ardından artımlı yedekler yüklenir.

- Fark (Differential) Yedek: Bir önceki tam yedekten sonra değişen veya eklenen verilerin yedeklenmesi işlemidir. Geri yükleme sırasında önce tam yedek, sonra fark yedeği yüklenir.

- LTO (Linear Tape Open): 1990'larda geliştirilen manyetik bant veri depolama teknolojisi standardıdır.

- NAS (Network Attached Storage): Disklerden oluşan veri depolama araçlarının, dosya sunucusuna ihtiyaç kalmadan yerel ağa bağlanarak oluşturulan veri depolama sistemi.

- Zaman Damgası: Elektronik verilerin üretildiği, değiştirildiği veya kaydedildiği zamanın tespit edilmesi amacıyla kullanılan kayıttır.

Yedekleme Altyapısı

Diyetisyen Randevu Sistemi'nin yedekleme altyapısı aşağıdaki bileşenlerden oluşur:

- Yedekleme Yazılımı: Veeam 8 Backup and Replication Enterprise Edition

- Yedekleme Sunucusu: Dell Server R420135H7P1N, Intel E5-2420 1.90 GHz CPU, 8 GB RAM, Windows 2012 R2 işletim sistemi, HB Adapter for tape

- Disk Array Controller Sistemi: Qnap TS-1679u-RP, 3 TB x 16 disk, Toplam kullanılabilir kapasite 37.9 TB Asustor AS7012R, 4 TB x 12 disk, Toplam kullanılabilir kapasite 36.2 TB

- Disk ve Tape Yedek Medyaları: Tape Yedekleme Kütüphanesi: Dell PowerVault TL2000/TL4000 Tape Library, 2 raf, raf başına 12 kartuş, Mevcut kartuş sayısı 15, kartuş modeli LTO6 (Min 2.5 TB, Maks 6 TB veri)

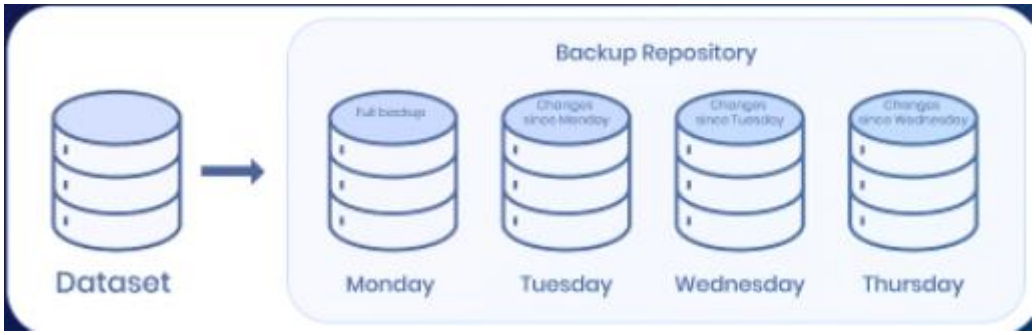
- Yerel Ağ (LAN) ve NAS: Yedekleme sunucusuna bağlı olan tape kütüphanesi NAS yapısına bağlı olan disk sistemleri

Yedekleme Şekli

Tam, artımlı ve fark yedekleme şekillerinin bir karışımı kullanılarak sistem yedeklenir. Bu yaklaşım, hem güvenilirliği artırır hem de yedekleme süresini ve depolama alanı ihtiyacını optimize eder. Özellikle, operasyonel hız gerektiren veriler disk tabanlı yedekleme çözümleriyle yedeklenirken, uzun süreli saklama ve büyük depolama kapasitesi gerektiren veriler manyetik tape tabanlı yedekleme çözümleriyle desteklenir.

Yedekleme işlemi tam yedekleme ile başlar ve ardından artımlı veya fark yedekleme yöntemleri kullanılarak günlük değişiklikler yedeklenir. Geri yükleme işlemi sırasında ise önce tam yedek yüklenir, ardından artımlı veya fark yedekler uygulanır.

Yedeklerin güvenliği ve yönetimi yedekleme yazılımı tarafından sağlanır ve gerektiğinde medya erişimi kayıtlarından yararlanılarak ilgili medyalara erişilir.



	Tam Yedekleme	Fark Yedekleme	Artımlı Yedekleme
Yedekleme Hızı	Çok düşük (D)	Düşük (D)	Çok yüksek (C)
Gereken Depolama Kapasitesi	Çok yüksek (D)	Yüksek (D)	Çok düşük (C)
Yedekten Geri Dönüş Hızı	Çok yüksek (C)	Düşük (D)	Çok düşük (D)

Yedekleme Sıklığı

Servis/Veri Grubu: Diyetisyen Randevu Veritabanı

- Tür: Tam Yedek
- Yedekleme Sıklığı: Haftalık
- Yedek Koruma Süresi: 6 Ay

Servis/Veri Grubu: Kullanıcı Profil Verileri

- Tür: Artımlı Yedek
- Yedekleme Sıklığı: Günlük
- Yedek Koruma Süresi: 30

Gün Servis/Veri Grubu: Randevu Geçmişi

- Tür: Fark Yedek
- Yedekleme Sıklığı: Günlük
- Yedek Koruma Süresi: 90 Gün

Servis/Veri Grubu: Ödeme Bilgileri

- Tür: Tam Yedek Yedekleme
- Sıklığı: Günlük
- Yedek Koruma Süresi: 1 Yıl

Servis/Veri Grubu: Sistem Ayarları

- Tür: Artımlı Yedek
- Yedekleme Sıklığı: Günlük
- Yedek Koruma Süresi: 30 Gün

Servis/Veri Grubu: İşlem Kayıtları

- Tür: Fark Yedek
- Yedekleme Sıklığı: Saatlik
- Yedek Koruma Süresi: 15 Gün

Yukarıdaki belirtilen yedekleme sıklıkları ve koruma süreleri, veri kaybı toleransı ve iş sürekliliği gereksinimleri dikkate alınarak belirlenmiştir. Bu plan, her veri grubu veya servis için uygun yedekleme stratejisini ve sıklığını sağlayarak veri merkezi operasyonlarının güvenilirliğini ve dayanıklılığını artırır.

Yedek Koruma/Saklama Süresi (Retention Period)

6 Aylık Saklama Süresi: Operasyonel verilerin kısa vadeli geri dönüş ihtiyaçlarını karşılamak için kullanılır. Genellikle operasyonel verilerin son 6 ayına ait yedekler bu süre boyunca saklanır ve ardından imha edilir.

12 Aylık Saklama Süresi: Orta vadeli yedek saklama için kullanılır. Bu kategoriye ait yedekler, operasyonel gereksinimler ve yasal düzenlemelere uygun olarak 6 ila 12 ay arasında saklanır ve sonra imha edilir.

24 Aylık Saklama Süresi: Uzun vadeli saklama ihtiyaçlarına yanıt vermek için kullanılır. Operasyonel ve yasal gereksinimler doğrultusunda 12 ila 24 ay arasında saklanır ve daha sonra imha edilir.

36 Aylık Saklama Süresi: Daha uzun vadeli yedek saklama için kullanılır. Operasyonel gereksinimler ve yasal düzenlemelere göre 24 ila 36 ay arasında saklanır ve ardından imha edilir.

60 Aylık Saklama Süresi: En uzun vadeli yedek saklama kategorisidir. Özellikle yasal düzenlemeler veya kurumsal politikalar gereği 36 ila 60 ay arasında saklanır ve daha sonra imha edilir.

Yedekleme İşleminin ve Yedeklerin Test Edilmesi

Yedekleme medyalarının güvenilirliğini ve erişilebilirliğini sağlamak için, Veri Merkezi'nde bulunan her servis ve veri grubu için düzenli olarak 6 ayda bir yedeklerden geri dönüş testleri yapılır. Bu testler, yedekleme işlemlerinin sağlıklı olarak gerçekleştiğini ve gerektiğinde yedeklerin başarılı bir şekilde geri yüklenebileceğini doğrulamak için yapılır. Yedeklerin başarılı olmaması durumunda, düzeltici önlemler alınır ve yedekler yeniden alınır.

Yedekleme Medyalarının Saklanması, Güvenliği ve İmhası

Yedekleme medyaları, fiziksel felaketlerden etkilenmeyecek güvenli bir uzak lokasyonda saklanır. Bu medyalar, güvenli ve kilitli kasalarda muhafaza edilir ve herhangi bir fiziksel hasar veya kötü niyetli müdahaleden korunurlar. Yedekleme medyalarının üzerindeki verilerin saklanması, güvenliği ve imhası, belirlenen sürelerde ve prosedürlere uygun olarak gerçekleştirilir.

Yedeklerden Geri Dönüş (Restoration & Recovery)

Veri kaybı durumunda veya geçmişe yönelik verilere ihtiyaç duyulduğunda, yedeklerden geri dönüş talebi BT Yardım Masası'na açılarak yapılır. Talep vakası, geri dönüş yapılacak verilerin detaylarını içerir ve ilgili ekipler tarafından işlenir. Bu süreçte, zaman damgası hizmeti kullanılarak verilerin güvenilirliği ve bütünlüğü sağlanır.

Kritik Başarı Unsurları

Veri Geri Dönüşü Sağlanabilirliği: Herhangi bir nedenle veri kaybı veya geçmişe yönelik verilere erişim ihtiyacı durumunda, Veri Merkezi'nde bulunan her servis ve veri grubu için belirlenen veri kaybı toleransı sınırları içindeki yedek verilerin operasyonel ortama geri dönüşünün sağlanması kritiktir.

Optimum Yedekleme Koşulları: Veri yedeklerinin, minimum yedekleme kapasitesi ihtiyacı, maksimum yedekleme hızı, maksimum geri dönüş kapasitesi ve maksimum geri dönüş hızı gibi optimum koşullar altında alınmasının sağlanması, başarılı bir yedekleme stratejisinin temelidir.

Yedek Veri Muhafazası: Yedek verilerin, en az yasal zorunluluklar ve kurumsal ihtiyaçlar çerçevesinde belirlenen süreler kadar sağlıklı bir şekilde muhafaza edilmesi, veri bütünlüğü ve erişilebilirliği açısından kritiktir.

Sağlık Kontrolleri ve Geri Dönüş Testleri: Yedek verilerin düzenli sağlık kontrollerinin yapılması ve yedeklerden geri dönüş testlerinin doğru ve eksiksiz olarak gerçekleştirilmesi, ihtiyaç duyulduğunda verilerin güvenilir bir şekilde geri dönüş yapılabilmesini sağlar.

Uzak Fiziki Lokasyonda Yedek Muhafazası: Doğal afetler, fiziksel hasarlar vb. gibi durumlarda Veri Merkezi'nde operasyonun durması durumunda, servislerin tekrar çalışır hale getirilmesinden sonra yedek verilerin geri dönüşünün sağlanabilmesi için yedeklerin uzak fiziksel lokasyonlarda muhafaza edilmesi önemlidir.

Fiziksel ve Bilgi Güvenliği: Yedek ve operasyonel verilerin fiziksel ve bilgi güvenliğinin sağlanması, yetkisiz erişim, veri bozulması veya manipülasyonu gibi risklerin önlenmesi açısından kritiktir.

İmha Edilen Verilerin Güvenliği: İmha edilen yedek verilerin bilgi güvenliğinin sağlanması, veri koruma politikalarına uygun olarak verilerin güvenli bir şekilde imha edilmesini ve yetkisiz erişime karşı korunmasını içerir.

Zaman Damgası Güvencesi: Yedek ve operasyonel verilerin, zaman damgası aracılığıyla kayıt altına alındığı tarihte orijinal haliyle var olduğunun ve sonradan değiştirilmediğinin garanti altına alınması, veri bütünlüğü ve doğruluğunun sağlanması açısından önemlidir.

Anahtar Başarı Göstergeleri

Tablo 3 Yedekleme Anahtar Başarı Göstergeleri Örneği

Anahtar Başarı Göstergesi	**Açıklama**	**Sorumlu**	**Metrik**	**Ölçme Yöntemi**
Yedekleme parametrelerinin belirlenmemesi	Tüm servis ve veri grupları için yedekleme parametrelerinin belirlenmiş olması gerekmektedir.	Bilgi & İletişim Tek. Md. – Sistem Yönetimi & Bilgi Güvenliği	Sıfır olmalı	Veri altyapısı ve sunucu kontrolleri
Yedeklenmeyen servis veya veri grubu	**Tüm servis ve veri gruplarının yedekleri alınmalıdır.**	Bilgi & İletişim Tek. Md. – Sistem Yönetimi	**Sıfır olmalı**	**Veri altyapısı ve sunucu kontrolleri**
Sağlık kontrolleri ve geri dönüş testlerinin doğru ve eksiksiz yapılmaması	**Yedek verilerin sağlık kontrolleri ve geri dönüş testleri her 6 ayda bir doğru ve eksiksiz olarak yapılmalıdır.**	Bilgi & İletişim Tek. Md. – Sistem Yönetimi & Bilgi Güvenliği	**Sıfır olmalı**	**Alınan yedeklerin kontrolü**
Uzak fiziki lokasyonda yedeklerin muhafaza edilmemesi	**Tüm kritik veriler uzak fiziki lokasyonda muhafaza edilmelidir.**	Bilgi & İletişim Tek. Md. – Sistem Yönetimi & Bilgi Güvenliği	**Sıfır olmalı**	**Veri altyapısı ve sunucu kontrolleri**
Uzak fiziki lokasyona veri aktarımı ve teslimine dair kayıtların eksik olması	**Tüm fiziksel veri transferleri kayıt altına alınmalı ve veri güvenliği sağlanmalıdır.**	Bilgi & İletişim Tek. Md. – Sistem Yönetimi & Bilgi Güvenliği	**Eksiksiz olmalı**	**Alınan yedeklerin kontrolü**

Yedek medyaların fiziksel ömrünü takip etmeye dair kayıtların eksik olması	**Tüm medya unsurlarının teknik ömrünü içeren bilgiler takip edilmelidir.**	Bilgi & İletişim Tek. Md. – Sistem Yönetimi & Bilgi Güvenliği	**Eksiksiz olmalı**	**Alınan yedeklerin kontrolü**
İmha edilen yedek verileri takip etmeye dair kayıtların eksik olması	**Kullanılan tüm operasyonel ve yedek veri medyaları, ömürleri tükendiğinde içindeki bilgiler tekrar elde edilemeyecek şekilde imha edilmelidir.**	Bilgi & İletişim Tek. Md. – Sistem Yönetimi & Bilgi Güvenliği	**Eksiksiz olmalı**	**İmha edilen medyaların kontrolü**
Geri dönüş talebi yapıldığında yedeklerin geri dönüş yapılamaması	**Yedek saklama süresi dahilinde bulunan tüm yedeklerde, istendiği anda geri dönüş yapılabilir olmalıdır.**	Bilgi & İletişim Tek. Md. – Sistem Yönetimi & Bilgi Güvenliği	**Sıfır olmalı**	**Geri dönüşü yapılan veri kontrolleri**
Zaman damgası aracılığıyla orijinalliği garanti altına alınmayan yedekler	**Zaman damgası hizmetinin süresinin dolup dolmadığı takip edilmeli ve zaman damgası sertifikasının geçersiz duruma düşmesi engellenmelidir.**	Bilgi & İletişim Tek. Md. – Sistem Yönetimi & Bilgi Güvenliği	**Sıfır olmalı**	**Alınan yedeklerin kontrolü**

Yetki/Sorumluluk Matrisi

Tablo 4 Yedekleme Yetki/Sorumluluk Matrisi

	B & IT Md. – Sistem Yönetimi	**B & IT Md. – Bilgi Güvenliği Yönetimi**	**B & IT Md.**	**Son Kullanıcı**
Yedekleme sıklığı, yedek koruma/saklama süresi ve hangi tür yedekleme kullanılacağını belirlemek	YG	YG	M	D
Yedekleri eksiksiz olarak almak	YG	B	M	-
Yedeklerin sağlık kontrolleri ve geri dönüş testlerini doğru ve eksiksiz olarak yapmak	YG	YG	M	-
Doğal afet, fiziksel hasar, vb. gibi sebeplerle yedekleri uzak fiziki lokasyona taşımak	YG	YG	M	-
Uzak fiziki lokasyondaki yedek medyayı teslim almak, saklanmasını ve güvenliğini sağlamak	M	M	B	-

Üzerinde yedek verilerin saklandığı medyaların fiziksel ömrünü takip etmek, imha edilecek medyaları ve yeni satın alma ihtiyaçlarını belirlemek	YG	YG	M	-
Gereken medyaları imha etmek	YG	YG	M	-
Yedeklerden geri dönüş ihtiyacı ortaya çıktığında doğru ve eksiksiz bilgi vererek, BT Yardım Masası aracılığıyla geri dönüş talep vakası açmak	B	-	B	YG
Yedeklerden geri dönüş talebi yapıldığında, geri dönüş işlemini gerçekleştirmek	YG	-	M	B
Zaman damgası sertifikasını takip etmek, gerektiğinde yenilemek	YG	YG	M	-

(YG: Yerine Getiren, M: Mesul, D: Danışılacak, B: Bilgi Verilecek)

3h Erişim kontrolü ve güvenlik (Access control and security)

Diyetisyen Randevu Sistemi Yazılımı için Kategorilere Göre Kontrol Adımları:

Veri Koruması:

- Uygulama tarafından kaydedilen veya kullanılan her türlü bilgiye yetkisiz erişimin engellenmesi sağlanmalıdır. Bu amaçla şu adımlar kontrol edilmelidir:
- Web, uygulama ve veritabanı sunucularının kritik bilgileri (sunucu adı, sürüm vb.) gizlenmelidir.
- Kullanılan yazılımların güvenlik yamaları düzenli olarak kontrol edilmeli ve en üst seviyede olmalıdır.
- Kullanılan uygulama çatılarının (örneğin, ASP.NET, PHP, STRUTS) güvenlik özellikleri etkinleştirilmelidir.
- Parolalar ve diğer hassas veriler açık metin olarak saklanmamalıdır.
- Geliştirme ortamından prodüksiyon ortamına aktarılırken gereksiz dosyalar silinmeli ve kaynak kod aktarımında gereksiz bilgiler kaldırılmalıdır.

Hata Yönetimi ve Kayıt Tutma:

- Uygulama hataları ve teknik detaylar son kullanıcıya gösterilmemelidir.
- Kritik işlemler uygulama ve sunucu seviyesinde kayıt altına alınmalıdır.

Erişim Kontrolü:

- Uygulamaların sunucuları, servis verdikleri dizinlerin içeriğini listelememelidir.

- Arama motorları tarafından görüntülenmemesi gereken dizinler için robots.txt kullanılmalıdır, ancak güvenlik riski oluşturabilecek dizinler bu dosyaya eklenmemelidir.
- HTTP metotlarına sadece gerektiğinde izin verilmelidir.
- Gereksiz dosyalara erişim engellenmeli ve gereksiz uygulamalar kaldırılmalıdır.
- Flash ve SilverLight uygulamalarında yapılandırma dosyalarının güvenliği kontrol edilmelidir.
- CSRF saldırılarına karşı önlemler alınmalıdır.
- Üçüncü şahısların bilgilerine yetkisiz erişim engellenmelidir.
- Sistem kullanıcısının yetkileri sınırlandırılmalıdır.
- Veritabanı kullanıcısının erişim hakları kontrol edilmelidir.
- Veritabanına sadece belirli IP adreslerinden bağlantı izni verilmelidir. Race-Condition'lara karşı senkronizasyon sağlanmalıdır.
- Web tabanlı istatistik uygulamalarına erişim kısıtlanmalıdır.
- Kısıtlı erişim gerektiren tüm kaynaklara erişim denetlenmelidir.
- Artık gerekmeyen yetkiler iptal edilmelidir.
- Kritik dizinlerin kolay tahmin edilebilir isimlere sahip olmamasına dikkat edilmelidir.

Kimlik Sınama:

- Öntanımlı kullanıcı hesapları kaldırılmalıdır.
- Sunucu tarafında kimlik doğrulaması yapılmalıdır.
- Kullanıcı adı ve parola doğrulaması tek tip hata mesajı ile gerçekleştirilmelidir.
- Tüm giriş ve erişim denemeleri kayıt altına alınmalıdır.
- Parola güncelleme işlemleri güvenlik önlemleriyle desteklenmelidir.
- Parola unuttum işlemleri güvenlik standartlarına uygun olmalıdır.
- Parola unuttum işlemleri sonrası kullanıcı bilgileri güvende olmalıdır.
- Kaba kuvvet saldırılarına karşı önlemler alınmalıdır.
- Web servislerinde kimlik doğrulaması yapılmalıdır.

Oturum Yönetimi:

- Hassas bilgilerin tarayıcılarda saklanmaması sağlanmalıdır.
- Oturum yönetimi için güçlü rastgele değerler kullanılmalıdır.

- Oturum bilgisi zaman aşımına uğratılmalıdır.
- Oturum bilgisi her girişte yeniden oluşturulmalıdır.
- Çerezlerin sınırları belirlenmelidir.
- Çerezler için güvenlik parametreleri tanımlanmalıdır.
- Başarılı girişler yönlendirme ile gerçekleştirilmelidir.
- Çıkış işlemi kullanıcıya sağlanmalıdır.
- Hassas bilgilerin arama motorları tarafından indekslenmemesi sağlanmalıdır.

Kriptoloji:

- Güçlü şifreleme algoritmaları kullanılmalıdır.
- Sunucu servis dışı bırakma ve MITM saldırılarına karşı önlemler alınmalıdır.
- Zayıf parolaların kullanımı engellenmelidir.
- Parolaların hash değerleri tuz ile oluşturulmalıdır.
- Başlangıç parolaları kullanıcılar tarafından değiştirilmelidir.

İletişim Güvenliği:

- Hassas veriler HTTPS üzerinden iletilmelidir.

Girdi Denetimi:

- Kullanıcı girdileri sunucu tarafında kontrol edilmelidir.
- Aritmetik işlemlerde sınırlar belirlenmelidir.
- Dosya yükleme işlemleri kontrol edilmelidir.
- Yönlendirme işlemlerinde pozitif girdi denetimi yapılmalıdır.
- HTTP cevap başlıklarında hassas karakterler kullanılmamalıdır.
- Sızma testleri düzenli olarak yapılmalıdır.
- SQL arama saldırılarına karşı önlemler alınmalıdır.

Çıktı Kodlama:

- Kullanıcı girdileri kontrol edilmeli ve düzgünleştirme işlemine tabi tutulmalıdır.
- SQL enjeksiyonuna karşı önlemler alınmalıdır.
- XSS saldırılarına karşı önlemler alınmalıdır.

- Dosya erişim işlemlerinde normalizasyon yapılmalıdır.
- LDAP ve XPath sorgularında güvenlik önlemleri alınmalıdır.
- Çerçeve engelleyici önlemler alınmalıdır.
- Web servislerinde güvenlik kontrolleri yapılmalıdır.

3i Küresel yazılım kontrolü (Global Software Control)

Diyetisyen Randevu Sitemi için küresel yazılım geliştirme pratikleri, çevik yazılım geliştirme ilkeleriyle birleşerek çeşitli zorluklarla karşılaşır. Özellikle, farklı coğrafi konumlardaki ekipler arasındaki iletişim ve iş birliği, geleneksel çevik yöntemlere meydan okuyabilir. Örneğin, "Scrum" çerçevesi, yüksek düzeyde etkileşim ve yüz yüze iletişim gerektirirken, küresel ekipler arasında bu tip etkileşim zor olabilir.

Geleneksel "Scrum" rolleri, birlikte yaşayan ve çalışan ekipler için tasarlanmıştır. Ancak, küresel yazılım geliştirme projelerinde, bazı ekip üyeleri farklı ülkelerde bulunabilir, bu da iletişimi zorlaştırabilir. Bununla birlikte, "Scrum" toplantıları olan "Daily Scrum", "Sprint Review" ve "Sprint Retrospective" gibi etkinlikler, yüz yüze gerçekleşmelidir.

Bu bağlamda, küresel yazılım geliştirme ortamlarında "Scrum" uygulamalarını uyarlamak gereklidir. Ancak, literatürde bu uyarlamanın nasıl yapılacağına dair genel bir yöntem bulunmamaktadır. Bu nedenle, bu zorluklarla karşılaşan ekipler, kendi gereksinimlerine ve deneyimlerine dayanarak bu pratikleri uyarlamak zorundadır.

Bu bağlamda, küresel yazılım yönetimi stratejileri, farklı coğrafi konumlardaki ekipler arasında etkili iletişim ve iş birliği sağlamak için özel olarak tasarlanmalıdır. Bu stratejiler, proje yönetimi süreçlerini, iletişim kanallarını ve toplantı formatlarını içerebilir, böylece farklı bölgelerdeki ekipler arasında verimli bir çalışma ortamı oluşturulabilir.

3j Sınır şartları (Boundary Conditions)

API İstekleri Toplu İşlenmeli: Harici sistemlere yönelik API istekleri, .NET Core'un asenkron programlama özellikleri kullanılarak toplu işlenmelidir. Bu, sistem performansını artırır ve kaynakları daha etkin kullanmamızı sağlar.

Veri Saklama ACID Uyumlu Olmalıdır: Kullanıcı verilerinin güvenli ve tutarlı bir şekilde saklanması için SQL veritabanı ACID uyumluluğunu sağlamalıdır. .NET Core'un ORM (Object-Relational Mapping) yetenekleri, veritabanı işlemlerini kolaylaştırarak bu gereksinimi karşılar.

API Etkileşimlerinin Denetimi: Tüm API etkileşimleri, güvenlik ve yetkilendirme kontrolleri ile denetlenmelidir. .NET Core'un sağladığı güvenlik middleware'leri ve SQL veritabanı tablolarındaki yetki kontrolleri bu denetimlerin sağlanmasına yardımcı olur.

Çoklu Kullanım Desteği: Sistem, .NET Core'un çoklu iş parçacığı desteği sayesinde birden fazla kullanıcının eşzamanlı erişimini desteklemelidir. Bu, sistem performansını artırır ve kullanıcı deneyimini iyileştirir.

Yetkilendirme ve Veri Erişimi Kontrolü: Kullanıcılar, rollerine bağlı olarak belirli verilere erişebilmelidir. .NET Core'un kimlik doğrulama ve yetkilendirme yetenekleri, kullanıcıların doğru verilere erişimini sağlamak için kullanılabilir.

Veri Depolama Seçimi: Veri depolama için SQL veritabanı tercih edilmelidir. Bu, ilişkisel verilerin güvenli ve yapılandırılmış bir şekilde saklanmasını sağlar ve .NET Core ile kolayca entegre edilebilir.

Kodlama Dili ve Test Kapsamı: Sistem, .NET Core gibi güçlü ve esnek diller kullanılarak kodlanmalıdır. Ayrıca, kodun en az %90 birim test kapsamına sahip olması sağlanmalıdır. Bu birim testler, sistem işlevselliğinin doğruluğunu ve güvenilirliğini sağlar.

Bu sınır şartları, Diyetsiyen Randevu Sistemi'nin güvenli, performanslı ve ölçeklenebilir bir şekilde geliştirilmesini sağlamak için önemlidir.

4 Alt Sistem Hizmetleri (Subsystem services)

Onion mimarisi, birçok katmana sahip bir mimari tasarım yaklaşımıdır. Her katman, sistemde belirli bir işlevi gerçekleştirir ve diğer katmanlarla sıkı bir şekilde entegre olur. Genellikle bu katmanlar iç içe geçmiş bir yapıya sahiptir ve veri iletişimi, güvenlik ve iş mantığı gibi farklı sorumluluk alanlarına sahiptir. Onion mimarisi, özellikle güvenlik ve ölçeklenebilirlik gibi önemli konularda avantajlar sağlar.Projemizin gereksinimlerine uygun olarak Onion mimarisini seçmenizdeki nedenler:

1. **Gizlilik ve Güvenlik**: Projemizde, kullanıcıların kişisel ve sağlık verilerini korumak çok önemlidir. Onion mimarisi, verileri farklı katmanlarda şifreleyerek ve şifreleme teknikleri kullanarak güvenli bir şekilde saklamamıza ve iletmemize olanak tanır. Bu, kullanıcıların gizliliğini ve güvenliğini sağlamak için önemli bir özelliktir.

2. **Ölçeklenebilirlik**: Diyetisten Klinik Randevu ve Yönetim Programı, potansiyel olarak büyük miktarda kullanıcıyı desteklemelidir. Onion mimarisi, bileşenleri kolayca ölçeklendirebilmemizi sağlar. Bu, sistemimizin büyüdükçe performansını korumasına ve daha fazla kullanıcıyı desteklemesine olanak tanır.

3. **Modülerlik ve Esneklik**: Projemizde, farklı işlevlere sahip bileşenlerin bir araya getirilmesi gerekebilir. Onion mimarisi, modüler bir yapı sunar, bu da farklı bileşenleri birbirinden bağımsız olarak geliştirebilmemize ve entegre edebilmemize olanak tanır. Bu, sistemi daha esnek hale getirir ve değişen gereksinimlere daha iyi uyum sağlar.

4. **Dağıtık Sistemler İçin Uygunluk**: Projemiz, kullanıcıların farklı cihazlardan ve farklı yerlerden erişim sağlamasını gerektirebilir. Onion mimarisi, dağıtık sistemler için uygun bir yapı sunar. Bu, kullanıcıların istedikleri zaman ve yerde sistemimize erişebilmelerini sağlar.

5. **Dayanıklılık ve Yüksek Erişilebilirlik**: Projemiz, kesintisiz bir hizmet sunmalıdır. Onion mimarisi, farklı katmanlarda yedekleme ve yedekleme mekanizmaları kullanarak sistemdeki bir arızadan etkilenme riskini azaltır. Bu, sistemimizin dayanıklılığını artırır ve yüksek erişilebilirlik sağlar.

Onion mimarisinin güvenlik açısından sağladığı artılar:

1. **Katmanlı Güvenlik**: Onion mimarisi, farklı katmanlardaki güvenlik önlemleri sayesinde sistem genelinde katmanlı bir güvenlik yaklaşımı sağlar. Bu, saldırganların sisteme sızma girişimlerini engellemek için birden fazla savunma katmanı oluşturur.

2. **Veri Gizliliği**: Onion mimarisi, verilerinizi farklı katmanlarda şifreler ve şifreleme teknikleri kullanarak korur. Bu, kullanıcıların kişisel bilgilerinin ve sağlık verilerinin gizliliğini sağlar.

3. **Saldiri Azaltma**: Onion mimarisi, dış dünyadan gelen saldırıların etkisini azaltır. Çünkü saldırganların her bir katmana sızmaları ve içerideki bilgilere erişmeleri daha zordur.

4. **Güvenli İletişim**: Onion mimarisi, kullanıcıların ve sistem bileşenlerinin güvenli bir şekilde iletişim kurmasını sağlar. Bu, veri alışverişi sırasında ortaya çıkabilecek güvenlik açıklarını azaltır.

5. **Doğrulama ve Yetkilendirme**: Onion mimarisi, kullanıcıların kimliklerini doğrulama ve yetkilendirme süreçlerini güçlendirir. Bu, yetkisiz erişimleri engeller ve veriye sadece yetkili kişilerin erişmesini sağlar.

6. **Distributed Denial of Service (DDoS) Koruması**: Onion mimarisi, dağıtılmış hizmet reddi saldırılarına karşı dirençli olabilir. Farklı katmanlardaki sunucular, ağ trafiğini etkili bir şekilde yöneterek sistemi DDoS saldırılarına karşı korur.

7. **İzolasyon ve Bölgeleme**: Onion mimarisi, farklı bileşenleri ve hizmetleri izole ederek, bir bileşenin güvenlik açığından etkilenen diğer bileşenleri sınırlar. Bu, sistemin bütünlüğünü ve güvenliğini korur.

8. **Güvenlik Denetimleri ve İzleme**: Onion mimarisi, güvenlik denetimleri ve izleme mekanizmalarını kolaylaştırır. Her bir katmandaki olayları izleyerek ve günlükleri analiz ederek, sistemin güvenlik durumunu sürekli olarak değerlendirebilir ve geliştirebilirsiniz.

Alt sistemler, Diyetisten Klinik Randevu ve Yönetim Programı'nın güvenliği, veri bütünlüğü, kullanılabilirliği ve işlevselliği için kritik öneme sahiptir. Her bir alt sistem, projenin başarılı bir şekilde işleyebilmesi için dikkate alınmalı ve uygun şekilde uygulanmalıdır.

Projemizle ilgili olarak bu alt sistemleri aşağıdaki şekillerde değerlendirebiliriz:

1. **Yetkilendirme Alt Sistemi**:

    - Kullanıcıların sisteme giriş yapması ve belirli işlevlere erişebilmesi için yetkilendirme altsistemi kullanılabilir. Örneğin, yöneticilerin tam erişim yetkisine sahip olması ve diyetisyenlerin sadece randevu takvimlerini düzenleyebilmesi gibi.

    - Ekran bazında yetkilendirme, kullanıcıların sadece belirli ekranlara erişebilmesini sağlar. Örneğin, hasta bilgilerine erişim yetkisi sadece diyetisyenlere verilebilir.

    - İşlev bazında yetkilendirme, kullanıcıların yalnızca belirli işlevleri gerçekleştirebilmesini sağlar. Örneğin, sadece yöneticilerin kullanıcı hesaplarını oluşturabilmesi gibi.

2. **Güvenlik Alt Sistemi**:

    - Kullanıcı işlemlerinin günlüğünü tutmak ve izlemek için güvenlik altsistemi kullanılabilir. Bu, sistemin güvenliğini artırır ve olası ihlalleri tespit etmeye yardımcı olur.

3. **Yedekleme Alt Sistemi**:

- Veri kaybını önlemek için düzenli olarak yedekleme altsistemi kullanılabilir. Bu, kullanıcı verilerinin güvenliğini sağlar ve sistemde oluşabilecek herhangi bir kesinti durumunda veri kaybını en aza indirir.

4. **Veri Transferi Alt Sistemi**:

   - Klinikler ve diyetisyenler arasında veri akışını sağlamak için veri transferi altsistemi kullanılabilir. Bu, randevu bilgilerinin ve hasta verilerinin güvenli bir şekilde paylaşılmasını sağlar.

5. **Arşiv Alt Sistemi**:

   - Kullanıcıların eski randevu kayıtlarına ve geçmiş sağlık verilerine erişebilmeleri için arşiv altsistemi kullanılabilir. Bu, geçmiş bilgilerin saklanmasını ve gerektiğinde erişilebilir olmasını sağlar.

6. **Dönüştürme Alt Sistemi**:

   - Var olan hasta bilgilerinin yeni sisteme aktarılması için dönüştürme altsistemi kullanılabilir. Bu, mevcut verilerin sisteme entegrasyonunu ve sorunsuz geçişini sağlar.

5 Kullanıcı Arayüz Tasarımları(GUI)

Diyetisten Klinik Randevu ve Yönetim Programı, diyetisyenlerin klinik işlemlerini etkili bir şekilde yönetmelerini ve randevu planlamasını kolaylaştırmayı amaçlamaktadır. Programın kullanıcı arayüzü, diyetisyenlerin iş akışını optimize etmek ve hasta deneyimini geliştirmek için kritik bir rol oynamaktadır.

Programın kullanıcı arayüzü, randevu planlama, hasta kaydı, raporlama ve iletişim gibi temel işlevleri içerecektir. Her bir işlevin ayrıntıları aşağıda belirtilmiştir:

Kullanıcı Arayüzü İşlevleri:

1.Randevu Planlama:

Kullanıcılar için kolayca erişilebilir bir randevu oluşturma ve düzenleme formu sağlanacak.

Formlar, ASP.NET Core Razor Pages veya MVC kullanılarak oluşturulacak.

Tarih ve saat seçme aracı, jQuery veya Vue.js gibi JavaScript kütüphaneleri kullanılarak entegre edilecek.

Randevu onayı ve iptali için kullanıcıya kolaylık sağlayan butonlar ve bilgilendirme mesajları eklenecek.

2.Hasta Kaydı:

Yeni hastalar için basit ve anlaşılır bir kayıt formu oluşturulacak.

Form doğrulaması ve veri geçerliliği, ASP.NET Core tarafında sağlanacak.

Hasta bilgileri, Entity Framework Core kullanılarak SQL Server veya PostgreSQL gibi veritabanlarına güvenli bir şekilde kaydedilecek ve güncellenebilecek.

3.Raporlama:

Kullanıcılar için özelleştirilebilir raporlama seçenekleri sunulacak.

Grafikler ve tablolar, Chart.js veya D3.js gibi JavaScript kütüphaneleri kullanılarak oluşturulacak.

Raporların PDF veya Excel formatında dışa aktarılması için iTextSharp veya EPPlus gibi .NET kütüphaneleri kullanılacak.

4.İletişim:

Kullanıcılar ve diyetisyenler arasında etkili iletişimi sağlamak için mesajlaşma özelliği entegre edilecek.

Mesajlaşma arayüzü, ASP.NET SignalR kullanılarak gerçek zamanlı iletişim sağlayacak.

Bildirimler, JavaScript veya ASP.NET Core tarafında yönetilerek kullanıcılara iletilere bildirim olarak gönderilecek.

Kullanıcı arayüzü tasarımında göz önünde bulundurulması gereken temel hususlar şunlardır:

Kullanıcı dostu bir tasarımın sağlanması için UI/UX tasarım prensiplerinin takip edilmesi.

Veri güvenliğinin sağlanması ve gizliliğin korunması için HTTPS kullanımı ve giriş bilgilerinin şifrelenmesi.

Arayüzdeki hata mesajlarının açık ve anlaşılır olması için i18n ve l10n tekniklerinin kullanılması.

Kullanıcıların kolayca erişebileceği bir navigasyon yapısının oluşturulması için menü ve breadcrumb gibi ögelerin kullanılması.

6 Nesne Tasarımı

6a Nesne Tasarımı Değişimleri

Nesne tabanlı programlama, yazılımın modüler ve esnek olmasını sağlayarak gelecekteki değişikliklere uyum sağlar. Projemiz için nesne tasarımı, programın ana yapı taşlarını oluşturur ve işlevselliğin sağlam bir temel üzerine inşa edilmesini sağlar. Bu tasarım yaklaşımı, programın klinik işlemlerini yönetmek, randevu planlamak, hasta bilgilerini saklamak ve raporlama işlevlerini içerir

Soyutlama (Abstraction):

Klinik işlemler ve randevu yönetimi için gerekli detayları soyutlayarak, sadece önemli bilgilerin ve işlevlerin ortaya çıkmasını sağlarız. Bu, programın gereksinimlerine odaklanarak gereksiz karmaşıklığı ortadan kaldırır.

Sarmalama / Paketleme (Encapsulation):

Klinik işlemlerin ve hasta bilgilerinin sarmalanması, dış dünyadan korunmasını sağlar ve veri bütünlüğünü korur. Bu, verinin sadece gerektiğinde erişilebilir olmasını sağlar ve veri manipülasyonunun kontrol altında tutulmasını sağlar

Açık Kapalı Prensibi (Open Closed Principle):

Programın modüllerinin genişlemeye açık ancak değişime kapalı olması, yeni özellikler eklenirken mevcut kodun değiştirilmesini engeller. Bu prensip, kodun kolayca genişletilebilir olmasını ve gelecekteki değişikliklere uyum sağlayabilmesini sağlar.

Fabrika Metodu (Factory Method):

Yeni randevuların ve hasta kayıtlarının oluşturulması için fabrika metodu kullanılabilir. Bu desen, nesnelerin oluşturulmasını merkezi bir noktaya taşır ve kod tekrarını azaltır. Ayrıca, farklı türde randevular veya hastalar için farklı fabrika metodları kullanılabilir, bu da programın genişlemesini kolaylaştırır.

Gözlemci Deseni (Observer Pattern):

Randevu ve hasta bilgilerindeki değişikliklerin diğer modüllere otomatik olarak bildirilmesi için gözlemci deseni kullanılabilir. Bu desen, veri değişikliklerini izlemeyi kolaylaştırır ve programın modülerliğini artırır. Örneğin, bir randevunun iptal edilmesi durumunda, ilgili modüller otomatik olarak güncellenebilir.

Filtreleme Deseni (Filter Pattern):

Hasta veya randevu listelerinin belirli kriterlere göre filtrelenmesi için filtreleme deseni kullanılabilir. Bu desen, kullanıcıların verileri hızlıca sorgulamasını sağlar ve kullanıcı deneyimini iyileştirir. Örneğin, kullanıcılar belirli bir tarih aralığındaki randevuları filtreleyebilir veya belirli bir hastanın tüm randevularını görüntüleyebilir.

Bu tasarım ve kalıplar, "Diyetisten Klinik Randevu ve Yönetim Programı" projesinin esnek ve modüler bir şekilde geliştirilmesini sağlayacak, aynı zamanda gelecekteki değişikliklere kolayca uyum sağlayabilecek bir yapı oluşturacaktır.

6a Arayüz Dokümantasyon Kılavuzları

Bu bölüm, "Diyetisten Klinik Randevu ve Yönetim Programı"nın kullanıcı arayüzlerini anlatan kılavuz şeklinde bir dokümantasyon sunar.

Programın kullanıcı arayüzlerinin etkili, verimli ve tatmin edici olması, kullanıcıların işlevleri eksiksiz ve doğru bir şekilde yerine getirebilmesini sağlar. Bu kılavuz, kullanıcıların programı etkin bir şekilde kullanmalarına yardımcı olacak şekilde tasarlanmıştır.

Kullanıcıların ihtiyaç duyduğu fonksiyonları eksiksiz ve doğru bir şekilde yerine getirmesi.

Arayüzlerin kolayca öğrenilebilmesi ve hatırlanabilmesi.

Kullanıcıların programın kullanımını hızlıca öğrenip, verimli bir şekilde kullanabilmesi.

Kullanıcı dostu kodun yazılabilmesi ve anlaşılabilirliği.

Dokümantasyon, kullanıcıların programı kolayca öğrenmelerini ve kullanmalarını destekleyen her türlü kaynağı içerir.

Arayüzler, kullanıcıların mümkün olduğunca dokümantasyona başvurmadan işlemlerini gerçekleştirebilmeleri için basit ve sezgisel olmalıdır.

Ayrıntılı dokümantasyon sağlanmalı ve sık gerçekleştirilen işlemler için adım adım kılavuzlar sunulmalıdır.

Açık kaynak kodlu arayüzler için, kodun içine girebilecek kullanıcılar için yorumlarla desteklenmiş kaynak kodları sağlanmalıdır.

Diyetisten Klinik Randevu ve Yönetim Programı Kullanım Kılavuzu

Bu dokümantasyon kılavuzları, kullanıcıların programı verimli bir şekilde kullanmalarını sağlayarak kullanıcı deneyimini iyileştirmeyi amaçlamaktadır.

6b Paketler

"Diyetisten Klinik Randevu ve Yönetim Programı" projenizde, .NET Core ve C# kullanarak geliştirilen bir yazılım olduğunu göz önünde bulundurarak, kullanmayı düşündüğünüz paketleme ve lisanslama yöntemlerini aşağıda ele alacağız.

Paketleme Stratejisi:

Projemizde kullanılacak olan paketler, .NET Core ekosistemi için özel olarak geliştirilmiş NuGet paketleri olacaktır. Bu paketler, projenin işlevselliğini artırmak ve geliştirme sürecini hızlandırmak için kullanılacaktır.

Özellikle, Entity Framework Core gibi ORM (Object-Relational Mapping) araçlarını, ASP.NET Core Identity gibi kimlik yönetimi araçlarını ve Swagger/OpenAPI gibi API belgelendirme araçlarını projemizde kullanmayı planlıyoruz. Bu paketler, veritabanı yönetimi, kullanıcı kimlik doğrulaması ve API belgelendirme gibi temel ihtiyaçları karşılamak için kullanılacaktır.

Lisanslama Stratejisi:

Projemizde kullanılacak yazılımın lisanslama stratejisi, MIT Lisansı gibi açık kaynak ve özgür lisans modellerine dayanacaktır. Bu lisans modeli, projenin kullanımını ve dağıtımını serbestleştirirken, aynı zamanda projenin geliştiricilere açık olmasını sağlayarak katkıda bulunmayı teşvik eder.

Projemizin tüm kaynak kodları ve belgeleri, GitHub gibi açık kaynak platformlarında barındırılacak ve geliştiricilere erişim sağlanacaktır. Bu şekilde, projeye katkıda bulunmak isteyen diğer geliştiricilerin katkılarını kolayca kabul edebileceğiz.

Bu paketleme ve lisanslama stratejileri, projenin gelişimini destekleyecek ve topluluk katılımını teşvik edecek şekilde tasarlanmıştır.

6c Arayüz (Interface) Sınıfları

Projemizin teknik altyapısını sağlamlaştırmak ve dış dünya ile etkileşimini optimize etmek için arayüz (interface) ve API stratejimizi belirleyeceğiz. Bu kısımda, arayüzlerin ve API'lerin detaylı tasarımını, kullanılacak teknolojileri ve belgelendirme sürecini ele alacağız.

Arayüz (Interface) ve API Tasarımı:

Arayüz (Interface) Tanımlama ve Kullanımı: Projemizin farklı katmanları arasındaki iletişimi sağlamak için arayüzlerden yararlanacağız. Arayüzler, projenin bileşenlerini birbirinden bağımsız hale getirecek ve gelecekteki değişikliklere uyum sağlayacak şekilde tasarlanmalıdır.

Arayüzler, projenin katmanları arasında standart bir iletişim arayüzü sağlayacak şekilde tanımlanacak. Örneğin, kullanıcı arayüzü katmanı ile iş mantığı katmanı arasındaki etkileşimi kolaylaştırmak için arayüzler tanımlanabilir.

Her arayüz, projenin ilgili bileşeninin işlevselliğini belirleyecek metodları içerecek ve bu metodların nasıl kullanılacağını açıklayacak. Örneğin, bir kullanıcı yönetim sistemi arayüzü, kullanıcı oluşturma, güncelleme, silme gibi temel işlevleri tanımlayabilir.

API Tasarımı ve Uygulaması: Projemizin dış dünya ile etkileşimini sağlamak için RESTful API'ler kullanacağız. API'ler, projenin dış dünya ile iletişimini standart HTTP protokolü üzerinden sağlayacak ve farklı platformlar arasında veri alışverişi yapmayı mümkün kılacak.

API rotaları, HTTP metotları (GET, POST, PUT, DELETE) kullanılarak tanımlanacak ve her rotanın parametreleri ve dönüş değerleri belirlenecek. Örneğin, bir kullanıcı yönetim sistemi API'si, kullanıcı bilgilerini getirme, güncelleme, silme gibi rotaları içerebilir.

API'lerin belgelendirilmesi için OpenAPI veya Swagger gibi araçlar kullanılacak ve API spesifikasyonları otomatik olarak oluşturulacak. Bu belgelendirme süreci, API kullanıcılarının API'leri nasıl kullanacaklarını anlamalarını ve hata ayıklama süreçlerini kolaylaştıracaktır.

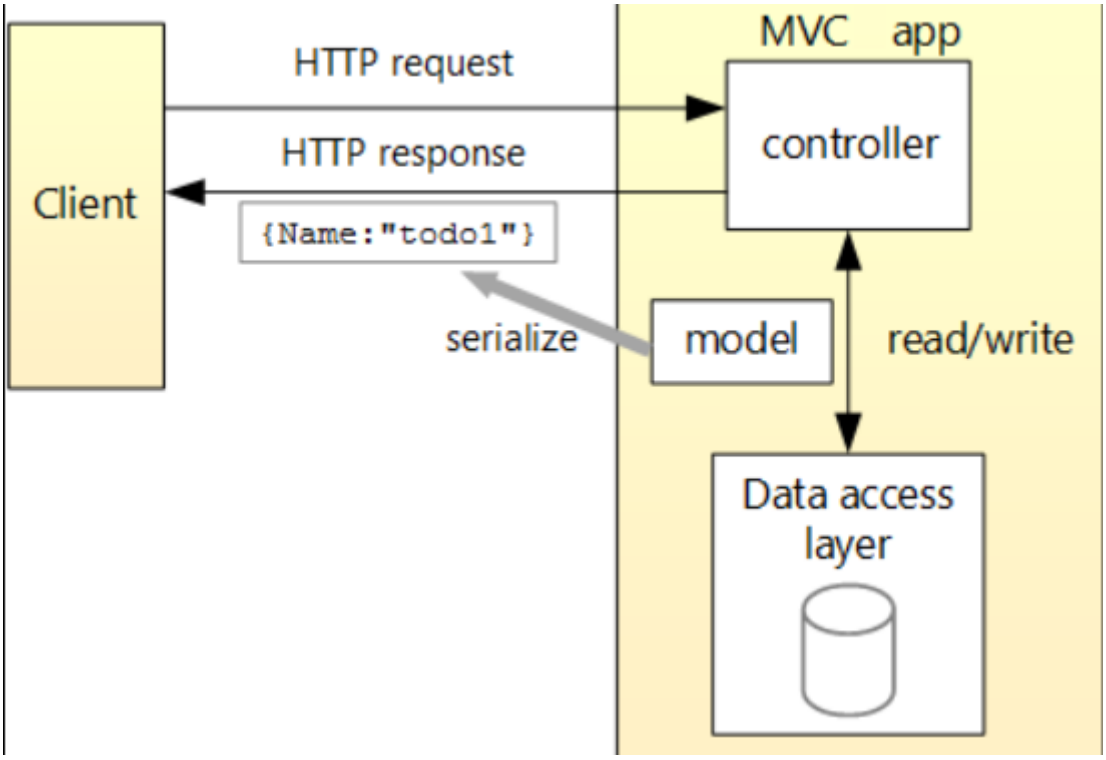
Arayüz ve API Özellikleri:

Metod ve Parametre Tanımları: Her arayüz ve API rotası için kullanılacak olan metodlar ve parametreler net bir şekilde tanımlanacak. Özellikle, API rotalarının hangi parametreleri kabul edeceği, hangi dönüş değerlerini sağlayacağı ve hata durumlarında nasıl davranacağı detaylı bir şekilde belirlenecek. Örneğin, bir kullanıcı oluşturma API rotası, kullanıcı adı, e-posta, şifre gibi parametreleri alabilir ve başarılı bir oluşturma durumunda 201 HTTP durum kodunu döndürebilir.

Versiyonlama Stratejisi: Projemizin geliştirme sürecinde değişiklikler yapılabilir ve bu değişiklikler API'lerin geriye dönük uyumluluğunu etkileyebilir. Bu nedenle, API rotalarının versiyonlama stratejisi belirlenecek ve API değişikliklerinin mevcut kullanıcıları etkilememesi sağlanacak. Örneğin, API rotalarına bir versiyon numarası eklenerek, kullanıcılar yeni bir versiyona geçiş yapabilirler.

Güvenlik Stratejisi: API'lerin güvenliği önemlidir. Projemizde kullanılacak olan API'lerin yetkilendirme, kimlik doğrulama ve veri güvenliği gibi konularda nasıl korunacağı belirlenecek. Örneğin, API rotaları için JWT tabanlı bir kimlik doğrulama mekanizması kullanılabilir ve API rotaları yalnızca yetkilendirilmiş kullanıcılar tarafından erişilebilir olabilir.

Arayüz ve API stratejisinin belirlenmesindeki temel motivasyon, projenin teknik altyapısını sağlamlaştırmak, dış dünya ile etkileşimini optimize etmek ve gelecekteki değişikliklere adapte olmasını sağlamaktır. Doğru bir arayüz ve API tasarımı, projenin geliştirme sürecini yönetilebilir ve optimize edilebilir hale getirecek ve uzun vadede projenin başarısını artıracaktır.



Şekil 11 Web API