



softITo 3. DÖNEM 3. GRUP
HASTANE YÖNETİM SİSTEMİ
BİTİRME PROJESİ RAPORU

ASP.NET Core Web API & MVC ile Çok Katmanlı Hastane Otomasyon Sistemi

Yusuf Çelikkaya

Temmuz, 2026
softITo Yazılım Bilişim Akademisi, İstanbul

İÇİNDEKİLER

1. GİRİŞ.....	3
1.1. Projenin Amacı	3
1.2. Proje Kapsamı ve Kısıtları.....	3
2. MATERYAL VE YÖNTEM	4
2.1. Kullanılan Teknolojiler	4
2.2. Sistem Mimarisi (Çok Katmanlı Yapı).....	4
2.3. Güvenlik ve Yetkilendirme.....	4
2.4. Performans Optimizasyonu	4
2.5. Uçtan Uca Haberleşme ve Dokümantasyon	4
3. SİSTEM MODÜLLERİ.....	6
3.1. Hasta ve Doktor Yönetimi	6
3.2. Randevu ve Muayene Süreci	6
3.3. Yatan Hasta ve Oda Yönetimi	6
3.4. Faturalandırma ve Ödeme	6
3.5. Kullanıcı ve Yetki Yönetimi.....	6
4. KURULUM	7
4.1. Sistem Gereksinimleri	7
4.2. Projenin Klonlanması	7
4.3. Veritabanı Bağlantısının Ayarlanması	7
4.4. Bağımlılıkların Yüklenmesi ve Derleme	7
4.5. Projenin Çalıştırılması.....	7
4.6. Proje Dizin Yapısı	7
5. EKRAN GÖRÜNTÜLERİ	8
5.1. MVC Ön Yüz (Kullanıcı Arayüzü).....	8
5.2. API / Swagger ve Dapper Entegrasyonu	21
6. SONUÇ.....	23
7. KAYNAKÇA.....	24

1. GİRİŞ

1.1. Projenin Amacı

Hastane Yönetim Sistemi, hastane operasyonlarını -hasta kaydı, doktor ataması, randevu planlaması, muayene ve reçete süreçleri, yatan hasta ve oda yönetimi ile faturalandırma işlemleri- tek bir dijital platform üzerinden yönetmek amacıyla geliştirilmiş, uçtan uca bir otomasyon çözümüdür.

Projenin temel amacı; sağlık kurumlarının hasta ve doktor bilgilerini merkezi bir sistemden yönetebilmesini, JWT tabanlı kimlik doğrulama ile güvenli erişim sağlamasını ve yüksek performanslı bir altyapı üzerinden kesintisiz hizmet vermesini mümkün kılmaktır. Sistem, arka planda RESTful bir Web API koşturarak, ön yüzde ise ASP.NET Core MVC ile kullanıcı dostu bir arayüz sunmaktadır.

Proje, kurumsal mimari standartlarına ve temiz kod (Clean Code) prensiplerine uygun şekilde geliştirilmiş; kritik sorgularda performansı artırmak amacıyla Stored Procedure ve Dapper (Micro-ORM) gibi araçlarla desteklenmiştir.

1.2. Proje Kapsamı ve Kısıtları

Sistem; hasta ve doktor yönetimi, poliklinik ve randevu işlemleri, muayene/reçete kayıtları, yatan hasta ve oda takibi, faturalandırma ile kullanıcı/yetki yönetimi modüllerini kapsamaktadır. Proje bir eğitim/bitirme çalışması niteliğinde olduğundan; ödeme altyapısı simüle edilmiş olup gerçek bir banka/ödeme kuruluşu entegrasyonu içermemektedir. Ayrıca sistem, tek bir hastane/kurum senaryosu üzerinden tasarlanmış olup çoklu kurum (multi-tenant) desteği kapsam dışında bırakılmıştır.

2. MATERYAL VE YÖNTEM

2.1. Kullanılan Teknolojiler

Projenin geliştirilmesinde sektör standartlarına uygun, güncel ve performans odaklı teknolojiler tercih edilmiştir:

Kategori	Teknoloji / Araç	Detay
Backend	C# 12, .NET 8.0	Güçlü, güvenli ve performanslı sunucu tarafı.
API & Arayüz	ASP.NET Core Web API, MVC	RESTful servisler ve kullanıcı dostu arayüz.
Veri Yönetimi	MS SQL Server, EF Core, Dapper	İlişkisel veritabanı ve ORM yaklaşımları.
Güvenlik	JWT, Azure SDK	Modern kimlik doğrulama standartları.
Araçlar	Swagger, HttpClient, Git	API dokümantasyonu ve versiyonlama.

2.2. Sistem Mimarisi (Çok Katmanlı Yapı)

Proje, anlaşılabilirlik ve sürdürülebilirlik açısından Çok Katmanlı (N-Tier) mimari standartlarına uygun olarak üç ana katmandan oluşacak şekilde tasarlanmıştır:

- HospitalAppApi: RESTful servislerin barındırıldığı, iş kurallarının ve veri doğrulamalarının yürütüldüğü backend katmanı.
- HospitalAppMvc: Web API ile HttpClient aracılığıyla haberleşen, son kullanıcıya sunulan arayüz (ASP.NET Core MVC) katmanı.
- HospitalAppDppr: Dapper (Micro-ORM) kullanılarak kritik ve performans gerektiren veri erişim işlemlerinin yürütüldüğü katman.

Bu ayırım sayesinde sunum, iş mantığı ve veri erişim sorumlulukları birbirinden bağımsız hale getirilmiş; katmanlar arası bağımlılık azaltılarak test edilebilirlik ve bakım kolaylığı sağlanmıştır.

2.3. Güvenlik ve Yetkilendirme

Sistemde oturum yönetimi ve servis erişimi JWT (JSON Web Token) tabanlı kimlik doğrulama ile sağlanmaktadır. Böylece her istemci isteği, sunucu tarafında doğrulanabilir ve yetkisiz erişimler engellenebilir. Ayrıca bulut tabanlı güvenlik ve kimlik senaryolarında Azure SDK entegrasyonu kullanılmıştır. Kullanıcılar; Yönetici, Doktor ve Hasta olmak üzere farklı rollerle sisteme erişmekte, her rolün yetki alanı ayrı ayrı yönetilmektedir.

2.4. Performans Optimizasyonu

Yoğun kullanılan ve kritik CRUD işlemlerinde yanıt sürelerini iyileştirmek amacıyla Stored Procedure yapıları tercih edilmiştir. Entity Framework Core'un tam ORM yaklaşımının getirdiği ek yükün istenmediği noktalarda ise Dapper Micro-ORM kullanılarak doğrudan ve yüksek hızlı veri erişimi sağlanmıştır. Bu hibrit yaklaşım (EF Core + Dapper), hem geliştirme hızını hem de kritik senaryolarda çalışma zamanı performansını dengelemektedir.

2.5. Uçtan Uca Haberleşme ve Dokümantasyon

MVC arayüzü, HttpClient kullanılarak Web API'den veri almakta ve API'ye veri göndermektedir. Tüm servis uç noktaları (endpoint'ler), Swagger (OpenAPI) aracılığıyla dokümente edilmiş ve test edilebilir hale getirilmiştir; bu sayede API'nin bağımsız olarak da doğrulanması mümkün kılınmıştır.

3. SİSTEM MODÜLLERİ

Sistem, sağlık kurumu işleyişini uçtan uca kapsayacak şekilde aşağıdaki temel modüllerden oluşmaktadır:

3.1. Hasta ve Doktor Yönetimi

- Hasta Kaydı: T.C. kimlik, kan grubu ve iletişim bilgileriyle yeni hasta kaydı oluşturma ve düzenleme.
- Doktor Yönetimi: Doktorların unvan ve poliklinik bilgileriyle sisteme tanımlanması, ilgili branşlara atanması ve takibi.

3.2. Randevu ve Muayene Süreci

- Poliklinik ve doktora göre online randevu oluşturma, listeleme, iptal/tamamlama işlemleri.
- Doktor panelinde tanı, reçete ve tahlil sonuçlarının işlendiği muayene kayıt ekranı.
- Semptom kontrol modülü ile hastanın şikayetine göre doğru polikliniğe yönlendirilmesi.

3.3. Yatan Hasta ve Oda Yönetimi

- Standart, VIP ve Yoğun Bakım (ICU) odalarının doluluk durumunun anlık takibi.
- Poliklinik randevusu sonrası uygun odaya hasta yatırışı yapılması ve taburcu süreçlerinin yönetimi.

3.4. Faturalandırma ve Ödeme

- Ayakta veya yatarak tedavi gören hastalar için detaylı faturalandırma.
- Ödendi/Ödenmedi durumuna göre filtreleme ve online ödeme (simüle) takibi.
- Kurumsal, kaşeli PDF fatura ve muayene raporu çıktısı üretimi.

3.5. Kullanıcı ve Yetki Yönetimi

- Yönetici, Doktor ve Hasta rollerine sahip kullanıcı hesaplarının merkezi olarak yönetilmesi.
- Rol bazlı yetkilendirme ile her kullanıcının yalnızca kendi yetki alanındaki verilere erişimi.

4. KURULUM

4.1. Sistem Gereksinimleri

- .NET 8.0 SDK
- MS SQL Server (LocalDB veya standart kurulum)
- IDE: Visual Studio 2022 (tavsiye edilen) veya Visual Studio Code

4.2. Projenin Klonlanması

Proje deposu aşağıdaki komutla yerel makineye klonlanır:

```
git clone https://github.com/Ysfcelikkaya/Bitirme_Projesi.git
cd Bitirme_Projesi/HospitalApp
```

4.3. Veritabanı Bağlantısının Ayarlanması

appsettings.json dosyalarındaki ConnectionStrings:Default bölümünün, yerel SQL Server ayarlarına uygun şekilde güncellenmesi gerekmektedir.

4.4. Bağımlılıkların Yüklenmesi ve Derleme

Projeyi derlemek için terminalde aşağıdaki komut çalıştırılır:

```
dotnet build
```

4.5. Projenin Çalıştırılması

Önce Web API projesi, ardından MVC projesi ayrı terminal pencerelerinden başlatılır:

```
# API projesi için
cd HospitalAppApi/HospitalAppApi
dotnet run

# Yeni bir terminalde MVC projesi için
cd ../../HospitalAppMvc/HospitalAppMvc
dotnet run
```

Kurulum tamamlandıktan sonra tarayıcı üzerinden MVC uygulamasına veya API portu üzerinden Swagger dokümantasyon sayfasına erişilebilir.

4.6. Proje Dizin Yapısı

```
HospitalApp
├── HospitalAppApi/    # RESTful servislerin barındırıldığı backend katmanı
├── HospitalAppMvc/    # Web API ile haberleşen son kullanıcı arayüz katmanı
└── HospitalAppDppr/   # Dapper (Micro-ORM) ile veri erişimi yapılan katman
```

5. EKRAN GÖRÜNTÜLERİ

Bu bölümde, geliştirilen sistemin uçtan uca akışını gösteren temel ekranlar görsel olarak sunulmuştur.

5.1. MVC Ön Yüz (Kullanıcı Arayüzü)

1. Ana Karşılama Ekranı

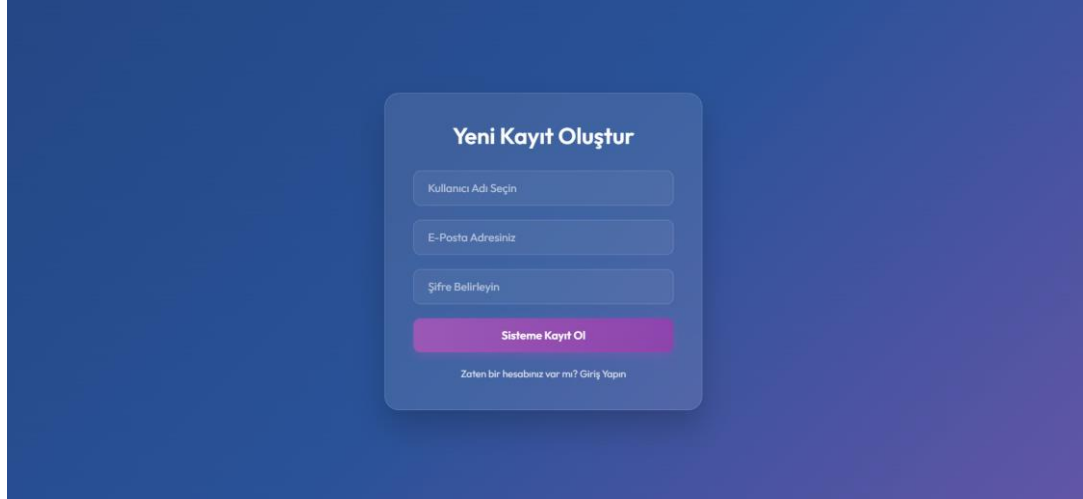
Hastaların sisteme giriş yapabileceği, kayıt olabileceği ve e-randevu ekranlarına yönlendirildiği modern ana sayfa.



Şekil 5.1: Ana Karşılama Ekranı

2. Şifre Yenileme Ekranı

Kullanıcıların unuttukları şifreleri güvenli şekilde sıfırlayabildikleri sayfa.



Şekil 5.2: Şifre Yenileme Ekranı

3. Kullanıcı Kayıt Ekranı

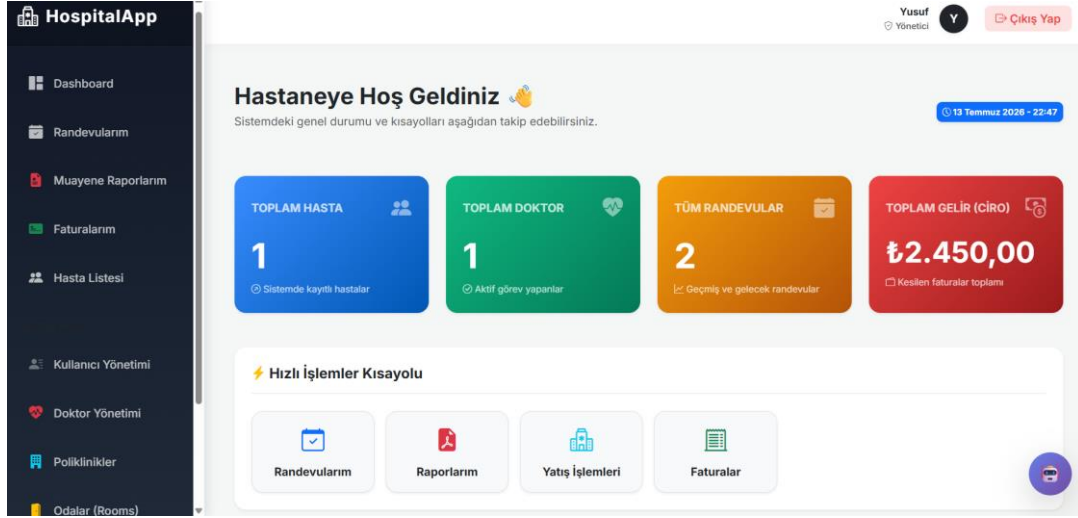
Sisteme ilk defa giriş yapacak hastalar için hazırlanmış kullanıcı dostu kayıt olma sayfası.



Şekil 5.3: Kullanıcı Kayıt Ekranı

4. Yönetim Paneli (Admin Dashboard)

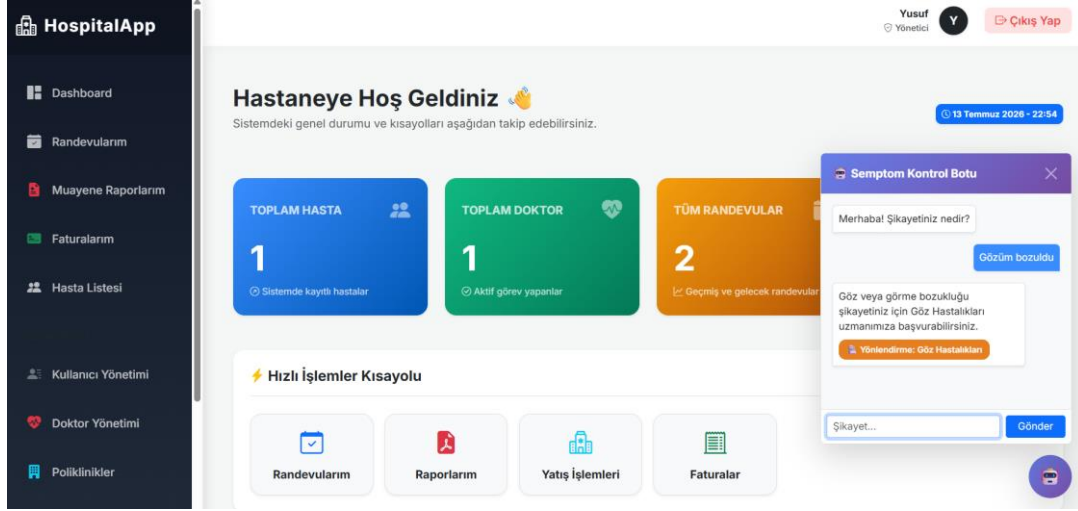
İstatistiklerin, randevuların ve gelirlerin takip edildiği kapsamlı kontrol paneli.



Şekil 5.4: Yönetim Paneli (Admin Dashboard)

5. Semptom Kontrol Botu

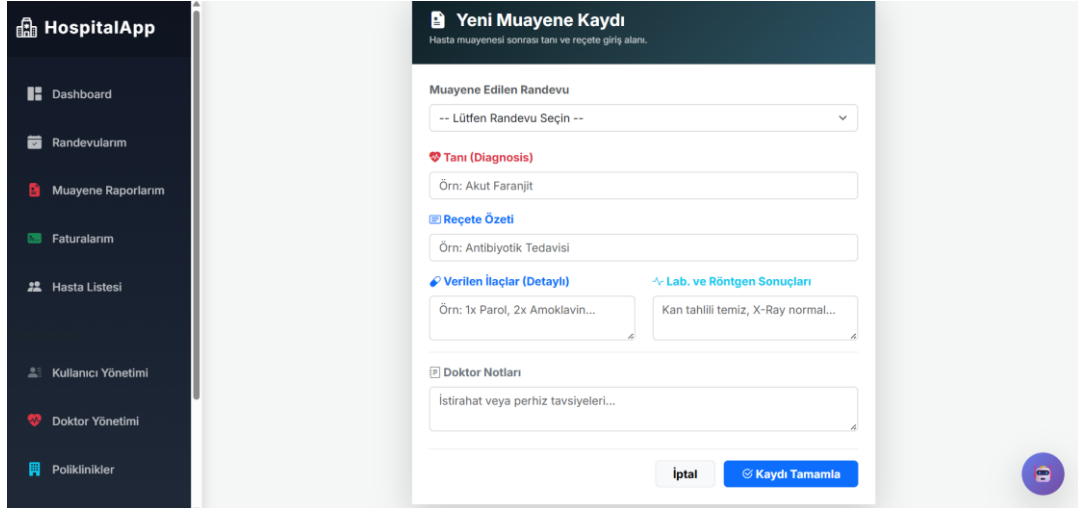
Hastaların şikayetlerini dinleyip doğru polikliniğe yönlendiren akıllı asistan modülü.



Şekil 5.5: Semptom Kontrol Botu

6. Muayene ve Reçete Ekranı

Doktorların tanı koyduğu, reçete yazdığı ve tahlil sonuçlarını girdiği ekran.



Şekil 5.6: Muayene ve Reçete Ekranı

7. Yeni Fatura Kesme İşlemi

Ayakta veya yatarak tedaviler için detaylı faturalandırma ve ödeme yönetim ekranı.

Şekil 5.7: Yeni Fatura Kesme İşlemi

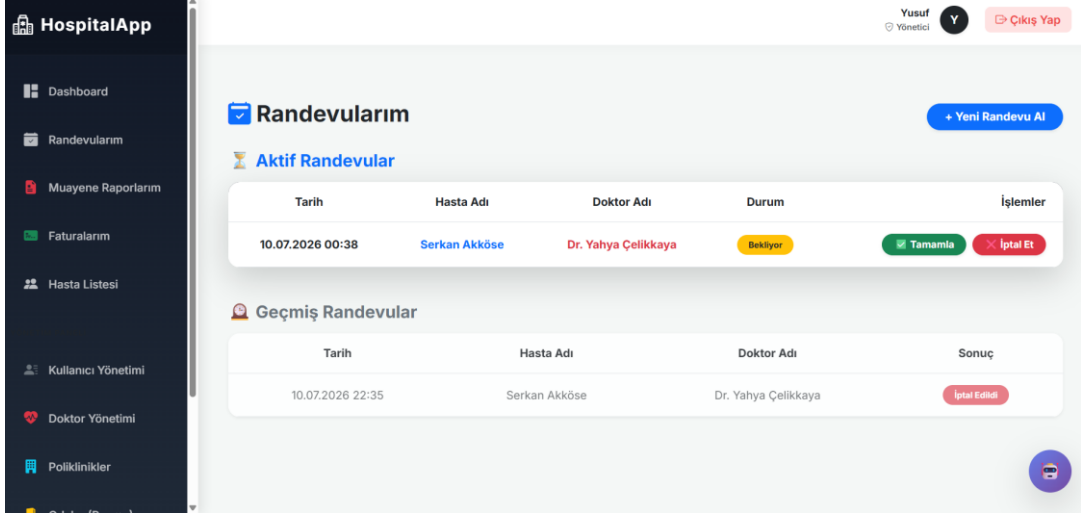
8. Yeni Doktor ve Poliklinik Tanımlama

Yeni uzman hekimlerin unvan ve poliklinik bilgileriyle eklendiği sayfa.

Şekil 5.8: Yeni Doktor ve Poliklinik Tanımlama

9. Randevularım Listesi

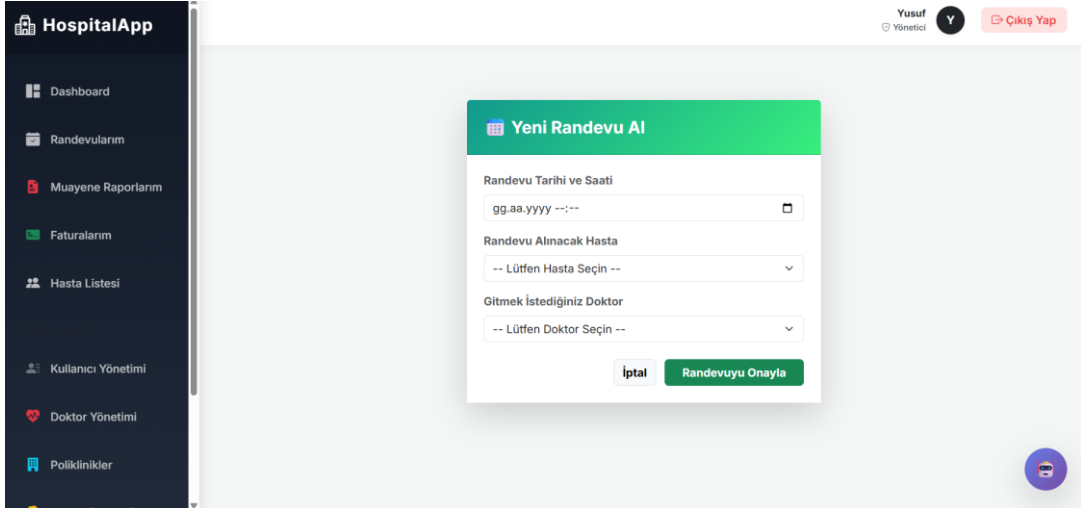
Geçmiş ve aktif randevuların durumlarının takip edildiği liste.



Şekil 5.9: Randevularım Listesi

10. Yeni Randevu Alma Ekranı

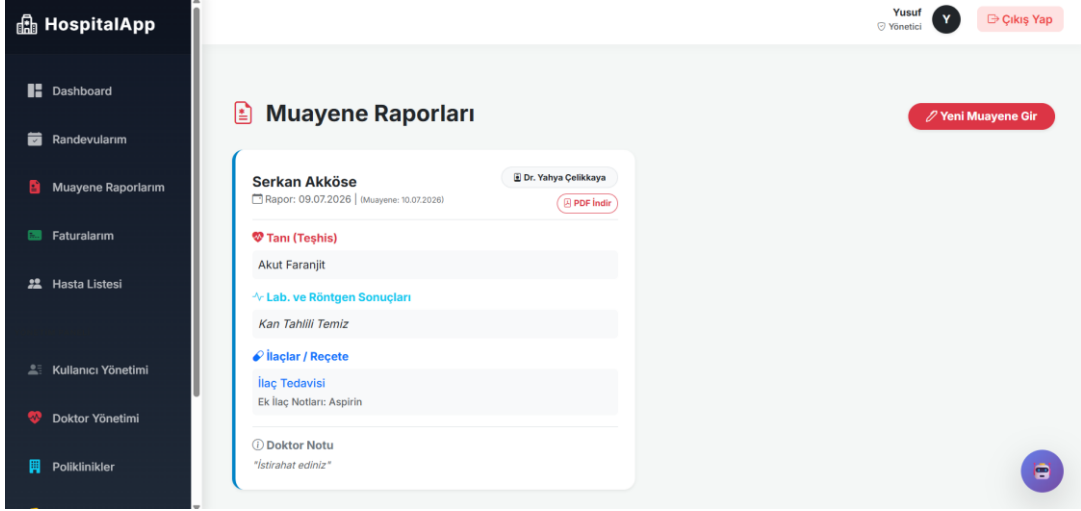
Poliklinik ve doktora göre anında randevu oluşturulan panel.



Şekil 5.10: Yeni Randevu Alma Ekranı

11. Detaylı Muayene Raporu

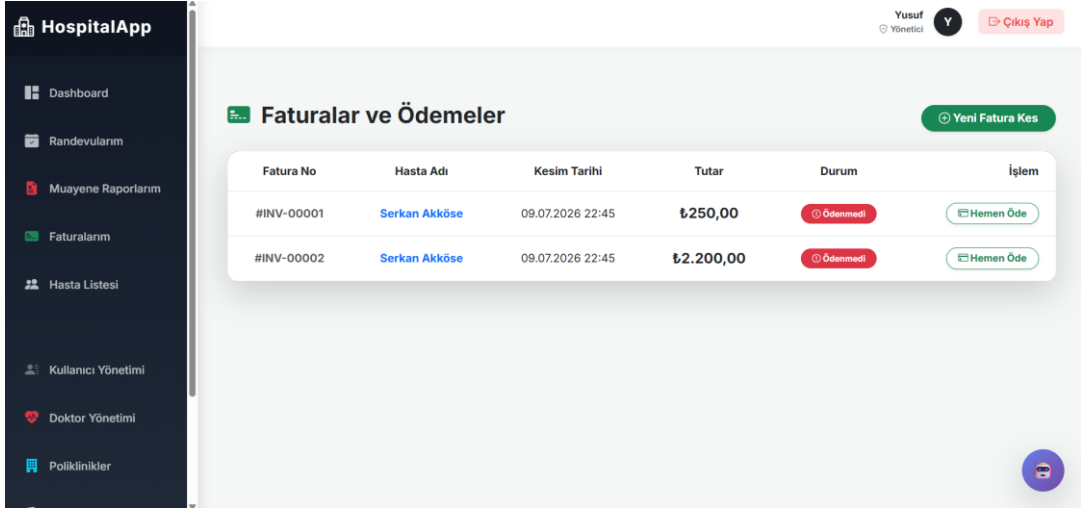
Tanı, ilaç, laboratuvar sonucu ve doktor notlarının yer aldığı çıktı alınabilir rapor.



Şekil 5.11: Detaylı Muayene Raporu

12. Faturalar ve Ödemeler Listesi

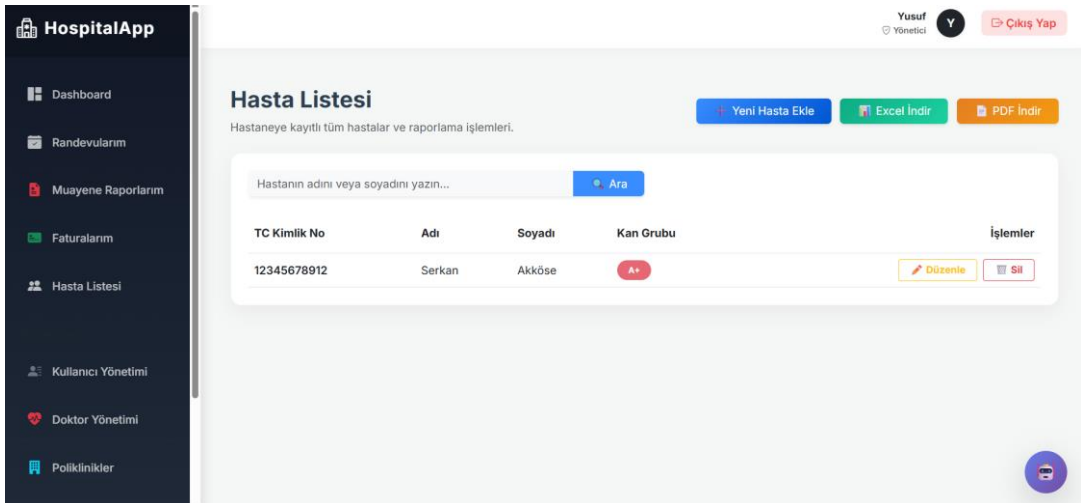
Geçmiş faturaların ve ödeme durumlarının listelendiği ekran.



Şekil 5.12: Faturalar ve Ödemeler Listesi

13. Hasta Listesi ve Yönetimi

Kayıtlı hastaların filtrelenip Excel/PDF olarak indirilebildiği liste.



Şekil 5.13: Hasta Listesi ve Yönetimi

14. Hasta Profil Tanımlama Ekranı

Yeni hastaların detaylı bilgilerle sisteme kaydedildiği form.

Şekil 5.14: Hasta Profil Tanımlama Ekranı

15. Dinamik PDF Rapor Çıktısı

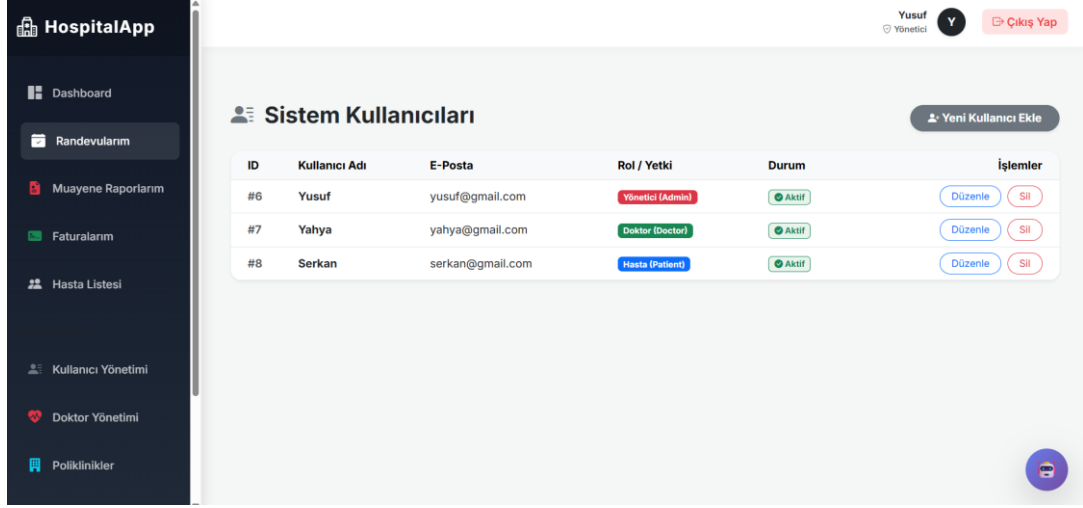
Sistem üzerinden anlık oluşturulan kurumsal PDF dökümleri.

TC Kimlik	Ad	Soyad	Kan Grubu
12345678912	Serkan	Akköse	A+

Şekil 5.15: Dinamik PDF Rapor Çıktısı

16. Sistem Kullanıcıları ve Yetkilendirme

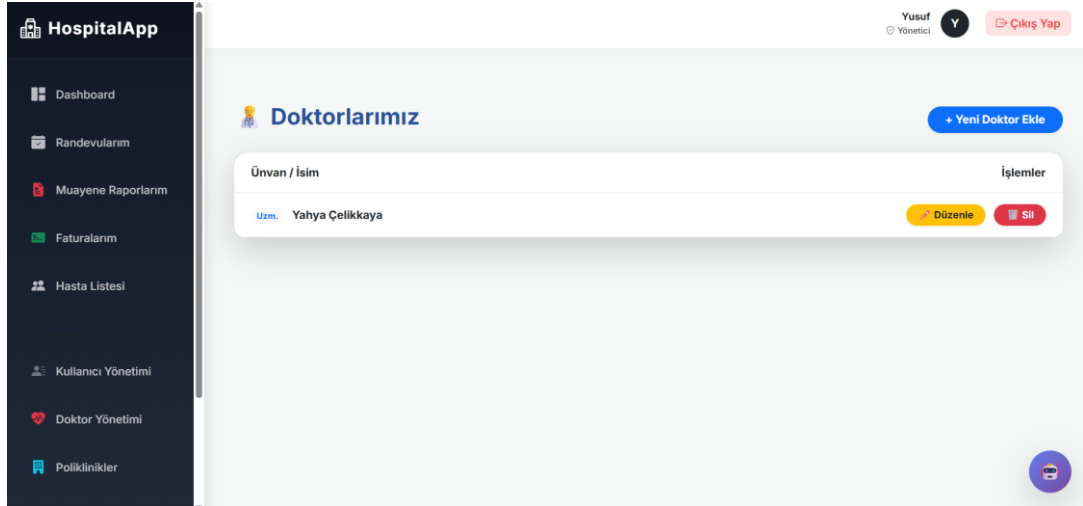
Tüm kullanıcı rollerinin yönetildiği merkezi liste.



Şekil 5.16: Sistem Kullanıcıları ve Yetkilendirme

17. Hastane Doktorları Listesi

Tüm uzman doktorların ve polikliniklerinin görüntülendiği sayfa.



Şekil 5.17: Hastane Doktorları Listesi

18. Yeni Kullanıcı Hesabı Açma

Yeni kullanıcı hesaplarının tanımlandığı modal ekran.

Şekil 5.18: Yeni Kullanıcı Hesabı Açma

19. Poliklinikler Listesi

Hastanedeki tüm tıbbi birimlerin yönetildiği sayfa.

Bölüm Adı	Açıklama	İşlemler
Kardiyoloji	Kalp	Düzenle Sil
Göz Hastalıkları	Göz	Düzenle Sil

Şekil 5.19: Poliklinikler Listesi

20. Yeni Poliklinik Ekleme Modalı

Yeni bir hastane bölümünün kaydedildiği ekran.

Şekil 5.20: Yeni Poliklinik Ekleme Modalı

21. Yatan Hasta Odaları Listesi

Standart, VIP ve ICU odalarının doluluk durumlarının takip edildiği ekran.

Oda No	Tipi	Durum	İşlemler
101	Standard	Dolu	Düzenle Sil
202	VIP	Boş	Düzenle Sil
301	ICU	Boş	Düzenle Sil
102	Standard	Boş	Düzenle Sil

Şekil 5.21: Yatan Hasta Odaları Listesi

22. Yeni Oda Tanımlama Modalı

Yeni odaların kapasite ve tip özellikleriyle eklendiği ekran.

Şekil 5.22: Yeni Oda Tanımlama Modalı

23. Oda Durumu Düzenleme

Odanın doluluk durumunun anlık değiştirilebildiği ekran.

HospitalApp

Yusuf Y Yönetici Çıkış Yap

Odayı Düzenle

Oda Numarası
101

Oda Tipi
Standart Oda

Alt Olduğu Poliklinik
Kardioloji

☒ Bu Oda Dolu Mu?

İptal Değişiklikleri Kaydet

Şekil 5.23: Oda Durumu Düzenleme

24. Yatan Hastalar Listesi

Yatan hastaların oda ve taburcu bilgilerinin yönetildiği sayfa.

HospitalApp

Yusuf Y Yönetici Çıkış Yap

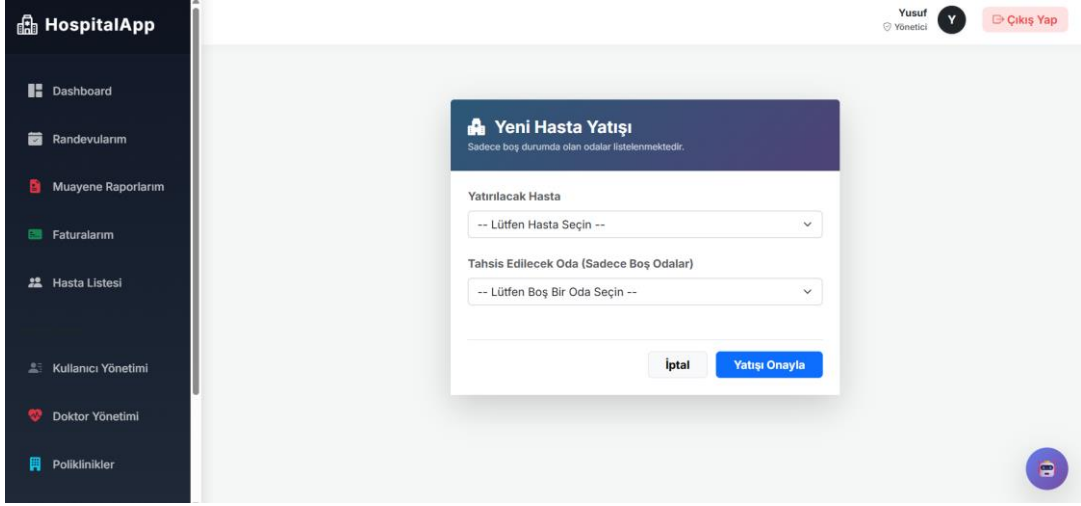
Yatan Hastalar (Yatışlar) + Yeni Hasta Yatır

Hasta Adı	Oda No	Yatış Tarihi	Çıkış Tarihi	Durum	İşlemler
Serkan Akköse	101	09.07.2026 22:45	-	Yatıyor	Taburcu Et

Şekil 5.24: Yatan Hastalar Listesi

25. Yeni Hasta Yatışı Modalı

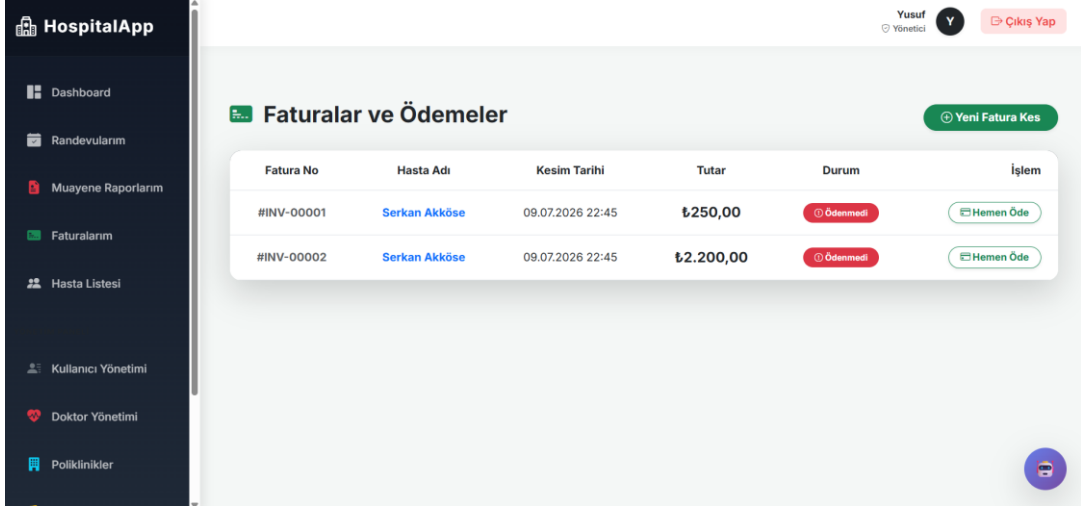
Hastanın boş odalara göre yatışının yapıldığı tahsis ekranı.



Şekil 5.25: Yeni Hasta Yatışı Modalı

26. Fatura ve Ödeme Durumu Takibi

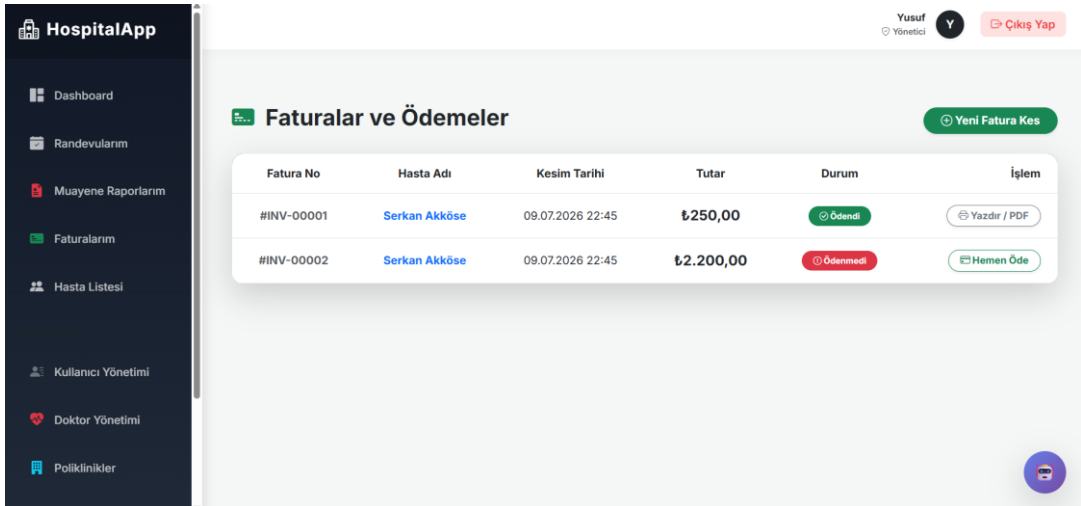
Faturaların ödeme durumuna göre filtrelenebilir takip edildiği liste.



Şekil 5.26: Fatura ve Ödeme Durumu Takibi

27. Online Ödeme ve Tahsilat

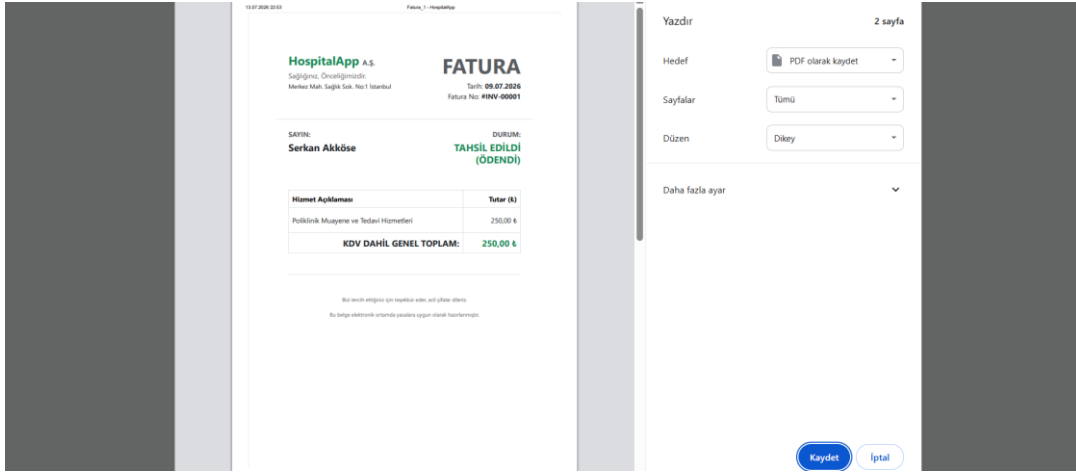
Fatura ödemesi sonrası sistemde anlık durum güncellemesi.



Şekil 5.27: Online Ödeme ve Tahsilat

28. Kurumsal Fatura PDF Çıktısı

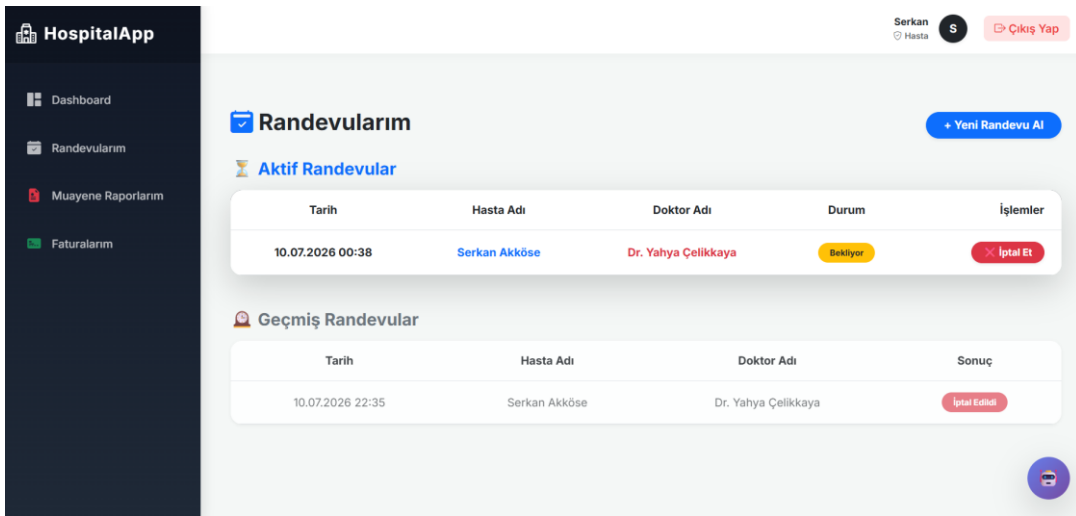
Hastane logolu, kaşeli PDF fatura dökümü.



Şekil 5.28: Kurumsal Fatura PDF Çıktısı

29. Hasta Paneli: Randevularım

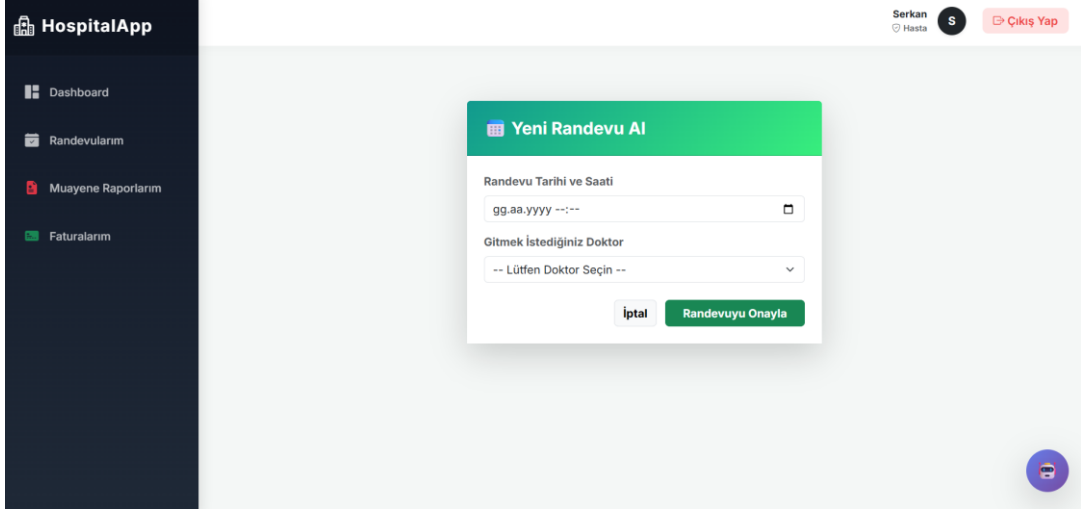
Hastanın kendi randevularını görebildiği profil sayfası.



Şekil 5.29: Hasta Paneli: Randevularım

30. Hasta Paneli: Yeni Randevu Al

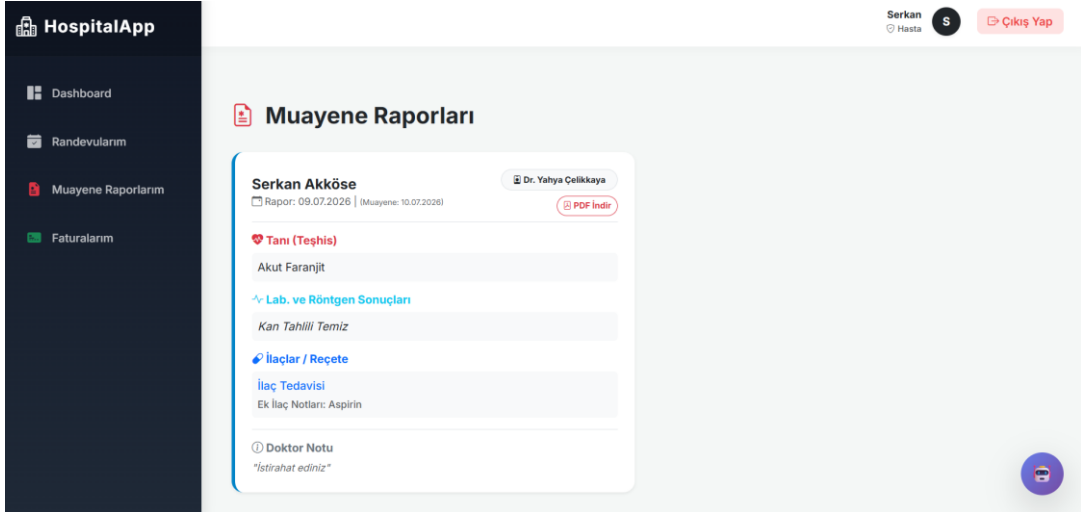
Hastaların kendi kendine randevu alabildiği takvim paneli.



Şekil 5.30: Hasta Paneli: Yeni Randevu Al

31. Hasta Paneli: Muayene Raporlarım

Hastanın kendi tetkik, teşhis ve reçetelerine erişim ekranı.



Şekil 5.31: Hasta Paneli: Muayene Raporlarım

32. Tıbbi Muayene PDF Raporu

Doktor imzalı resmi tıbbi muayene raporu çıktısı.

HOSPITAL APP

Modern Sağlık Bilgi Sistemi

Rapor Tarihi: 09.07.2026 22:37

Rapor No: #MR-00001

TIBBİ MUAYENE RAPORU

HASTA BİLGİLERİ

Serkan Akköse

SORUMLU HEKİM

Dr. Yahya Çelikkaya

Muayene Tarihi: 10.07.2026

Klinik Tanı (Teşhis)

Akut Faranjit

Laboratuvar ve Görüntüleme (Röntgen) Sonuçları

Kan Tahilli Temiz

Reçete ve İlaç Tedavisi

Şekil 5.32: Tıbbi Muayene PDF Raporu

33. Hasta Paneli: Faturalarım

Hastanın masraf ve borç durumunu görebildiği finans ekranı.

HospitalApp

Serkan S
Çıkış Yap

- Dashboard
- Randevularım
- Muayene Raporlarım
- Faturalarım

Faturalar ve Ödemeler

Fatura No	Hasta Adı	Kesim Tarihi	Tutar	Durum	İşlem
#INV-00001	Serkan Akköse	09.07.2026 22:45	₺250,00	Ödendi	Yazdır / PDF
#INV-00002	Serkan Akköse	09.07.2026 22:45	₺2.200,00	Ödenmedi	Hemen Öde

Şekil 5.33: Hasta Paneli: Faturalarım

5.2. API / Swagger ve Dapper Entegrasyonu

Swagger
Select a definition HospitalAppApi v1

Hospital API v1 OAS 3.0

https://localhost:7150/swagger/v1/swagger.json

Authorize

Admissions

GET /api/Admissions

POST /api/Admissions

PUT /api/Admissions/discharge/{id}

AIChat

POST /api/AIChat/Analyze

Appointments

Şekil 5.34: Swagger (OpenAPI) üzerinden test edilebilir hale getirilmiş servis uç noktaları.

Auth			^
POST	/api/Auth/Login		🔒
POST	/api/Auth/Register		🔒
POST	/api/Auth/ResetPassword		🔒
Departments			^
GET	/api/Departments		🔒
POST	/api/Departments		🔒
GET	/api/Departments/{id}		🔒
PUT	/api/Departments/{id}		🔒
DELETE	/api/Departments/{id}		🔒
Doctors			^
GET	/api/Doctors		🔒
POST	/api/Doctors		🔒

Şekil 5.35: Swagger arayüzünde detaylandırılmış bir servis uç noktası örneği.

Swagger			Select a definition	HospitalAppDppr v1
HospitalAppDppr 1.0 OAS 3.0				
https://localhost:7241/swagger/v1/swagger.json				
AdmissionsRead			^	
GET	/api/AdmissionsRead		v	
AppointmentsRead			^	
GET	/api/AppointmentsRead		v	
Dashboard			^	
GET	/api/Dashboard/GetHospitalStatistics		v	
DepartmentsRead			^	
GET	/api/DepartmentsRead		v	
DoctorsRead			^	

Şekil 5.36: Dapper Micro-ORM kullanılarak yapılan yüksek performanslı veri erişimi örneği.

6. SONUÇ

Bu çalışmada, hastane operasyonlarının uçtan uca dijitalleştirilmesini amaçlayan bir Hastane Yönetim Sistemi geliştirilmiştir. Sistem; ASP.NET Core Web API ve MVC mimarisi üzerine kurulmuş, JWT tabanlı güvenli kimlik doğrulama ile desteklenmiş ve kritik veri işlemlerinde Stored Procedure ile Dapper kullanılarak performans odaklı bir altyapı oluşturulmuştur.

Geliştirme sürecinde; hasta ve doktor yönetimi, randevu ve muayene süreçleri, yatan hasta/oda takibi ve faturalandırma gibi modüller Çok Katmanlı (N-Tier) mimari standartlarına uygun şekilde birbirinden bağımsız katmanlara ayrılarak tasarlanmıştır. Bu yapı, sistemin sürdürülebilirliğini, test edilebilirliğini ve ileride yeni modüllerle genişletilebilirliğini kolaylaştırmaktadır.

Sonuç olarak proje; hem kurumsal bir sağlık otomasyon sisteminin gerçek dünya gereksinimlerini karşılayacak şekilde tasarlanmış olması, hem de modern .NET teknolojileri, güvenlik standartları ve performans optimizasyon tekniklerinin bir arada uygulanmış olması bakımından, SoftITo Backend Bitirme Projesi kapsamında kurumsal yazılım geliştirme sürecinin uçtan uca deneyimlenmesini sağlamıştır.

7. KAYNAKÇA

- [1] Microsoft, "ASP.NET Core dokümantasyonu", Microsoft Learn, <https://learn.microsoft.com/aspnet/core/> (Erişim Tarihi: 17.07.2026).
- [2] Microsoft, ".NET 8.0 dokümantasyonu", Microsoft Learn, <https://learn.microsoft.com/dotnet/> (Erişim Tarihi: 17.07.2026).
- [3] Microsoft, "Entity Framework Core dokümantasyonu", Microsoft Learn, <https://learn.microsoft.com/ef/core/> (Erişim Tarihi: 17.07.2026).
- [4] Dapper Contributors, "Dapper - a simple object mapper for .NET", GitHub Deposu, <https://github.com/DapperLib/Dapper> (Erişim Tarihi: 17.07.2026).
- [5] Swagger / SmartBear, "OpenAPI Specification", <https://swagger.io/specification/> (Erişim Tarihi: 17.07.2026).
- [6] M. Jones, J. Bradley, N. Sakimura, "RFC 7519 - JSON Web Token (JWT)", Internet Engineering Task Force (IETF), 2015, <https://datatracker.ietf.org/doc/html/rfc7519>.
- [7] Microsoft, "Azure SDK for .NET dokümantasyonu", Microsoft Learn, <https://learn.microsoft.com/dotnet/azure/sdk/azure-sdk-for-dotnet> (Erişim Tarihi: 17.07.2026).