OBJECT ORIENTED PROGRAMMING : HASTANE BİLGİ SİSTEMİ

Proje üyeleri:

Numarası: 1030521204 İsim-Soyisim: Aykut Ufuk

Numarası: 1030521078 İsim-Soyisim: Mehmet Uruş

Numarası: 1030521022 İsim-Soyisim: Enis Mehmet Korkut

Numarası: 1030521020 İsim-Soyisim: Muhammet Dülek

Tarih: 30/12/2022

1. Gereksinim (Requirements) iş akışı

Uygulama domeninin (application domain) anlatılması:

Hastane Bilgi Sistemi, hastalar ve sistem için geliştirilmiş bir yazılımdır. Hastaların muayene için gelecekleri poliklinikleri, kendi kişisel bilgilerini, muayene olmak istedikleri doktorları ve muayene olmak istedikleri tarih ve saati seçmelerini sağlar. Aynı zamanda randevu günü geldiği vakit hastalara yapılan tetkikler sonucunda belirtilen hastalıklara göre de hasta için reçete oluşturulmasını ve bu reçetelerin kağıda dökülmesine olanak sağlar. Bu Hastane Bilgi Sistemi hastalar için faydalı olabileceği gibi ,muayene için gelmiş olan hastaların kişisel ve randevu bilgilerini de sistem tarafından tutulmasında yardımcı olur.

Sözlük (Glossary) oluşturma:

Sınıf = İçeresinde belirli görevler ve fonksiyonların bulunduğu, kendine has ismi, başlangıç ve bitiş blokları olan yazılım kümesidir.

Bakım = Yazılımın belirli periyotlarda kontrolünün yapılıp gerekli düzenlemelerle yazılımda değişim yada gelişimin yapılmasıdır.

Güncelleme = Eski bir sürümün yerini alan yeni, geliştirilmiş veya sabit bir yazılım sürümüdür.

Taşınabilirlik = Bir uygulamanın bir bilgisayar ortamından diğerine ne kadar kolay aktarılabileceğinin bir ölçüsüdür

Güvenilirlik = Yazılımın belirli koşullar altında, yazılımdan beklenen özellikleri yerine getirebilmesidir.

Minimum Karmaşıklık = Algoritmanın performansını hesaplamak için kullanılır.

Kritiklik Düzeyi = Var olan nesnelerin önem seviyesinin belirlenmesidir.

Project summary = Proje Özetinin verildiği yer.

Purpose, scope, and objectives = Proje amaç , kapsam ve hedeflerinin belirtilmesi.

Assumptions and constraints = Proje için varsayımlar ve kısıtlamalar , teslimat tarihi, bütçe ve kaynakların belirtildiği kısımdır.

Project deliverables = Müşteriye teslim edilecek tüm ürünler burada listelenmiştir, teslim tarihleri ile birlikte.

Schedule and budget summary = Program ve bütçe özeti. Genel program burada sunulmuştur, genel bütçe ile birlikte.

Evolution of the project management plan = Proje yönetim planının gelişiminin anlatıldığı kısımdır.

Reference materials =Proje yönetim planında atıfta bulunulan tüm belgeler burada listelenmiştir.

Definitions and acronyms = Tanımlar ve kısaltmalar proje yönetiminin herkes tarafından aynı şekilde anlaşılması için kullanılır.

External interfaces =Proje üyeleri müşteri organizasyonunun diğer üyeleri ile etkileşime girmesidir.

Internal structure = Kalkınma organizasyonun yapısının tarif edildiği yerdir.

Estimation plan = Proje süresinin ve maliyetinin tahmini yapıldığı yerdir.

Staffing plan = Gerekli personel sayıları ve türleri birlikte listelenir.

Resource acquisition plan = Gerekli kaynakları elde etme yolu, donanım, yazılım, hizmet sözleşmeleri ve idari hizmetler dahil olmak üzere burada verilmiştir.

Metrics collection plan = Olması gereken metrikler (toplanılan bilgiler) burada listelenmiştir.

Project close-out plan = Proje tamamlandıktan sonra yapılması gerekenler, personelin yeniden atanması ve eserlerin arşivlenmesi dahil, burada sunulmuştur

Process Model = Süreç modeli ,Yaşam döngüsünün ayrıntılı bir açıklaması verilmiştir.

Infrastructure plan = Altyapı planının anlatıldığı yerdir.

Product acceptance plan = Ürün kabul planı. Tamamlanan yazılım ürününün geçmesini sağlamak için kabul testi, kabul kriterleri hazırlanmalı, müşteri kriterleri kabul etmelidir.

Configuration management plan = Yapılandırma yönetim planı detaylı anlatıldığı yerdir.

Quality assurance plan = Kalite güvence planı. Test de dahil olmak üzere kalite güvencesinin tüm yönleri, standartlar ve incelemeler bu bölümde yer almaktadır.

Reviews and audits plan = İnceleme ve denetim planı. İncelemelerin nasıl yapıldığına dair ayrıntılar bu bölümde sunulmuştur.

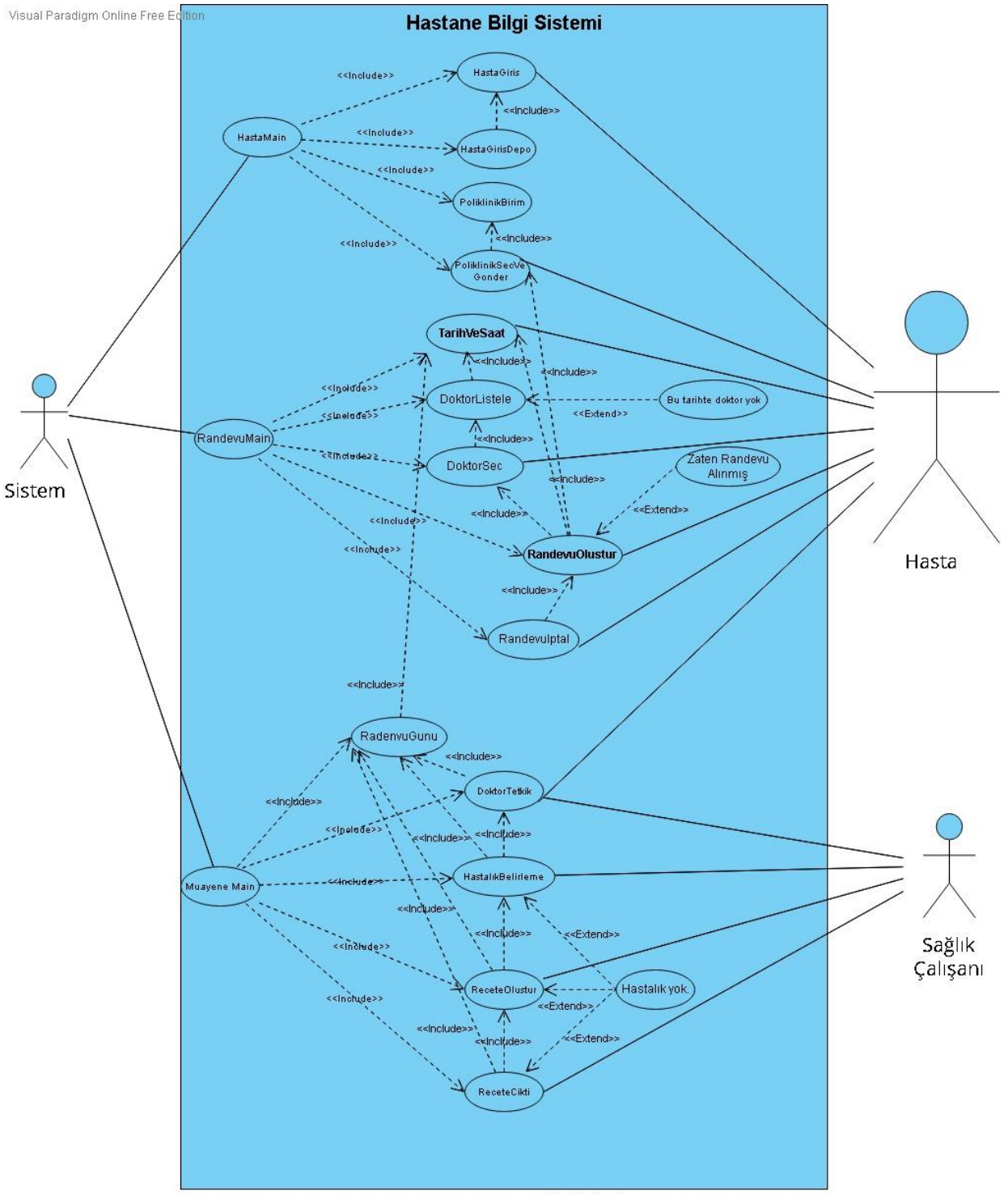
Subcontractor management plan = Taşeron yönetim planının yapıldığı yerdir.

Process improvement plan = Süreç iyileştirme planı, süreç iyileştirme stratejilerinin yapıldığı yerdir.

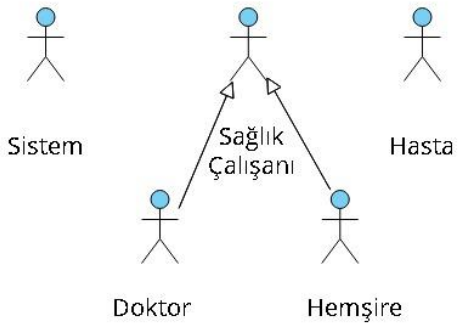
Unified Process = Birleşik Yazılım Geliştirme Süreci veya Birleşik Süreç, yinelemeli ve artımlı bir yazılım geliştirme süreci çerçevesidir.

Use Case = Yazılım programının kendisiyle ve yazılım programının kullanıcıları arasındaki ilişkiyi modeller.

USE CASE:



Use Case Aktörleri



USE CASE CASE DİAGRAMLARINA AİT TANIMLAMALAR:

HastaGiris USE CASE
BRIEF DESCRIPTION Randevu için gelmiş olan hastaların, ikamet ettiği adres, kimlik numaraları, telefon numaraları gibi bilgilerin girişi yapılır.
ENTRY CONDITION Kişinin ad, soyad ve kimlik numarasını yazmış olması
EXIT CONDITION Bu bilgileri döndermesi
STEP-BY-STEP DESCRIPTION 1.Kullanıcı bilgileri alınır: 1.1 İsim,soyiism alınır 1.2 Adres alınır. 1.3 Kimlik numarası alınır. 1.4 Telefon numarası alınır. 2. Bu bilgileri dönder.
EXCEPTIONAL CASES Bulunmamaktadır.

HastaGirisDepo USE CASE
BRIEF DESCRIPTION HastaGiris use case'ini dönen bilgileri depolar.
ENTRY CONDITION HastaGiris use case'inden bu bilgileri alması
EXIT CONDITION Bu bilgileri depolaması
STEP-BY-STEP DESCRIPTION 1.Hasta bilgilerini depolamak için HastaGiris use case'ini aktifleştir 2. HastaGiris use case'inden dönen bilgiyi depola.
EXCEPTIONAL CASES Bulunmamaktadır.

PoliklinikBirim USE CASE
BRIEF DESCRIPTION Hastanenin polikliniklerini hastanın seçimi için listeler.
ENTRY CONDITION Polikliniklerin diziye index olarak aktarılması
EXIT CONDITION Polikliniklerin diziye aktarılmış halinin listelenmesi.
STEP-BY-STEP DESCRIPTION 1. Hastane veritabanından poliklinikleri alır. 2.Poliklinikleri diziye aktarır. 3. Bu polikliniklerin ekranda sırasıyla çıktısını gösterir. 4. Poliklinik dizisini dönderir.
EXCEPTIONAL CASES Bulunmamaktadır.

PoliklinikSecVeGonder USE CASE
BRIEF DESCRIPTION PoliklinikBirim use case'ini diziye ulaşmak için aktifleştirir ve poliklinik seçimi yapılır. Seçilen polikliniğe sahip olan dizi index'ini dönderir.
ENTRY CONDITION PoliklinikBirim use case'inden listenin dönmesi.
EXIT CONDITION Poliklinik seçimi için gerekli index'in seçimi ve bu index'in dönderilmesi
STEP-BY-STEP DESCRIPTION 1. PoliklinikBirim use case'ini aktifleştir. 2. İstenilen poliklinik birimini index olarak seç. 3. Seçili index'e sahip olan polikliniği dönderir.
EXCEPTIONAL CASES Bulunmamaktadır.

HastaMain USE CASE
BRIEF DESCRIPTION Sisteme hastaya ait geçerli tüm bilgilerin ve seçtiği polikliniği dönderir.
ENTRY CONDITION Dahil edilen sınıflardan bilgilerin dönderilmesi
EXIT CONDITION Bu bilgileri sisteme depolanması.
STEP-BY-STEP DESCRIPTION 1.HastaGiris use case'ini aktifleştir. 2.HastaGirisDepo use case'ini aktifleştir . 3.PoliklinikBirim use case'ini aktifleştir. 4.PoliklinikSecVeGonder use case'ini aktifleştir.
EXCEPTIONAL CASES Bulunmamaktadır.

TarihVeSaat USE CASE
BRIEF DESCRIPTION Hastanın randevu için tarih ve saat seçimi.
ENTRY CONDITION Geçerli tarih ve saat girilmesi..
EXIT CONDITION Seçilen tarih ve saatin dönderilmesi
STEP-BY-STEP DESCRIPTION 1.Tarih Seç 1.1 Yıl seç. 1.2 Ay seç. 1.3 Gün seç 2 .Saat seç. 3. Seçilen tarihi ve saati dönder.
EXCEPTIONAL CASES Bulunmamaktadır.

DoktorListele USE CASE

BRIEF DESCRIPTION
TarihVeSaat use case'i aktifleştirerek dönderilen tarihe göre uygun olan doktorları listeler.

ENTRY CONDITION
Belirli tarih ve saat bulunması.

EXIT CONDITION
Belirli tarih ve saatteki doktorların dizi olarak listelenmesi.

STEP-BY-STEP DESCRIPTION
1.TarihVeSaat use case'ini aktifleştir ve buradaki tarih ve saati al.
2.Girilen tarih ve saate göre doktor varsa listele.
3. Doktor yoksa "Doktor bulunamadı" yaz.
4. Başka tarih ve saat seç ve doktor varsa doktorları diziye aktar.

EXCEPTIONAL CASES
1. Bu tarihte doktorun olmaması.

DoktorSec USE CASE

BRIEF DESCRIPTION
DoktorListele use case'ini aktifleştirerek elde edilen doktorlar dizisinden seçim yaptırır ve bunu indeks olarak dönderir.

ENTRY CONDITION
Elimizde doktor listesinin bulunması.

EXIT CONDITION
Doktor seçiminin yapılmış olması.

STEP-BY-STEP DESCRIPTION
1. DoktorListele use case'ini aktifleştir.
2. Listeden doktoru seç.
3. Seçilen doktor indeksini dönder.

EXCEPTIONAL CASES
Bulunmamaktadır.

RandevuOlustur USE CASE

BRIEF DESCRIPTION
TarihVeSaat,PoliklinikSecVeGonder,DoktorSec use case'inden seçilen tarih ve saat, poliklinik ve doktora göre randevu oluşturulur, sisteme kaydedilir.

ENTRY CONDITION
TarihVeSaat,PoliklinikSecVeGonder,DoktorSec use case'inden tarih ve saat,poliklinik ve doktorun seçilmiş olması.

EXIT CONDITION
Randevunun sisteme dönderilmesi / kaydedilmesi.

STEP-BY-STEP DESCRIPTION
1.TarihVeSaat use case'i aktifleştirilerek tarih ve saat alınır.
2.DoktorSec use case'ini aktifleştirir ve seçilen doktor alınır.
3. PoliklinikSecVeGonder use case'i aktifleştirilerek seçilen polikliniği al.
4. Bu bilgileri birleştir ekranda gösterir.
5. Hastanın aynı poliklinikten randevusu yoksa randevu oluşturulur
6.Aynı poliklinikten randeuvusu varsa "Zaten randevu alınmıştır" yaz.
7. Alınan randevuyu sisteme dönder / kaydet.

EXCEPTIONAL CASES
1. Zaten randevunun alınmış olması.

RandevuIptal USE CASE

BRIEF DESCRIPTION
RandevuOlustur use case'ini aktifleştirerek önceden oluşturulan randevuyu iptal ettirmeye yarar.

ENTRY CONDITION
Önceden bir randevunun oluşturulmuş olması.

EXIT CONDITION
Randevunun sistemden silinmesi.

STEP-BY-STEP DESCRIPTION
1.RandevuOlustur use case'ini aktifleştir.
2. Randevunun kayıtlı olduğu adresi veri tabanından sil.

EXCEPTIONAL CASES
Bulunmamaktadır.

RandevuMain USE CASE

BRIEF DESCRIPTION
Randevu için gerekli olan bilgileri sistem için tutar.

ENTRY CONDITION
Dahil edilen sınıflardan bilgilerin dönderilmesi

EXIT CONDITION
Bu bilgileri sisteme aktarması

STEP-BY-STEP DESCRIPTION
1.TarihVeSaat use case'i ile randevu için tarih ve saat belirle.
2.DoktorListele use case'i ile belirlenen zamandaki uygun doktorları listele.
3.DoktorSec use case'i ile listelenen doktorlar dizisinden doktor seç.
4.RandevuOlustur use case'i ile seçilen doktor sonucunda bu randevuyu sisteme kaydet.
5.RandevuIptal use case'i ile istenirse bu randevuyu iptal et.
6. "İşlem başarılı" yaz.

EXCEPTIONAL CASES
Bulunmamaktadır.

RandevuGunu USE CASE

BRIEF DESCRIPTION
TarihveSaat use case'ini aktifleştirerek randevu zamanının gelip gelmediğini gösterir.

ENTRY CONDITION
TarihveSaat use case'inden randevu zamanının alınması.

EXIT CONDITION
Zamana göre değer dönderilmesi.

STEP-BY-STEP DESCRIPTION
1.TarihVeSaat use case'ine randevu zamanını değişkene aktarmak için eriş.
2. Bugün ile randevu zamanını karşılaştır.
3. Bugünün tarihi != TarihVeSaat ise "Randevu zamanı gelmemiş veya randevu zamanı geçmiş" yaz. ve ZamanGeldi 0'a eşitle.
5.Bugünün tarihi = = TarihVeSaat ise devam et ve ZamanGeldi 1'a eşitle.
6. ZamanGeldi değerini dönder.

EXCEPTIONAL CASES
Bulunmamaktadır.

DoktorTetkik USE CASE
BRIEF DESCRIPTION Doktorun hasta üzerinde yaptığı incelemeleri input olarak yazması.
ENTRY CONDITION Randevunu gününün gelmesi.
EXIT CONDITION Doktorun input girmesi.
STEP-BY-STEP DESCRIPTION 1. RandevuGunu use case'ine erişerek zamanın gelip gelmediğine bakar. 2. Randevu zamanı değilse "randevu günü gelmedi veya geçmiş" yaz. 3 Randevu zamanı geldiyse form.hastatetkik textboxları doldur.
EXCEPTIONAL CASES 1. Randevu günü değil.

HastalikBelirleme USE CASE
BRIEF DESCRIPTION DoktorTetkik use case'ini aktifleştirerek ordan gelen bilgilere göre doktorun hastalığı girmesi.
ENTRY CONDITION Randevu gününün gelmesi ve DoktorTetkik use case'inde form.hastatetkik'in doldurulması.
EXIT CONDITION Hastalık belirtisinin yazılması.
STEP-BY-STEP DESCRIPTION 1. RandevuGunu use case'ine erişerek zamanın gelip gelmediğine bakar. 2. Randevu zamanı değilse "randevu günü gelmedi veya geçmiş" yaz. 3. Randevu zamanıysa form.doktortetkik.hastaliktextbox = = "yok" de "hastalık yok" yaz. 4. form.doktortetkik.hastaliktextbox ! = "yok" hastalığa göre form.hastalik doldur
EXCEPTIONAL CASES 1.Hastalık yoktur. 2. Randevu günü değil.

ReceteOlustur USE CASE
BRIEF DESCRIPTION Sağlık çalışanın hastalığa göre reçete yazması.
ENTRY CONDITION Randevu gününün gelmesi ve hastalığın olması.
EXIT CONDITION Recetenin bilgisinin yazılması.
STEP-BY-STEP DESCRIPTION 1. RandevuGunu use case'ine erişerek zamanın gelip gelmediğine bakar. 2. Randevu zamanı değilse "randevu günü gelmedi veya geçmiş" yaz. 3.Randevu zamanı geldiyse form.hastalik'a bak. 4.form.hastalik = = "yok" ise "hastalık yok" yaz. 5.form.hastalik ! = "yok "ise girilen form.hastalik'a göre form.recete'yi doldur.
EXCEPTIONAL CASES 1.Hastalık bulunamadı 2. Randevu günü değil.

ReceteCikti USE CASE
BRIEF DESCRIPTION ReceteOlustur use case'ini aktifleştirerek bilgilerin çıktısını oluştururur.
ENTRY CONDITION Randevu gününün gelmesi, gastalığın olması ve ReceteOlustur use case'inin çalışması.
EXIT CONDITION Reçete çıktısının oluşturulması
STEP-BY-STEP DESCRIPTION 1. RandevuGunu use case'ine erişerek zamanın gelip gelmediğine bakar. 2. Randevu zamanı değilse "randevu günü gelmedi veya geçmiş" yaz. 3.Randevu zamanı geldiyse ReceteOlustur use case'ini aktifleştirerek form.recete bilgilerini alır. 4. Bu bilgileri çıkartır/raporlar/ekrana verir.
EXCEPTIONAL CASES 1. Hastalık yoktur. 2. Randevu günü değil.

MuayeneMain USE CASE
BRIEF DESCRIPTION Muayene için gerekli bilgilerin alınıp sistemde tutulması.
ENTRY CONDITION Dahil edilen sınıflardan bilgilerin dönmesi.
EXIT CONDITION Bu bilgileri sisteme aktarması.
STEP-BY-STEP DESCRIPTION 1.RandevuGunu use case'ini aktifleştirerek randevu zamanını kontrol et. 2.DoktorTetkik use case'ini aktifleştirerek bilgileri forma gir. 3.HastalıkBelirleme use case'ini aktifleştirerek belirlenen hastalık varsa yaz. 4. ReceteOlustur use case'ini aktifleştirerek reçete oluştur. 5. ReceteCikti use case'ini aktifleştirerek reçete çıktısı üret.
EXCEPTIONAL CASES Bulunmamaktadır.

İşlevsel Gereksinimler:

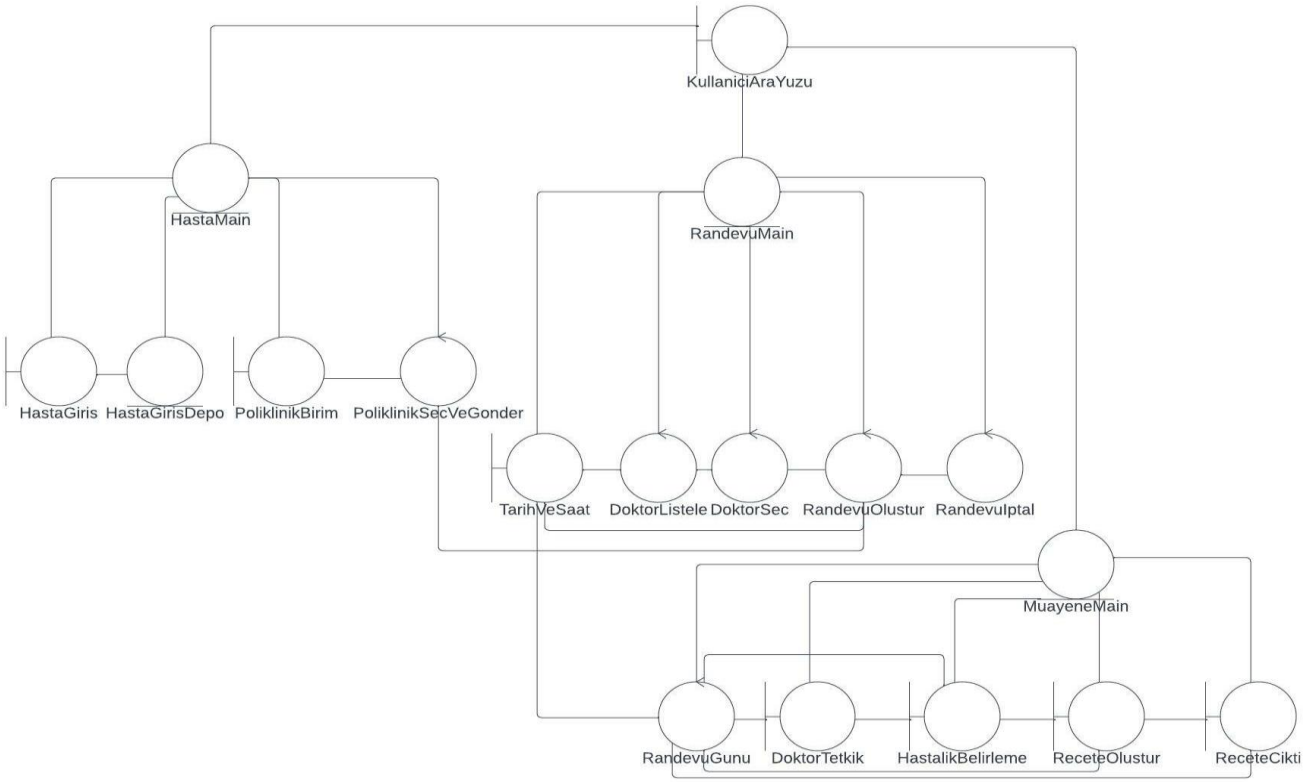
- ✓ Hastanın kişisel bilgi girişini sağlar.
- ✓ Hastanın tedavi olacağı polikliniği seçmesini sağlar.
- ✓ Randevu zamanını seçmesini sağlar.
- ✓ Hastanın tedavi olacağı doktoru seçmesini sağlar.
- ✓ Hastanın randevu onayını yapmasını sağlar.
- ✓ Hastanın randevusunu iptal etmesini sağlar.
- ✓ Randevu günü gelen hastanın tetkiklerinin sisteme yazılması ve hastalığının belirlenmesini sağlar.
- ✓ Reçete oluşturulmasını ve bu reçetenin çıkarılmasını sağlar.

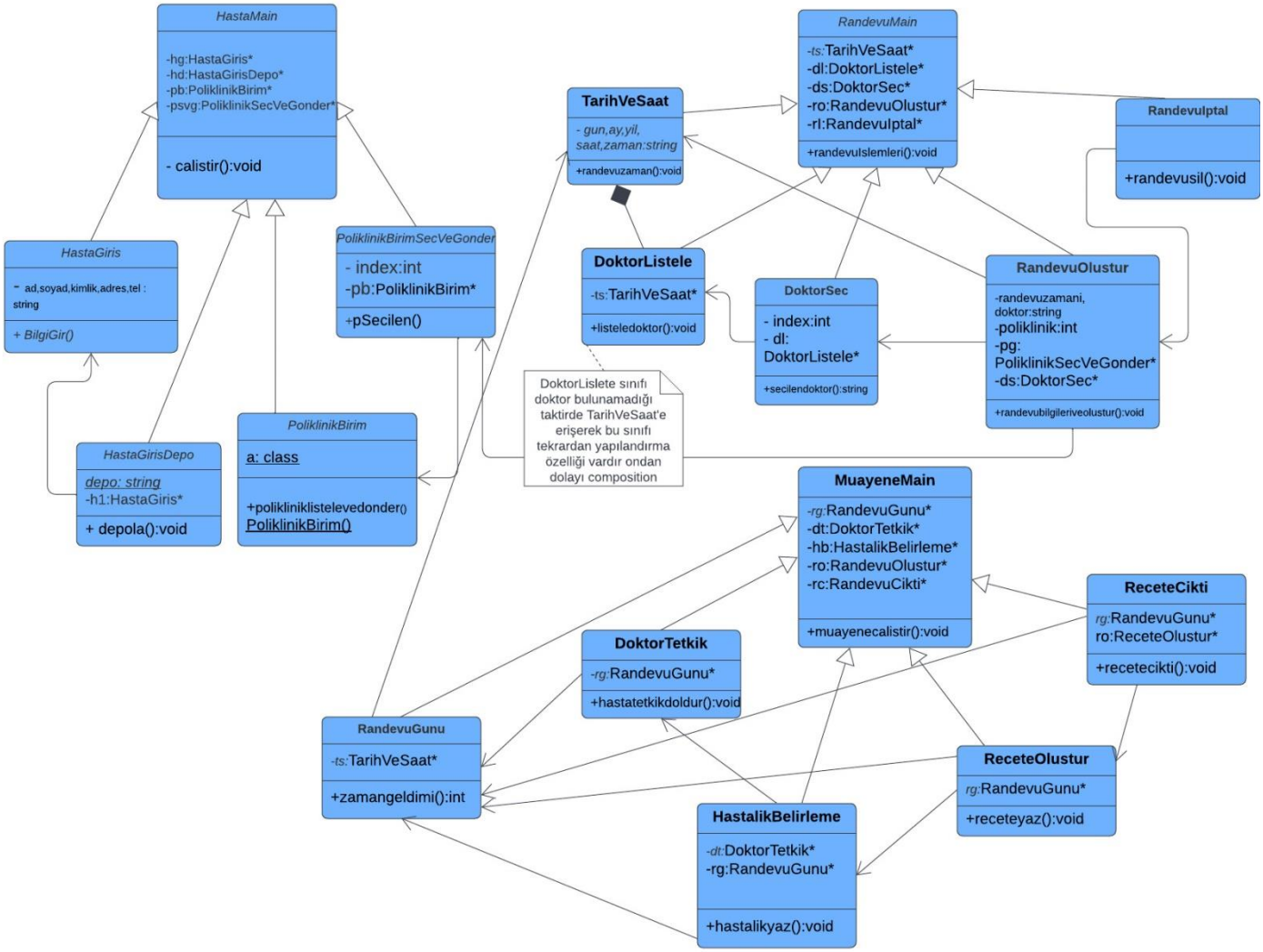
İşlevsel olmayan gereksinimler:

- ✓ Bakımı kolay olmalıdır.
- ✓ Yazılımın hızlı çalışması.
- ✓ Yazılımın güvenilirliğinin olması.
- ✓ Yazılımın kullanıcılarla iletişimi maksimum düzeyde olmalı.
- ✓ Yazılımın hata verme sürekliliği minimum düzeyde olmalı.

2. Analiz (Analysis) iş akışı

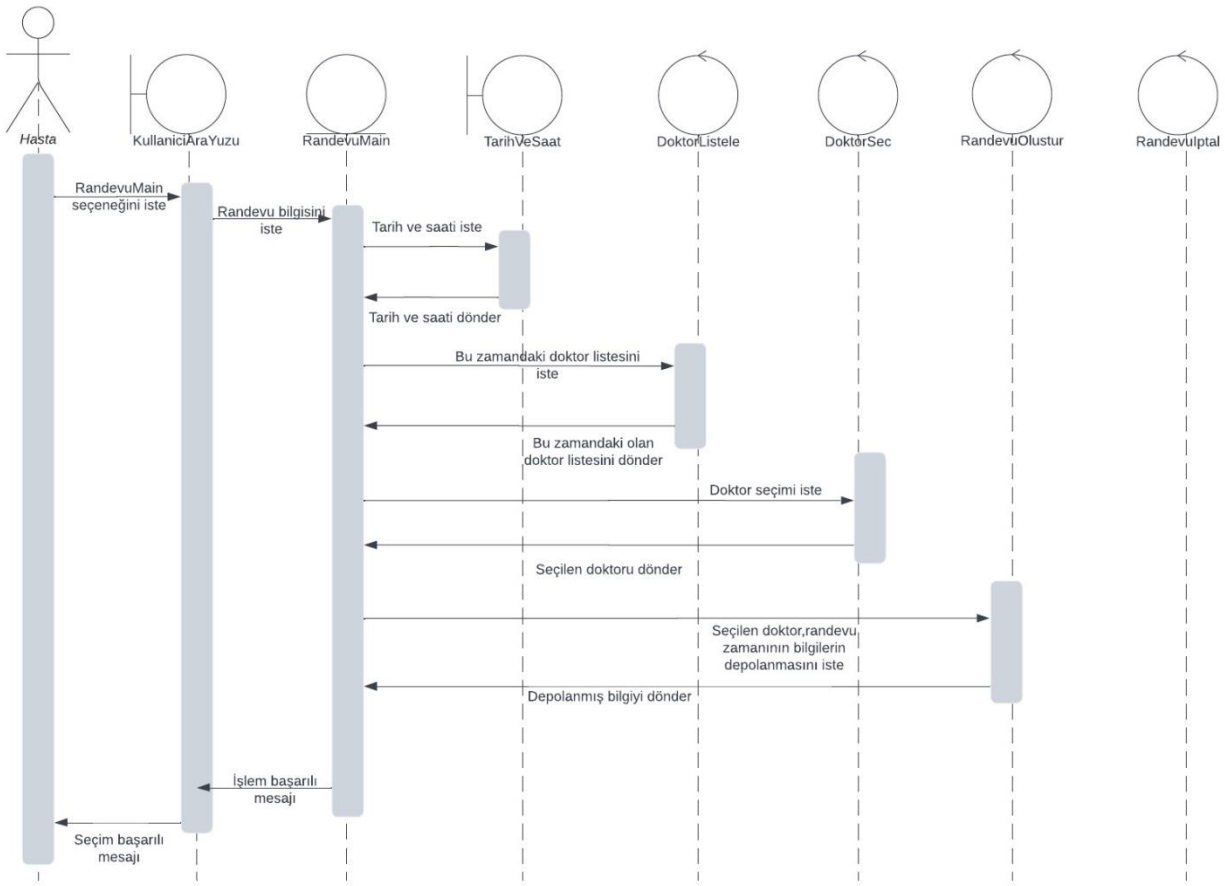
Class diagramlarının çizilmesi , Varlık sınıflarının , Varlık özelliklerinin belirlenmesi, sınır (tampon) sınıfları (boundary class) ve control classların belirlenmesi



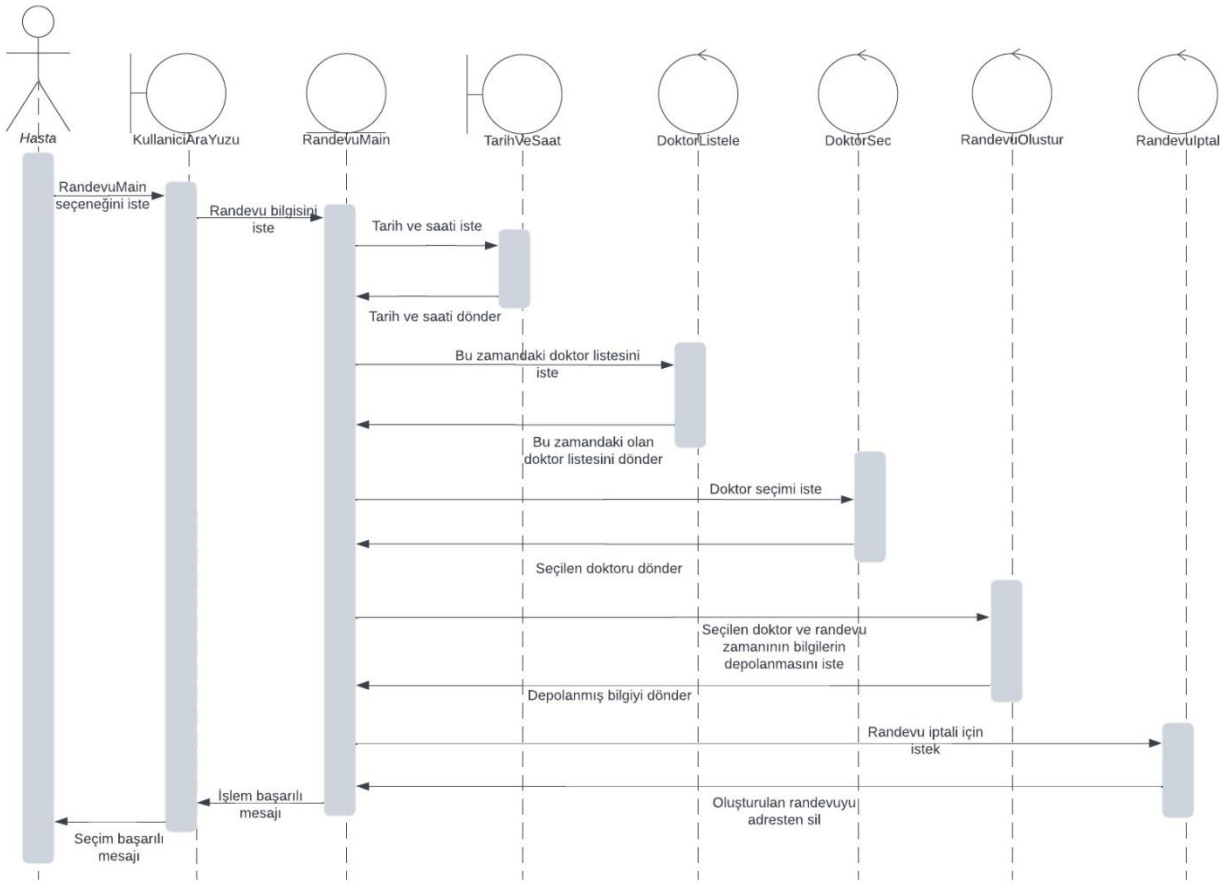

HastaMain
-hg:HastaGiris*
-hd:HastaGirisDepo*
-pb:PoliklinikBirim*
-psvg:PoliklinikSecVeGonder*
- calistir():void
RandevuMain
-ts:TarihVeSaat*
-dl:DoktorListele*
-ds:DoktorSec*
-ro:RandevuOlustur*
-rl:RandevuIptal*
+randevuIslemleri():void
TarihVeSaat
- gun,ay,yil, saat,zaman:string
+randevuzaman():void
RandevuIptal
+randevusil():void
HastaGiris
- ad,soyad,kimlik,adres,tel : string
+ BilgiGir()
PoliklinikBirimSecVeGonder
- index:int
-pb:PoliklinikBirim*
+pSecilen()
DoktorListele
-ts:TarihVeSaat*
+listeledoktor():void
DoktorSec
- index:int
- dl: DoktorListele*
+secilendoktor():string
RandevuOlustur
-randevuzamani, doktor:string
-poliklinik:int
-pg: PoliklinikSecVeGonder*
-ds:DoktorSec*
+randevubilgileriveolustur():void
DoktorLislete sınıfı doktor bulunamadığı taktirde TarihVeSaat'e erişerek bu sınıfı tekrardan yapılandırma özelliği vardır ondan dolayı composition
HastaGirisDepo
depo: string
-h1:HastaGiris*
+ depola():void
PoliklinikBirim
a: class
+polikliniklistelevedonder()
PoliklinikBirim()
MuayeneMain
-rg:RandevuGunu*
-dt:DoktorTetkik*
-hb:HastalikBelirleme*
-ro:RandevuOlustur*
-rc:RandevuCikti*
+muayenecalistir():void
ReceteCikti
rg:RandevuGunu*
ro:ReceteOlustur*
+recetecikti():void
DoktorTetkik
-rg:RandevuGunu*
+hastatetkikdoldur():void
RandevuGunu
-ts:TarihVeSaat*
+zamangeldimi():int
ReceteOlustur
rg:RandevuGunu*
+receteyaz():void
HastalikBelirleme
-dt:DoktorTetkik*
-rg:RandevuGunu*
+hastalikyaz():void

Her bir senaryoya ait sequence diagramların verilmesi (use-case realization) :

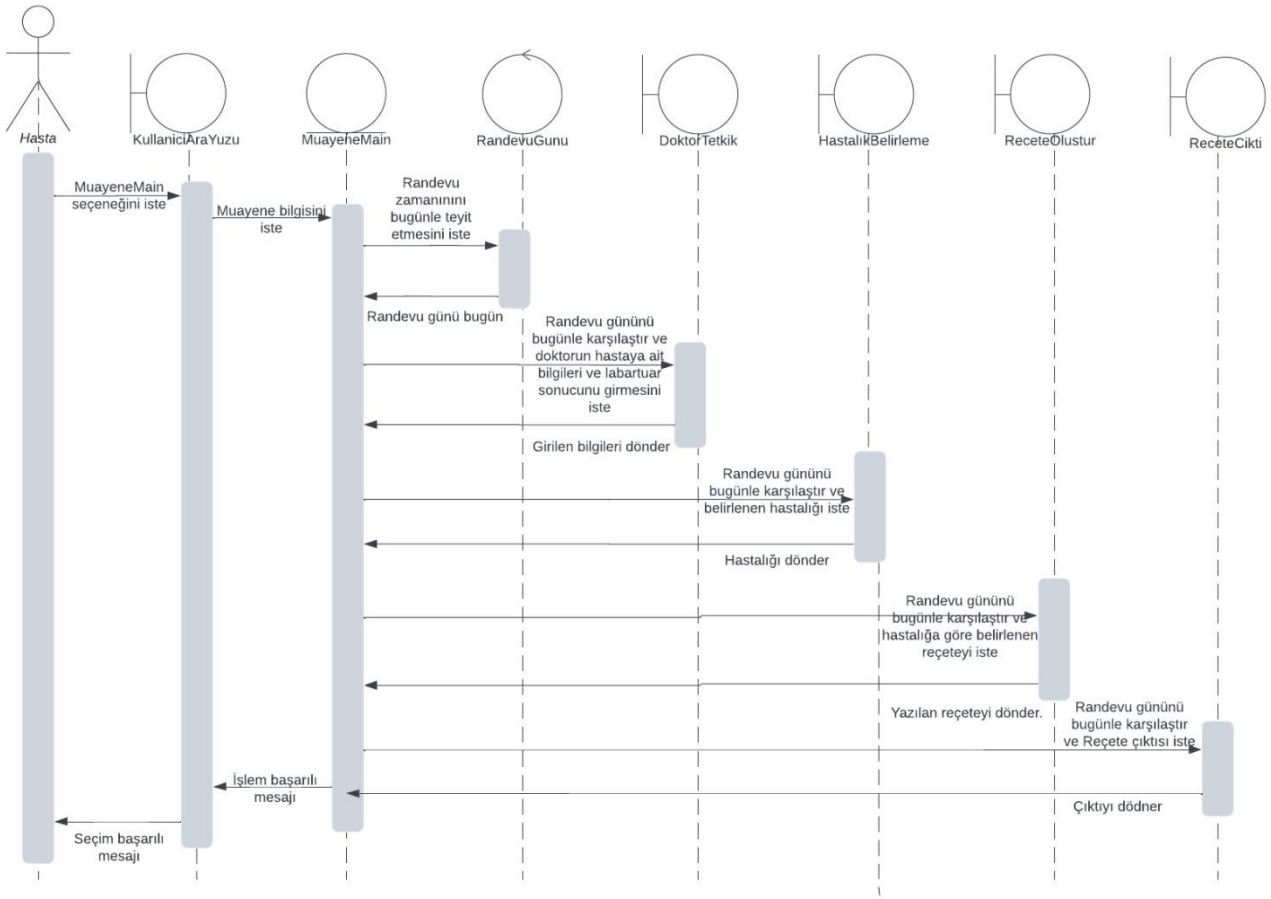
Senaryo: RandevuMain - Randevu oluşturuldu



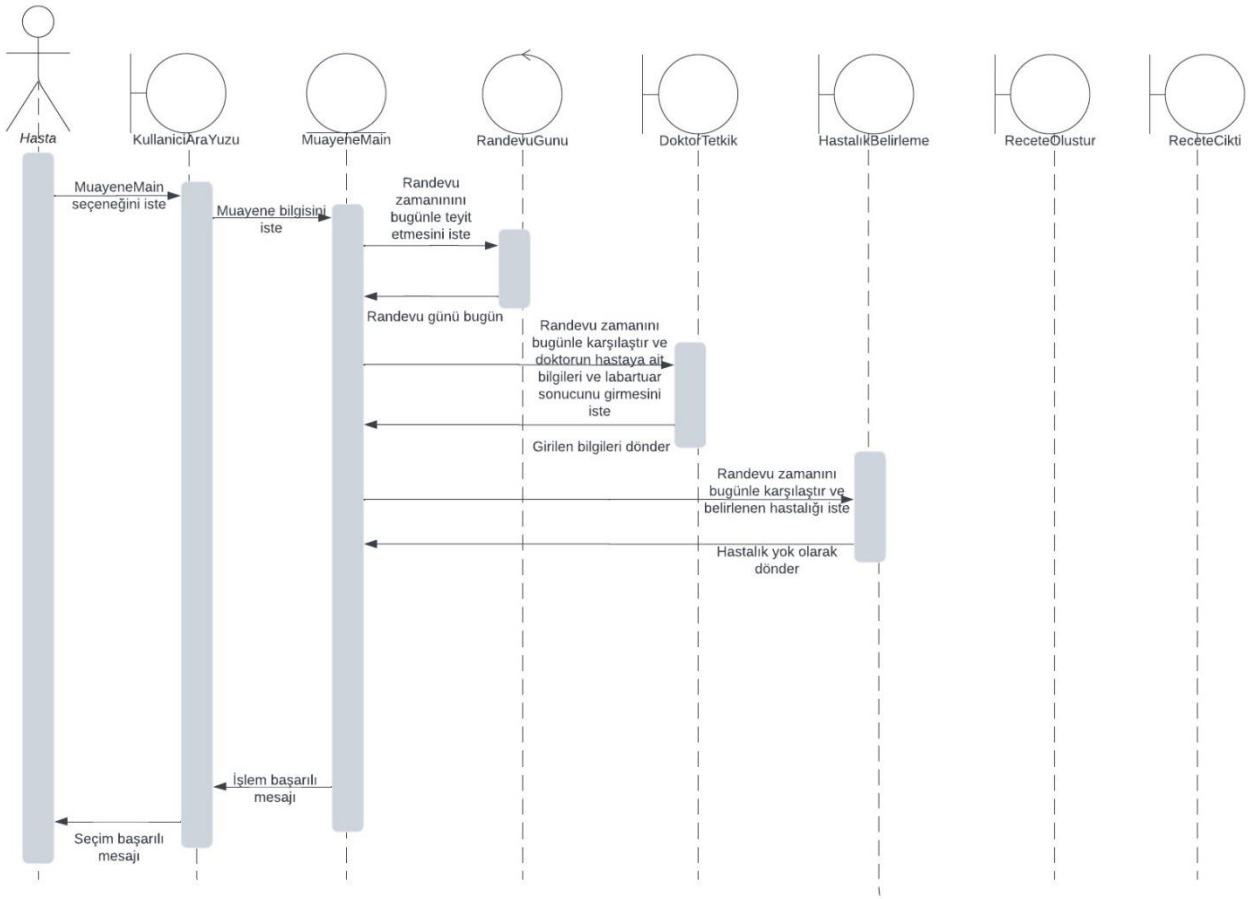
Senaryo: RandevuMain - Randevu iptal edildi



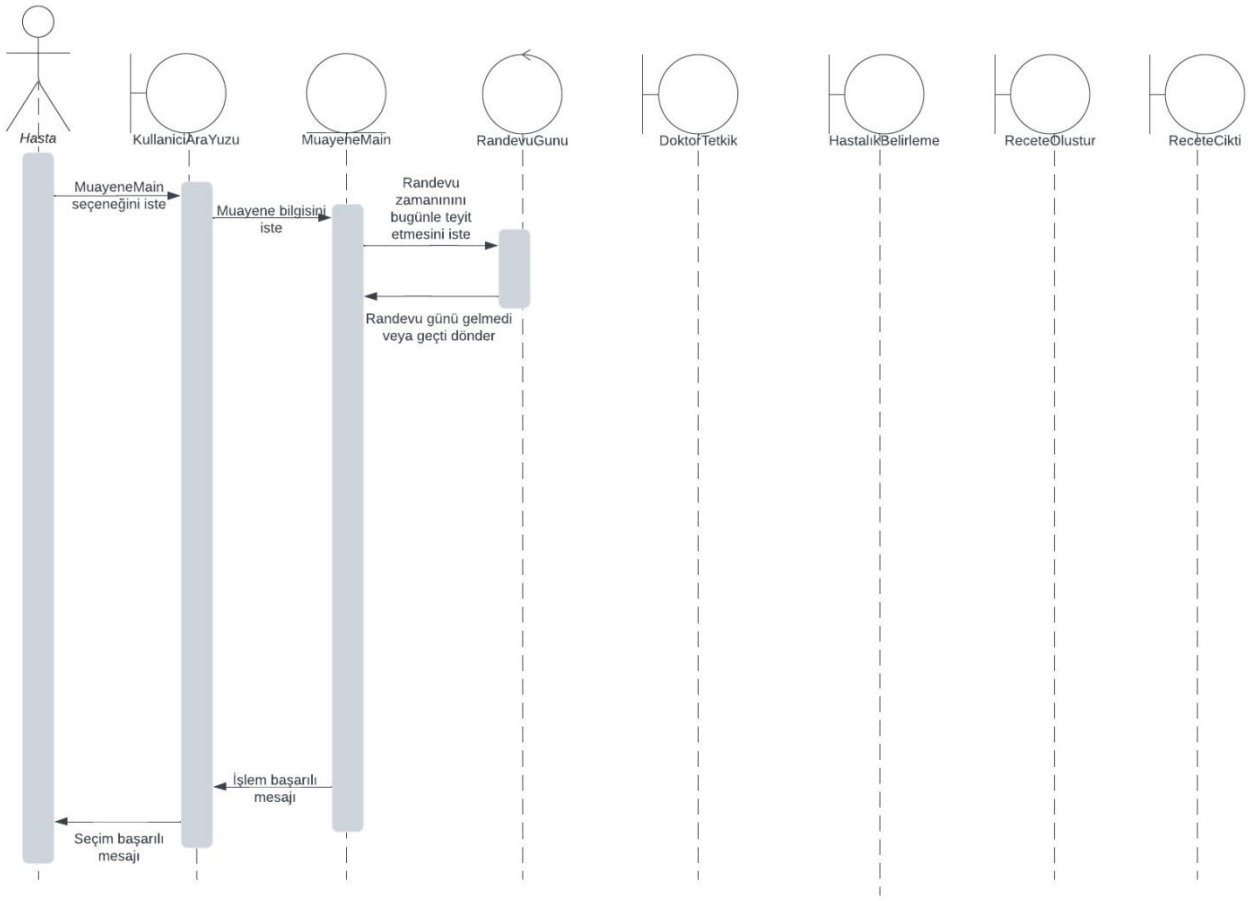
Senaryo: MuayeneMain - Randevu günü ve hastalık var



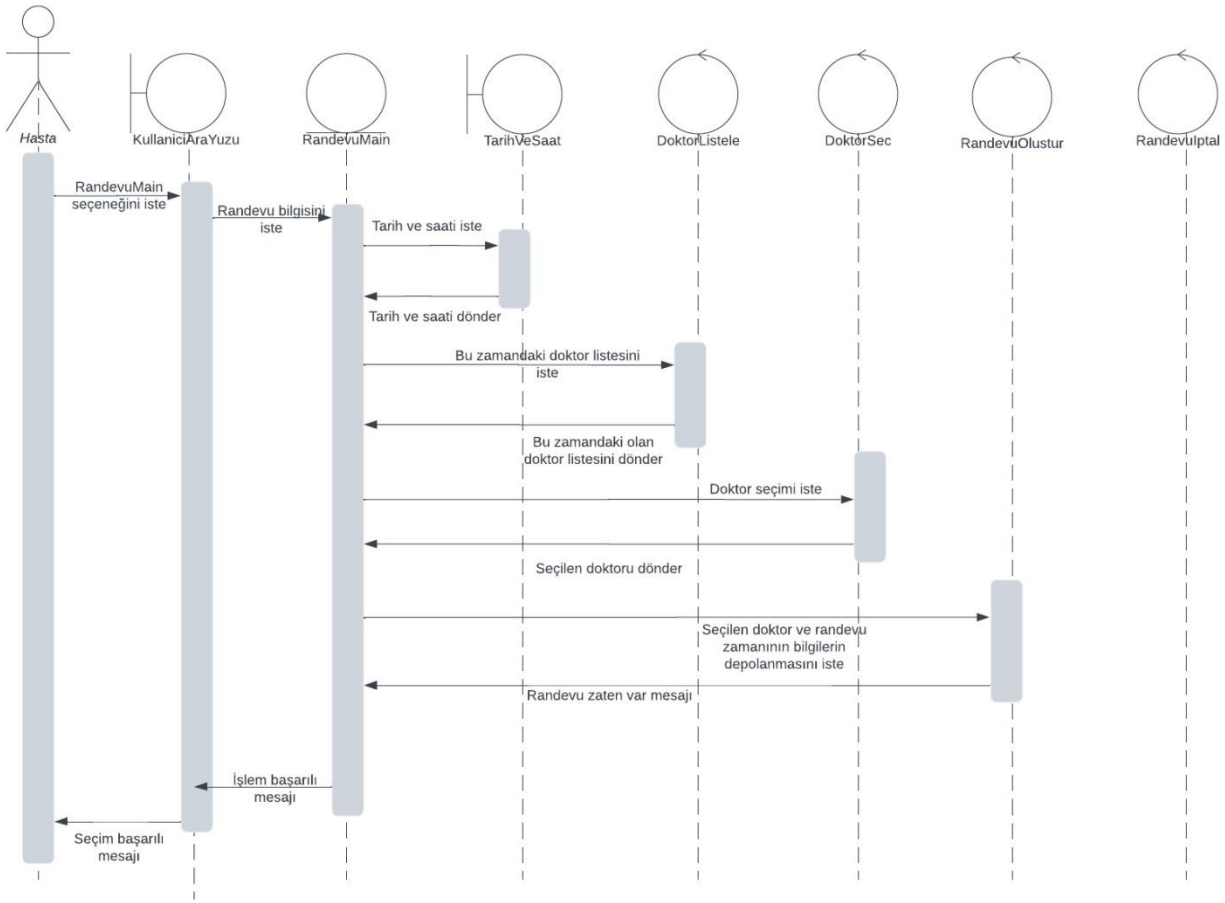
Senaryo: MuayeneMain - Randevu günü ve hastalık yok



Senaryo: MuayeneMain - Randevu günü değil



Senaryo: RandevuMain - Randevu zaten var



CRC KARTLARI:

CLASS **HastaGiris**
RESPONSIBILITY **1.**Hastanın ad soyad al 2.Hasta kimlik al. 3.Hastanın telefon al. 4.Hastanın adres al. 5.Bilgileri dönder.
COLLABORATION 1.HastaMain

CLASS **HastaGirisDepo**
RESPONSIBILITY **1.** Hasta bilgilerini almak için HastaGiris'e mesaj gönder. 2. Hasta bilgilerini depola. 3. Depolanmış bilgiyi kaydet
COLLABORATION 1. HastaGiris 2.HastaMain

CLASS **PoliklinikBirim**
RESPONSIBILITY 1.Hastane veritabanından poliklinik birimlerini al. 2. Bu poliklinikleri dizide depola ve dönder/ekrana ver.
COLLABORATION 1. PoliklinikSecVeGonder. 2.HastaMain

CLASS **PoliklinikSecVeDonder**
RESPONSIBILITY 1.Poliklinikleri almak için PoliklinikBirim'e mesaj gönder 2.Poliklinik seç. 3. Seçilen polikliniği dönder
COLLABORATION 1. PoliklinikBirim 2.HastaMain

CLASS **HastaMain**
RESPONSIBILITY 1. Hasta bilgilerini almak için HastaGiris'e mesaj gönder 2. Girilen hasta bilgilerini depolamak için HastaGirisDepo'ya mesaj gönder. 3.Poliklinik birimlerini görmek için PoliklinikBirim'e mesaj gönder. 4.Poliklinik seçimi için PoliklinikSecVeGonder'e mesaj gönder.
COLLABORATION 1. HastaGiris 2. HastaGirisDepo 3. PoliklinikBirim 4. PoliklinikSecVeGonder

CLASS **TarihVeSaat**
RESPONSIBILITY 1. Tarih gir. 2. Saat gir. 3. Tarih ve saati, zaman değişkeni olarak atama işlemi yap.
COLLABORATION 1. RandevuMain

CLASS **DoktorListele**
RESPONSIBILITY 1. Randevu zamanı için TarihVeSaat'e mesaj gönder. 2. Bu zamandaki doktorları listesini ekrana ver/dönder. 3. Bu zamanda doktor yoksa tekrar seçim yap ve ekrana ver.
COLLABORATION 1. TarihVeSaat 2. RandevuMain

CLASS **DoktorSec**
RESPONSIBILITY **1.** Randevu zamanı uygun doktorları görmek için DoktorListe'ye mesaj gönder. 2. Doktor seç. 3. Seçilen doktoru dönder.
COLLABORATION 1. DoktorListele 2. RandevuMain

CLASS **RandevuOlustur**
RESPONSIBILITY **1.** Randevu zamanı için TarihVeSaat'e mesaj gönder. 2. Bu tarih ve saati al. 3. Tarih ve saati depola. 4. Seçilen doktor için DoktorSec'e mesaj gönder. 5. Seçilen doktoru al. 6. Doktoru depola. 7. Seçilen poliklinik için PoliklinikSecVeGonder'e mesaj gönder. 8. Polikliniği depola. 9. Randevu zamanı,doktor ve polikliniği randevu'da depola. 10. Randevu PoliklinikDatabase'de varsa "zaten randevu oluşturulmuş" yaz. 11. Randevu eklenmemişse PoliklinikDatabase'e ekle ve ekranda randevu bilgisini göster.
COLLABORATION 1. DoktorSec 2.TarihVeSaat 3.PoliklinikSecVeDonder 4.RandevuMain

CLASS **RandevuIptal**
RESPONSIBILITY **1.** Randevu adresini almak için RandevuOlustur'a mesaj gönder. 2. Bu adresi sistemden sil.
COLLABORATION 1. RandevuOlustur 2. RandevuMain

CLASS **RandevuMain**
RESPONSIBILITY 1. Randevunun zaman bilgilerini almak için TarihVeSaat'e mesaj gönder. 3. Bu zamandaki uygun doktorları görmek için DoktorListele'ye mesaj gönder. 4. Seçili doktoru görmek için DoktorSec'e mesaj gönder. 5. Oluşturulan randevuyu görmek için RandevuOlustur'a mesaj gönder. 6.Randevu iptali için RandevuIptal'e mesaj gönder.
COLLABORATION 1.TarihVeSaat 2. DoktorListele 3. DoktorSec 4. RandevuOlustur 5. RandevuIptal

CLASS **RandevuGunu**
RESPONSIBILITY 1. Randevu zamanını için TarihVeSaat'e mesaj gönder. 2. Bugünün tarihi ile randevu zamanını karşılaştır. 3. Bugün randevu zamanıysa 1 dönder. 4. Randevu zamanı değilse 0 dönder.
COLLABORATION 1.TarihVeSaat 2 MuayeneMain

CLASS **DoktorTetkik**
RESPONSIBILITY 1.Randevu zamanıysa hastaya tetkik sonucu için form.hastatetkik doldur. 2. Randevu zamanı değilse "Randevu günü değil" yaz.
COLLABORATION 1. RandevuGunu 2. MuayeneMain

CLASS **HastalikBelirleme**
RESPONSIBILITY 1. Randevu zamanıysa hasta tetkik sonuçları için DoktorTetkik'e mesaj gönder. 2. Belirlenen hastalığı form.hastalik'a yaz. 3. Randevu zamanı değilse "Randevu günü değil" yaz.
COLLABORATION 1. DoktorTetkik 2. MuayeneMain

CLASS **ReceteOlustur**
RESPONSIBILITY 1. Randevu zamanıysa belirlenen hastalık sonucu için HastalikBelirleme'ye mesaj gönder. 2. Belirlenen hastalık için oluşturulan reçete bilgilerini form.recete'ye doldur. 3. Randevu zamanı değilse "Randevu günü değil" yaz.
COLLABORATION 1. HastalikBelirleme 2. RandevuGunu 3. MuayeneMain

CLASS **ReceteCikti**
RESPONSIBILITY 1. Randevu zamanıysa Reçete bilgisi için ReceteOlustur'a mesaj gönder. 2. form.recete'ye eriş ve bunun çıktısını ekrana yazdır/dönder. 3. Randevu zamanı değilse "Randevu günü değil" yaz.
COLLABORATION 1. ReceteOlustur 2.RandevuGunu 3. MuayeneMain

CLASS **MuayeneMain**
RESPONSIBILITY 1. Randevu zamanı için RandevuGunu'ye mesaj gönder. 2. Hastaya yapılan tetkik için DoktorTetkik'e mesaj gönder. 3. Belirlenen hastalık için HastalikBelirleme'ye mesaj gönder. 4. Recete bilgileri için ReceteOlustur'a mesaj gönder. 5. Reçete çıktısı için ReceteCikti'ya mesaj gönder.
COLLABORATION 1. RandevuGunu 2. DoktorTetkik 3. HastalikBelirleme 4. ReceteOlustur 5. ReceteCikti

IEEE formatında SPMP oluşturulması:

1 Overview

1.1 Project Summary.

1.1.1 Purpose, Scope, and Objectives.

Bu projenin amacı hastane bilgi sistemini geliştirmektir. Ürün, hastaların giriş yapacakları polikliniklerde, uygun tarihlerde doktor seçimi yaparak randevu oluşturulması ile başlayan süreçte , randevu gününde doktorların hastalara gerekli tetkikleri uygulaması ve bu tetkikler sonucunda görülen bulgulardan hastalık veya hastalıklar belirlenerek ilaç reçete edilmesi süreçlerini anlatır.

1.1.2 Assumptions and Constraints.

Varsayımlar ve kısıtlamalar şunları içerir;

Ürünü kullanan kullanıcılar (hastalar, hastane çalışanları) üründen memnun olmalıdır.

Son teslim tarihine uymalıdır.

Bütçe kısıtlamasını karşılamalıdır.

Ürün güvenilir olmalıdır.

Daha sonra eklenebilecek ek fonksiyonlara bağlı olarak mimarinin açık olması gerekir.

Ürün kullanışlı, kullanıcı dostu olmalıdır.

Ürün randevu sistemi her kesim tarafından kullanılacağı için arayüzü sade ve anlaşılır olmalıdır.

Ürün sürekli güncel olmalı, doktor listeleri ve iptal edilen randevular gibi durumlarda sistem hata vermemelidir.

1.1.3 Project Deliverables.

Son teslim tarihine kadar ürünün tamamı, kullanım kılavuzu ve gerekli belgelerle birlikte teslim edilecektir.

1.1.4 Schedule and Budget Summary.

Gereksinim İş Akışı (4 grup üyesi $-)

Analiz İş Akışı (4 grup üyesi $-)

Tasarım İş Akışı (4 grup üyesi $-)

Gerçekleştirme İş Akışı (4 grup üyesi $-)

Test İş Akışı (4 grup üyesi $-)

1.2 Evolution of the Project Management Plan.

Proje yönetim planı geliştirme sürecinde ayrı bir plan aşaması yoktur. Plan her aşamada yapılır. Aynı zamanda ayrı bir test ve dokümantasyon aşaması da yoktur her aşamada test yapılmalı ve bu işlemler doküman hale getirilmelidir. Müşterinin isteği doğru anlaşılmalıdır. Proje yönetim planını doğru ve güncel tutmak için tüm değişiklikler, geliştirmeler belgelenmelidir.

2 Reference Materials.

Visual Paradigm

LucidChart

3 Definitions and Acronyms.

HBS : HASTANE BİLGİ SİSTEMİ

4 Project Organization.

4.1 External Interfaces.

Bu ürün, proje üyeleri (Aykut, Mehmet, Muhammet, Enis) tarafından müşteri istek ve ihtiyaçları doğrultusunda yapılmıştır.

4.2 Internal Structure.

Bu proje kalkınma organizasyonunun yapısı ve birçok yazılım geliştirme kuruluşlarının belirlediği gruplar baz alınarak tek bir proje üzerinde çalışan geliştirme ve destek grupları tipine sahiptir. Proje geliştirme ekibi Aykut, Mehmet, Muhammet ve Enis'tir.

4.3 Roles and Responsibilities .

Mehmet: Gereksinim İş Akışı, Analiz İş Akışı

Aykut: Analiz İş Akışı, Tasarım İş Akışı

Muhammet: Gereksinim İş akışı, Analiz İş Akışı

Enis: Tasarım İş Akışı

5 Managerial Process Plans.

5.1 Start-up Plan.

5.1.1 Estimation Plan.

Toplam yazılım geliştirme ve ürün oluşturma süreci, ürünün son teslim tarihine (30 Aralık 2022 saat 16.00) ve tahmini maliyetine uygun olarak tahmini plan yapılmıştır.

5.1.2 Staffing Plan.

Proje görev dağılımı ve ihtiyaç duyulan süreler şu şekildedir.

Gereksinim İş Akışı iki kişi Mehmet ve Muhammet görev almıştır ihtiyaç duyulan süre ise 3 gündür.

Analiz İş Akışı üç kişi Mehmet ve Muhammet ve Aykut görev almıştır ihtiyaç duyulan süre ise 6 gündür.

Tasarım İş Akışı iki kişi Aykut ve Enis görev almıştır ihtiyaç duyulan süre ise 5 gündür.

5.1.3 Resource Acquisition Plan.

Proje için gerekli olan kaynaklar, donanımlar ve hizmetler bu alanda tahmini planlanmıştır.

5.1.4 Project Staff Training Plan

Bu proje için ek proje personel eğitim planı yapılmamıştır. Gerekli olan tüm eğitimler proje üyelerinin lisans eğitimlerinde verilmiştir.

5.2 Work Plan.

5.2.1 Work Activities

İş faaliyetleri aşağıdaki şekilde belirtilmiştir.

Gün 1-3: Gereksinim İş Akışı tamamlanması.

Gün 4-9: Analiz İş Akışı tamamlanması.

Gün 10-14: Tasarım İş Akışı tamamlanması.

5.2.2 Schedule Allocation

Çalışmalar ve paketler arasındaki bağlılıklar şu şekildedir. İlk olarak Gereksinim İş akışı yapılmıştır ve buradan yola çıkılarak Analiz İş Akışı yapılmıştır. Tasarım İş Akışı ile proje tamamlanırken Gerçekleştirme İş Akışı bulunmamaktadır.

5.2.3 Resource Allocation.

Bu proje için ekstra kaynak tahsisi yapılmamıştır.

5.2.4 Budget Allocation

Bu proje ve kapsamından dolayı bütçe tahsisi yapılmamıştır.

5.3 Control Plan.

Gereksinim kontrol planı gereksinimlere uygun bir şekilde yapılmıştır. Gerçekleştirme İş Akışı proje kapsamı dışında bulunduğu için bu alanda kalite kontrol, risk kontrol ve bütçe kontrol planları yapılmamıştır bu yüzden Requirements Control Plan yapılmış olup Schedule, Budget, Quality , Reporting ,Metrics Collection planları bulunmamaktadır.

5.4 Risk Management Plan.

Bu proje ve kapsamından dolayı risk yönetim planı yapılmamıştır.

5.5 Project Close-out Plan.

Proje tamamlandıktan sonra gerekli dokümanlar, planlar müşteri sunulmuş ve arşive kaydedilmiştir.

6 Technical Process Plans.

6.1 Process Model.

Projede kullanılan diyagram ve iş modelleri şu şekildedir.

Use-case diyagramları

Class diyagramları

Sequance Diyagramları

6.2 Methods, Tools, and Techniques.

İş akışları Unified Process ile gerçekleştirilmiştir. Ürün taslakları C# ile çıkarılmıştır.

6.3 Infrastructure Plan.

Ürün, kişisel bir bilgisayarda Windows platformu altında çalışan Visual paradigm ve lucid chart kullanılarak geliştirilmiştir.

6.4 Product Acceptance Plan.

Ürün kabul planı Müşterinin isteklerini doğru anlayıp detaylı gereksinim aşamasından sonra Unified Process aşamalarına uygun olarak tasarlanmıştır.

7 Supporting Process Plan

7.1 Configuration Management Plan.

Ürün yapılandırma yönetim planı verilen dokümantasyonlarda detaylı bir şekilde anlatılmıştır.

7.2 Testing Plan.

Proje kapsamı doğrultusunda Test İş Akışı bulunmadığı için test planı da bulunmamaktadır.

7.3 Documentation Plan.

Dokümantasyon Unified Process de belirtildiği şekilde hazırlanmıştır.

7.4 Quality Assurance Plan

7.5 Reviews and Audits Plan.

7.6 Problem Resolution Plan.

Proje sürecinde proje Gerçekleştirme İş Akışı bulunmadığından dolayı çıkabilecek hatalar daha çok önceki İş Akış modellerinde öngörülmüştür. Çıkabilecek problemler proje üyelerinden Aykut'a iletilecektir.

7.7 Subcontractor Management Plan.

Proje kapsamında yer almamaktadır.

7.8 Process Improvement Plan.

Proje iyileştirme süreçleri şu şekilde ilerlemiştir. Proje İş Akışı modelleri proje üyeleri arasında iki kişiye bölünerek, Analiz İş Akışı ise 3 kişi kapsamında hazırlanmış olup sonrasında proje üyeleri ile komple tekrar edilmiştir ve doküman haline getirmiştir. Bu şekilde proje ve üyeleri arasında aksaklıklar kapatılmaya çalışılmıştır.

8. Additional Plans

Herhangi ek bir plan bulunmamaktadır.

IEEE 829-2008 formatında test planlarının hazırlanması:

-Giriş

-Test Ögeleri

-Yazılımın Risk Unsurları

-Test Edilmeyen Özellikler

-Test Edilmeyen Özellikler

-Yaklaşım

-Ögelerin Çalışma/Çalışmama Kriterleri

-Sorumluluklar

-Askıya Alma Kriterleri ve Devam Ettirme Gereksinimleri

-Takvim

1. Giriş

- Test edilecek ögeler ve özellikleri

2. Test Ögeleri

- Sınıflar ve veriler

3.Yazılımın Risk Unsurları

- Yoğunluktan dolayı hata vermesi
- Bakımların zamanında yapılmaması
- Güncelleme ile yeni sorunların çıkması
- Farklı cihazlarla giriş sorunu

4.Test Edilmeyen Özellikler

- Sınıfların hız testi
- Yazılımın bakım testi
- Taşınabilirliğinin testi

5.Test Edilen Özellikler

- Güvenilirliği
- Yazılımın kullanışlılığı
- Yazılımın fonksiyonlarının amacına uygunluğu

6. Yaklaşım

- Test aktivite planı hazırlanır.
- Ana etkinlikleri,araçları ve teknikleri belirlenir.
- Minimum karmaşıklık belirlenir
- Kritiklik düzeyine göre test aşamaları belirlenir
- Muhtemel riskler göz önünde bulundurularak test edilir
- Gerekirse ek test aşamaları belirlenir
- Bu aşamalar raporlanır

7. Ögelerin Çalışma/Çalışmama Kriterleri

- Sınıfın hatasız çalışması
- Sınıfın belirlenen zamanda çalışması(response time)
- Verinin doğru formatta girilmesi
- Verinin doğru bilgide girilmesi

8. Sorumluluklar

- Hastanın doktor ile tedavi amacı ile buluşması
- Hastanın hastaneden randevu alabilmesi
- Doktorun reçete vermesi

9. Askıya Alma Kriterleri ve Devam Ettirme Gereksinimleri

- Randevu alamama ve sonrasında doktorun uygun olması
- Reçete alamama ve sonrasında sistemsel hatanın düzeltilmesi
- Doktorun yerinde olmaması ve sonrasında randevu tarihinin başka bir tarihe ertelenmesi

10.Takvim

- Test aşamalarının belirlenmesi
- Belirlenen test aşamalarının gruplanması
- Belirlenen test aşamalarının kontrollerinin yapılması
- Test aşamalarının raporlanması

3. Tasarım (Design) iş akışı

Sınıfların iç detaylarının belirlenmesi (veri tip ve formatları, metotlar, metot interfaceleri, değişken vemetotların erişim seviyelerinin belirlenmesi (information hiding), ve her metoda ait pseudo-code verilmesi:

```
class HastaGiris {
private string ad,soyad,kimlik,adres,tel;
public string BilgiGir(){
write "bilgileri girin";
read 'ad' ; read 'soyad' ; read 'kimlik ; read 'tel' ; read 'adres' ;
bilgi = ad +"/"+ soyad +"/"+ kimlik+"/"+ tel+"/"+ adres;
return bilgi;}
}
class HastaGirisDepo{
HastaGiris h1 = new HastaGiris();
static string depo;
public void depola ( ) {
depo=h1.BİlgiGir();
}
}
class PoliklinikBirim{
static PoliklinikBirim a;
static PoliklinikBirim(){
a = new PoliklinikBirim(); }
public static PoliklinikBirim baglantidonder(){
return a; } /* tüm hastalar poliklinik listesine tek nesne üzerinden erişsin, her hasta her poliklinik seçimi için farklı farklı nesneler açıp hafızayı doldurmaması amacıyla yapılmıştır. */
public string polikliniklistelevedonder ( ) {
 veritabanını aktifleştir hastane.polikliniksayisi;
public string[ ] poliklinik  = new string[polikliniksayisi];
```

```
string poliklinikler=” “ ;
for(int i=0;i<length.poliklinik;i++)
 poliklinik[i]=database hastane.poliklinik[i];
for (int i=0; ;i<lenght.poliklinik;i++)
 poliklinikler+=poliklinik[i]+”\n”;
write poliklinikler;
return poliklinikler;
}
```

```
class PoliklinikSecVeGonder {
PoliklinikBirim pb = PoliklinikBirim.baglantidonder();
private int index;
public strimg pSecilen(){
read index;
return pb.poliklinik[index];}
} }
```

```
class HastaMain{
HastaGiris hg = new HastaGiris();
HastaGirisDepo hdepo = new HastaGirisDepo ();
PoliklinikBirim pb = PoliklinikBirim.baglantidonder();
PoliklinikSecVeGonder psvg = new PoliklinikSecVeGonder();
Private void calistir(){
hg.BİlgiGir(); hdepo.depola(); pb.polikliniklistelevedonder(); psvg.pSecilen(); }}
```

```
class TarihVeSaat{
private string gun;
private string ay;
private string yil;
private string saat;
private string zaman;
public void randevuzaman ( ) {
yaz "randevu için  sırasıyla gun,ay,yıl,saat giriniz";
oku gun; oku ay; oku yil; oku saat;
zaman= gun + “/” + ay + “/” + yil + “/” + saat; }

class DoktorListele {
TarihVeSaat ts = new TarihVeSaat ( ) ;
public void listeledoktor(){
veritabanını aktifleştir Poliklinik.Database.Doktorlar;
if( ts.randevuzaman = = Poliklinik.Database.Doktorlar.Zaman)
  for(;doktorzaman = =Poliklinik.Database.Doktorlar.Zaman && doktor = =Poliklinik.Database.Doktorlar;){
   int i = 0; doktorlar[i] = doktor ; i++; }
string doktorlistesi = “ “;
for(int i = 0; i<doktorlar.length;i++) doktorlistesi+= doktorlar[i] + “\n” ;
 else {
yaz “doktor yok”
ts.randevuzaman; listeledoktor(); } }
class DoktorSec {
DoktorListele dl = new DoktorListele();
private int index;
public string secilendoktor(){
yaz “İstenilen doktoru listeden seç”;
yaz dl.doktorlistesi;
oku index;
```

return dl.doktorlar[index];}

}

class RandevuOlustur{

TarihVeSaat ts = new TarihVeSaat ( ) ;

PoliklinikSecVeGonder pg = new PoliklinikSecVeGonder ( ) ;

DoktorSec ds = new DoktorSec( );

private string randevuzamani , doktor; private int poliklinik;

public void randevubilgileriveolustur(){

randevuzamani=ts.randevuzaman();

doktor= ds.secilendoktor();

poliklinik=pg.pSecilen();

convert poliklinik to string;

randevu= randevuzamani + doktor + poliklinik;

yaz " randevu zamanı :" randevuzamani , "randevu doktoru:" doktor, "randevu polikliniği:" poliklinik;

if (randevu = = PoliklinikDatabase)

 yaz "Randevu zaten oluşturulmuş";

else

randevu ekle PoliklinikDatabase;

} }

class RandevuIptal( ){

public void randevusil(){ delete randevu from PoliklinikDatabase;} }

class RandevuMain{

TarihVeSaat ts = new TarihVeSaat ( ) ;

DoktorListele dl = new DoktorListele();

DoktorSec ds = new DoktorSec( );

RandevuOlustur ro = new RandevuOlustur();

RandevuIptal rI = new RandevuIptal();

public void randevuIslemleri(){

ts. randevuzaman();

dl.listeledoktor();

ds. secilendoktor();

ro. randevubilgileriveolustur();

rI. randevusil(); /* Eğer bu işlem istenirse gerçekleştir*/

yaz “Randevu işlemleri başarılı”; } }

class RandevuGunu{

public int zamangeldimi( ){

TarihVeSaat ts = new TarihVeSaat ( );

if(dateTime.now = = ts.randevuzaman)

return 1;

else

return 0; } }

class DoktorTetkik{

RandevuGunu rg = new RandevuGunu();

public void hastatetkikdoldur(){

if(rg.zamangeldimi = = 1){

doldur form.hastatetkik; }

else

yaz “Randevu günü değil”; } }

```
class HastalikBelirleme {
DoktorTetkik dt = new DoktorTetkik ();
RandevuGunu rg = new RandevuGunu();
public void hastalikyaz(){
if(dt.form.hastatetik.hastaliktextbox = = “yok”)
 yaz “hastalik yok”;
else
 if(rg.zamangeldimi = = 1){
  aktifleştir form.dt.hastatetik ve doldur form.hastalik;
  }
 else
  yaz “Randevu günü değil”; }

class ReceteOlustur {
RandevuGunu rg = new RandevuGunu();
public void receteyaz(){
if(rg.zamangeldimi = = 1)
aktifleştir form.hastalik ve doldur form.recete;
else
yaz “Randevu günü değil”; } }
class ReceteCikti{
RandevuGunu rg = new RandevuGunu();
ReceteOlustur ro = new ReceteOlustur();
public void recetecikti(){
if(rg.zamangeldimi = = 1)
aktifleştir form.recete ve yazdır form.recete;
else
yaz “Randevu günü değil”;}}
```

class MuayeneMain{

RandevuGunu rg = new RandevuGunu();

DoktorTetkik dt = new DoktorTetkik();

HastalikBelirleme hb = new HastalikBelirleme();

ReceteOlustur ro = new ReceteOlustur();

ReceteCikti rc = new ReceteCikti ();

public void muayenecalistir(){

rg. zamangeldimi();

dt. hastatetkikdoldur();

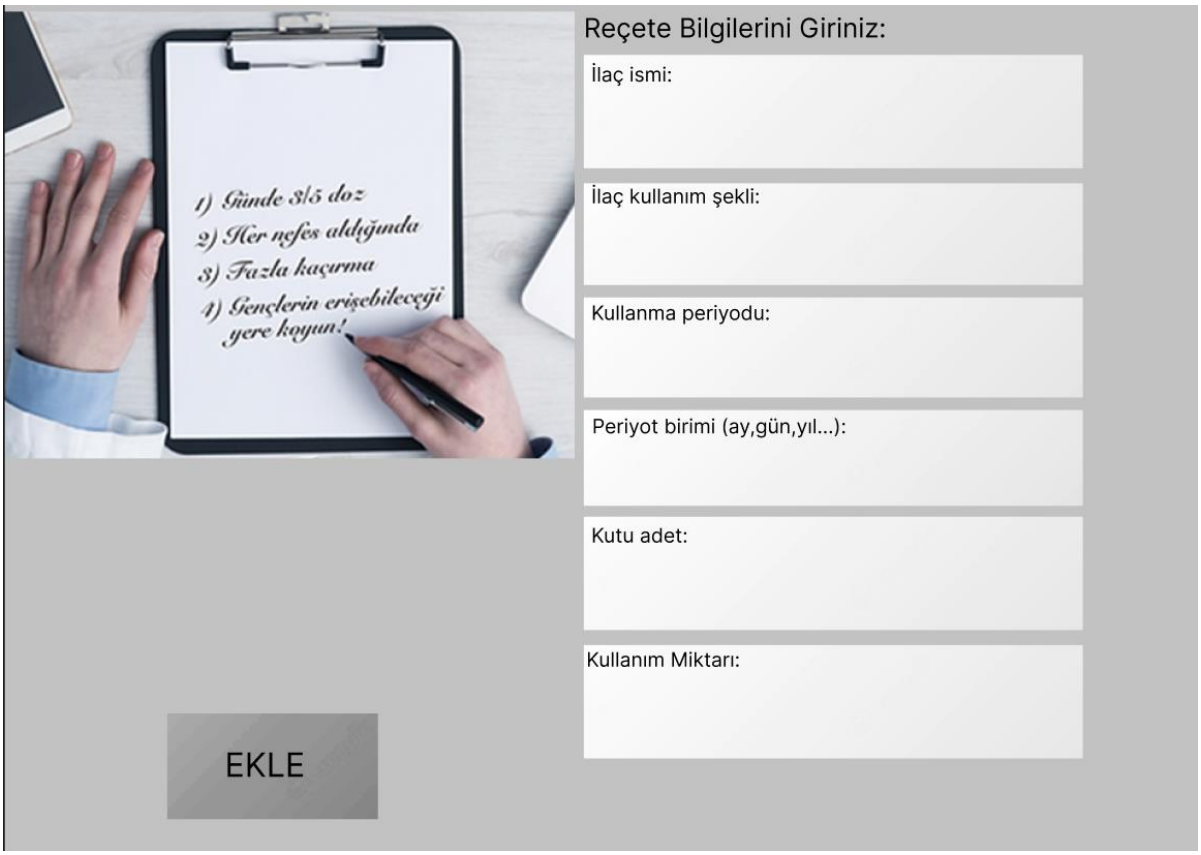
hb. hastalikyaz();

ro. receteyaz();

rc. recetecikti();

yaz “muayene işlemleri gerçekleşti”; } }

Form Arayüzlerinin Tasarımı:



Hasta Tetkik Bilgilerini Giriniz:

Öz Geçmişi:

Soy geçmişi:

Şikayeti:

Hikayesi:

Labaratuar Bulguları:

Fiziki Bulgu:

EKLE

Hastaya ait bulunan hastalığı yazınız:

Karar:

Hasta Tetkik Formuna Dön

Tedavi Öğüt:

EKLE