**T.C. CUMHURİYET ÜNİVERSİTESİ  
MÜHENDİSLİK FAKÜLTESİ  
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**VERİTABANI TASARIM VE UYGULAMA FİNAL RAPORU**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **GrupID** | | *3* | | | |
| **Öğrenciler** | | **No** | | **Ad Soyad** | **Öğretim Türü** |
| *2018141069*  *2018141025*  *2018141003*  *2018141019*  *2018123060* | | *Fatih KAYA*  *Karani KARAMAN*  *Ramazan İŞCANAN*  *Seyit Can BAĞ*  *Esma EL OSMAN* | *İkinci Öğretim*  *İkinci Öğretim*  *İkinci Öğretim*  *İkinci Öğretim*  *Birinci Öğretim* |
| **Projenin Konusu** | | *Hastane Randevu Sistemi* | | | |
| **Gereksinim Analizi** | | * *Sistemimizi 4 kişi yönetebilmeli Hastalar, Başhekim ve Hasta Kayıt elemanı ve Admin.* * *Sistemimizi yönetenlerin genel bilgileri bir tabloda tutulmalı* * *Mesleki bilgileri ve diğer bilgileri ise kendilerine ait tablolarda tutulmalı* * *Sistemimizdeki Kullanıcıların bilgileri: Benzersiz Bir ID ismi(Kullanıcı Id) , TcNo, Telefon Numarası, Doğum Tarihi, Adı Soyadı, Mail Adresi, Şifre ve Cinsiyet Gibi bilgiler içermeli* * *Sistemimizde Doktorlar, Hastaları, Hasta Kayıt Elemanları ve Süper Admin ,*   *Kullanıcı Tablosundan kendilerine ait olan bilgileri alabilmeliler*   * *Sistemimizdeki Admin Tablosunun : Benzersiz Bir ID ismi(Admin Id) , unvanı gibi bilgiler barındırmalı* * *Sistemimizde Doktorlar, Hastalar, Hasta Kayıt Elemanları Admin tarafından yönetilebilmeliler.* * *Sistemimizdeki Doktorların: Benzersiz Bir ID ismi(Doktor Id) , Unvan gibi alanları bulunmalı* * *Bir doktor kendi bölümünden olan birden fazla hastaya sırası ile bakabilmeli* * *Sistemimizde Hastaların: Benzersiz Bir ID ismi(Hasta Id) , Baba Adı, Anne Adı gibi bilgiler olmalı* * *Sistemimizde Hasta Kayıt Elemanların: Benzersiz Bir ID ismi(Hasta Kayıt Elemanı Id) , hastanenin telefon numarası* * *Hasta Kayıttaki eleman birden fazla hastaya randevu alabilmeli* * *Bu 3 farklı kişilerin 3 farklı seviyede yetkileri olmalı* * *Sistemimizde Roller : Benzersiz Bir ID ismi(Rol Id), Rol Name isminde sütunları olmalı ve bunlar kullanıcı genel bilgilerini tutan user tablosu ile ilişkili olmalı , ve yetkileri de ona göre olmalı* * *Sistemimizde Hastanenin Bölümleri olmalı , bu bölümler doktorlar ile ilişkilendirilmeli(yani kalp polikliniğinin doktoruna beyin cerrahisi hasta randevu alamamalı), ve aynı zamanda da her bölümün birden fazlada polikliniği olmalı ve bu poliklinik de hem doktorlar ile hem de departmanı ile ilişkili olmalı* * *Sistemimize de Departman Tablosunda: Benzersiz Bir ID ismi(Departman Id) , Departman Adı bulunmalı* * *Sistemimize de Poliklinik Tablosunda: Benzersiz Bir ID ismi(Poliklinik Id) , Poliklinik Adı bulunmalı* * *Sistemimizde Randevu alanı: Benzersiz Bir ID ismi(Randevu Id), Randevu saati, Randevu günü , Randevu alınma tarihi olmalı* * *Randevuyu kimin kime aldığı (mesela hasta kendisine alabilir yada hasta kayıttaki adam da bir hastaya alabilir.) bilgisi de tutulmalı (yani hasta ve hasta kayıt elemanının mı randevu oluşturduğu bilinmeli)* * *Tedavi sonrası Doktorun Hastaya girdiği ilaçlar görünmeli* * *Sistemimizde Reçete alanı: Benzersiz Bir ID ismi(Reçete Id), Hastalık Adı, ilaç, açıklama şeklinde alanlar olmalı* * *Randevu Kontrol sistemi de olmalı mesela hasta randevu alıp da muayene olmadı ise kara listeye alınmalı ve sistemden belli bir süre randevu verilmemeli* * *Sistemimizde Kara Liste : Benzersiz Bir ID ismi(Kara Liste Id), Hastaneyi meşgul etme sayısı, açıklama şeklinde alanlar olmalı* * *Kara Liste Tablomuz Doktorlar ve Hastalar ile ilişkili olmalı, eğer hasta gelmez ise doktor o hastayı işaretleyebilmeli.* | | | |
| **Kavramsal Tasarım** | | *C:\Users\Kaya\Downloads\E-R Diagram (3).jpg*  Şekil .1.1 Veritabanı E/R Diyagramı  ***Veri Tabanı E/R Diagramı***    *Tablolar*  *User , Rol , Admin , Patient , PatientRegister, Appointment, Doctor,Departman,Policlinic,Prescribe,BlackList Tabloları Bulunmaktadır.*  *İlişkiler*  *User :*   * *User Tablosunda kullanıcıların genel bilgileri bulunmaktadır.* * *Admin , Patient , PatientRegister, Doctor bu tablolar kullanıcıları ifade etmektedir.* * *Bu kullanıcıların Bilgilerini birebir ilişki olarak informations\_of ilişkisi ile yukarıda belirtmiş olduğum kullanıcılar bilgilerine erişilebilmektedir.* * *user\_for ilişkisi ile her kullanıcının rolünü belirlemiş olduk*   *Role :*   * *Sistemimizin Yönetilebilir olması açısından bu sistemi kullanacak kişilerin belli bir yetki ile sınırlandırılması gerekmektedir bunu da Role tablosu ile sağladık.*   *Admin :*   * *Sistemimizde ki tüm kullanıcılar hakkında işlem yapabilen yegane kullanıcıdır.* * *Manages\_of ilişkisi ile Patient, PatientRegister , Doctor tabloları hakkında yönetme, silme, ekleme güncelleme hakkında sahip, Hastane Başhekim yada Hastane Yöneticisi düzeyinde bir kullanıcıdır.*   *Patient :*   * *İsminden de anlaşılacağı gibi hasta tablosudur. Hasta hakkında bilgileri tutar* * *User tablosu ile informations\_of ilişkisine sahiptir. Bu sayede kendisine ait olan bilgileri getirebilmektedir.* * *makes\_an ilişkisi ile Appointments tablosu hasta randevu ilişkisini sağlamaktadır.* * *Count ilişkisi ile BlackList Tablosunun hangi hastayı kara listeye alacağımızı belirleriz.* * *Taken\_by ilişlisi ile Prescribe tablosunun hangi hastanın ilacı olduğunu anlarız.*   *PatientRegister :*   * *Makes\_an ilişkisi ile randevu alamayan hastalara randevu alma işlemini gerçekleştirir.* * *Hasta ile alakalı tüm Crud işlemlerini gerçekleştirir.*   *Doctor :*   * *Doctor tablosu ise hastanın randevu aldığı kişidir* * *Bir departmanda Birden fazla doktor çalışabilir.* * *Bir doktor Bir policlinikde çalışabilir.* * *Doktor writes prescirbe ilişkisi ile doktorun Hastaya yazdığı reçete gelir.* * *Doktor randevuya gelmeyen hastaları yönetmek için blacklist ile counter ilişkisi içindedir.* * *Bir doktor birden fazla hastaya bakabilir, bu yüzden doktor ile randevular arasında 1-N ‘ lik bir ilişki var.*   *Departman(Klinik) :*   * *Bir Departmanda Birden fazla Policlinik olabilir.*   *Policlinik :*   * *Bir policlinik de bir doktor bulunabilir* * *Bir policlinik bir departmana ait olabilir*   *Appointments:*   * *Bir hasta kayıtçı birden fazla kişiye randevu alabilir* * *Bir hasta birden fazla kez kendine randevu alabilir.*   ***\*\*Appointment Açıklama***  *Appointment:Randevu tarihi(Date) ve Randevuya bakacak doktorun id si(DoktorId) beraber kompozit key şeklinde primary key olarak kullanılır.Bunun sebebi şudur bir doktor bir tarih-zaman da ancak bir kişiye bakabilir yani aynı anda ikinci bir hastayı muayeneye alamaz.Yani şöyle bir durum var bir tarih-zamanda iki farklı doktor randevuya sahip olabilir yada iki farklı tarih-zamanda bir doktor randevulara bakabilir ama bir doktor aynı anda iki randevuya giremez yani burada bi tekillik oluşur.Bu tekillik kompozit key mantığıyla kullanıldığında Id niyetine kullanılabilir.*  ***\*\*User Tablosu Açıklama***  *User Tablosu Id Kolonu:User tablosu sistemdeki kullanıcıların hasta,doktor,veznedar olup olmadığına bakmaksızın genel olarak herkeste olacak olan bilgilerin tutulduğu tablo olarak kullanılmaktadır.Ve burada primary key olarak Id kolunu bulunmaktadır ve bu kolon diğer özel tablolara(Hasta,Doktor vs) tablolara bire bir olarak bağlanmaktadır yani bir kullanıcı kaydının bir tane hasta kaydı olabilir.TC numarası ise gene User tablosunda unique olarak başka bir kolonda tutulmaktadır.Bunun sebebi TC numarası bir kişilik bilgisidir tıpkı ad,soyad,doğum tarihi gibi.Bunun diğer özelleştirilmiş tablolarda tutulması mantık olarak doğru değildir.Çünkü o tablolarda kişinin o özel işi neyse (örn:Doktor ise polikliniği vs) onlar tutulmaktadır ancak TC bir kişilik bilgisi olduğundan dolayı diğer tablolarda primary key niyetine kayda geçilmesi doğru değildir.* | | | |
| **Mantıksal Tasarım** | | *İlişkisel şemayı E/R diyagramını normalize etmeden önce basitleştirmemize ve anlamamıza yardımcı olan yoldur, çünkü birincil ve yabancı anahtarları belirtir ve bize üç tür tablo arasındaki ilişkileri gösterir (1’e 1’lik, 1’e n’lik ve n’e n’lik ilişkiler) sonraki adımda tablolardaki niteliklerin arasındaki fonksiyonel bağımlılıkları (functional dependency) bulmamız şarttır. Anomaly’leri ortadan kaldırmak amacıyla ilişkisel şemayı normalleştirme için FD’leri kullanılır. Son olarak normalizasyona dönüştürmek için normalizasyonun 4 formunu kullanarak*  *(\*1.Normal form*  *\*2.Normal form*  *\*3. Normal form*  *\*Boyce-Codd Normal Form )*  *“Kötü” ilişkilerin niteliklerini daha küçük ilişkilere bölüyoruz.*  C:\Users\Kaya\Desktop\rapor dosyalar\Mantıksal Tasarım\İlişkisel şema1.jpg  Şekil 1.1.2 İlişkisel Şema          Şekil 1.1.3 Fonksiyonel Şema      Şekil 1.1.4 Normalizasyon | | | |
| **Fiziksel Tasarım** | | --CREATE DATABASE [HospitalAppDb]  USE [HospitalAppDb]  GO  /\*\*\*\*\*\* Departman Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[Department](  [Id] [int] IDENTITY(1,1) NOT NULL,  [DepartmanName] [nvarchar](50) NOT NULL,  CONSTRAINT [PK\_Department] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Policlinic Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[Policlinic](  [Id] [int] NOT NULL,  [PoliclinicName] [nvarchar](50) NOT NULL,  [DepartmentId] [int] NOT NULL,  CONSTRAINT [PK\_Policlinic] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Randevu Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[Appointment](  [Date] [smalldatetime] NOT NULL,  [RegistrarId] [int] NULL,  [DoctorId] [int] NOT NULL,  [PatientId] [int] NOT NULL,  [Confirmed] [bit] NOT NULL,  [CreatingDate] [smalldatetime] NOT NULL,  CONSTRAINT [PK\_Appointment] PRIMARY KEY CLUSTERED  (  [Date] ASC,  [DoctorId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Karaliste Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[BlackList](  [Id] [int] IDENTITY(1,1) NOT NULL,  [DeceptionCount] [smalldatetime] NOT NULL,  [DoctorId] [int] NOT NULL,  [PatientId] [int] NOT NULL,  [Description] [nvarchar](500) NOT NULL,  CONSTRAINT [PK\_BlackList] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Doktor Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[Doctor](  [Id] [int] NOT NULL,  [DepartmentId] [int] NOT NULL,  [Apellation] [nvarchar](50) NOT NULL,  [SuperAdminId] [int] NOT NULL,  CONSTRAINT [PK\_Doctor] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Hasta Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[Patient](  [Id] [int] NOT NULL,  [MotherName] [nvarchar](50) NOT NULL,  [FatherName] [nvarchar](50) NOT NULL,  [SuperAdminId] [int] NOT NULL,  CONSTRAINT [PK\_Patient] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Veznedar Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[PatientRegistrar](  [Id] [int] NOT NULL,  [TellerNumber] [int] NOT NULL,  [SuperAdminId] [int] NOT NULL,  CONSTRAINT [PK\_PatientRegistrar] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Reçete Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[Prescribe](  [Id] [int] IDENTITY(1,1) NOT NULL,  [DiseaseName] [nvarchar](50) NOT NULL,  [Medicine] [nvarchar](50) NOT NULL,  [DoctorId] [int] NOT NULL,  [PatientId] [int] NOT NULL,  [Description] [nvarchar](500) NOT NULL,  CONSTRAINT [PK\_Prescribe] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Rol Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[Rol](  [Id] [int] IDENTITY(1,1) NOT NULL,  [RolName] [nvarchar](50) NOT NULL,  CONSTRAINT [PK\_Rol] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Admin Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[SuperAdmin](  [Id] [int] NOT NULL,  [Apellation] [nvarchar](50) NOT NULL,  CONSTRAINT [PK\_SuperAdmin] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Genel Kullanıcı Bilgilerinin Olduğu User Tablosunu Oluşturuyoruz \*\*\*\*\*\*/  CREATE TABLE [dbo].[User](  [Id] [int] IDENTITY(1,1) NOT NULL,  [TCNo] [bigint] NOT NULL,  [RolId] [int] NOT NULL,  [Name] [nvarchar](50) NOT NULL,  [SurName] [nvarchar](50) NOT NULL,  [Email] [nvarchar](50) NOT NULL,  [BirthDate] [smalldatetime] NOT NULL,  [Telephone] [bigint] NOT NULL,  [Gender] [bit] NOT NULL,  [Password] [nvarchar](50) NOT NULL,  CONSTRAINT [PK\_User] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Departmanları Policlinikler ile Getiren bir View Oluşturuyoruz \*\*\*\*\*\*/  CREATE VIEW [dbo].[View\_GetWithPol]  AS  SELECT dbo.Department.Id AS DepId, dbo.Department.DepartmanName, dbo.Policlinic.Id AS PolId, dbo.Policlinic.PoliclinicName  FROM dbo.Department INNER JOIN  dbo.Policlinic ON dbo.Department.Id = dbo.Policlinic.DepartmentId  GO  /\*\*\*\*\*\* Tablolar için Gerekli olan CheckConstrait,DefaultValue vb tanımlamaları yapıyoruz \*\*\*\*\*\*/  ALTER TABLE [dbo].[BlackList] ADD CONSTRAINT [DF\_BlackList\_DoctorId] DEFAULT ((-1)) FOR [DoctorId]  GO  ALTER TABLE [dbo].[Appointment] WITH NOCHECK ADD CONSTRAINT [FK\_Appointment\_Doctor] FOREIGN KEY([DoctorId])  REFERENCES [dbo].[Doctor] ([Id])  GO  ALTER TABLE [dbo].[Appointment] NOCHECK CONSTRAINT [FK\_Appointment\_Doctor]  GO  ALTER TABLE [dbo].[Appointment] WITH CHECK ADD CONSTRAINT [FK\_Appointment\_Patient] FOREIGN KEY([PatientId])  REFERENCES [dbo].[Patient] ([Id])  ON DELETE CASCADE  GO  ALTER TABLE [dbo].[Appointment] CHECK CONSTRAINT [FK\_Appointment\_Patient]  GO  ALTER TABLE [dbo].[Appointment] WITH NOCHECK ADD CONSTRAINT [FK\_Appointment\_PatientRegistrar] FOREIGN KEY([RegistrarId])  REFERENCES [dbo].[PatientRegistrar] ([Id])  GO  ALTER TABLE [dbo].[Appointment] NOCHECK CONSTRAINT [FK\_Appointment\_PatientRegistrar]  GO  ALTER TABLE [dbo].[BlackList] WITH NOCHECK ADD CONSTRAINT [FK\_BlackList\_Doctor] FOREIGN KEY([DoctorId])  REFERENCES [dbo].[Doctor] ([Id])  GO  ALTER TABLE [dbo].[BlackList] NOCHECK CONSTRAINT [FK\_BlackList\_Doctor]  GO  ALTER TABLE [dbo].[BlackList] WITH CHECK ADD CONSTRAINT [FK\_BlackList\_Patient] FOREIGN KEY([PatientId])  REFERENCES [dbo].[Patient] ([Id])  ON DELETE CASCADE  GO  ALTER TABLE [dbo].[BlackList] CHECK CONSTRAINT [FK\_BlackList\_Patient]  GO  ALTER TABLE [dbo].[Doctor] WITH CHECK ADD CONSTRAINT [FK\_Doctor\_Department] FOREIGN KEY([DepartmentId])  REFERENCES [dbo].[Department] ([Id])  ON DELETE CASCADE  GO  ALTER TABLE [dbo].[Doctor] CHECK CONSTRAINT [FK\_Doctor\_Department]  GO  ALTER TABLE [dbo].[Doctor] WITH CHECK ADD CONSTRAINT [FK\_Doctor\_SuperAdmin] FOREIGN KEY([SuperAdminId])  REFERENCES [dbo].[SuperAdmin] ([Id])  GO  ALTER TABLE [dbo].[Doctor] CHECK CONSTRAINT [FK\_Doctor\_SuperAdmin]  GO  ALTER TABLE [dbo].[Doctor] WITH CHECK ADD CONSTRAINT [FK\_Doctor\_User1] FOREIGN KEY([Id])  REFERENCES [dbo].[User] ([Id])  ON DELETE CASCADE  GO  ALTER TABLE [dbo].[Doctor] CHECK CONSTRAINT [FK\_Doctor\_User1]  GO  ALTER TABLE [dbo].[Patient] WITH CHECK ADD CONSTRAINT [FK\_Patient\_SuperAdmin] FOREIGN KEY([SuperAdminId])  REFERENCES [dbo].[SuperAdmin] ([Id])  GO  ALTER TABLE [dbo].[Patient] CHECK CONSTRAINT [FK\_Patient\_SuperAdmin]  GO  ALTER TABLE [dbo].[Patient] WITH CHECK ADD CONSTRAINT [FK\_Patient\_User1] FOREIGN KEY([Id])  REFERENCES [dbo].[User] ([Id])  ON DELETE CASCADE  GO  ALTER TABLE [dbo].[Patient] CHECK CONSTRAINT [FK\_Patient\_User1]  GO  ALTER TABLE [dbo].[PatientRegistrar] WITH CHECK ADD CONSTRAINT [FK\_PatientRegistrar\_SuperAdmin] FOREIGN KEY([SuperAdminId])  REFERENCES [dbo].[SuperAdmin] ([Id])  GO  ALTER TABLE [dbo].[PatientRegistrar] CHECK CONSTRAINT [FK\_PatientRegistrar\_SuperAdmin]  GO  ALTER TABLE [dbo].[PatientRegistrar] WITH CHECK ADD CONSTRAINT [FK\_PatientRegistrar\_User1] FOREIGN KEY([Id])  REFERENCES [dbo].[User] ([Id])  ON DELETE CASCADE  GO  ALTER TABLE [dbo].[PatientRegistrar] CHECK CONSTRAINT [FK\_PatientRegistrar\_User1]  GO  ALTER TABLE [dbo].[Policlinic] WITH CHECK ADD CONSTRAINT [FK\_Policlinic\_Department] FOREIGN KEY([DepartmentId])  REFERENCES [dbo].[Department] ([Id])  ON DELETE CASCADE  GO  ALTER TABLE [dbo].[Policlinic] CHECK CONSTRAINT [FK\_Policlinic\_Department]  GO  ALTER TABLE [dbo].[Policlinic] WITH CHECK ADD CONSTRAINT [FK\_Policlinic\_Doctor] FOREIGN KEY([Id])  REFERENCES [dbo].[Doctor] ([Id])  GO  ALTER TABLE [dbo].[Policlinic] CHECK CONSTRAINT [FK\_Policlinic\_Doctor]  GO  ALTER TABLE [dbo].[Prescribe] WITH NOCHECK ADD CONSTRAINT [FK\_Prescribe\_Doctor] FOREIGN KEY([DoctorId])  REFERENCES [dbo].[Doctor] ([Id])  GO  ALTER TABLE [dbo].[Prescribe] NOCHECK CONSTRAINT [FK\_Prescribe\_Doctor]  GO  ALTER TABLE [dbo].[Prescribe] WITH CHECK ADD CONSTRAINT [FK\_Prescribe\_Patient] FOREIGN KEY([PatientId])  REFERENCES [dbo].[Patient] ([Id])  ON DELETE CASCADE  GO  ALTER TABLE [dbo].[Prescribe] CHECK CONSTRAINT [FK\_Prescribe\_Patient]  GO  ALTER TABLE [dbo].[SuperAdmin] WITH CHECK ADD CONSTRAINT [FK\_SuperAdmin\_User1] FOREIGN KEY([Id])  REFERENCES [dbo].[User] ([Id])  ON DELETE CASCADE  GO  ALTER TABLE [dbo].[SuperAdmin] CHECK CONSTRAINT [FK\_SuperAdmin\_User1]  GO  ALTER TABLE [dbo].[User] WITH CHECK ADD CONSTRAINT [FK\_User\_Rol] FOREIGN KEY([RolId])  REFERENCES [dbo].[Rol] ([Id])  GO  ALTER TABLE [dbo].[User] CHECK CONSTRAINT [FK\_User\_Rol]  GO  /\*\*\*\*\*\* Hastanın Karaliste kaydının olup olmadığını sorgulayan bir procedure yazıyoruz \*\*\*\*\*\*/  CREATE PROCEDURE [dbo].[sp\_CheckInBlackListByPatientId]  @patientId int  AS  BEGIN  select top 1 \* from BlackList where PatientId = @patientId and DeceptionCount>GETDATE()  END  GO  /\*\*\*\*\*\* Yeni bir karaliste tanımlandığı zaman aynı hasta için eski karalisteyi silen bir procedure yazıyoruz \*\*\*\*\*\*/  CREATE TRIGGER [dbo].[previousBlacklistDelete]  ON [dbo].[BlackList]  AFTER INSERT  AS  BEGIN  declare @patientId int  select @patientId = PatientId from inserted  delete from BlackList where DeceptionCount<GETDATE() and PatientId=@patientId  END  GO  ALTER TABLE [dbo].[BlackList] ENABLE TRIGGER [previousBlacklistDelete]  GO  /\*\*\*\*\*\* Bir doktor silindiği zaman üzerinde id sinin bulunduğu karaliste reçete ve geçmişteki randevulara id sinin yerine 0 basan trigger Not:Program akışı kesilmeyecek şekilde tasarlandı \*\*\*\*\*\*/  CREATE trigger [dbo].[doctor\_Delete]  on [dbo].[Doctor]  after delete as  begin  declare @Id int  select  @Id = Id from deleted  update BlackList set DoctorId = 0 where DoctorId=@Id  update Prescribe set DoctorId = 0 where DoctorId=@Id  update Appointment set DoctorId = 0 where DoctorId=@Id  End  GO  ALTER TABLE [dbo].[Doctor] ENABLE TRIGGER [doctor\_Delete]  GO  /\*\*\*\*\*\* Bir veznedar silindiği zaman veznedarın hastalar için almış olduğu randevular kısmına veznedar id si yerine null basan trigger Not:Program akışı kesilmeyecek şekilde tasarlandı\*\*\*\*\*\*/  create trigger [dbo].[patientRegistrar\_Delete]  on [dbo].[PatientRegistrar]  after delete as  begin  declare @Id int  select  @Id = Id from deleted  update Appointment set RegistrarId = null where RegistrarId=@Id  End  GO  ALTER TABLE [dbo].[PatientRegistrar] ENABLE TRIGGER [patientRegistrar\_Delete]  GO  HospiDiagram  Şekil 1.2 Veritabanı Diyagram | | | |
| **Özel Sorgular** | | ***View***  ***Açıklama:*** *Departmanlar ve Departmanlara ait Poliklinikleri beraber döndüren bir View alt tarafta ise view’ın c# tarafında entityFramework aracılığı ile çağrılışı var*  *SELECT dbo.Department.Id AS DepId, dbo.Department.DepartmanName, dbo.Policlinic.Id AS PolId, dbo.Policlinic.PoliclinicName*  *FROM dbo.Department INNER JOIN*  *dbo.Policlinic ON dbo.Department.Id = dbo.Policlinic.DepartmentId*  viewef1  viewef2  viewef3  ***Trigger***  ***Açıklama:*** *Yeni bir karaliste kaydı eklenince eski karaliste kaydını otomatik olarak silen bir trigger*  ALTER TRIGGER [dbo].[previousBlacklistDelete]  ON [dbo].[BlackList]  AFTER INSERT  AS  BEGIN  declare @patientId int  select @patientId = PatientId from inserted  delete from BlackList where DeceptionCount<GETDATE() and PatientId=@patientId  END  ***StoredProcedure***  ***Açıklama:*** *Hastanın herhangi bir karaliste kaydı olup olmadığını kontrol eden bir procedure ve alt tarafta ise procedure’ün c# tarafında entityFramework aracılığı ile çağrılışı var.*  ALTER PROCEDURE [dbo].[sp\_CheckInBlackListByPatientId]  @patientId int  AS  BEGIN  select top 1 \* from BlackList where PatientId = @patientId and DeceptionCount>GETDATE()  END  stored | | | |
| **Arayüz Tasarımı ve Ekran Görüntüleri** | | *Projenin Arayüz tasarımında Asp.Net Core Mvc 3.1 Teknoloji kullanılmıştır, Yazılım dilinde C# kullanılmıştır ve Bootstrap ön yüz çatısı ile desteklenmiştir.*  *Proje; Hasta paneli, Veznedar(Hasta Kayıt Personeli) paneli, Doktor paneli ve Admin paneli olmak üzere 4 kısımdan oluşmaktadır.*  *Hasta Paneli; Yeni randevu alma, geçmiş randevular, gelecek randevular, reçete bilgileri ve hastanın profil bilgisi sayfaları mevcuttur.*  *Veznedar paneli; Yeni hasta ekleme, hasta güncelleme, Hastaya randevu verme, Verdiği randevular ve profil bilgisi sayfaları mevcuttur.*  *Doktor Paneli; Gün içi randevu listesi ve profil bilgileri mevcuttur.*  *Admin Paneli; Doktorlar(ekleme,güncelleme ve silme) sayfası, Hastalar(ekleme,güncelleme ve silme) sayfası, Hastalar(ekleme,güncelleme ve silme) sayfası, Departmanlar(ekleme,güncelleme ve silme) sayfaları ve profil bilgileri vardır.*  ***1.Panel Girişleri***  *C:\Users\Kaya\Desktop\rapor dosyalar\img\İndex Panel\1.panel.png*  Şekil 1.1 Panel Girişleri  C:\Users\Kaya\Desktop\rapor dosyalar\img\İndex Panel\hasta_giris.png  Şekil 1.2 Hasta Giriş Ekranı  *C:\Users\Kaya\Desktop\rapor dosyalar\img\İndex Panel\hasta_giris_uye.png*  Şekil 1.3 Hasta Üyelik Oluşturma Ekranı  *C:\Users\Kaya\Desktop\rapor dosyalar\img\İndex Panel\veznedar_giris.png*  Şekil 1.4 Veznedar Giriş Ekranı  C:\Users\Kaya\Desktop\rapor dosyalar\img\İndex Panel\doktor_giris.png  Şekil 1.5 Doktor Giriş Ekranı  *C:\Users\Kaya\Desktop\rapor dosyalar\img\İndex Panel\admin giris.png*  Şekil 1.6 Admin Giriş Ekranı  ***2.Hasta Paneli***  *C:\Users\Kaya\Desktop\rapor dosyalar\img\Hasta\hasta_randevu1.png*  Şekil 2.1 Hasta Yeni Randevu Alma Ekranı (Bölüm seçme)  C:\Users\Kaya\Desktop\rapor dosyalar\img\Hasta\hasta_randevu2.png  Şekil 2.2 Hasta Yeni Randevu Alma Ekranı (Doktor seçme)    Şekil 2.3 Hasta Yeni Randevu Alma Ekranı (Tarih seçme)  C:\Users\Kaya\Desktop\rapor dosyalar\img\Hasta\hasta_randevu4.png  Şekil 2.4 Hasta Yeni Randevu Alma Ekranı (Saat seçme)  *C:\Users\Kaya\Desktop\rapor dosyalar\img\Hasta\hasta_randevu5.png*  Şekil 2.5 Hasta Yeni Randevu Alma Ekranı (Kaydetme)  C:\Users\Kaya\Desktop\rapor dosyalar\img\Hasta\hasta_gecmis.png  Şekil 2.6 Geçmiş Randevular  C:\Users\Kaya\Desktop\rapor dosyalar\img\Hasta\hasta_gelecek.png  Şekil 2.7 Gelecek Randevular  C:\Users\Kaya\Desktop\rapor dosyalar\img\Hasta\hasta_recete.png  Şekil 2.8 Hastanın Reçeteleri  C:\Users\Kaya\Desktop\rapor dosyalar\img\Hasta\hasta_profile.png  Şekil 2.9 Hastanın Profili  *\*\*hastanın Profil ekranında herhangi bir değişiklik yapılamaz, sadece bilgileri gözükür. Güncelleme Admin ve veznedar tarafından yapılabilir.*  ***3.Veznedar(Hasta Kayıt Personeli)***  ***C:\Users\Kaya\Desktop\rapor dosyalar\img\Veznedar\veznedar_hastaekleme.png***  Şekil .1 Veznedar, Hasta Ekleme Sayfası  C:\Users\Kaya\Desktop\rapor dosyalar\img\Veznedar\veznedar_hastaguncelleme.png  Şekil 3.2 Hasta Bilgileri Güncelleme (T.C no giriş)  C:\Users\Kaya\Desktop\rapor dosyalar\img\Veznedar\veznedar_hastaguncelleme2.png  Şekil 3.3 Hasta Bilgileri Güncelleme (Kaydet)    Şekil 3.4 Veznedar, Hastaya Yeni Randevu Verme Sayfası  C:\Users\Kaya\Desktop\rapor dosyalar\img\Veznedar\veznedar_hastarandevu2.png  Şekil 3.5 Veznedar, Hastaya Yeni Randevu Verme Sayfası (Tarih Seçme)  \*\* Hasta randevu alma sayfasında T.C Kimlik no girişinden sonra 2 adım atlanmıştır. Sunum esnasında tüm adımlar detaylı anlatılacak.  C:\Users\Kaya\Desktop\rapor dosyalar\img\Veznedar\veznedar_hastarandevu3.png  Şekil 3.6 Veznedar, Hastaya Yeni Randevu Verme Sayfası (Saat seçme)  C:\Users\Kaya\Desktop\rapor dosyalar\img\Veznedar\veznedar_hastarandevu4.png  Şekil 3.7 Veznedar, Hastaya Yeni Randevu Verme Sayfası (Kaydetme)  C:\Users\Kaya\Desktop\rapor dosyalar\img\Veznedar\veznedar_hastarandevu5.png  Şekil 3.8 Veznedarın Verdiği Randevu Listesi  C:\Users\Kaya\Desktop\rapor dosyalar\img\Veznedar\veznedar_profile.png  Şekil 3.9 Veznedar Profil Bilgileri  ***4.Doktor Paneli***  ***C:\Users\Kaya\Desktop\rapor dosyalar\img\Doktor\doktor_randevular.png***  Şekil .1 Doktor, Gün İçi Randevu Listesi  C:\Users\Kaya\Desktop\rapor dosyalar\img\Doktor\doktor_randevu_onaylama.png  Şekil 4.2 Hasta Randevu onaylama (Reçete yazma) Ekranı  C:\Users\Kaya\Desktop\rapor dosyalar\img\Doktor\doktor_randevu_silme.png  Şekil 4.3 Hastanın Randevusunu Silme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Doktor\doktor_profile.png  Şekil 4.4 Doktor Profil Bilgileri  ***5.Admin Paneli***  ***C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_doktorlar.png***  Şekil .1 Doktorlar Sayfası  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_doktor_ekleme.png  Şekil 5.2 Yeni Doktor Ekleme Sayfası  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_doktor_update.png  Şekil 5.3 Doktor Güncelleme Sayfası  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_doktor_delete.png  Şekil 5.4 Doktor Silme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_hastalar.png  Şekil 5.5 Hastalarım Sayfası  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_hasta_ekleme.jpg  Şekil 5. Yeni Hasta Ekleme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_hasta_update.png  Şekil 5. Hasta Güncelleme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_hasta_delete.png  Şekil 5. Hasta Silme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_veznedar.png  Şekil 5. Veznedarlar Sayfası  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_veznedar_ekleme.png  Şekil 5. Yeni Veznedar Ekleme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_veznedar_update.png  Şekil 5. Veznedar Güncelleme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_veznedar_delete.png  Şekil 5. Veznedar Silme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_departman.png  Şekil 5. Hastane Departmanları (Klinikler)  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_departman_ekleme.png  Şekil 5. Yeni Departman Ekleme    Şekil 5. Departman Güncelleme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_departman_delete.png  Şekil 5. Departman Silme  C:\Users\Kaya\Desktop\rapor dosyalar\img\Super Admin\admin_profili.png  Şekil 5. Admin Profil Bilgileri  *\*\*Program üzerinde admin üzerinde herhangi bir değişiklik yapılamaz. Admin 1 kişidir, gerekli görülen ekleme ve güncellemeler veritabanı üzerinden yapılır.* | | | |
| **Grup Üyelerinin Projeye olan Katkısı** | | *Fatih KAYA %100 Performans*  *Karani KARAMAN %100 Performans*  *Ramazan İŞCANAN %100 Performans*  *Seyit Can BAĞ %100 Performans*  *Esma EL OSMAN %100 Performans* | | | |