Veritabanı Model Analizi ve Normalizasyon Raporu

APPROVAL

```
using ilterisg.Models.Enums;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Approvals", Schema = "dbo")]
    public class Approval
    {
        [Key]
        public int Id { get; set; }

        [Required]
        public string RecordId { get; set; } // Onaylanacak kaydın ID'si

        [Required]
        [StringLength(100)]
        public string TableName { get; set; } // Hangi tabloya ait?

        [StringLength(100)]
        public string? ApproverRole { get; set; }

        [StringLength(450)]
        [ForeignKey("ApprovedByUser")]
        public string? ApprovedByUserId { get; set; }

        [Required]
        public ApprovalStatus Status { get; set; } = ApprovalStatus.Pending;

        [Required]
        public ApprovalType ApprovalType { get; set; } // Yeni eklenen alan: Hesap mı, ödeme mi?

        public DateTime? ApprovedDate { get; set; }
```

```
        public string? Notes { get; set; }

        // Navigation Property
        public virtual ApplicationUser? ApprovedByUser { get; set; }
    }
}
```

Yapay Zeka Notu:Model iyi yapılandırılmış iyileştirme olarak

```
public virtual ApplicationUser? ApprovedByUser { get; set; }
public virtual ApplicationUser? CreatedByUser { get; set; }
```

bu alanlar eklenebilir gereksiz sütun yok denildi.

BİLDİRİMLER

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Bildirimler", Schema = "dbo")]
    public class Bildirimler
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int BildirimId { get; set; }

        [Required]
        [MaxLength(450)]
        public string RefUserId { get; set; } // Bildirimin gönderildiği kullanıcı (ApplicationUser.Id)

        [ForeignKey(nameof(RefUserId))]
        public virtual ApplicationUser ApplicationUser { get; set; }
```

```
        [Required]
        [MaxLength(500)]
          public string Mesaj { get; set; } // Bildirim mesajı (örneğin, "Toplantı X 15 Nisan 2025 tarihinde oluşturuldu!")

        public DateTime GonderimTarihi { get; set; } = DateTime.Now;

        public bool OkunduMu { get; set; } = false; // Okundu/Okunmadı durumu

        [MaxLength(100)]
        public string BildirimTuru { get; set; } // Örneğin: "Etkinlik Oluşturma", "Etkinlik Silme"

        public int? RefEgitimId { get; set; } // İlgili eğitim (opsiyonel, eğitimle ilgiliyse)

        [ForeignKey(nameof(RefEgitimId))]
        public virtual Egitimler? Egitim { get; set; }

        public int? RefEtkinlikId { get; set; } // İlgili etkinlik (opsiyonel, etkinlik ile ilgiliyse)

        [ForeignKey(nameof(RefEtkinlikId))]
        public virtual Etkinlikler? Etkinlik { get; set; }

        [MaxLength(500)] // URL uzunluğu için uygun bir sınır
        public string? RedirectUrl { get; set; } // Yönlendirme URL'si
    }
}
```

ReftableName string değil enum yapmak ve her reftable için bir enum belirlemek stringden daha güvenli olabilir, ama daha uğraştırıcı olur diye öneri geldi

```
        [Required]
        [MaxLength(450)]
          public string RefUserId { get; set; } // Bildirimin gönderildiği kullanıcı (ApplicationUser.Id) bildirimi alan kişi için ilgili alan var ama bildirimi gönderen kişi için yok

// Opsiyonel audit eklentisi
```

```
        [MaxLength(450)]
        public string? CreatedByUserId { get; set; } // Kim gönderdi?
        [ForeignKey(nameof(CreatedByUserId))]
        public virtual ApplicationUser? CreatedByUser { get; set; }
```

Gereksiz sütun bulunamadı.

BLOGPOST

```
using Microsoft.AspNetCore.Mvc.ModelBinding;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    public class BlogPost
    {
        [Key]
        public int Id { get; set; }

        [Required]
        [MaxLength(200)]
        public string Title { get; set; }

        [Required]
        public string Content { get; set; }

        public DateTime CreatedAt { get; set; } = DateTime.Now;

        public bool IsPublished { get; set; } = true;

        //  Görsel için
        [MaxLength(500)]
```

```
        public string? ImageUrl { get; set; }

        //  Özet (opsiyonel ama önerilir)
        [MaxLength(500)]
        public string? Summary { get; set; }

        //  Popülerlik takibi için
        public int ViewCount { get; set; } = 0;

        //  Kullanıcı bilgileri
        [BindNever]
        [MaxLength(450)]
        public string AuthorUserId { get; set; }

        [BindNever]
        [ForeignKey(nameof(AuthorUserId))]
        public virtual ApplicationUser Author { get; set; }
    }
}
```

%90 iyi yapılandırılmış ve sorunsuz sadece IsPublished alanı boolean değilde Enum olarak yapılandırılsaydı arşivlendi, yayınlandı, kaldırıldı gibi çoklu işlem opsiyonları olabilirdi. Güncellemeler için UpdatedUserId ve UpdateDate gibi alanlar eklenebilir.

DestekFeedback

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using ilterisg.Models.Enums;

namespace ilterisg.Models
{
    [Table("DestekTalepleri", Schema = "dbo")]
```

```
public class DestekTalepleri
{
    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    public int TalepId { get; set; }

    [Required]
    [MaxLength(200)]
    public string KonuBasligi { get; set; }

    [Required]
    public string Icerik { get; set; }

    [Required]
    public int RefKullaniciId { get; set; }

    [ForeignKey(nameof(RefKullaniciId))]
    public virtual Kullanicilar Kullanici { get; set; }

    public int? RefFirmaId { get; set; } // Nullable, firma opsiyonel

    [ForeignKey(nameof(RefFirmaId))]
    public virtual Firmalar? Firma { get; set; }

    [MaxLength(450)] // AspNetUsers.Id ile eşleşmesi için
    public string? CozumuYapanId { get; set; } // Nullable, çözüm yapan admin (ApplicationUser.Id)

    [ForeignKey(nameof(CozumuYapanId))]
    public virtual ApplicationUser? CozumuYapan { get; set; }

    [MaxLength(450)] // AspNetUsers.Id ile eşleşmesi için
        public string? SahiplenenAdminId { get; set; } // Nullable, talebi sahiplenen admin
(ApplicationUser.Id)

    [ForeignKey(nameof(SahiplenenAdminId))]
    public virtual ApplicationUser? SahiplenenAdmin { get; set; }
```

```
        [Required]
        public DateTime OlusturmaTarihi { get; set; } = DateTime.Now;

        public DateTime? GuncellemeTarihi { get; set; } // Nullable, güncelleme opsiyonel

        [Required]
        public DestekTalebiDurumu Durum { get; set; } = DestekTalebiDurumu.Acik;

        [Required]
        public OnemDurumu OnemDurumu { get; set; } = OnemDurumu.Orta;

        [Required]
        public KonuTuru KonuTuru { get; set; }

        public virtual ICollection<DestekMesajlari> Mesajlar { get; set; } = new List<DestekMesajlari>();

        public virtual ICollection<Dokumanlar> Dokumanlar { get; set; } = new List<Dokumanlar>();
        public virtual DestekFeedback Feedback { get; set; } // Tek bir feedback varsayımı
    }
}
```

Sadeleştirme Önerisi: RefKullaniciId ve RefFirmaId'i kaldır – bunlar transitive (Talep üzerinden erişilebilir). Bu, 3NF'ye tam uyum sağlar (non-key attributes sadece PK'ye dependent olur). Sorgularda: feedback.Talep.Kullanici diye eriş.
Tarih: DateTime.Now yerine DateTime.UtcNow kullan – timezone sorunlarını önler.
Yildiz: Int yeterli, ama enum yap (e.g., Rating enum 1-5) veri tutarlılığı için – opsiyonel.
Audit Genişletme: Eğer feedback editlenecekse UpdatedAt/Yorum ekle. Index'ler migration'da: (RefTalepId, Tarih DESC) – talep bazlı son feedback'ler için hızlı.
Genişletme: Eğer fotoğraflı feedback varsa ImageUrl ekle, ama mevcut haliyle minimal iyi.

DestekTalepler tablosundan Erişim yapılabilir RefKullanıcıId ve RefFirmaId'ye
Önerilen Yapı;

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("DestekFeedback", Schema = "dbo")]
    public class DestekFeedback
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int FeedbackId { get; set; }

        [Required]
        public int RefTalepId { get; set; }
        [ForeignKey(nameof(RefTalepId))]
        public virtual DestekTalepleri Talep { get; set; } = null!; // Talep üzerinden Kullanici/Firma eriş

        [MaxLength(450)]
        public string? CozumuYapanId { get; set; } // Nullable, çözüm yapan admin
        [ForeignKey(nameof(CozumuYapanId))]
        public virtual ApplicationUser? CozumuYapan { get; set; }

        [Required]
        [Range(1, 5)]
        public int Yildiz { get; set; } // Veya enum yap

        [MaxLength(500)]
        public string? Yorum { get; set; } // Opsiyonel yorum

        [Required]
        public DateTime Tarih { get; set; } = DateTime.UtcNow; // UtcNow'a çevir

        // Opsiyonel audit eklentisi (eğer edit varsa)
        public DateTime? UpdatedAt { get; set; }
    }
}
```

DESTEKTALEPLERİ

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using ilterisg.Models.Enums;

namespace ilterisg.Models
{
    [Table("DestekTalepleri", Schema = "dbo")]
    public class DestekTalepleri
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int TalepId { get; set; }

        [Required]
        [MaxLength(200)]
        public string KonuBasligi { get; set; }

        [Required]
        public string Icerik { get; set; }

        [Required]
        public int RefKullaniciId { get; set; }

        [ForeignKey(nameof(RefKullaniciId))]
        public virtual Kullanicilar Kullanici { get; set; }

        public int? RefFirmaId { get; set; } // Nullable, firma opsiyonel

        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar? Firma { get; set; }
```

```
        [MaxLength(450)] // AspNetUsers.Id ile eşleşmesi için
        public string? CozumuYapanId { get; set; } // Nullable, çözüm yapan admin (ApplicationUser.Id)

        [ForeignKey(nameof(CozumuYapanId))]
        public virtual ApplicationUser? CozumuYapan { get; set; }

        [MaxLength(450)] // AspNetUsers.Id ile eşleşmesi için
                public string? SahiplenenAdminId { get; set; } // Nullable, talebi sahiplenen admin
(ApplicationUser.Id)

        [ForeignKey(nameof(SahiplenenAdminId))]
        public virtual ApplicationUser? SahiplenenAdmin { get; set; }

        [Required]
        public DateTime OlusturmaTarihi { get; set; } = DateTime.Now;

        public DateTime? GuncellemeTarihi { get; set; } // Nullable, güncelleme opsiyonel

        [Required]
        public DestekTalebiDurumu Durum { get; set; } = DestekTalebiDurumu.Acik;

        [Required]
        public OnemDurumu OnemDurumu { get; set; } = OnemDurumu.Orta;

        [Required]
        public KonuTuru KonuTuru { get; set; }

        public virtual ICollection<DestekMesajlari> Mesajlar { get; set; } = new List<DestekMesajlari>();

        public virtual ICollection<Dokumanlar> Dokumanlar { get; set; } = new List<Dokumanlar>();
        public virtual DestekFeedback Feedback { get; set; } // Tek bir feedback varsayımı
    }
}
```

Tarihler: DateTime.Now yerine DateTime.UtcNow kullan – timezone uyumsuzlukları önler (kullanıcılar global olabilir).
İyi yapılandırılmış gereksiz alan mevcut değil.

DÖKÜMANLAR

```
using ilterisg.Models.Enums;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
   [Table("Dokumanlar", Schema = "dbo")]
   public class Dokumanlar
   {
      [Key]
      [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
      public int DokumanId { get; set; }

      [Required]
      public int RefId { get; set; } // Bağlanacağı kaydın ID’si (ödeme, kullanıcı, sözleşme, etkinlik vb.)

      [Required]
      [MaxLength(100)]
       public string RefTableName { get; set; } // Hangi tabloya bağlı? (Odemeler, Sozlesmeler, Etkinlikler vb.)

      [Required]
      [MaxLength(255)]
      public string DosyaAdi { get; set; }

      [Required]
```

```
        public string DosyaYolu { get; set; }
        [MaxLength(500)]
        public string? KlasorYolu { get; set; } // Yeni alan: Klasör hiyerarşisi (örneğin, "Egitimler/Modul1")

        [Required]
        public DateTime YuklemeTarihi { get; set; } = DateTime.Now;

        [MaxLength(50)]
                public DokumanTuru DokumanTuru { get; set; } // "Dekont", "Kimlik", "Sözleşme", "EtkinlikDokuman" vb.

        [MaxLength(500)]
        public string? Aciklama { get; set; }

        [MaxLength(128)] // AspNetUsers.Id uzunluğuna uygun
        public string? UserId { get; set; } // Kullanıcı ID'sini saklamak için yeni alan
    }
}
```

Yapı zaten iyi, her modül için reusable – zorunlu değişiklik yok. Ama normalization ve maintainability için:

RefTableName: String yerine enum yap (e.g., DokumanRefType: Odemeler, Kullanicilar, Etkinlikler vb.). Bu, veri tutarsızlığını önler ve int discriminator'a çevirir (EF TPH için). Eğer çok fazla tip varsa, string kalabilir ama validation attribute ile kısıtla.

YuklemeTarihi: DateTime.Now yerine DateTime.UtcNow kullan – timezone sorunlarını önler (global dosya upload'ları için).

UserId: FK yap ve navigation ekle (ApplicationUser) – EF ile dokuman.User diye eriş. [MaxLength(128)] AspNetUsers.Id'ye uymaz, 450 yap (Guid/string için). Altta önerilen örnek yapı mevcut;

```
using ilterisg.Models.Enums;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
```

```
namespace ilterisg.Models
{
    [Table("Dokumanlar", Schema = "dbo")]
    public class Dokumanlar
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int DokumanId { get; set; }

        [Required]
        public int RefId { get; set; } // Bağlanacağı kaydın ID’si (polymorphic)

        [Required]
        public DokumanRefType RefTableName { get; set; } // String'den enum'a (yeni enum tanımla)

        [Required]
        [MaxLength(255)]
        public string DosyaAdi { get; set; }

        [Required]
        [MaxLength(1000)] // Uzun path'ler için genişlet
        public string DosyaYolu { get; set; }

        [MaxLength(500)]
        public string? KlasorYolu { get; set; } // Opsiyonel hiyerarşi

        [Required]
        public DateTime YuklemeTarihi { get; set; } = DateTime.UtcNow; // UtcNow'a çevir

        [Required]
        public DokumanTuru DokumanTuru { get; set; } // Enum zaten iyi

        [MaxLength(500)]
        public string? Aciklama { get; set; }

        // FK ve navigation ekle
```

```
        [MaxLength(450)] // AspNetUsers.Id için doğru uzunluk
        [ForeignKey(nameof(User))]
        public string? UserId { get; set; }
        public virtual ApplicationUser? User { get; set; }
    }

    // Yeni enum örneği (Enums klasöründe, mevcut modüllere göre genişlet)
    public enum DokumanRefType
    {
        Odemeler,
        Kullanicilar,
        Sozlesmeler,
        Etkinlikler,
        Egitimler
        // Yeni modül ekle: , YeniModul
    }
}
```

DOKUMAN PAYLAŞIM

```
using System.ComponentModel.DataAnnotations;

namespace ilterisg.Models
{
    public class DokumanPaylasim
    {
        [Key]
        public int PaylasimId { get; set; } // Primary key
        public string Token { get; set; } // Unique token for sharing
        public List<int> DokumanIdList { get; set; } // List of document IDs, stored as JSON
        public string RefTableName { get; set; } // Reference table (e.g., Firmalar, Kullanicilar)
        public int RefId { get; set; } // Reference ID (e.g., FirmaId or KullaniciId)
```

```
        public string PaylasanKullaniciId { get; set; } // ID of the user who created the share
        public bool SureliMi { get; set; } // Is the share time-limited?
        public DateTime? GecerlilikTarihi { get; set; } // Expiry date for time-limited shares
        public bool SadeceGoruntulemeMi { get; set; } // View-only permission
        public DateTime OlusturmaTarihi { get; set; } // Creation date
    }
}
```

DokumanIdList alanını kaldırıp bir Junction tabloda saklamak daha sağlam bir yapı sağlar Id 1, 2, 3 gibi tutmak yerıne paylaşım id ile junction, ara tabloda tutmak daha sağlam

Önerilen yapı alttadır;

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("DokumanPaylasim", Schema = "dbo")]
    public class DokumanPaylasim
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int PaylasimId { get; set; } // Primary key

        [Required]
        [MaxLength(450)] // Token uzunluğu için (Guid string)
        public string Token { get; set; } // Unique token, service'te generate et

        [Required]
        [MaxLength(100)]
        public string RefTableName { get; set; } // Reference table
```

```
        [Required]
        public int RefId { get; set; } // Reference ID

        [Required]
        [MaxLength(450)]
        [ForeignKey(nameof(PaylasanKullanici))]
        public string PaylasanKullaniciId { get; set; } // FK

        public bool SureliMi { get; set; } = false; // Default false

        public DateTime? GecerlilikTarihi { get; set; } // Validation: SureliMi ise required

        public bool SadeceGoruntulemeMi { get; set; } = true; // Default view-only, güvenlik için

        [Required]
        public DateTime OlusturmaTarihi { get; set; } = DateTime.UtcNow; // UtcNow default

        // Navigation Properties
        public virtual ApplicationUser PaylasanKullanici { get; set; } = null!;

        // Many-to-many: Doküman listesi için junction collection
         public virtual ICollection<DokumanPaylasimDokuman> DokumanPaylasimDokumanlar { get; set; }
= new List<DokumanPaylasimDokuman>();
    }

    // Yeni junction model (ayrı tablo: DokumanPaylasimDokumanlar)
    [Table("DokumanPaylasimDokumanlar", Schema = "dbo")]
    public class DokumanPaylasimDokuman
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        public int PaylasimId { get; set; }
```

```
        [ForeignKey(nameof(PaylasimId))]
        public virtual DokumanPaylasim Paylasim { get; set; } = null!;

        [Required]
        public int DokumanId { get; set; }
        [ForeignKey(nameof(DokumanId))]
        public virtual Dokumanlar Dokuman { get; set; } = null!;

        // Opsiyonel: Per-doküman permission (e.g., bu doküman view-only mı?)
        public bool SadeceGoruntulemeMi { get; set; } = true;
    }
}
```

EGİTİM GRUPLARI - EGİTİMGRUPKULLANICILAR - KULLANİCİ_EGİTİMGRUBU

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("EgitimGruplari", Schema = "dbo")]
    public class EgitimGruplari
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int GrupId { get; set; }

        [Required]
        [MaxLength(100)]
        public string GrupAdi { get; set; }

        [MaxLength(500)]
        public string? Aciklama { get; set; }
```

```
        public TehlikeSinifi TehlikeSinifi { get; set; }

        public int? RefFirmaId { get; set; } // Firma ile ilişki

        public bool SilindiMi { get; set; } = false;

        // Navigation properties
        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }

            public virtual ICollection<EgitimGrupKullanicilar> EgitimGrupKullanicilar { get; set; } = new
List<EgitimGrupKullanicilar>();

        // Eğitimlerle ilişki
        public virtual ICollection<Egitimler> Egitimler { get; set; } = new List<Egitimler>();

        // Sınav oturumlarıyla ilişki
                public virtual ICollection<SinavOturumlari> SinavOturumlari { get; set; } = new
List<SinavOturumlari>();
            public virtual ICollection<Kullanici_EgitimGrubu> KullaniciEgitimGruplari { get; set; } = new
List<Kullanici_EgitimGrubu>();
    }
}



using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("EgitimGrupKullanicilar", Schema = "dbo")]
    public class EgitimGrupKullanicilar
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
```

```
        public int Id { get; set; }

        [Required]
        public int RefGrupId { get; set; }

        [Required]
        public int RefKullaniciId { get; set; }

        [ForeignKey(nameof(RefGrupId))]
        public virtual EgitimGruplari Grup { get; set; }

        [ForeignKey(nameof(RefKullaniciId))]
        public virtual Kullanicilar Kullanici { get; set; }
    }
}



using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Kullanici_EgitimGrubu", Schema = "dbo")]
    public class Kullanici_EgitimGrubu
    {
        [Key, Column(Order = 0)]
        public int RefKullaniciId { get; set; }

        [Key, Column(Order = 1)]
        public int RefGrupId { get; set; }

        [ForeignKey(nameof(RefKullaniciId))]
        public virtual Kullanicilar Kullanici { get; set; }

        [ForeignKey(nameof(RefGrupId))]
        public virtual EgitimGruplari Grup { get; set; }
```

```
    }
}
```

EgitimGruplari ve JuNCTion tablesi EgitimGrupKullanicilar iyi yapılandırılmış normalızasyon için ekleme ve çıkarmaya gerek yok Kullanici_EgitimGrubu isimli duplicate bir junction tablo daha bulundu o kaldırılacak.

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Egitimler", Schema = "dbo")]
    public class Egitimler
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int EgitimId { get; set; }

        [MaxLength(200)]
        public string Ad { get; set; } // Eğitim adı

        public DateTime EgitimTarihi { get; set; } // Eğitim tarihi

        public int Sure { get; set; } // Süre (saat cinsinden)

        public TehlikeSinifi TehlikeSinifi { get; set; } // Enum: AzTehlikeli, Tehlikeli, CokTehlikeli

        public int? EgitimTuruId { get; set; } // Eğitim türü (foreign key), nullable

        public DateTime? YenilemeTarihi { get; set; } // Periyodik eğitimler için yenileme tarihi

        public int? GrupId { get; set; } // Eğitim grubu
```

```
        public int? EgitmenId { get; set; } // Yeni alan: Eğitmen

        public bool TamamlandiMi { get; set; } = false; // Eğitimin tamamlanıp tamamlanmadığı

        public bool SilindiMi { get; set; } = false; // Soft delete için silme durumu

        // Navigation properties
        [ForeignKey(nameof(EgitimTuruId))]
        public virtual EgitimTurleri EgitimTuru { get; set; }

        [ForeignKey(nameof(GrupId))]
        public virtual EgitimGruplari Grup { get; set; } // Yeni navigation property

        // Bir eğitime birden fazla firma katılabilir (Firma_Egitim tablosu üzerinden)
        public virtual ICollection<Firma_Egitim> FirmaEgitimler { get; set; } = new List<Firma_Egitim>();

        // Bir eğitime birden fazla kullanıcı/çalışan katılabilir (Kullanici_Egitim tablosu üzerinden)
                public virtual ICollection<Kullanici_Egitim> KullaniciEgitimler { get; set; } = new
List<Kullanici_Egitim>();
    }
}
```

Eğitim sistemimizin eski kalıntı bir tablo yeni yapılandırılmış Sınav sistemimizde tablo hiçbir şekilde kullanılmıyor, kaldırılabilir.

EGİTİMTURUMATERYALLERİ

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
```

```
    [Table("EgitimTuruMateryalleri", Schema = "dbo")]
    public class EgitimTuruMateryalleri
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        public int EgitimTuruId { get; set; }

        [Required]
        public int DokumanId { get; set; } // Materyalin Dokumanlar tablosundaki ID'si

        [ForeignKey(nameof(EgitimTuruId))]
        public virtual EgitimTurleri EgitimTuru { get; set; }

        [ForeignKey(nameof(DokumanId))]
        public virtual Dokumanlar Dokuman { get; set; }
    }
}
```

Gereksiz Alan?: Hayır, minimal – ekstra metadata (e.g., sıralama) yoksa bu haliyle yeterli. Eğer materyalin "sırası" (order) önemliyse, int SıraNo ekle (ama zorunlu değil).

Potansiyel İyileştirmeler (Değişiklik Gerekir mi?):
Yapı zaten production-ready, zorunlu değişiklik yok – ama long-term için ufak eklemeler öneririm (normalization'ı bozmaz, audit/performans artırır):

Audit Alanları: CreatedAt/UpdatedAt ekle (DateTime.UtcNow default) – materyal atamalarının ne zaman yapıldığını track et (raporlama için faydalı).
Soft Delete: SilindiMi bool ekle (default false) – materyali silmek yerine gizle (veri kaybı önler, raporlarda filtrele).
Index'ler: Migration'da composite unique index (EgitimTuruId, DokumanId) ekle – duplicate atamayı önler, sorgu hızı için (e.g., materyalin bir türe birden fazla atanmasını engelle).
Genişletme: Eğer materyal-tür ilişkisinde ek info (e.g., zorunlu mu/opsiyonel mi) varsa, enum Durum ekle. Ama mevcut haliyle kategorileştirme yeterli.

EGİTİMTURUSINAVLARI - EGİTİM TURLERİ - SINAVLAR

Sınavlar soruların tutulduğu taslak sınavı belirler, eğiti turu sınavları sınav içeriğindeki soruları belirler.

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Sinavlar", Schema = "dbo")]
    public class Sinavlar
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int SinavId { get; set; }

        [Required(ErrorMessage = "TrainingTypeRequired")]
        public int EgitimTuruId { get; set; }

        [Required(ErrorMessage = "ExamNameRequired")]
        [MaxLength(100, ErrorMessage = "ExamNameMaxLength")]
        [MinLength(3, ErrorMessage = "ExamNameMinLength")]
        public string SinavAdi { get; set; }

        [Required(ErrorMessage = "RepeatCountRequired")]
        [Range(1, 10, ErrorMessage = "RepeatCountRange")]
        public int TekrarlamaSayisi { get; set; } = 3;

        [Required(ErrorMessage = "PassingScoreRequired")]
        [Range(0, 100, ErrorMessage = "PassingScoreRange")]
        public int GecmePuani { get; set; } = 70;

        public bool SilindiMi { get; set; } = false;
```

```
        // Navigation properties
        [ForeignKey(nameof(EgitimTuruId))]
        public virtual EgitimTurleri EgitimTuru { get; set; }

                        public virtual ICollection<EgitimTuruSinavlari> Sorular { get; set; } = new
List<EgitimTuruSinavlari>();
    }
}

using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("EgitimTurleri", Schema = "dbo")]
    public class EgitimTurleri
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int EgitimTuruId { get; set; }

        [Required]
        [MaxLength(100)]
        public string Ad { get; set; } // Tür adı

        [MaxLength(500)]
        public string? Aciklama { get; set; } // Tür açıklaması

        [MaxLength(50)]
        public string? EgitimKodu { get; set; } // İBYS eğitim kodu (örneğin, "140")

        public bool SilindiMi { get; set; } = false; // Soft delete için

        // Navigation properties
        public virtual ICollection<Egitimler> Egitimler { get; set; } = new List<Egitimler>();
```

```
        public virtual ICollection<EgitimTuruMateryalleri> Materyaller { get; set; } = new List<EgitimTuruMateryalleri>();

        public virtual ICollection<EgitimTuruSinavlari> Sinavlar { get; set; } = new List<EgitimTuruSinavlari>();
    }
}
```

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("EgitimTuruSinavlari", Schema = "dbo")]
    public class EgitimTuruSinavlari
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        public int EgitimTuruId { get; set; }

        public int? SinavId { get; set; }

        [Required]
        [MaxLength(500)]
        public string SoruMetni { get; set; }

        [Required]
        [MaxLength(100)]
        public string SecenekA { get; set; }
```

```
        [Required]
        [MaxLength(100)]
        public string SecenekB { get; set; }

        [Required]
        [MaxLength(100)]
        public string SecenekC { get; set; }

        [Required]
        [MaxLength(100)]
        public string SecenekD { get; set; }

        [Required]
        [MaxLength(50)]
        public string DogruCevap { get; set; }

        // Navigation properties
        [ForeignKey(nameof(EgitimTuruId))]
        public virtual EgitimTurleri? EgitimTuru { get; set; }

        [ForeignKey(nameof(SinavId))]
        public virtual Sinavlar Sinav { get; set; }
    }
}
```

Güçlü Yönler:

İlişkiler ve Esneklik: EgitimTuruId required FK ile her soru bir türe zorunlu bağlı (kategorileştirme için mükemmel). SinavId nullable ile taslak modu (sınava atanmamış sorular) destekleniyor – admin taslak oluşturup sonra Sinavlar'a atayabilir. Navigation'lar bidirectional (EgitimTurleri.Sinavlar collection'ı EF Include'larını kolaylaştırır, e.g., Include(et => et.Sinavlar) ile sorular getir).

Veri Bütünlüğü: [Required]/[MaxLength] validation'ları sağlam (SoruMetni geniş, seçenekler kısa). DogruCevap string ama pratik (A/B/C/D için). EgitimTurleri'nde SilindiMi soft delete standart – veri kaybı önler.

Mantık Uyumu: Taslak soruları için ideal (SinavId null = taslak). Collections (Materyaller, Sinavlar)

many-to-many/many-to-one'i doğru yönetiyor, eğitim sistemi scalable (yeni tür ekle, sorular otomatik kategorize).
Gereksiz Alan?: Hayır, hepsi core – EgitimKodu gibi opsiyonel alanlar (IBYS entegrasyonu için) faydalı, çıkarılmaz.

Potansiyel İyileştirmeler (Değişiklik Gerekir mi?):
Yapı zaten iyi, zorunlu değişiklik yok – ama normalization ve maintainability için:

DogruCevap: String yerine enum yap (e.g., CevapSecenegi: A, B, C, D) – veri tutarlılığı artar, validation kolaylaşır (string free-text olursa "A " vs "a" hatası). Opsiyonel
Audit Eklemeleri: CreatedAt/UpdatedAt ekle (DateTime.UtcNow default) – taslak/soru değişikliklerini track et (raporlama için). EgitimTuruSinavlari'na da SilindiMi ekle (soft delete tutarlılığı).

EGİTİMTURUMATERYALLERİ

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("EgitimTuruMateryalleri", Schema = "dbo")]
    public class EgitimTuruMateryalleri
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        public int EgitimTuruId { get; set; }

        [Required]
        public int DokumanId { get; set; } // Materyalin Dokumanlar tablosundaki ID'si

        [ForeignKey(nameof(EgitimTuruId))]
```

```
        public virtual EgitimTurleri EgitimTuru { get; set; }

        [ForeignKey(nameof(DokumanId))]
        public virtual Dokumanlar Dokuman { get; set; }
    }
}
```

Güçlü Yönler:

Many-to-Many İlişki: EgitimTuruId ve DokumanId [Required] FK'ler ile zorunlu bağ, navigation'lar EF sorgularını kolaylaştırır (e.g., egitimTur.Materyaller.Include(m => m.Dokuman) ile materyalleri getir, DosyaYolu'nu kullan).
Veri Bütünlüğü: [Key]/[DatabaseGenerated] otomatik ID, [Required] ile boş atama yok – validation sağlam, hata riski düşük.
Esneklik: Eğitim materyallerini tür bazlı kategorize etmek için ideal (Dokumanlar'daki polymorphic yapıyla entegre, e.g., RefTableName="EgitimTurleri"). Minimal tasarım, gereksiz karmaşıklık yok.
Gereksiz Alan?: Hayır, core – ekstra info (e.g., materyalin zorunlu mu olduğu) yoksa bu hali yeterli.

Potansiyel İyileştirmeler (Değişiklik Gerekir mi?):
Yapı zaten production-ready, zorunlu değişiklik yok – ama scalability ve traceability için ufak ekler öneririm (normalization'ı bozmaz):

Audit Alanları: CreatedAt/UpdatedAt ekle (DateTime.UtcNow default) – materyal atamasının ne zaman yapıldığını track et (rapor/denetim için faydalı).
Soft Delete: SilindiMi bool ekle (default false) – materyali silmek yerine gizle, ilişkileri koru (veri bütünlüğü için).
Index'ler: Migration'da unique composite index (EgitimTuruId, DokumanId) ekle – duplicate atamayı önle, sorgu hızını artır (e.g., bir materyalin aynı türe birden fazla atanmasını engelle).
Genişletme: Eğer materyallerin sırası önemliyse int SıraNo ekle (default 0) – UI'de sıralama için (opsiyonel, overkill değil).

SınavOTURUMLARI

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("SinavOturumlari", Schema = "dbo")]
    public class SinavOturumlari
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int OturumId { get; set; }

        public int? RefEgitimId { get; set; }

        [ForeignKey(nameof(RefEgitimId))]
        public virtual Egitimler Egitim { get; set; }

        public int SinavId { get; set; }

        [ForeignKey(nameof(SinavId))]
        public virtual Sinavlar Sinav { get; set; }

        public int? KullaniciId { get; set; } // Nullable yapıldı

        [ForeignKey(nameof(KullaniciId))]
        public virtual Kullanicilar Kullanici { get; set; }

        [MaxLength(500)]
        public string OturumToken { get; set; }

        public SinavOturumDurumu Durum { get; set; }

        public DateTime? BaslamaTarihi { get; set; }
```

```
        public DateTime? BitisTarihi { get; set; }

        public int? Puan { get; set; }

        public bool TcDogrulandiMi { get; set; } = false;

        public int? GrupId { get; set; }

        [Column(TypeName = "bit")]
        public bool DavetGonderildiMi { get; set; } = false;

        [ForeignKey(nameof(GrupId))]
        public virtual EgitimGruplari Grup { get; set; }

        public int? RefEtkinlikId { get; set; }

        [ForeignKey(nameof(RefEtkinlikId))]
        public virtual Etkinlikler Etkinlik { get; set; }

                public virtual ICollection<KullaniciCevaplari> KullaniciCevaplari { get; set; } = new
List<KullaniciCevaplari>();
    }

    public enum SinavOturumDurumu
    {
        [Display(Name = "NotStarted")]
        Baslamadi,
        [Display(Name = "Ongoing")]
        DevamEdiyor,
        [Display(Name = "Completed")]
        Tamamlandi
    }
}
```

Her personelin girdiği sınav oturumlarını temsil eder her sınava giren ve bitiren kişi için ayrı kayıt açılır. Normalizasyona tamamen uygun SinavOuturumDurumu enumu ise 3NF normalizasyona uygun

EGİTMENLER

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
   [Table("Egitmenler", Schema = "dbo")]
   public class Egitmenler
   {
      [Key]
      [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
      public int EgitmenId { get; set; }

      public int KullaniciId { get; set; }

      public int FirmaId { get; set; }

      public bool SilindiMi { get; set; } = false;

      [ForeignKey("KullaniciId")]
      public virtual Kullanicilar Kullanici { get; set; }

      [ForeignKey("FirmaId")]
      public virtual Firmalar Firma { get; set; }
   }
}
```

Değerlendirme:

Egitmenler tablosu, kullanıcı (Kullanicilar) ile firma (Firmalar) arasında bağlantı (ilişki) görevi görüyor.

Yani aslında bu tablo bir ilişki (junction) tablosu gibi davranıyor.

SilindiMi alanı hariç, fazladan bilgi barındırmıyor.
Bu durumda tablo 3NF (Third Normal Form) kurallarına uygundur.

Olası İyileştirmeler:

Eğer bir eğitmen sadece bir firmaya bağlı olabiliyorsa, bu yapı doğru.

Ancak aynı eğitmen birden fazla firmada görev alabiliyorsa, Egitmenler tablosu zaten olması gereken ilişki tablosu konumundadır.

SilindiMi yerine soft delete için genellikle ortak bir “BaseEntity” sınıfında IsDeleted, CreatedDate, UpdatedDate gibi alanlar tanımlanabilir.

Sonuç:İyi yapılandırılmış gereksiz alan yok iyileştirme için alanlar eklenebilir.

Şuan Kullanılmıyor.

LanguageResoure

```
using System;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    public class LanguageResource
    {
        [Key]
        public long Id { get; set; }

        [Required]
        [StringLength(200)]
        public string ResourceKey { get; set; }
```

```
        [Required]
        [StringLength(10)]
        public string LanguageCode { get; set; }

        [Required]
        [StringLength(1000)]
        public string Value { get; set; }

        public DateTime LastUpdated { get; set; } = DateTime.UtcNow;
    }
}
```

Localization için ilgili kayıtların tutulduğu tablo, güncellemeye gerek yok.

MSLEKKODLARI

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{

    public class MeslekKodlari
    {
        [Key]
        public int Id { get; set; }

        [Required]
        [MaxLength(20)]
        public string IscoKodu { get; set; }  // ISCO-08 Kodu
```

```
        [Required]
        [MaxLength(255)]
        public string Ad { get; set; }  // Meslek Adı
    }
}
```

Veri seti tablosu güncellemeye gerek yok.

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Mevzuat", Schema = "dbo")]
    public class Mevzuat
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        [MaxLength(50)]
        public string MevzuatKodu { get; set; }

        [Required]
        [MaxLength(255)]
        public string MevzuatAdi { get; set; }

        public string Aciklama { get; set; } // LONGTEXT (MySQL) / nvarchar(max) (SQL Server)

        public DateTime? YayinTarihi { get; set; }

        public DateTime CreatedAt { get; set; } = DateTime.Now;
```

```
    [InverseProperty("Mevzuat")]
        public virtual ICollection<RiskDegerlendirme> RiskDegerlendirmeler { get; set; } = new List<RiskDegerlendirme>();
  }
}
```

Veri seti tablosu tsv dosyası set ediiyor güncellemeye gerek yok.

MUAYENE

```
using Microsoft.AspNetCore.Mvc.ModelBinding;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
  public class Muayene
  {
    [NotMapped]
    public string HastaAdSoyad
      => Kullanici?.AdSoyad ?? string.Empty;
    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    public int? MuayeneId { get; set; }

    [Required]
    [Display(Name = "Personel")]
    public int? KullaniciId { get; set; }

    [ForeignKey(nameof(KullaniciId))]
    [BindNever]
    public Kullanicilar? Kullanici { get; set; }
```

```
public DateTime? Tarih { get; set; } // DateTime'dan DateTime? olarak güncellendi

[Required, StringLength(500)]
public string Sikayet { get; set; } = "";

[Required, StringLength(500)]
public string Bulgular { get; set; } = "";



public string OnTani { get; set; } = "";

[Required, StringLength(250)]
public string TetkikOnerileri { get; set; } = "";

[Required, StringLength(250)]
public string TedaviOnerisi { get; set; } = "";

// Artık DB’de saklanacak:
[StringLength(1000)]
public string Hikaye { get; set; } = "";

[StringLength(1000)]
public string Sonuclar { get; set; } = "";

//[StringLength(1000)]
//public string Recete { get; set; } = "";



[StringLength(1000)]
public string Sonuc { get; set; } = "";

// Oluşturan kullanıcıyı action içinde ata, formdan gelmesin:
[BindNever]
public string? OlusturanUserId { get; set; }

[BindNever]
```

```
        public ApplicationUser? OlusturanUser { get; set; }

        // → Navigation property: bir muayeneye birden fazla teşhis
        public List<MuayeneTeshis> MuayeneTeshisler { get; set; } = new();
        public List<MuayeneRecete> MuayeneReceteler { get; set; } = new();
        [NotMapped]
        public string? ManuelTc { get; set; } = KALKTI
    }
}
```

ManuelTC alanı normalizasyona aykırı Kullanıcı tc kimlik nosu dahil her bilgi KullanıcıId ile kullanıcılar tablosundan erişilebiliyor.

```
    public class Ilac
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int IlacId { get; set; }

        [Required]
        public string Ad { get; set; } = null!;

        [Required]
        [StringLength(50)]
        public string Barkod { get; set; } = null!;

        [Required]
        [StringLength(20)]
        public string ATCKodu { get; set; } = null!;

        [Required]
```

```
    [StringLength(100)]
    public string ATCAdi { get; set; } = null!;

    [Required]
    public KullanimSekliEnum KullanimSekli { get; set; } = KullanimSekliEnum.Belirtilmemis;
}
```

Veri seti tablosu revizeye gerek yok.

MUAYENERECETE

```
public class MuayeneRecete
{
    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    [BindNever]
    public int Id { get; set; }

    // E-reçete için ana reçete kimliği (DVO: protokolNo)
    [Required]
    [StringLength(50)]
    public string ReceteId { get; set; } = "";

    // Muayene ile ilişki
    public int? MuayeneId { get; set; }
    [ForeignKey(nameof(MuayeneId))]
    [ValidateNever]
    public Muayene? Muayene { get; set; }

    // İlaç ile ilişki
    [Required]
    public int IlacId { get; set; }
```

```
[ForeignKey(nameof(IlacId))]
[ValidateNever]
public Ilac Ilac { get; set; } = null!;

// Hasta (Yazılan kişi) - Opsiyonel (Medula için)
public int? KullaniciId { get; set; } // [Required] kaldırıldı, nullable yapıldı
[ForeignKey(nameof(KullaniciId))]
[ValidateNever]
public Kullanicilar? Kullanici { get; set; } // Nullable

// Yazan doktor (ApplicationUser ID'si için - her zaman bilinsin)
[StringLength(450)]
public string OlusturanUserId { get; set; } = string.Empty; // Doktor'un appUser.Id'si

// Genel reçete açıklaması (DVO: ereceteAciklamaListesi)
[StringLength(500)]
public string? Aciklama { get; set; } = "";

// İlaç özel açıklaması (DVO: ereceteIlacAciklamaListesi)
[StringLength(500)]
public string? IlacAciklama { get; set; } = "";

// Doz 1 x Doz 2 (DVO: kullanimDoz1, kullanimDoz2)
[Required, Range(0.01, 9999)]
[Column(TypeName = "decimal(8,2)")]
public decimal Doz1 { get; set; }

[Required, Range(0.01, 9999)]
[Column(TypeName = "decimal(8,2)")]
public decimal Doz2 { get; set; }

// Kullanım periyodu (DVO: kullanimPeriyot, kullanimPeriyotBirimi)
[Required, Range(1, 3650)]
public int KullanimPeriyot { get; set; } = 1;

[Required]
```

```
                public KullanimPeriyotBirimiEnum KullanimPeriyotBirimi { get; set; } = KullanimPeriyotBirimiEnum.Gun;

    // Adet (DVO: adet)
    [Required, Range(1, 999)] // Min 1 olarak güncellendi (0 mantıksız)
    public int Adet { get; set; } = 1;

    // Kullanım şekli (DVO: kullanimSekli)
    [Required]
    public KullanimSekliEnum KullanimSekli { get; set; } = KullanimSekliEnum.Belirtilmemis;

    // E-reçete için ek alanlar
    [StringLength(20)]
    public string? Barkod { get; set; } // DVO: barkod (Ilac.Barkod ile yedekli)

    // DVO'dan gelen reçete seviyesindeki ek alanlar
    [Required]
    public ReceteTuruEnum ReceteTuru { get; set; }

    [Required]
    public DateTime ReceteTarihi { get; set; }

    [StringLength(20)]
    public string? ErecetеNo { get; set; }

    [StringLength(20)]
    public string? TakipNo { get; set; }

    [StringLength(10)]
    public string? SeriNo { get; set; }

    public int? TesisKodu { get; set; }

    public int? DoktorBransKodu { get; set; }

    public long? DoktorSertifikaKodu { get; set; }
```

```
    [StringLength(50)]
    public string? DoktorAdi { get; set; }

    [StringLength(50)]
    public string? DoktorSoyadi { get; set; }

    public long? DoktorTcKimlikNo { get; set; }

    [Required]
    public long TcKimlikNo { get; set; } // Hasta TC'si (Medula'dan gelir, zorunlu)

    // Yeni eklenen alan: Medula'dan silinip silinmediğini belirten bool
    public bool MedulaSilindiMi { get; set; } = false;
    public DateTime? MedulaSilmeTarihi { get; set; }

    // Computed properties
    [NotMapped]
    public string DozOzet => $"{Doz1} x {Doz2} / {KullanimPeriyot} {KullanimPeriyotBirimi}";

    [NotMapped]
      public string TamAciklama => string.IsNullOrEmpty(IlacAciklama) ? (Aciklama ?? "") : $"{Aciklama
?? ""} - {IlacAciklama}";
  }
```

Sonuç: İyi yapılandırılmış güncelleme gerekmiyor.

MuayeneRecete tablosu (ve ilişkili modeller) genel olarak çok iyi yapılandırılmış—özellikle Medula entegrasyonu (e-reçete DVO standartları) için düşünülmüş, pratik bir tasarım. Nullable'lar, foreign key'ler ve computed property'ler (DozOzet, TamAciklama) gibi detaylar, hem veritabanı verimliliğini hem de kod okunabilirliğini artırıyor. İki fark (KullaniciId'nin opsiyonel olması ve TC'nin zorunlu fallback'i) dışında, akış tutarlı: Hasta detayları opsiyonelken TC her zaman anchor noktası, bu da ghost hasta senaryolarını destekliyor.
Güçlü Yönler:

Standart Uyumu: E-reçete alanları (EreceteNo, TakipNo, SeriNo, TesisKodu vb.) doğrudan DVO'dan

uyarlanmış—gereksiz şişkinlik yok, Medula için hazır.
Esneklik: MuayeneId nullable (muayenesiz reçete için), KullaniciId opsiyonel (sistem dışı hasta için TC yeterli). Doz/Adet/Kullanim gibi alanlar Range validation'lı, veri kalitesini koruyor.
Navigation Property'ler: Ilac, Muayene, Kullanici gibi FK'lar EF Core ile sorunsuz join'lere izin veriyor (Include'larla raporlama kolay).
Ek Alanlar: MedulaSilindiMi/SilmeTarihi gibi flag'ler, silme takibini entegre etmiş—pratik, audit için ideal.
Ilac Modeli: Basit ve yeterli; Barkod/ATC entegrasyonu için hazır.
Kullanicilar & Kullanici_Firma: Standart user-firma ilişkisi, composite key ile many-to-many'yi temiz yönetiyor. RefUserId ile AspNetUsers'a bağlanması, auth entegrasyonunu güçlendiriyor.

Gereksiz Alan Var mı?
Hayır, açıkça gereksiz alan yok—hepsi işlevsel ve Medula/reçete akışına hizmet ediyor. Ama ufak optimizasyon önerileri (zorunlu değil, mevcut haliyle bile mükemmel):

Doktor Alanları (DoktorAdi, DoktorSoyadi, DoktorTcKimlikNo, BransKodu, SertifikaKodu): Bunlar denormalize edilmiş (tekrarlanabilir)—eğer doktor her zaman mevcut (OlusturanUserId'den çekilebilir) ise, view-time'da computed olarak hesapla (DB'de saklama). Ama DVO için flat field zorunluysa (Medula XML'inde direkt TC/adı istiyor), kalması doğru. Potansiyel gereksizlik: Eğer her reçetede aynı doktor varsa, index'le ama silme.
Barkod: Ilac.Barkod ile yedekli, ama fallback için mantıklı—gereksiz değil, silme.
Aciklama & IlacAciklama: StringLength 500, yeterli; birleştirilebilir (TamAciklama gibi), ama ayrı tutmak UI için iyi.
Genel: Column(TypeName) ve BindNever'lar gibi attribute'ler temiz—veri tipi tutarlı (decimal doz için). Eğer tablo büyüyecekse, index'ler ekle (ReceteId, TcKimlikNo, IlacId'ye).

Kısa Cevap:

Hayır, tamamen TC kimlik numarasıyla yönetmek doğru ve sürdürülebilir değildir.
TC kimlik alanı sadece kimlik doğrulama veya eşleştirme amacıyla kullanılabilir,
ama sistem içi ilişki yönetimi (firma-personel, yetkilendirme, muayene geçmişi, reçete kayıtları) için KullaniciId kullanılmalıdır.

⚙ Detaylı Açıklama

1. Kullanici_Firma yapısının amacı

Bu tablo, firma-personel ilişkisini tanımlıyor:

```
public class Kullanici_Firma
{
   public int RefFirmaId { get; set; }
   public int RefKullaniciId { get; set; }
}
```

Bu model, sistemin içinde “bu kullanıcı şu firmada çalışıyor” bilgisini taşır.
Yani ilişki yönetimi KullaniciId üzerinden yapılır — çünkü:

Firma tablosunun anahtarı FirmaId

Kullanıcı tablosunun anahtarı KullaniciId

Dolayısıyla bu ilişki, veritabanı düzeyinde yabancı anahtar (FK) bağı ile tanımlıdır.

⚠ 2. TC kimlik numarası üzerinden yönetim neden hatalı olur?

Eğer sadece TC ile kontrol etmeye çalışırsan, birkaç ciddi problem ortaya çıkar:

a) Veri bütünlüğü bozulur

TC bir metinsel değerdir, ilişkisel bir FK değildir.
Yani:

Kullanici_Firma tablosunda TcKimlikNo ile bağ kuramazsın (FK atayamazsın).

Bir kişi aynı TC ile yanlışlıkla iki kez eklenirse (duplicate kayıt),
sistem bunları ayırt edemez.
→ Reçete, muayene veya personel atamaları karışır.

b) Performans düşer

TC üzerinden sorgular (join, where) daima string karşılaştırması gerektirir.
Büyük veri setlerinde (JOIN on NVARCHAR) hem indeks kullanılamaz,
hem sorgular yavaşlar.

c) Yabancı uyruk / geçici kimlik senaryosu

Sisteme yabancı uyruklu çalışan, geçici kimlik numarası olan ya da TC'si hatalı girilmiş kişiler geldiğinde KullaniciId tabanlı sistem bunları yönetebilir ama TC tabanlı sistem yönetemez.

d) Uygulama mantığı tutarsızlaşır

Birçok servis (PopulatePersonelList, GetDoktorFirmaIdsAsync, MuayeneKaydet, ReceteOlustur) zaten KullaniciId ile çalışıyor.
Bu kodları TC'ye çevirmek demek, tüm ilişki zincirini (FK navigation property) bozmak olur.

3. Doğru Yaklaşım:

İlişkisel olarak her zaman KullaniciId kullanılmalı,
ama kimlik doğrulama veya harici sistemlerle entegrasyon (örneğin Medula) aşamasında TcKimlikNo kullanılabilir.

Yani:

SenaryoKullanılacak Alan
Firma-personel eşleşmesiKullanici_Firma.RefKullaniciId
Muayene, Reçete, Eğitim ilişkileriKullaniciId
Medula veya e-Reçete gönderimiTcKimlikNo
TC'siz dış hastalarKullaniciId = null, ManuelTc geçici değer

4. En İdeal Mimari (Önerilen)

Firma yönetimi, yetkilendirme, personel filtreleme → KullaniciId

E-reçete, dış hasta, Medula işlemleri → TcKimlikNo

İlişkisel bütünlük → Kullanici_Firma tablosu üzerinden

Sonuç

KriterDeğerlendirme

Tamamen TC ile yönetim mümkün mü? Hayır — ilişkisel bütünlük ve performans sorunları olur.

Kullanici_Firma ne işe yarıyor?Firma-personel bağlantısını KullaniciId ile tanımlar.

TC kimlik nerede kullanılmalı?Sadece Medula/e-Reçete gibi dış sistem işlemlerinde veya kayıtlı olmayan hastalarda.

Doğru yönetim modeliFirma personeli KullaniciId ile, kayıt dışı hasta ManuelTc ile yönetilir.

İstersen bu yapıyı şematik olarak (ilişki diyagramı şeklinde) çizip gösterebilirim —
örneğin Kullanici, Firma, Kullanici_Firma, Muayene, MuayeneRecete ilişkilerini net şekilde.
İster misin o diyagramı da oluşturayım?

EK2FORM

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using ilterisg.Models.Enums;

namespace ilterisg.Models
{
    [Table("Ek2Formlar", Schema = "dbo")]
    public class Ek2Form
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Ek2FormId { get; set; }

        [Required(ErrorMessage = "Kullanıcı ID zorunludur.")]
        public int KullaniciId { get; set; }

        [Required(ErrorMessage = "Firma ID zorunludur.")]
```

```
    public int RefFirmaId { get; set; }

    public int? RefOSGBId { get; set; }

    [Required(ErrorMessage = "Oluşturan kullanıcı ID zorunludur.")]
    public string OlusturanUserId { get; set; }

    [Required(ErrorMessage = "Oluşturma tarihi zorunludur.")]
    public DateTime OlusturmaTarihi { get; set; }

    [Required(ErrorMessage = "Form durumu zorunludur.")]
      public Ek2FormDurum Durum { get; set; } = Ek2FormDurum.AcildiBekliyor; // Varsayılan: Açıldı, bekliyor

    // İşyeri Bilgileri
    [Required(ErrorMessage = "Firma adı zorunludur.")]
    public string FirmaAdi { get; set; }
    public string? SgkSicilNo { get; set; }
    public string? FirmaAdresi { get; set; }
    public string? TelefonFaks { get; set; }
    public string? EPosta { get; set; }
    public string? CalisanBeyani { get; set; }

    // Çalışan Bilgileri
    [Required(ErrorMessage = "Ad soyad zorunludur.")]
    public string AdSoyad { get; set; }
    [Required(ErrorMessage = "T.C. Kimlik No zorunludur.")]
    public string TcKimlikNo { get; set; }

    public string? DogumYeri { get; set; }
    [Required(ErrorMessage = "Doğum tarihi zorunludur.")]
    public DateTime DogumTarihi { get; set; }
    [Required(ErrorMessage = "Cinsiyet zorunludur.")]
    public string Cinsiyet { get; set; }
    public string? EgitimDurumu { get; set; }
    public string? MedeniDurum { get; set; }
```

```
        public int? CocukSayisi { get; set; }
        public string? EvAdresi { get; set; }
        public string? TelefonNo { get; set; }
        public string? Meslek { get; set; }
        public string? YaptigiIs { get; set; }
        public string? CalistigiBolum { get; set; }
                public virtual ICollection<Ek2OncekiIsYeri> OncekiIsYerleri { get; set; } = new
List<Ek2OncekiIsYeri>();
        public string? Ozgecmis { get; set; }
        public string? KanGrubu { get; set; }
        public string? KonjenitalKronikHastalik { get; set; }
        public bool TetanozBagisiklama { get; set; } = false;
        public bool HepatitBagisiklama { get; set; } = false;
        public string? DigerBagisiklama { get; set; }
        public string? SoygecmisAnne { get; set; }
        public string? SoygecmisBaba { get; set; }
        public string? SoygecmisKardes { get; set; }
        public string? SoygecmisCocuk { get; set; }

        // Tıbbi Anamnez
        public bool BalgamliOksuruk { get; set; } = false;
        public bool NefesDarligi { get; set; } = false;
        public bool GogusAgrisi { get; set; } = false;
        public bool Carpinti { get; set; } = false;
        public bool SirtAgrisi { get; set; } = false;
        public bool Ishalkabizlik { get; set; } = false;
        public bool EklemlerdeAgri { get; set; } = false;
        public bool KalpHastaligi { get; set; } = false;
        public bool SekerHastaligi { get; set; } = false;
        public bool BobrekRahatsizligi { get; set; } = false;
        public bool Sarilik { get; set; } = false;
        public bool MideUlseri { get; set; } = false;
        public bool IsitmeKaybi { get; set; } = false;
        public bool GormeBozuklugu { get; set; } = false;
        public bool SinirSistemiHastaligi { get; set; } = false;
        public bool DeriHastaligi { get; set; } = false;
```

```
public bool BesinZehirlenmesi { get; set; } = false;
public string? HastanedeYatisTani { get; set; }
public string? AmeliyatNeden { get; set; }
public string? IsKazaDetay { get; set; }
public string? MeslekHastaligiSonuc { get; set; }
public string? MaluliyetDetay { get; set; }
public string? TedaviDetay { get; set; }
public bool SigaraIciyorMu { get; set; } = false;
public string? SigaraBirakmaZamani { get; set; }
public string? SigaraIcmeSuresi { get; set; }
public int? SigaraGunlukAdet { get; set; }
public bool AlkolAliyorMu { get; set; } = false;
public string? AlkolBirakmaZamani { get; set; }
public string? AlkolIcmeSuresi { get; set; }
public string? AlkolSiklik { get; set; }

// Fizik Muayene
public string? GozMuayene { get; set; }
public string? KbbMuayene { get; set; }
public string? DeriMuayene { get; set; }
public string? KardiyovaskulerMuayene { get; set; }
public string? SolunumMuayene { get; set; }
public string? SindirimMuayene { get; set; }
public string? UrogenitalMuayene { get; set; }
public string? KasIskeletMuayene { get; set; }
public string? NorolojikMuayene { get; set; }
public string? PsikiyatrikMuayene { get; set; }
public string? Tansiyon { get; set; }
public int? Nabiz { get; set; }
public double? Boy { get; set; }
public double? Kilo { get; set; }
public double? VucutKitleIndeksi { get; set; }

// Laboratuvar Bulguları
public string? KanAnalizi { get; set; }
public string? IdrarAnalizi { get; set; }
```

```
        public string? RadyolojikAnaliz { get; set; }
        public string? Odyometri { get; set; }
        public string? SFT { get; set; }
        public string? PsikolojikTestler { get; set; }

        // Kanaat ve Sonuç
        public bool CokTehlikeliIslereUygun { get; set; } = false;
        public bool YuksekteCalismaUygun { get; set; } = false;
        public bool GeceVardiyaliCalismaUygun { get; set; } = false;
        public string? Kanaat { get; set; }
        [Required(ErrorMessage = "İşyeri hekimi ad soyad zorunludur.")]
        public string IsyeriHekimiAdSoyad { get; set; }
        public string? DiplomaBilgileri { get; set; }

        // Navigation Properties (Nullable)
        [ForeignKey(nameof(KullaniciId))]
        public virtual Kullanicilar? Kullanici { get; set; }

        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar? Firma { get; set; }

        [ForeignKey(nameof(RefOSGBId))]
        public virtual OSGBler? OSGB { get; set; }

        [ForeignKey(nameof(OlusturanUserId))]
        public virtual ApplicationUser? OlusturanUser { get; set; }
    }
}
```

Normalizasyona uygun form dışı alanlar sadece bağlantıları belli edebilmek için doğru bir şekilde tanımlanmış.

EK2OncekıIsYerı

EK2Formda bırden fazla eski işyeri olabileceği için Junction Table olarak yapılandırıldı o alan

```
using System;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Ek2OncekiIsYerleri", Schema = "dbo")]
    public class Ek2OncekiIsYeri
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        public int? Ek2FormId { get; set; }

        public string? IsKolu { get; set; }
        public string? YaptigiIs { get; set; }
        public DateTime? GirisTarihi { get; set; }
        public DateTime? CikisTarihi { get; set; }

        [ForeignKey(nameof(Ek2FormId))]
        public virtual Ek2Form? Ek2Form { get; set; }
    }
}
```

Normalizasyona uygun sadece Ek2FormID nullable int yerine int olabılilir her Ek2 Öncek İ yeri alanı bir forma bağlı olacağı için.

EMAİL SETTİNGS - OPSİYONEL - YERLEŞİK EMAİL SİSTEMİ İÇİN OLUŞTURULDU DEVAM EDİLMEYECEKSE SİLİNEBİLİR

```
using System.ComponentModel.DataAnnotations;

namespace ilterisg.Models
{
    public class EmailSettings
    {
        [Key]
        public int Id { get; set; }

        [Required]
        public string UserId { get; set; }

        [EmailAddress]
        public string EmailAddress { get; set; }

        public string Password { get; set; } // OAuth kullanılmayan durumlarda

        public string ImapServer { get; set; }
        public string ImapPort { get; set; }
        public bool ImapEnableSsl { get; set; }

        public string PopServer { get; set; }
        public string PopPort { get; set; }
        public bool PopEnableSsl { get; set; }

        public string SmtpServer { get; set; }
        public string SmtpPort { get; set; }
        public bool SmtpEnableSsl { get; set; }

        public string Protocol { get; set; }

        // OAuth için token’lar
        public string AccessToken { get; set; }
        public string RefreshToken { get; set; }
```

```
        public string TokenProvider { get; set; } // "Gmail", "Outlook", "Yahoo" vb.

        public ApplicationUser User { get; set; }
    }
}
```

ETKİNLİK-ETKNLİK KULLANICI

```
using ilterisg.Models.Enums;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Etkinlikler", Schema = "dbo")]
    public class Etkinlikler
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        [MaxLength(200)]
        public string Ad { get; set; } // Etkinlik adı

        [Required]
        public DateTime BaslangicTarihi { get; set; } // Etkinlik başlangıç tarihi

        public DateTime? BitisTarihi { get; set; } // Etkinlik bitiş tarihi (opsiyonel)

        [MaxLength(1000)]
        public string Aciklama { get; set; } // Etkinlik açıklaması (opsiyonel)
```

```
        [Required]
        public string KullaniciId { get; set; } // Etkinliği oluşturan kullanıcı (foreign key)

        public EtkinlikTuru EtkinlikTuru { get; set; } // Etkinlik türü (Etkinlik, Toplanti, Ziyaret, Diger, Sinav)

        public int? RefFirmaId { get; set; } // Firma bağlantısı (Toplanti, Ziyaret, Diger, Sinav için)

        public bool SilindiMi { get; set; } = false; // Soft delete için
        public bool IsClosed { get; set; } = false; // Etkinlik kapatma durumu
        public bool FirmaOnay { get; set; } = false; // Firma onayı
        public bool PersonelOnay { get; set; } = false; // Personel onayı

        [Required]
        public DateTime OlusturulmaTarihi { get; set; } = DateTime.Now; // Yeni alan

        // Navigation property: Kullanıcı
        [ForeignKey(nameof(KullaniciId))]
        public virtual ApplicationUser Kullanici { get; set; }

        // Navigation property: Firma
        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }

        // Navigation property: Etkinliğe atanan personeller
                public virtual ICollection<Etkinlik_Kullanici> EtkinlikKullanicilar { get; set; } = new
List<Etkinlik_Kullanici>();

        // Navigation property: Etkinliğe bağlı sınav oturumları (Yeni)
                 public virtual ICollection<SinavOturumlari> SinavOturumlari { get; set; } = new
List<SinavOturumlari>();
    }
}



using System.ComponentModel.DataAnnotations.Schema;
```

```
namespace ilterisg.Models
{
   [Table("Etkinlik_Kullanici", Schema = "dbo")]
   public class Etkinlik_Kullanici
   {
      public int Id { get; set; } // Yeni birincil anahtar
      public int RefEtkinlikId { get; set; }
      public int RefKullaniciId { get; set; }

      [ForeignKey(nameof(RefEtkinlikId))]
      public virtual Etkinlikler Etkinlik { get; set; }

      [ForeignKey(nameof(RefKullaniciId))]
      public virtual Kullanicilar Kullanici { get; set; }
   }
}
```

Gereksiz alan yok güzel yapıalndırılmış
Ufak Öneriler (Zorunlu Değil, Opsiyonel İyileştirmeler):

Etkinlikler'e Index Ekle: Migration'da [Index(nameof(BaslangicTarihi))] veya composite index (EtkinlikTuru + RefFirmaId) koy—tarih/firma bazlı listeleme için hızlanır.
Junction'a Ek Potansiyel: Eğer katılım rolü (katılımcı/koordinatör) veya katılım tarihi lazımsa, KatilimTarihi DateTime? ve Rol string? ekle—ama şu an minimalist iyi.
Validation: Etkinlikler'e [Required]'leri genişlet (örneğin, BaslangicTarihi için), ama mevcut haliyle de model binding hatasız çalışır.
Genişleme: SinavOturumlari navigation'ı güzel—sınav etkinlikleri için one-to-many hazır.

FAALİYETALANI

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
```

```
namespace ilterisg.Models
{
    [Table("FaaliyetAlanlari", Schema = "dbo")]
    public class FaaliyetAlani
    {
        [Key]
        public int Id { get; set; }

        [Required, MaxLength(200)]
        public string MeslekGrubu { get; set; }

        [Required, MaxLength(50)]
        public string NaceKodu { get; set; }

        [Required, MaxLength(500)]
        public string NaceFaaliyetAdi { get; set; }
    }
}
```

Veri seti tablosu tsv dosyası set ediliyor güncellemeye gerek yok

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using ilterisg.Models.Enums;

namespace ilterisg.Models
{
    [Table("FAQ", Schema = "dbo")]
    public class Faq
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int FaqId { get; set; }
```

```
    [Required]
    [StringLength(200)]
    public string Baslik { get; set; }

    [Required]
    public string Icerik { get; set; }

    [Required]
    public KonuTuru KonuTuru { get; set; }

    [Required]
    public DateTime OlusturmaTarihi { get; set; } = DateTime.Now;

    public DateTime? GuncellemeTarihi { get; set; }

    [Required]
    [MaxLength(128)]
    public string OlusturanId { get; set; } // Admin'in AspNetUsers.Id'si
}

[Table("FaqFeedback", Schema = "dbo")]
public class FaqFeedback
{
    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    public int FeedbackId { get; set; }

    [Required]
    public int FaqId { get; set; }

    [Required]
    [MaxLength(128)]
    public string UserId { get; set; } // Oy veren kullanıcının AspNetUsers.Id'si

    [Required]
```

```
        public bool YararliOlduMu { get; set; }

        public DateTime Tarih { get; set; } = DateTime.Now;

        [ForeignKey("FaqId")]
        public virtual Faq Faq { get; set; }
    }
}
```

Gereksiz alan yok normalizasyon için bölüm eklemeye veya çıkarmaya gerek yok.

FATURA BİLGİLERİ

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("FaturaBilgileri", Schema = "dbo")]
    public class FaturaBilgileri
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        public int FirmaId { get; set; } // Firma ID’si (Firmalar tablosu ile ilişkilendirme)

        [Required]
        [MaxLength(500)]
        public string Adres { get; set; } // Fatura adresi

        [Required]
        [MaxLength(100)]
```

```
        public string Sehir { get; set; } // Şehir

        [Required]
        [MaxLength(100)]
        public string Ulke { get; set; } // Ülke

        [MaxLength(20)]
        public string? PostaKodu { get; set; } // Posta kodu (opsiyonel)

        [MaxLength(255)]
        public string IletisimAdi { get; set; } // İletişim adı (örneğin, ad soyad)

        [ForeignKey(nameof(FirmaId))]
        public virtual Firmalar? Firma { get; set; } // Nullable navigation property
    }
}
```

Iyzıco ihtiyacı olan alanlar evcut ekstradan Kullanıcılar içinde alan eklenebilir firma bilgisi tutuluyor şuan yalnızca.

FEATURED CONTENT

```
// Models/FeaturedContent.cs
namespace ilterisg.Models
{
    public class FeaturedContent
    {
        public int Id { get; set; }
        public string Section { get; set; } // "LatestPosts", "RecommendedPosts", "PopularPosts"
        public int BlogPostId { get; set; }
        public BlogPost BlogPost { get; set; }
        public int DisplayOrder { get; set; } // Sıralama için
    }
```

}

Blog postları sıralaması için Popüler Önerilen postlar gibi alanlar için tablo
Yapı genel olarak çok iyi.
Sadece Section alanı için enum kullanımı ve PublishedAt gibi bir alan eklemen, uzun vadede yönetimi kolaylaştırır.

FİRMA DEPARTMAN

Proje açılırken oluşturuldu henüz yapılandırılması yok

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Firma_Departman", Schema = "dbo")]
    public class Firma_Departman
    {
        [Key, Column(Order = 0)]
        public int RefFirmaId { get; set; }

        [Key, Column(Order = 1)]
        public int RefDepartmanId { get; set; }

        // Firma tablosuna ait FK
        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }

        // Departmanlar tablosuna ait FK
```

```
        [ForeignKey(nameof(RefDepartmanId))]
        public virtual Departmanlar Departman { get; set; }
    }
}
```

FİRMA EGİTİM

GEREKSİZ ESKI EGITIM SISTEMINDEN KALMA BIR TABLO KALDIRILABILIR

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Firma_Egitim", Schema = "dbo")]
    public class Firma_Egitim
    {
        [Key, Column(Order = 0)]
        public int RefFirmaId { get; set; } // Firmalar tablosuna foreign key

        [Key, Column(Order = 1)]
        public int RefEgitimId { get; set; } // Egitimler tablosuna foreign key

        // Navigation properties
        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }

        [ForeignKey(nameof(RefEgitimId))]
        public virtual Egitimler Egitim { get; set; }
```

```
    }
}
```

999

FİRMA SEKTOR

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Firma_Sektor", Schema = "dbo")]
    public class Firma_Sektor
    {
        [Key, Column(Order = 0)]
        public int RefFirmaId { get; set; }

        [Key, Column(Order = 1)]
        public int RefSektorId { get; set; }

        // Navigation Property: Firmalar tablosuyla ilişki
        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }

        // Navigation Property: Sektorler tablosuyla ilişki
        [ForeignKey(nameof(RefSektorId))]
        public virtual Sektorler Sektor { get; set; }
    }
}
```

Proje başında açıldı suan kullanımı yok

FİRMA DAVETLER

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    public class FirmaDavetler
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int DavetId { get; set; }

        [Required]
        [MaxLength(256)]
        public string Email { get; set; }

        [Required]
        public int RefOSGBId { get; set; }

        [MaxLength(450)]
        public string? RefUserId { get; set; }

        [Required]
        public DateTime DavetTarihi { get; set; } = DateTime.Now;

        public bool? OnayDurumu { get; set; }

        [MaxLength(450)]
        public string? InvitationToken { get; set; }
```

```
        [ForeignKey(nameof(RefOSGBId))]
        public virtual OSGBler OSGB { get; set; }

        [ForeignKey(nameof(RefUserId))]
        public virtual ApplicationUser? User { get; set; }
    }
}
```

Normalizasyon 3NF’ye uygun
FK yapısı Doğru tanımlanmış
Navigation property Eksiksiz
Gereksiz alan Yok
Genişletilebilirlik İyi düzeyde (enum / index eklenebilir)
Migration uyumu Sorunsuz

Öneri

Email alanı için doğrulama
[EmailAddress] attribute’u eklenirse form bazlı validasyon kolaylaşır:

```
[Required, MaxLength(256), EmailAddress]
public string Email { get; set; }
```

FİRMAKATILIMTALEBİ

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using ilterisg.Models.Enums;

namespace ilterisg.Models
{
    [Table("FirmaKatilimTalebi", Schema = "dbo")]
```

```
    public class FirmaKatilimTalebi
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int IstekId { get; set; }

        [Required]
        public int RefFirmaId { get; set; }

        [Required]
        [MaxLength(450)]
        public string RefUserId { get; set; }

        [Required]
        public DateTime IstekTarihi { get; set; } = DateTime.Now;

        public ApprovalStatus? OnayDurumu { get; set; }

        [MaxLength(500)]
        public string? BasvuruNotu { get; set; }

        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }

        [ForeignKey(nameof(RefUserId))]
        public virtual ApplicationUser Kullanici { get; set; }
    }
}
```

Bu tablo tam anlamıyla doğru normalize edilmiş,
hiç gereksiz alan barındırmıyor,
FK ilişkileri doğru kurgulanmış,
ve Entity Framework tarafında sorunsuz çalışır.

KULLANICILAR-FİRMALAR-OSGBLER

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Kullanicilar", Schema = "dbo")]
    public class Kullanicilar
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int KullaniciId { get; set; }

        [Required]
        [MaxLength(100)]
        public string AdSoyad { get; set; }

        [Required]
        [MaxLength(11)]
        public string TcKimlikNo { get; set; }

        [Required]
        public DateTime DogumTarihi { get; set; }

        [Required]
        [MaxLength(50)]
        public string? Cinsiyet { get; set; }

        [MaxLength(250)]
        public string? Adres { get; set; }
```

```
        public DateTime? KayitTarihi { get; set; }

        [MaxLength(450)]
        public string RefUserId { get; set; }

        [ForeignKey(nameof(RefUserId))]
        public virtual ApplicationUser ApplicationUser { get; set; }

                public virtual ICollection<Kullanici_Egitim> KullaniciEgitimler { get; set; } = new
List<Kullanici_Egitim>();

                 public virtual ICollection<Kullanici_Firma> KullaniciFirmalar { get; set; } = new
List<Kullanici_Firma>();
        public virtual ICollection<Ek2Form> Ek2Formlar { get; set; } = new List<Ek2Form>();
        public ICollection<Muayene> Muayeneler { get; set; } = new List<Muayene>();

        /// <summary>
        /// Kullanıcının çevrimiçi olup olmadığını belirtir.
        /// </summary>
        public bool IsActive { get; set; }
        public DateTime? LastLoginTime { get; set; }
        /// Kullanıcının şifre değiştirmesi gerekip gerekmediğini belirtir.
        /// </summary>
        public bool ForcePasswordChange { get; set; } // Yeni alan

        // Yeni alan: Profil fotoğrafı için Dokumanlar tablosuna referans
        public int? ProfilFotografiDokumanId { get; set; }

        [ForeignKey(nameof(ProfilFotografiDokumanId))]
        public virtual Dokumanlar? ProfilFotografi { get; set; }
        // Yeni alan: Meslek Kodu
        public int? MeslekKoduId { get; set; }
        [ForeignKey(nameof(MeslekKoduId))]
        public virtual MeslekKodlari? MeslekKodu { get; set; }
    }
}
```

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Firmalar", Schema = "dbo")]
    public class Firmalar
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int FirmaId { get; set; }

        [Required]
        [MaxLength(150)]
        public string FirmaAdi { get; set; }

        [Required]
        [MaxLength(100)]
        public string VergiDairesi { get; set; }

        [Required]
        [MaxLength(20)]
        public string VergiNumarasi { get; set; }

        [MaxLength(50)]
        public string? SskSicilNo { get; set; }

        [MaxLength(250)]
        public string? Adres { get; set; }

        public int? CalisanSayisi { get; set; }

         public int? NaceKoduId { get; set; } // Yeni foreign key sütunu, NaceKodlari tablosunun Id sütununa
bağlanacak
```

```
        [ForeignKey(nameof(NaceKoduId))]
        public virtual NaceKodlari? NaceKodu { get; set; }

        public int? FaaliyetAlaniId { get; set; } // Yeni foreign key
        [ForeignKey(nameof(FaaliyetAlaniId))]
        public virtual FaaliyetAlani? FaaliyetAlani { get; set; } // Yeni navigation property

        public bool? IsActive { get; set; }
        public bool? IsApproved { get; set; }

        [Column("KayıtTarihi")]
        public DateTime? KayitTarihi { get; set; }

        // TehlikeSinifi sütununu kaldırdık, bu bilgi NaceTehlikeSiniflari tablosunda tutulacak
        public bool? IsOSGB { get; set; }

        public string? ApplicationUserId { get; set; }

        [ForeignKey(nameof(ApplicationUserId))]
        public virtual ApplicationUser? ApplicationUser { get; set; }

        public virtual ICollection<Firma_Departman> Firma_Departman { get; set; }
            = new HashSet<Firma_Departman>();

        public virtual ICollection<Firma_Sektor> Firma_Sektor { get; set; }
            = new HashSet<Firma_Sektor>();
        public virtual ICollection<OSGB_Firmalar> OSGB_Firmalar { get; set; }
            = new HashSet<OSGB_Firmalar>();

        public virtual ICollection<Firma_Egitim> FirmaEgitimler { get; set; }
            = new HashSet<Firma_Egitim>();

        public virtual ICollection<Ek2Form> Ek2Formlar { get; set; } = new HashSet<Ek2Form>();
    }
}
```

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    public class OSGBler
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int OSGBId { get; set; }

        [Required]
        [MaxLength(255)]
        public string OSGBAdi { get; set; }

        [Required]
        public string Adres { get; set; }

        [Required]
        [MaxLength(20)]
        public string Telefon { get; set; }

        [Required]
        [Column("KayıtTarihi")]
        public DateTime KayitTarihi { get; set; }

        [Required]
        public int RefFirmaId { get; set; }

        [Required]
        public string ApplicationUserId { get; set; }

        [MaxLength(100)]
        public string? SubMerchantKey { get; set; } // Nullable SubMerchantKey
```

```
        [MaxLength(34)] // IBAN uzunluğu genelde 34 karakter
        public string? Iban { get; set; } // Nullable IBAN
        public string? SubMerchantExternalId { get; set; } // Yeni alan
        public string? HesapAdi { get; set; } // Yeni eklenen nullable HesapAdi alanı

        // Navigation property'ler
        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }

        [ForeignKey(nameof(ApplicationUserId))]
        public virtual ApplicationUser ApplicationUser { get; set; }

                public virtual ICollection<OSGB_Firmalar> OSGB_Firmalar { get; set; } = new
List<OSGB_Firmalar>();
        public virtual ICollection<Ek2Form> Ek2Formlar { get; set; } = new List<Ek2Form>();



    }
}
```

Çok güzel — bu üç tablo (Firmalar, OSGBler, Kullanicilar) sisteminin en temel “çekirdek” modelleri, yani senin domain’inde işveren, OSGB ve kullanıcı varlıklarını temsil ediyorlar.
Bu yüzden normalizasyonun burada doğru olması, tüm sistemin veri bütünlüğünü etkiliyor.
Kodlarını inceledim, üçü de çok iyi normalize edilmiş.
Ancak profesyonel olarak inceleyip artılarını ve küçük geliştirme fırsatlarını da aşağıda özetledim

1. Firmalar Tablosu

Genel Değerlendirme

Alan	Durum	Açıklama
FirmaId	Otomatik PK.	
FirmaAdi, VergiDairesi, VergiNumarasi	Temel kimlik bilgileri, doğru tür ve kısıtlar.	
SskSicilNo, Adres	Opsiyonel, atomik değerler.	

CalisanSayisiSayısal değer, doğru tip.
NaceKoduId, FaaliyetAlaniIdDış anahtarlar, ilişkisel bütünlük sağlanmış.
IsActive, IsApproved, IsOSGBDurum alanları, mantıklı.
KayitTarihiTarih alanı, nullable ve uygun.
ApplicationUserIdFirma sahibini/hesabını bağlar, FK doğru.
ICollection navigation’lar1:N ilişkiler doğru modellenmiş.
⚙ Normalizasyon Analizi

1NF: Her alan atomik (örneğin adres, tek string).

2NF: Bütün alanlar sadece FirmaId’ye bağımlı.

3NF: Transitif bağımlılık yok (örneğin NACE kodu başka tabloya taşınmış — çok doğru).

⚠ Küçük İyileştirme Fikirleri

IsActive, IsApproved gibi boolean çiftleri yerine
bir enum daha açıklayıcı olur:

public enum FirmaDurumu { Pasif, Beklemede, Aktif, Onayli }

[Column("KayıtTarihi")] alan adını Türkçe yazman gerekmez; veritabanında karışıklık yapabilir.
İngilizce (KayitTarihi) olarak bırakmak genelde daha temiz olur.

Sonuç

3NF düzeyinde, tamamen doğru yapı.
Her alan doğru bağımlılığa sahip, hiçbir tekrar veya gereksizlik yok.

1NF: Tüm alanlar atomik.

2NF: Her alan KullaniciId’ye tam bağımlı.

3NF: Transitif bağımlılık yok — örneğin “meslek kodu” ilişkisel tabloya taşınmış, mükemmel.

⚠ Küçük İyileştirme Fikirleri

Cinsiyet string olarak tutuluyor; bu ileride “Erkek”, “ERKEK”, “e” gibi dağılabilir.
Bunun yerine küçük bir enum kullanmak daha doğru olur:

public enum Cinsiyet { Erkek, Kadin, Diger }

TcKimlikNo benzersiz olmalı:

[Index(nameof(TcKimlikNo), IsUnique = true)]

RefUserId [Required] olarak tanımlanabilir — aksi halde orphan user (ApplicationUser’sız) oluşabilir.

4. İlişkiler Arası Normalizasyon Kontrolü

İlişkiTürAçıklama
Firma ↔ OSGB1:NHer firma birden fazla OSGB’ye sahip olabilir.
Firma ↔ KullaniciN:N (Kullanici_Firma üzerinden)Doğru, normalizasyonu sağlıyor.
OSGB ↔ KullaniciDolaylı (Firmalar üzerinden)İlişki zinciri doğru.
Firma ↔ Eğitim / Departman / SektörN:NDoğru şekilde ara tablolarla kurulmuş.

Tüm bu bağlantılar, EF Core’un ilişki çözümleme algoritmasına tamamen uygun.
Hiçbir “many-to-many” doğrudan tutulmuyor — aratablolarla ayrılmış.
Bu da tam normalize edilmiş anlamına gelir.

Sonuç: Üç Tablo Arasındaki Normalizasyon Özeti

TabloNormalizasyon DüzeyiGereksiz AlanÖneri / İyileştirme
Firmalar3NF YokEnum ile durum yönetimi yapılabilir.
OSGBler 3NF YokDefault tarih eklenebilir.
Kullanicilar3NF YokCinsiyet enum’a taşınmalı, TC kimlik unique olmalı.

Genel Değerlendirme

Bu üç tablo birlikte çalıştığında:

Veri tekrarı hiç yok,

FK yapıları doğru ve mantıklı,

Alan bağımlılıkları doğru anahtarlara bağlı,

Sistem tam 3. normal form'da.

Kısacası: Tam normalize, kurumsal seviyede güçlü bir veri modeli.
Sadece küçük "kalite iyileştirmeleri" (enum'lar, unique index) önerilebilir.

İstersen bu üç tabloyu birlikte (Firmalar ↔ OSGBler ↔ Kullanicilar) içeren bir ER diyagramı (ilişki görseli) hazırlayayım;
bağlantıların nasıl kurulduğunu net şekilde görebilirsin.
İster misin o diyagramı da oluşturayım?

```
using Microsoft.AspNetCore.Mvc.ModelBinding;
using Microsoft.AspNetCore.Mvc.ModelBinding.Validation;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    public class MuayeneTeshis
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        [BindNever]
        public int Id { get; set; }
```

```
        public int? MuayeneId { get; set; }
        [ForeignKey(nameof(MuayeneId))]
        [ValidateNever]
        public Muayene? Muayene { get; set; }

        // SADECE ReceteId string → navigation YOK
        [StringLength(50)]
        public string? ReceteId { get; set; }

        // KALDIR: public MuayeneRecete? Recete { get; set; }

        [Required, StringLength(50)]
        public string Kod { get; set; } = "";

        [Required, StringLength(1000)]
        public string Adi { get; set; } = "";

        public long? TcKimlikNo { get; set; } // Hasta TC'si (Medula'dan gelir, zorunlu)
    }
}
```

Tamamen normalize gereksiz alan yok Muayene var ise MuayeneID ile bağlanıyor her reçete ReceteID ile bağlanıyor bu ileride ProtokolNo ile değişebilir Reçetede birden fazla tan olacağı için bu senaryoyada uygun

TcKimlikNo alanını ayrıca tutmak, kullanıcı olmayan kişiler için teşhisleri ayırt etmek —
tamamen doğru, profesyonel ve sağlık sistemleri için standart bir uygulamadır.

Yani bu tasarımı kesinlikle korumalısın.

NACEKODLARI

```
using System.ComponentModel.DataAnnotations;
```

```
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("NaceKodlari", Schema = "dbo")]
    public class NaceKodlari
    {
        [Key]
        public int Id { get; set; }

        [Required]
        public string SektorKodu { get; set; } // LONGTEXT (MySQL) / nvarchar(max) (SQL Server)

        [Required]
        public string SektorTanim { get; set; } // LONGTEXT (MySQL) / nvarchar(max) (SQL Server)

        [Required]
        public string MeslekKodu { get; set; } // LONGTEXT (MySQL) / nvarchar(max) (SQL Server)

        [Required]
        public string MeslekTanim { get; set; } // LONGTEXT (MySQL) / nvarchar(max) (SQL Server)

        [MaxLength(50)]
        public string NaceRev21Kodu { get; set; }

        [Required]
        public string NaceRev21Tanim { get; set; } // LONGTEXT (MySQL) / nvarchar(max) (SQL Server)

        public virtual NaceTehlikeSiniflari NaceTehlikeSinifi { get; set; }
            public virtual ICollection<RiskDegerlendirme> RiskDegerlendirmeler { get; set; } = new
HashSet<RiskDegerlendirme>();
        public virtual ICollection<Firmalar> Firmalar { get; set; } = new HashSet<Firmalar>();
    }
}
```

Veri Seti tablosu değişie gerek yok

NaceTehlikeSınıfları

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("NaceTehlikeSiniflari", Schema = "dbo")]
    public class NaceTehlikeSiniflari
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; } // Birincil anahtar olarak Id ekledik

        public int NaceKoduId { get; set; } // NaceRev21Kodu yerine NaceKoduId kullanıyoruz

        public TehlikeSinifi TehlikeSinifi { get; set; } // Enum: AzTehlikeli, Tehlikeli, CokTehlikeli

        [ForeignKey(nameof(NaceKoduId))]
        public virtual NaceKodlari NaceKodu { get; set; } // NaceKodlari ile 1:1 ilişki
    }
}
```

Veri Seti tablosu değişime gerek yok

ÖDEMELER

Iyzıco İşlemleri için Ödemeler tablosu

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Odemeler", Schema = "dbo")]
    public class Odemeler
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int OdemeId { get; set; }

        [Required]
        public int RefSozlesmeId { get; set; }

        [Column(TypeName = "decimal(18, 2)")]
        public decimal Tutar { get; set; }

        [Required]
        public DateTime OdemeTarihi { get; set; }

        [Required]
        public string OdemeYontemi { get; set; }

        [Required]
        public bool YapildiMi { get; set; }

        [Required]
        public string OnayDurumu { get; set; }

        public DateTime? OnayTarihi { get; set; }
        // iyzico'dan dönen PaymentId'yi saklamak için yeni alan
        public string? IyzicoPaymentId { get; set; }

        // Navigation Property: Sozlesmeler tablosuyla FK ilişkisi
```

```
        [ForeignKey(nameof(RefSozlesmeId))]
        public virtual Sozlesmeler Sozlesme { get; set; }
    }
}
```

OSGB_Firmalar

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("OSGB_Firma", Schema = "dbo")]
    public class OSGB_Firmalar
    {
        [Key, Column(Order = 0)]
        public int RefOSGBId { get; set; } // OSGBler tablosuna foreign key

        [Key, Column(Order = 1)]
        public int RefFirmaId { get; set; } // Firmalar tablosuna foreign key

        // Navigation property: OSGBler tablosu
        [ForeignKey(nameof(RefOSGBId))]
        public virtual OSGBler OSGB { get; set; }

        // Navigation property: Firmalar tablosu
        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }
    }
}
```

Tamamen Normalize OSGB Firmalarını tutmak için bir junction table

PERSONEL DAVETLER

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    public class PersonelDavetler
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int DavetId { get; set; }

        [MaxLength(450)]
        public string? RefUserId { get; set; }

        [Required]
        public int RefFirmaId { get; set; }

        [Required]
        [MaxLength(50)]
        public string Rol { get; set; }

        [MaxLength(256)]
        public string? Email { get; set; } // Yeni e-posta alanı

        public DateTime DavetTarihi { get; set; } = DateTime.Now;

        public bool? OnayDurumu { get; set; }

        [MaxLength(450)]
        public string? InvitationToken { get; set; }
        public int? MeslekKoduId { get; set; } // Yeni nullable alan

        [ForeignKey(nameof(RefUserId))]
```

```
        public virtual ApplicationUser? Kullanici { get; set; }

        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }
    }
}
```

Normalize mi, Eksik varmı?
Hayır, kritik eksik yok—temel davet sistemi (firma + rol + token + onay) tam kapsanmış. Ama ufak audit/esneklik için şu alanlar eklenebilir (zorunlu değil, mevcut haliyle de iş görür):

OlusturanUserId: Davet eden kullanıcıyı track etmek için (string, max 450, FK to AspNetUsers). Neden? Audit/raporlama için (kim kimi davet etti)—şu an RefUserId davet edilen, ama oluşturan ayrı olmalı.
ExpiryDate: DateTime? (davet token süresi için, default DateTime.Now.AddDays(7)). Neden? Token'lar sonsuz kalmasın, güvenlik için.
DavetNotu: string? (max 500, opsiyonel mesaj). Neden? Email'de kişiselleştirme için, ama basit tutmak istersen gereksiz.

Öneriler (Opsiyonel İyileştirmeler):

Enum Kullanımı:

Rol: String yerine enum (örneğin, public RolEnum Rol { get; set; }—Doktor, Personel, Admin vb.). Neden? Validation ve dropdown'lar için daha güvenli, string hatalarını önler.
OnayDurumu: Bool? yerine enum (örneğin, public DavetDurumu? OnayDurumu { get; set; }—Beklemede, KabulEdildi, Reddedildi, SüresiDoldu). Neden? Reddetme gibi ekstra durumlar için esnek, bool yetersiz kalabilir.

POZİSYONLAR

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
```

```
namespace ilterisg.Models
{
    [Table("Pozisyonlar", Schema = "dbo")]
    public class Pozisyonlar
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        [MaxLength(100)]
        public string PozisyonAdi { get; set; }
    }
}
```

Açıldı ama yapıalndırma uygulanmadı planlama aşamasında kuruldu

RANDEVU

```
// Dosya: Models/Randevu.cs
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    public enum RandevuDurum
    {
        Beklemede,
        Onaylandi,
```

```
    Reddedildi,
    Iptal
}

public class Randevu
{
    [Key]
    public int RandevuId { get; set; }

    [Required]
    [Display(Name = "Hasta")]
    public int HastaId { get; set; }

    [ForeignKey(nameof(HastaId))]
    public Kullanicilar Hasta { get; set; }

    [Required]
    [Display(Name = "Doktor")]
    public string DoktorId { get; set; }

    // Eğer ApplicationUser kullanıyorsanız
    [ForeignKey(nameof(DoktorId))]
    public ApplicationUser Doktor { get; set; }

    [Required]
    [Display(Name = "Başlangıç Zamanı")]
    public DateTime BaslangicZamani { get; set; }

    [Required]
    [Display(Name = "Bitiş Zamanı")]
    public DateTime BitisZamani { get; set; }

    [Required]
    [Display(Name = "Durum")]
    public RandevuDurum Durum { get; set; }
}
```

}

Ertugrulun Kurdugu andevu sistemi

Sonuç:
Evet, normalizasyon iyi – randevu yönetimi için sağlam temel (redundansi yok, ilişkiler sağlıklı). Bu haliyle deploy-ready, ama audit/validation tweak'leriyle (özellikle zaman çakışması için) daha robust olur.

RiskANALİZDAVETLER

```
using System;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("RiskAnalizDavetler", Schema = "dbo")]
    public class RiskAnalizDavet
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        public Guid AnalizGrupId { get; set; } // Davet edilen risk analizi grubu

        [Required]
        [MaxLength(450)]
        public string DoktorUserId { get; set; } // Davet edilen işyeri hekiminin UserId'si
        [ForeignKey(nameof(DoktorUserId))]
        public virtual ApplicationUser? DoktorUser { get; set; } // İlişkili kullanıcı
```

```
        public int? FirmaId { get; set; } // Opsiyonel: Firma bağlantısı
        [ForeignKey(nameof(FirmaId))]
        public virtual Firmalar? Firma { get; set; } // Nullable firma ilişkisi

        [Required]
        public Guid Token { get; set; } // Benzersiz erişim token'ı (Guid'e çevir, otomatik generate)

        [Required]
        public DateTime DavetTarihi { get; set; } // Davet oluşturulma tarihi

        public DateTime? SonGecerlilikTarihi { get; set; } // Token'ın geçerlilik süresi

        public bool KullanildiMi { get; set; } = false; // Token kullanıldı mı?

        [Required]
        [MaxLength(450)]
        public string DavetEdenUserId { get; set; } // Davet eden kullanıcının UserId'si
        [ForeignKey(nameof(DavetEdenUserId))]
        public virtual ApplicationUser? DavetEdenUser { get; set; } // Davet eden kullanıcı

        // Audit ekleri
        public DateTime CreatedAt { get; set; } = DateTime.UtcNow;
        public DateTime? UpdatedAt { get; set; }

        public bool SilindiMi { get; set; } = false; // Soft delete

        // Opsiyonel: Notlar
        [MaxLength(500)]
        public string? Notlar { get; set; }
    }
}
```

Normalizasyona uygun Davet eden kişi isg uzmanı değil Firma ise FirmaID Doluyor ama DavetEdenUSERID'de mevcut firma sorumlusu ıd atanıp firma ıd alanı silinerek revize edilebilir.

RİSKANALZ KAYDI

Risk Analizlerinin Bulundugu Tablo

```
using ilterisg.Models.Enums; // SahaEnum için
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
namespace ilterisg.Models
{
    [Table("RiskAnalizKayitlari", Schema = "dbo")]
    public class RiskAnalizKaydi
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }
        public int? FirmaId { get; set; } // Nullable yapıldı
        [ForeignKey(nameof(FirmaId))]
        public Firmalar? Firma { get; set; } // Nullable yapıld
        public string? UserId { get; set; }
        [ForeignKey(nameof(UserId))]
        public ApplicationUser? User { get; set; }
        [MaxLength(200)]
        public string RiskAdi { get; set; } = string.Empty;
        [MaxLength(500)]
        public string RiskAciklamasi { get; set; } = string.Empty;
        [MaxLength(500)]
        public string? OnerilenOnlem { get; set; }
        [Range(0.2, 10, ErrorMessage = "Olasılık 0.2 ile 10 arasında olmalıdır.")]
        public double? Olasilik { get; set; }
        [Range(1, 5, ErrorMessage = "Şiddet 1 ile 5 arasında olmalıdır.")]
        public int? Siddet { get; set; }
        [NotMapped]
        public int? RiskSkoru => Olasilik.HasValue && Siddet.HasValue ? (int?)(Olasilik.Value * Siddet.Value)
: null; // Cast to int for 5x5 compatibility
```

```
    [MaxLength(50)]
    public string AnalizMetodu { get; set; } = "5x5";
    [Range(0.5, 10)]
    public double? Maruziyet { get; set; }
    [Range(1, 100)]
    public double? FinneySiddet { get; set; }
    [NotMapped]
    public double? FinneySkor => Olasilik.HasValue && Maruziyet.HasValue && FinneySiddet.HasValue
        ? Olasilik.Value * Maruziyet.Value * FinneySiddet.Value : null;
    public int? RiskDegerlendirmeId { get; set; }
    [ForeignKey(nameof(RiskDegerlendirmeId))]
    public RiskDegerlendirme? KaynakRisk { get; set; }
    public SahaEnum Saha { get; set; } = SahaEnum.GenelSaha;
    [MaxLength(50)]
    public string TeminSuresi { get; set; } = "3 ay";
    [MaxLength(500)]
    public string? Notlar { get; set; }
    public int? MevzuatId { get; set; }
    [ForeignKey(nameof(MevzuatId))]
    public Mevzuat? Mevzuat { get; set; }
    public DateTime Tarih { get; set; } = DateTime.Now;
    public Guid? AnalizGrupId { get; set; }
    [NotMapped]
    public int Adim { get; set; }
    [ForeignKey(nameof(RefBildirimId))]
    public RiskBildirimleri? Bildirim { get; set; }
    public int? RefBildirimId { get; set; }
    [MaxLength(50, ErrorMessage = "Revizyon numarası 50 karakterden fazla olamaz.")]
    public string? RevizyonNo { get; set; } // Yeni eklenen nullable alan
    public DateTime? RevizyonTarihi { get; set; } // Yeni eklenen nullable alanmigr
    public string? RiskZarari { get; set; }
      public string? OneriYapanDoktorId { get; set; } // Öneriyi yapan İşyeri Hekimi'nin ApplicationUser
Id'si
    [ForeignKey(nameof(OneriYapanDoktorId))]
    public ApplicationUser? OneriYapanDoktor { get; set; } // Nullable foreign key
```

```
        public bool IsApproved { get; set; } = true; // Varsayılan olarak true (normal analiz için), öneriler için false
    }
}
```

RevizyonNo ve Revizyon tariji JUNCTİON Tableye taşınmalı 100 adet risk varsa yüzüde guncellenmemelı revizyon için. Önerilen yapı revize kısmının junction table ıle yönetilmesi.

RİSK BİLDİRİMLERİ - RİSK BİLDİRİM MESJALARI

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using ilterisg.Models.Enums;

namespace ilterisg.Models
{
    [Table("RiskBildirimleri", Schema = "dbo")]
    public class RiskBildirimleri
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int BildirimId { get; set; }

        [Required]
        [MaxLength(200)]
        public string KonuBasligi { get; set; } = "Risk Bildirimi";

        [Required]
        public string Icerik { get; set; }

        [Required]
```

```
    [MaxLength(450)] // AspNetUsers.Id ile eşleşiyor
    public string GonderenId { get; set; } // İSG Uzmanı UserId

    [ForeignKey(nameof(GonderenId))]
    public virtual ApplicationUser Gonderen { get; set; }

    [Required]
    [MaxLength(450)] // AspNetUsers.Id ile eşleşiyor
    public string AliciId { get; set; } // İşveren UserId

    [ForeignKey(nameof(AliciId))]
    public virtual ApplicationUser Alici { get; set; }

    [Required]
    public int RefFirmaId { get; set; } // İlgili firma

    [ForeignKey(nameof(RefFirmaId))]
    public virtual Firmalar Firma { get; set; }

    public int? OsgbId { get; set; } // İlgili OSGB (nullable, opsiyonel)

    [ForeignKey(nameof(OsgbId))]
    public virtual OSGBler OSGB { get; set; }

    [Required]
    public DateTime OlusturmaTarihi { get; set; } = DateTime.Now;

    public DateTime? GuncellemeTarihi { get; set; } // Nullable, güncelleme opsiyonel

    [Required]
    public DestekTalebiDurumu Durum { get; set; } = DestekTalebiDurumu.Acik;

                public virtual ICollection<RiskBildirimMesajlari> Mesajlar { get; set; } = new
List<RiskBildirimMesajlari>();
    public virtual ICollection<Dokumanlar> Dokumanlar { get; set; } = new List<Dokumanlar>();
  }
```

```
}



using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("RiskBildirimMesajlari", Schema = "dbo")]
    public class RiskBildirimMesajlari
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int MesajId { get; set; }

        [Required]
        public int RefBildirimId { get; set; } // RiskBildirimleri.BildirimId ile bağlanacak

        [ForeignKey(nameof(RefBildirimId))]
        public virtual RiskBildirimleri RiskBildirimi { get; set; }

        [Required]
        public string MesajIcerigi { get; set; }

        [Required]
        [MaxLength(450)] // AspNetUsers.Id ile eşleşiyor
        public string GonderenId { get; set; } // ApplicationUser Id

        [ForeignKey(nameof(GonderenId))]
        public virtual ApplicationUser Gonderen { get; set; }

        [Required]
        public DateTime GonderimTarihi { get; set; } = DateTime.Now;
    }
}
```

4. Gereksiz Alan Var mı?

Hayır.
Her alanın bir amacı var:

AliciId ve GonderenId birlikte gerekli (mesaj yönü için).

OsgbId opsiyonel ama farklı kurumsal türleri destekler.

FirmaId zorunlu — kurumsal bağlam.

Durum, Tarih, Icerik hepsi atomik ve işlevsel.

Hiçbir alan “fazlalık” ya da “türev” değil.
Hiçbir bilgi başka alandan türetilemiyor — bu da 3NF’nin özüdür.

Sonuç
TabloNormalizasyonGereksiz AlanNot
RiskBildirimleri 3NF YokTüm alanlar anlamlı ve ayrı işlevde.
RiskBildirimMesajlari 3NF YokSade ve efektif.
Kısa Özet:

Modeller tamamen normalize (3. normal forma kadar).

OsgbId’nin nullable olması doğru, çünkü opsiyonel.

AliciId ve GonderenId ayrı FK olarak tasarlanması doğru modelleme yaklaşımı.

Gereksiz veya silinmesi gereken hiçbir alan yok.

Sadece “performans” amaçlı birkaç index önerisi yapılabilir.

Yani bu iki tablo üretim düzeyinde normalize,
veri bütünlüğü sağlam,
iş modeli açısından mükemmel orantılı bir tasarıma sahip.

RiskDeğerlenDirme
Veri seti tablosu revizeye gerek yok

```
using ilterisg.Models.Enums;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    // Veritabanında dbo şemasında RiskDegerlendirme tablosuna karşılık gelir
    [Table("RiskDegerlendirme", Schema = "dbo")]
    public class RiskDegerlendirme
    {
        // Birincil anahtar (primary key), otomatik artan (identity)
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        // NaceKodlari tablosuna foreign key, zorunlu alan
        public int? NaceKoduId { get; set; }

        // Riskin tanımı (örn. "Yüksekten düşme"), zorunlu alan, maksimum 200 karakter
        [Required]
        [MaxLength(200)]
        public string RiskTanimi { get; set; }

            // Riskin oluşmasına neden olan faktörler (örn. "Kaygan zemin, yetersiz aydınlatma"), opsiyonel, maksimum 500 karakter
```

```
    [MaxLength(500)]
    public string RiskFaktorleri { get; set; }

    // Riski azaltmak için alınacak önlemler (örn. "Kaymaz zemin, uyarı levhaları"), opsiyonel, maksimum 500 karakter
    [MaxLength(500)]
    public string KontrolOnlemleri { get; set; }

    // Fine-Kinney metodolojisi için olasılık (P), 0.2-10 arası, opsiyonel
    [Range(0.2, 10)]
    public double? OlasilikFineKinney { get; set; } // double olarak değiştirildi

    // Fine-Kinney metodolojisi için şiddet (S), 1-100 arası, opsiyonel
    [Range(1, 100)]
    public double? SiddetFineKinney { get; set; } // double olarak değiştirildi

    // Fine-Kinney metodolojisi için frekans (F), 0.5-10 arası, opsiyonel
    [Range(0.5, 10)]
    public double? FrekansFineKinney { get; set; } // double olarak değiştirildi

    // Fine-Kinney risk skoru (P × S × F), opsiyonel
    public double? RiskSkoruFineKinney { get; set; } // double olarak değiştirildi

    // Fine-Kinney risk seviyesi (örn. "Düşük Risk", "Yüksek Risk"), opsiyonel, maksimum 50 karakter
    [MaxLength(50)]
    public string RiskSeviyesiFineKinney { get; set; }

    // 5x5 metodolojisi için olasılık (L), 1-5 arası, opsiyonel
    [Range(1, 5)]
    public int? Olasilik5x5 { get; set; }

    // 5x5 metodolojisi için şiddet (I), 1-5 arası, opsiyonel
    [Range(1, 5)]
    public int? Siddet5x5 { get; set; }

    // 5x5 risk skoru (L × I), opsiyonel
```

```
        public int? RiskSkoru5x5 { get; set; }

        // 5x5 risk seviyesi (örn. "Düşük Risk", "Çok Yüksek Risk"), opsiyonel, maksimum 50 karakter
        [MaxLength(50)]
        public string RiskSeviyesi5x5 { get; set; }

        // Mevzuat tablosuna foreign key, opsiyonel
        public int? MevzuatId { get; set; }

        // NaceKodlari tablosu ile ilişki (navigation property)
        [ForeignKey(nameof(NaceKoduId))]
        public virtual NaceKodlari NaceKodu { get; set; }

        // Mevzuat tablosu ile ilişki (navigation property)
        [ForeignKey(nameof(MevzuatId))]
        public virtual Mevzuat Mevzuat { get; set; }
        public SahaEnum? Saha { get; set; } // Nullable SahaEnum alanı
        public bool OnaylandiMi { get; set; } = false;
        // Mevcut modele RiskZarari ekle
        [MaxLength(500)]
        public string? RiskZarari { get; set; }
    }
}
```

RıskOnlemSonrasiDurum

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
```

```
{
    [Table("RiskOnlemSonrasiDurum", Schema = "dbo")]
    public class RiskOnlemSonrasiDurum
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required(ErrorMessage = "Risk analizi kaydı zorunludur.")]
        public int RiskAnalizKaydiId { get; set; }

        [ForeignKey(nameof(RiskAnalizKaydiId))]
        public RiskAnalizKaydi RiskAnalizKaydi { get; set; } = null!;

        [Required(ErrorMessage = "Analiz grup ID zorunludur.")]
        public Guid AnalizGrupId { get; set; }

        [MaxLength(500, ErrorMessage = "Mevcut durum 500 karakterden fazla olamaz.")]
        public string? MevcutDurum { get; set; }

        [MaxLength(500, ErrorMessage = "Önlem sonrası durum 500 karakterden fazla olamaz.")]
        public string? OnlemSonrasiDurum { get; set; } = string.Empty;

        [Range(0.2, 10, ErrorMessage = "Olasılık 0.2 ile 10 arasında olmalıdır (Fine-Kinney için). 5x5 için 1-5
kullanın.")]
        public double? Olasilik { get; set; } // Fine-Kinney: 0.2-10, 5x5: 1-5

        [Range(0.5, 10, ErrorMessage = "Maruziyet 0.5 ile 10 arasında olmalıdır (Fine-Kinney için).")]
        public double? Maruziyet { get; set; } // Fine-Kinney için, 5x5'te null

        [Range(1, 100, ErrorMessage = "Finney Şiddet 1 ile 100 arasında olmalıdır (Fine-Kinney için).")]
        public double? FinneySiddet { get; set; } // Fine-Kinney için, 5x5'te null

        [MaxLength(500, ErrorMessage = "Ek notlar 500 karakterden fazla olamaz.")]
        public string? EkNotlar { get; set; }
```

```
        [MaxLength(250, ErrorMessage = "Sorumlu kişi 250 karakterden fazla olamaz.")]
        public string? SorumluKisi { get; set; }

        [MaxLength(500, ErrorMessage = "Etkilenen kişiler 500 karakterden fazla olamaz.")]
        public string? EtkilenenKisiler { get; set; }

        public DateTime GuncellemeTarihi { get; set; } = DateTime.UtcNow;
    }
}
```

tamamen normalize sadece Guncelleme tarihi junction table ile yonetılebılır rıskanalız kaydı gibi

SeçiliHekimBilgileri

Tamamen normalize Medula için ilgili varsayılan bilgilerin kaydedildiği bir junction table

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    public enum SecimTuruEnum
    {
        [Display(Name = "TesisKoduSecimi")]
        TesisKodu = 1,
        [Display(Name = "BransKoduSecimi")]
        BransKodu = 2,
        [Display(Name = "SertifikaSecimi")]
        Sertifika = 3,
        [Display(Name = "MedulaSifresiSecimi")]
        MedulaSifresi = 4
    }
```

```
    [Table("SeciliHekimBilgileri", Schema = "dbo")]
    public class SeciliHekimBilgileri
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Required]
        public int KullaniciId { get; set; }

        [Required]
        public int IsyeriHekimiBilgileriId { get; set; }

        [ForeignKey("IsyeriHekimiBilgileriId")]
        public virtual IsyeriHekimiBilgileri IsyeriHekimiBilgileri { get; set; }

        [Required]
            public SecimTuruEnum SecimTuru { get; set; } // Hangi alan seçildi? (TesisKodu, BransKodu,
Sertifika)

        public DateTime SecimTarihi { get; set; } = DateTime.Now; // Seçim zamanı
    }
}
```

Sektorler

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
```

```
    [Table("Sektorler", Schema = "dbo")]
    public class Sektorler
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int SektorId { get; set; }

        [Required]
        [MaxLength(50)]
        public string SektorAdi { get; set; }

        // Navigation Property: Firma_Sektor tablosuyla ilişki
                        public virtual ICollection<Firma_Sektor> Firma_Sektor { get; set; } = new
HashSet<Firma_Sektor>();
    }
}
```

Planlamada yapılandırıldı hiçbir yere bağlı değil kullanılmıyor.

SektorRiski

```
using ilterisg.Models; // Sektorler sınıfı bu namespace'te
using System.ComponentModel.DataAnnotations.Schema;

public class SektorRiski
{
    public int Id { get; set; }

    public int SektorId { get; set; }
    [ForeignKey("SektorId")]
    public Sektorler Sektor { get; set; }
```

```
    public string RiskAdi { get; set; } = string.Empty;
    public string RiskAciklamasi { get; set; } = string.Empty;
    public string OnerilenOnlem { get; set; } = string.Empty;

    // 5x5 matrise özel alanlar
    public int Olasilik { get; set; }  // 1-5
    public int Siddet { get; set; }    // 1-5
}
```

Planlamada yapılandırıldı hiçbir yere bağlı değil kullanılmıyor.

YanlışBilgiBildirimi

```
namespace ilterisg.Models
{
    public class YanlisBilgiBildirimi
    {
        public int Id { get; set; }

        // Yanlış bilgi olduğu belirtilen alanın adı ve değeri
        public string? AlanAdi { get; set; } // Hata yoksa null olabilir
        public string? MevcutDeger { get; set; } // Hata yoksa null olabilir
        public string? DogruDeger { get; set; } // Hata yoksa null olabilir

        // Kullanıcının bilgilerin doğru olup olmadığını onaylaması
        public bool IsBilgilerDogru { get; set; } = true; // Varsayılan true, nullable değil

        // Bildirim tarihi ve işlem durumu
        public DateTime? TalepTarihi { get; set; } // Hata yoksa null olabilir, varsayılanı kaldırdık
        public bool? IsProcessed { get; set; } // Hata yoksa null olabilir, yönetici işlemi yoksa anlamı yok
```

```
        // Yönetici işlemleri için ek alanlar
        public string? YoneticiNotu { get; set; } // Yönetici devreye girmezse null
        public DateTime? IslemTarihi { get; set; } // Yönetici devreye girmezse null

        // ApplicationUser ile ilişki
        public string ApplicationUserId { get; set; } // Foreign Key, nullable değil
        public ApplicationUser? ApplicationUser { get; set; } // Navigation Property
    }
}
```

Tamemen normalizeye uygun gereksiz alan yok, gereksiz navigation proerty yok.

Öneri

```
        // Kullanıcının bilgilerin doğru olup olmadığını onaylaması
        public bool IsBilgilerDogru { get; set; } = true; // Varsayılan true, nullable değil
```

Bu alan kaldırılabilir bilgiler doğruysa hiçbir veri set edilmez değilse edilir tek.

Subeler

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Subeler", Schema = "dbo")]
    public class Subeler
    {
```

```
        [Key]
        public int SubeId { get; set; }

        [Required]
        [MaxLength(50)]
        public string SubeAdi { get; set; }

        [Required]
        public int SicilNo { get; set; }

        [Required]
        public int RefFirmaId { get; set; }

        // Navigation Property: Firmalar tablosuyla FK ilişkisi
        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }
    }
}
```

Planama tarafında açıldı. Kullanılmıyor.

SÖZLESMELER

```
using ilterisg.Models.Enums;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("Sozlesmeler", Schema = "dbo")]
    public class Sozlesmeler
```

```
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int SozlesmeId { get; set; }

        [Required]
        public int RefFirmaId { get; set; }

        [Required]
        public int RefOSGBId { get; set; }

        [Required]
        public DateTime BaslangicTarihi { get; set; }

        public DateTime? BitisTarihi { get; set; }

        [Required]
        public bool IsActive { get; set; } = false;

        [Required]
        public DateTime OlusturulmaTarihi { get; set; }

        [Required]
        public SozlesmeOnayDurumu OnayDurumu { get; set; } = SozlesmeOnayDurumu.Taslak;

        public string SozlesmeMetni { get; set; } // LONGTEXT (MySQL) / nvarchar(max) (SQL Server)

        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }

        [ForeignKey(nameof(RefOSGBId))]
        public virtual OSGBler OSGB { get; set; }
    }
}
```

Firmaların Taslak sonucu firmalarla yaptığı karşılıklı otomatik sözleşmeyi temsil ediyor gereksiz alan yok Normalize.

SozlesmeTaslakları

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ilterisg.Models
{
    [Table("SozlesmeTaslaklari", Schema = "dbo")]
    public class SozlesmeTaslaklari
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int TaslakId { get; set; }

        [Required]
        public int RefOSGBId { get; set; }

        public int? RefFirmaId { get; set; }

        [Required]
        public string TaslakMetni { get; set; } // LONGTEXT (MySQL) / nvarchar(max) (SQL Server)

        [Required]
        [Column(TypeName = "decimal(18, 2)")]
        public decimal TaslakTutar { get; set; }

        [Required]
        public DateTime OlusturulmaTarihi { get; set; } = DateTime.Now;

        [ForeignKey(nameof(RefOSGBId))]
```

```
        public virtual OSGBler OSGB { get; set; }

        [ForeignKey(nameof(RefFirmaId))]
        public virtual Firmalar Firma { get; set; }

        public bool IsVarsayilan { get; set; } = false;
    }
}
```

Taslak tutulan bir junction table sadece Normalize.