ASP.NET CORE 8.0 MVC DERS NOTLARI-(8. HAFTA)

İçindekiler

ÜYELİK İŞLEMLERİ-**IDENTITY** KÜTÜPHANESİ ALT YAPISININ OLUŞTURULMASI

Geliştirmekte olduğumuz sitede Kullanıcı bilgilerinin Veritabanında şifrelenerek saklanması önemlidir. Gelişmiş bir veri güvenliği için Identity kütüphanesini kullanacağız. Bu kütüphaneyi kullanarak aşağıdaki uygulamaları gerçekleştireceğiz. Kullanıcı bilgilerini Identity yapısı üzerinde tutacağımızdan, daha önceden oluşturduğumuz Kullanıcı tablosunu kaldırmamız ve onunla bağlantılı diğer tabloları da elden geçirmemiz gerekecek.

- Register
- Rolleme
- Şifre Doğrulama
- Sms Doğrulama

- Mail Doğrulama
- Login
- Şifremi Unuttum
- Facebook, Twitter vs ile giriş

Ek Bilgi: Erişim Belirleyiciler/Access Modifier: Bir sınıfı yada metodu oluşturduğumuzda diğer yerlerden erişimi belirlemek için temel olarak 4 tane yöntem kullanılır

C# Access Modifiers (Erişim Belirleyiciler)

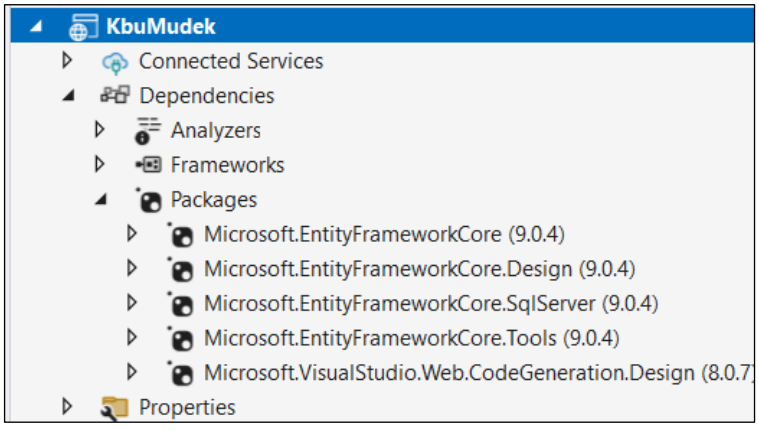
- public:Her yerden erişilebilir
- private :sadece tanımlandığı sınıfın içinden erişilebilir
- protected: Tanımlandığı sınıf ve ondan türeyen (miras alan) sınıflardan erişilebilir
- internal: Aynı projede (assembly) tanımlı her yerden. Yani katmanlı mimaride solution altındaki her projenin kendi içinde erişilebilir.
- protected internal Aynı proje içinde veya miras alan sınıflardan erişilebilir
- private protected Yalnızca tanımlandığı sınıfta veya aynı assembly içindeki miras alan sınıflardan erişilebilir.

Şu anda bizim işimizi public gördüğünden bütün sınıf ve metodlarımızı public oluşturacağız. Farklı bir durum çıkarsa konuyu orada tekrar değerlendiririz.

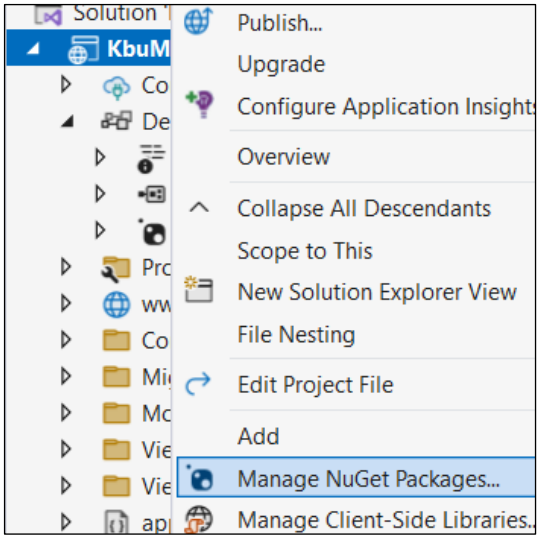
İlk olarak Identity kütüphanesinin paketlerini kuralım.

Identity Kütüphanesinin Kurulumu

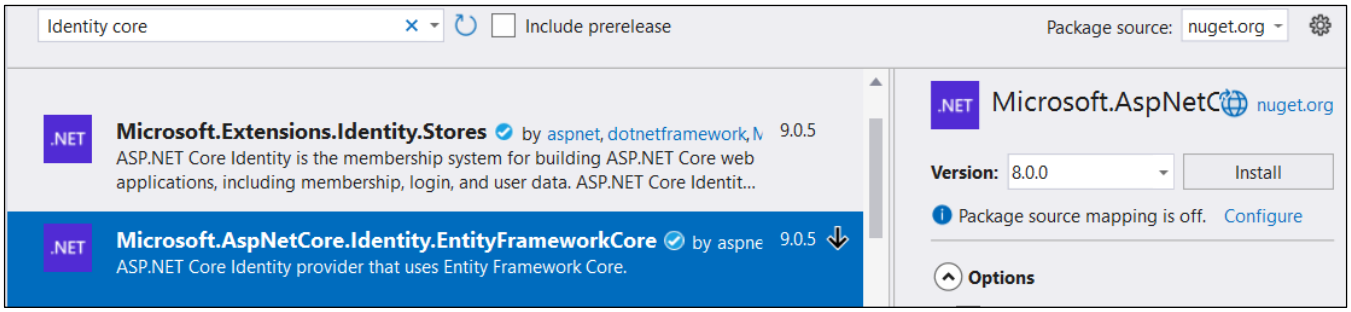
Daha önceden kurduğumuz paketler şunlardı. İlk 4 tanesini önceki ders notlarında kurmuştuk. Şimdi ise Identity kütüphanesini aynı şekilde kuralım.



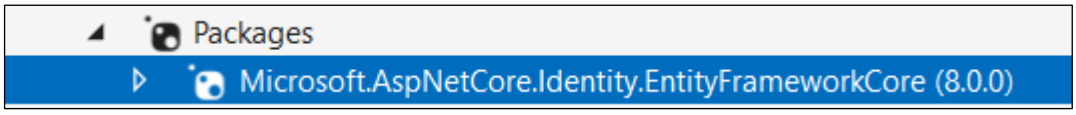
Projenin üzerine sağ tuşa tıklayıp “Manage NuGet Packages” komutunu çalıştıralım. “AspNet.Identity.Core” yazan kütüphaneyi seçip kuralım.



Paket olarak "..Identity.EntityFrameworkCore" kütüphanesini kuralım. Versiyon seçeneğinde 8.0.0 seçili olsun. Çünkü bizim projemiz Core'un 8.0 versiyonu üzerinde çalışacak. Eski versiyonlarda çıkan ekranda alt paketleri de gösterecektir. Onları da kabul edelim. 8.0 da gelmeyecektir.



Explorer'dan paket klasörünü açalım. Yüklediğimiz paketi görelim. Herhangi bir hata uyarısı veren Ünlem işareti görmemeliyiz. Versiyon uyumsuzluğu olmasın.



Eski versiyon Core'larda listede gözüken diğer paketleride yüklemek gerekir fakat 8.0 da bunlar zaten yüklenmiş olarak gelir. Sadece yukarıdaki Identity.EntityFrameworkCore (8.0.0) yüklemek yeterlidir.

Identity Kütüphanesinin Context Sınıfının Oluşturulması

Daha önceden Context sınıfımızı oluşturmuştuk. Context sınıfımız Model sınıf yapılarından Veritabanı yapımızı otomatik olarak oluşturduğumuz ara işlemlerini halleden bir sınıf yapısı idi. Şimdi artık Identity kütüphanesine uygun olarak kullanıcı ve rol tabloları vs oluşturacağımızdan bu işlemleri yapan Context sınıf yapısını değiştireceğiz.

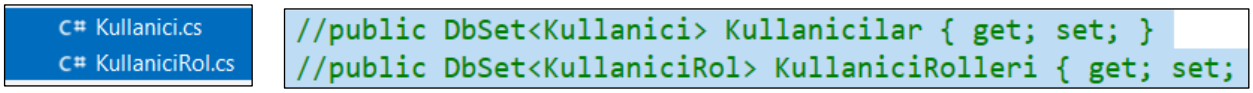
Burda Context sınıfımızı oluşturmak için IdentityDbContext sınıfını kullanacağız (yani ondan miras alacağız) ama bu sınıfın arka planda detaylarına indiğimiz zaman aslında oda temel yapısını önceki kullandığımız DbContext sınıfından miras aldığını görebiliriz. Bu ne anlama geliyor? Yani Biz IdentityDbContext burada

kullanmaya başladığımızda önceki yaptığımız tüm işlemleri yine yapabiliyoruz ama bunun üzerinde Kullanıcı ve Rol tablo vs işlemlerimizi de yapabileceğiz.

Yeni Context sınıfımızın yapısı:

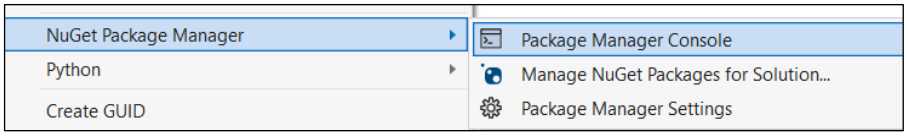
```
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;

namespace KbuMudek.Models
{
    public class Context:IdentityDbContext
```

Daha önce Model sınıflarımız içinde oluşturduğumuz Kullanıcı ve Rol sınıflarımızı kaldıralım. Çünkü bu tür kullanıcı girişini kontrol tabloları Identity kütüphanesine onun formatında otomatik olarak oluşturacağız. Aynı zamanda Context sınıfımız içindeki bu modellere karşılık gelen tabloları tanımlayan DbSet properties satırlarını da kaldıralım.

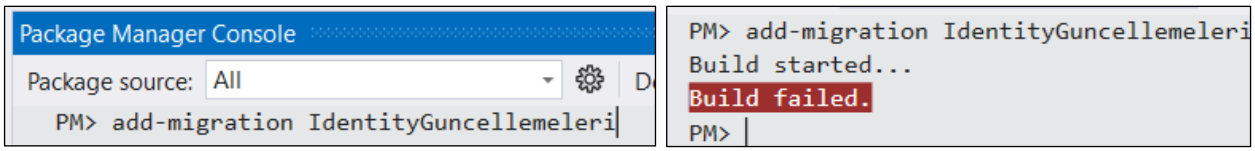


Artık yeni Identity kütüphanesi ile veritabanı yapımızı yeniden oluşturabiliriz. Bunun için tekrardan yeni bir Migration dosyası (Context sınıfındaki bilgilere göre Veritabanını oluşturan komutların olduğu dosya) oluşturalım ve Veritabanımızı güncelleyelim.

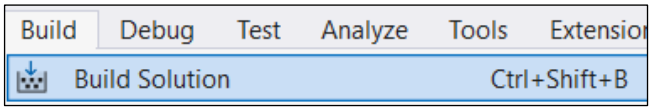
Migration dosyasını oluşturmak için üstteki “Tools” menüsünden “Package Manager Console” açalım.



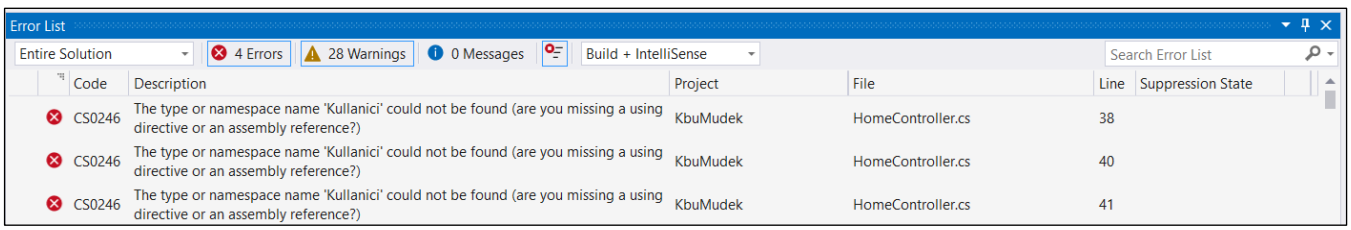
Açılan pencerede add-migration komutunu kullanarak ve oluşacak migration dosyasına da bir isim vererek çalıştıralım. Hata verdi hatanın sebebini bulalım.



Hatanın sebebini tahmin edebiliriz. İki tane modelimizi kaldırdık ve onları kullanan bir takım yerlerdeki kodlarımız kalmıştır. Bunları görebilmek için önce bir projeyi bir derleyelim.



Çok sayıda hatanın olduğunu görüyoruz.

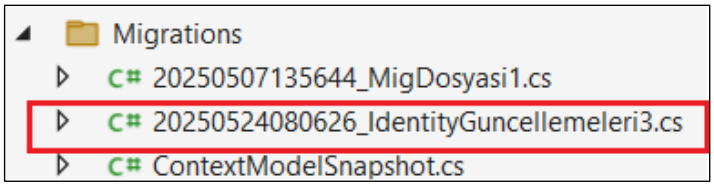


Bunları düzeltip tekrar derleyelim. Hata yoksa migration oluşturmayı bir daha deneyelim. Package Manager (PM) Console ekranını bir daha açalım ve üst oklara tıklayarak en son girdiğimiz komutu tekrar çalıştıralım.

```
PM> add-migration IdentityGuncellemeleri3
Build started...
Build succeeded.
An operation was scaffolded that may result in the loss of data. Please review the migration for accuracy.
To undo this action, use Remove-Migration.
PM>
```

Sarı renkli bir uyarı verdi. Veri kaybına yol açacak bir alt yapının oluşturulduğunu söylüyor. İstemiyorsanız geri alıp migration dosyasını kaldırın diyor. Kullanıcılar ve Rol tablolarını kaldırınca veri kaybı oluşacağını biz biliyorduk, sorun değil.

Migration dosyamız oluşmuş olarak görüyoruz. İçerisini açtığımızda eski Kullanici bilgilerine ait tabloların satırların olmadığını görüyoruz. Yeni Identity kütüphanesine ait User, Role, vs tabloları oluşturduğunu görüyoruz. Ayrıca dosya içinde bizim normal "Fakulteler, Bolumler" gibi tabloların da olduğunu görüyoruz.




```
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityRole",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityRoleClaim<string>",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUser",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserClaim<string>",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserLogin<string>",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserRole<string>",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserToken<string>",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityRoleClaim<string>",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserClaim<string>",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserLogin<string>",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserRole<string>",
modelBuilder.Entity("Microsoft.AspNetCore.Identity.IdentityUserToken<string>",
```

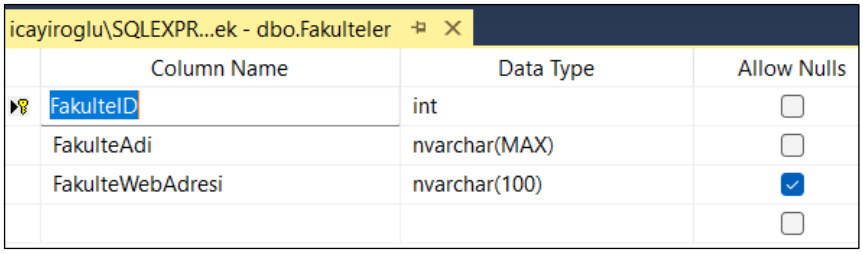
Identity Tablolarının Veritabanında Oluşturulması (Migration İşlemleri)

Artık Migration dosyamızı Veritabanına yansıtalım. Bunun için PM konsoluna "Update-Database" yazıp entera basalım. Fakat devamında Fakulteler tablosu için bir uyarı verdi. FakulteWebAdresi kolonu birden fazla tanımlanmış mı bakalım.

```
PM> Update-Database
Build started...
Build succeeded.
```

```
Column names in each table must be unique. Column name 'FakulteWebAdresi' in table 'Fakulteler' is specified more than once.
PM>
```

FakulteWebAdresi sütunu tahminen elle oluşturuldu. Migration dosyası kodla oluşturmaya çalışınca da hata verdi. Bu sütunu veritabanından manuel olarak silip update komutunu bir daha deneyelim.



Bu sefer aynı hatayı Bolumler tablosundada buldu. Demekki bu sütunları elle ekledik ve var olanı eklemeye çalışınca hata oluşuyor. Bu sütunu da kaldırıp tekrar update komutunu çalıştıralım.

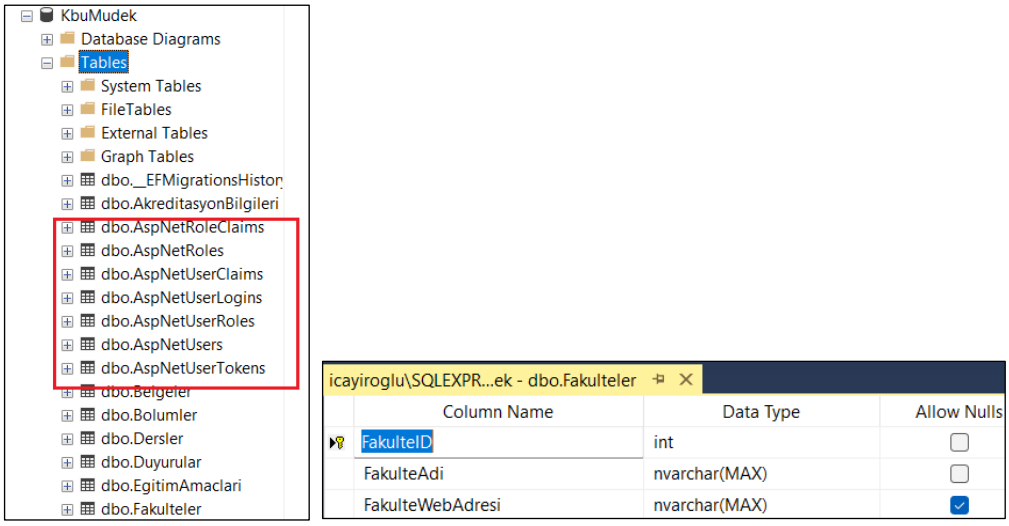
```
Column names in each table must be unique. Column name 'BolumWebAdresi' in table 'Bolumler' is specified more than once.
```

Bu iki hatayı düzelttikten sonra Migration çalıştı. *Demekki Veritabanındaki tablolara elle müdahale etmemek lazım. Bütün değişiklikleri Migration dosyaları üzerinden gerçekleştirmek gerekiyor. Diğer türlü çakışmalar başlarsa düzeltmek zorlaşır.*

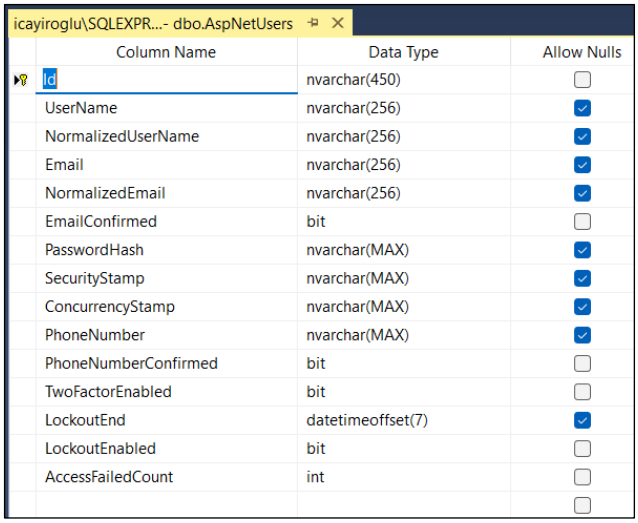
```
Applying migration '20250524080626_IdentityGuncellemeleri3'.
Done.
PM>
```

Veritabanını inceleyelim yeni eklenen ve kullanıcı işlemlerini yapan Identity tabloları gelmiş mi. Ayrıca eski Kullanıcı ve Rol tabloları kalkmış mı. Birde Fakulteler ve Bolumler tablosu içine eklenenen Web adresi sütunları eklenmiş mi, inceleyelim.

Kullanıcı işlemlerini yürüten Idendity tablolarının oluştuğunu görüyoruz. Ayrıca Kullanıcılar ve Roller tablolarıda gitmiş. Fakulteler ve Bolumler tablolarındaki WebAdresi sütunları da kodla eklenmiş oluyor.



Identity'nin Users tablosunu açıp inceleyelim. Gördüğümüz gibi bir çok alan çok daha gelişmiş bir şekilde hazır olarak oluşturulmuş. Bu tablolara elle veri girişi yapmayalım. Veri girişleri Asp sayfalarından Asenkron metodlar (Ek açıklama aşağıda verildi) kullanılarak yapılacak. Burada kullanıcı şifre kayıtları şifrenelenerek (Hash lenerek) yapılacak. O yüzden elle veri girişi yapılamaz. Identity kütüphanesinin tüm metodları Asenkron olarak çalışarak işlemleri yapar. Yani biz veriyi girdikten sonra işlemin bitmesini beklemez yeni bir işleme devam edebilir. Önceki işlem arka planda devam eder.



icayiroglu\SQLEXPR...- dbo.AspNetUsers

Column Name	Data Type	Allow Nulls
Id	nvarchar(450)	☐
UserName	nvarchar(256)	☑
NormalizedUserName	nvarchar(256)	☑
Email	nvarchar(256)	☑
NormalizedEmail	nvarchar(256)	☑
EmailConfirmed	bit	☐
PasswordHash	nvarchar(MAX)	☑
SecurityStamp	nvarchar(MAX)	☑
ConcurrencyStamp	nvarchar(MAX)	☑
PhoneNumber	nvarchar(MAX)	☑
PhoneNumberConfirmed	bit	☐
TwoFactorEnabled	bit	☐
LockoutEnd	datetimeoffset(7)	☑
LockoutEnabled	bit	☐
AccessFailedCount	int	☐
		☐

Burada bazı alanların anlamını açıklayalım. Alan Adı Tür Açıklama

Id nvarchar(450) Kullanıcıya özel, sistem tarafından otomatik üretilen benzersiz ID (GUID veya string olabilir).

UserName nvarchar(256) Kullanıcının giriş yaparken kullandığı kullanıcı adı.

NormalizedUserName nvarchar(256) UserName alanının büyük harfe dönüştürülmüş hali. Aramalarda hızlı karşılaştırma için kullanılır. Büyük/küçük harf duyarsız arama yapabilmek için kullanılır.

Email nvarchar(256) Kullanıcının e-posta adresi.

NormalizedEmail nvarchar(256) Email alanının büyük harfle normalize edilmiş hali. E-posta ile kullanıcı arama işlemlerinde kullanılır.

EmailConfirmed bit E-posta adresi doğrulanmış mı? (Doğrulama linki ile kontrol edilir.) true veya false.

PasswordHash nvarchar(MAX) Parolanın hash'lenmiş hali. Parola düz yazı olarak asla saklanmaz. Şifrelenmiş olarak saklanır.

SecurityStamp nvarchar(MAX) Kullanıcıya özel güvenlik belirteci. Şifre değiştiğinde veya oturum sıfırlandığında değişir. Değeri değiştiğinde kullanıcının tüm oturumları otomatik olarak sonlandırılır. Genelde parola değiştirme gibi işlemlerden sonra kullanılır.

ConcurrencyStamp nvarchar(MAX) Aynı anda yapılan güncellemeleri çakışmadan kontrol etmeye yarayan GUID.

PhoneNumber nvarchar(MAX) Kullanıcının telefon numarası (isteğe bağlıdır).

PhoneNumberConfirmed bit Telefon numarası doğrulandı mı? true veya false.

TwoFactorEnabled bit İki adımlı kimlik doğrulama (2FA) etkin mi? Hem mail hem Telefon doğrulamasını açar.

LockoutEnd datetimeoffset(7) Kullanıcı geçici olarak kilitlendiğinde, kilidin biteceği tarih/zaman.

LockoutEnabled bit Bu kullanıcının kilitlenebilmesine izin verilsin mi?

AccessFailedCount int Üst üste başarısız oturum açma girişimi sayısı. Belirli bir sayıyı geçerse LockoutEnd uygulanabilir.

---------- 000 ----------

Ara Not: Asenkron İşlem Nedir?

"Asenkron" (asynchronous), bir işlemin arka planda çalışması ve tamamlandığında haber vermesi anlamına gelir. Bu sayede ana akış durdurulmaz ve sistem diğer işlere devam edebilir. ASP.NET Core'da özellikle veritabanı, dosya sistemi ve ağ işlemleri için asenkron metotlar tercih edilir.

Temel Yapı:

```
public async Task<IActionResult> MyMethod()
{
    var result = await SomeAsyncOperation();
    return View(result);
}
```

Burada;

async: Metodun asenkron olduğunu belirtir.

await: Asenkron işlemin tamamlanmasını beklerken diğer işleri durdurmaz.

Task<T>: Gelecekte tamamlanacak işlemleri temsil eder.

Neden Kullanılır?

- Performansı artırır.
- Kullanıcı deneyimini iyileştirir (donma olmaz).
- ASP.NET Core gibi web sistemlerinde yüksek eş zamanlı istekler için idealdir.

Uyarılar

await yazmayı unutursanız işlem çalışır ama sonucu beklemez, hata alabilirsiniz.

async void yerine genelde async Task veya async Task<T> kullanılır.

---------- 000 ----------

Identity/Users ve Role Tablolarını Özelleştirme

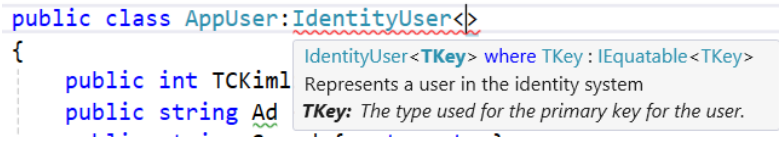
Dikkat edersek Identity'ye ait Kullanıcılar (Users) tablosunun içerisinde istediğimiz bazı sütunlar yoktur. Örneğin biz kullanıcıların Ad, Soyad ve TC Kimlik numarası gibi temel bilgileri de bu tablo içinde tutmak istiyoruz. Bunu nasıl yaparız şimdi onu görelim.

Ayrıca Identity'nin Users ve Role tablolarındaki Id sütunları string olarak tanımlanıyor. Oysa bizim projede tüm ID sütunlarını integer kullanmak istiyoruz. Bu yapıyı da nasıl değiştiririz görelim.

Öncelikle yeni bir kullanıcı modeli ekleyeceğiz. Fakat bu model Identity'nin IdentityUser sınıfını miras alacak. Yani bu ne demek? IdentityUser sınıfı içinde bizim için hazırlanmış herşey aynı zamanda çalışacak ve birde üzerine bizim eklediğimiz alanlar olacak.

Genellikle bu tip üzerine bindirilen Model'in adı AppUser olarak tanımlandığından bizde o şekilde tanımlayalım. Dikkat edersek Identity kütüphanesindeki alanların isimleri İngilizcedir. Biz bunların üzerine kendi alanlarımızı eklerken yine Türkçe olarak eklemeye devam edelim... Kendi sistemimizi bozmadan gidelim.

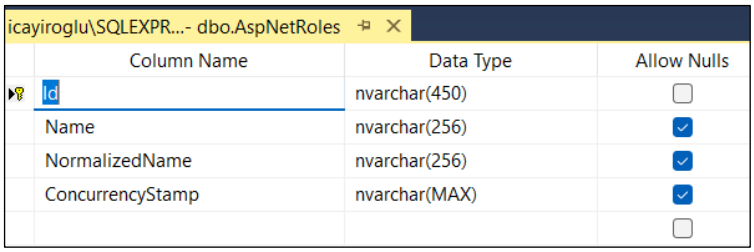
Burada bizim oluşturduğumuz AppUser sınıfı IdentityUser sınıfının üzerine bindirilecektir. <int> ifadesi ise TKey değişkenin yani anahtar alanın tipini belirlemektedir.



AppUser.cs

```
using Microsoft.AspNetCore.Identity;

namespace KbuMudek.Models
{
    public class AppUser:IdentityUser<int>
    {
        public int AppUserId { get; set; }
        public string Ad { get; set; }
        public string Soyad { get; set; }
        public DateTime DogumTarihi { get; set; }
        public int UyruguID { get; set; }
        public string Adres { get; set; }
        public bool AktifMi { get; set; }
        public string Resim { get; set; }

    }
}
```

Aynı özelleştirmeyi Role tablosu içinde yapalım. İncelediğimizde bizim için Role adının bulunması yeterli gözükmektedir. Sadece Id tipinin integer olması formatımıza uygun değil. Onu değiştirelim, integer yapalım.

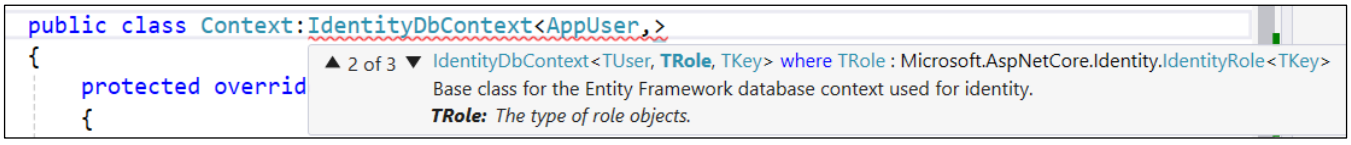


AppRole.cs

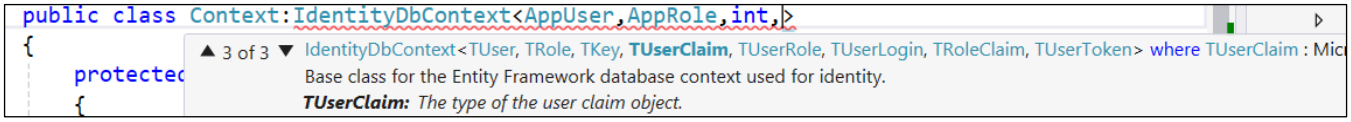
```
using Microsoft.AspNetCore.Identity;

namespace KbuMudek.Models
{
    public class AppRole:IdentityRole<int>
    {
    }
}
```

Bu iki tane sınıfımızı oluşturduk fakat bunlar Veritabanındaki AspNetUsers ve AspNetRoles tablolarını etkilemesini nasıl anlayacak. Bunu tarif etmek için Context sınıfımızı oluşturduğumuz yerde bunların üzerine Overload (bindirme) ekleyeceğimizi söylemeliyiz.

Context başlığımızdaki çıkan help ekranını incelediğimizde bizden User, Role ve Key leri istediğini görüyoruz.



Eğer Key'den sonra virgül eklersek başka Override yapabileceğimiz sınıfları da göstermektedir. Bizi ilgilendiren bu 3 tane override yeterlidir.



Context.cs

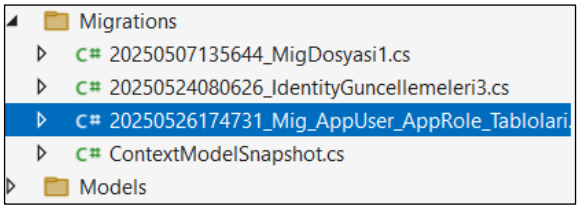
```
namespace KbuMudek.Models
{
    public class Context:IdentityDbContext<AppUser,AppRole,int>
    {
        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            optionsBuilder.UseSqlServer("server=ICAYIROGLU\\SQLEXPRESS; database=KbuMudek;
integrated security=true; TrustServerCertificate=true");
        }
```

Migration oluşturma komutumuzu çalıştıralım.

```
PM> Add-Migration Mig_AppUser_AppRole_Tablolari
```

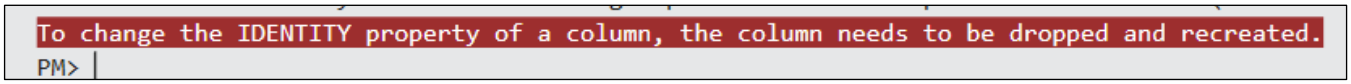
Migration dosyamız oluştu. Bazı bilgilerin kaybolabileceğine dair uyarıda verdi. Bunu bekliyorduk zaten.

```
PM> add-migration Mig_AppUser_AppRole_Tablolari
Build started...
Build succeeded.
An operation was scaffolded that may result in the loss of data. Please review the migration for accuracy.
To undo this action, use Remove-Migration.
PM>
```



Şimdi Migrationdaki bilgileri Veritabanına yansıtalım. **Update-Database** komutunu PM penceresinde uygulayalım

Build kısmında hata vermedi (yani kodlarda bir hata yok) fakat veritabanında güncelleme yaparken hata verdi. Kolonun Identity özelliğini değiştirirken hata oluştu bunun için silinmesi ve yeniden oluşturulması gerekiyor.



Bu sorunu manuel ve kısmi çözüm arayarak çözmek çok uğraştırır ve neredeyse imkansızdır. Çünkü Id sütunlarını Alter etmek (değiştirmek) imkansızdır. O yüzden veritabanını baştan silmek ve yeni yapıya göre migration'nımızı tekrar oluşturmak gerekir. Eğer veritabanında verileriniz varsa buda imkansız hale gelir. O yüzden daha verileriniz oluşmadan bu sutünları Integer olarak dönüştürmek gerekir.

AppUser ve AppRole Tablolarını Ekleme

O zaman şimdi veritabanının tamamını kaldıralım (dikkat veriler varsa yapılmamalı), migration dosyamızı da silelim ve migration'ımızı yeniden oluşturup güncellemesini yapalım. Bunun için sırayla uygulayacağımız komutlar şunlardır.

- Drop-Database
- Remove-Migration
- Add-Migration Mig_AppUser_AppRole_Uygulandi
- Update-Database

Drop-Database komutunda ciddi olarak uyarıyor.

```
PM> Drop-Database
Build started...
Build succeeded.
Confirm
Are you sure you want to perform this action?
Performing the operation "Drop-Database" on target "database 'KbuMudek' on server 'ICAYIROGLU\SQLEXPRESS'".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"):
```

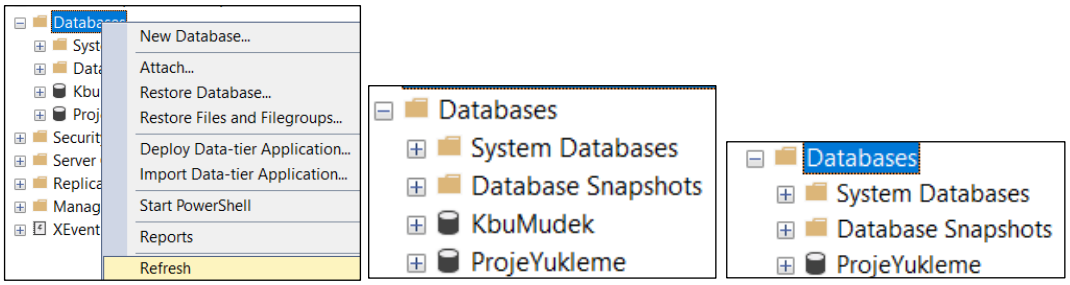
Bu kısaltmaların açıklamaları;

Y Yes – Bu işlemi bir kereye mahsus onayla ve devam et.
A Yes to All – Bu işlemle ilgili tüm sonraki onayları da otomatik olarak kabul et. Yani her adımda tekrar sorma.
N No – Bu işlemi reddet, yani devam etme.
L No to All – Bu tür tüm işlemleri reddet, hiçbiri için devam etme.
S Suspend – İşlemi beklemeye al, (çoğu zaman bu devre dışıdır veya etkisizdir).
? Help – Bu açıklamaları tekrar göster.

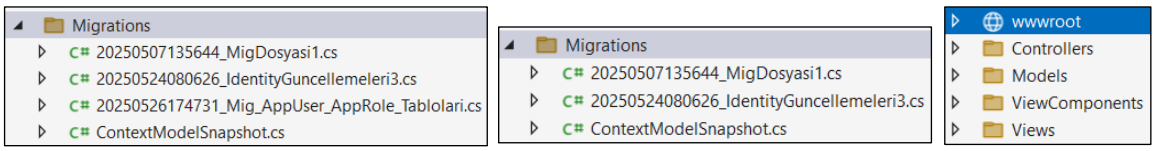
Biz Y komutunu uygulayalım.

```
Dropping database 'KbuMudek' on server 'ICAYIROGLU\SQLEXPRESS'.
Successfully dropped database 'KbuMudek'.
PM>
```

Bakalım veritabanımız silinmiş mi? Refresh yapalım. Önceki hali ve sonraki hali;

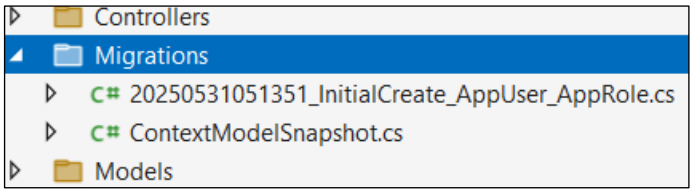


Sırada Remove-Migration komutumuz var. Uygulayalım. Dikkat edersek klasörde en son üretilen Migration dosyasını sildi ama hala daha önceki migration dosyaları da var. O yüzden tüm klasörü manuel olarak silelim. Zaten klasörü ve dosyaları yeniden kendisi oluşturur.

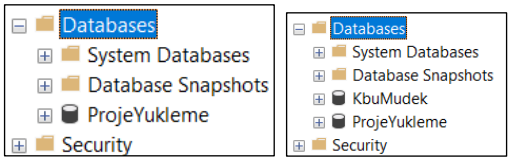


Artık Migration'larımız yok. İlk migration dosyamızı oluşturalım. Dosya adını InitialCreate (başlangıç oluşturulan dosyası) adını verelim. AppUser ve AppRole modellerimizinde uygulandığını anlatmak için sonuna o ifadeleri de ekleyelim.

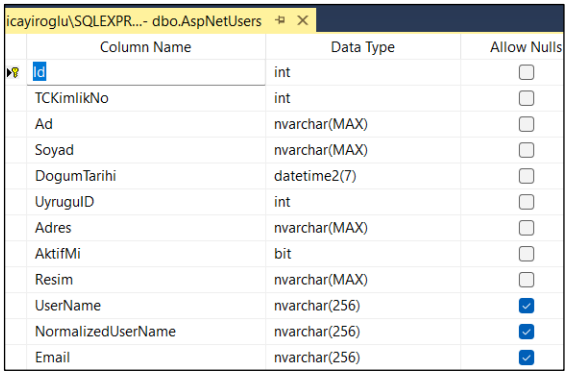
Add-Migration InitialCreate_AppUser_AppRole



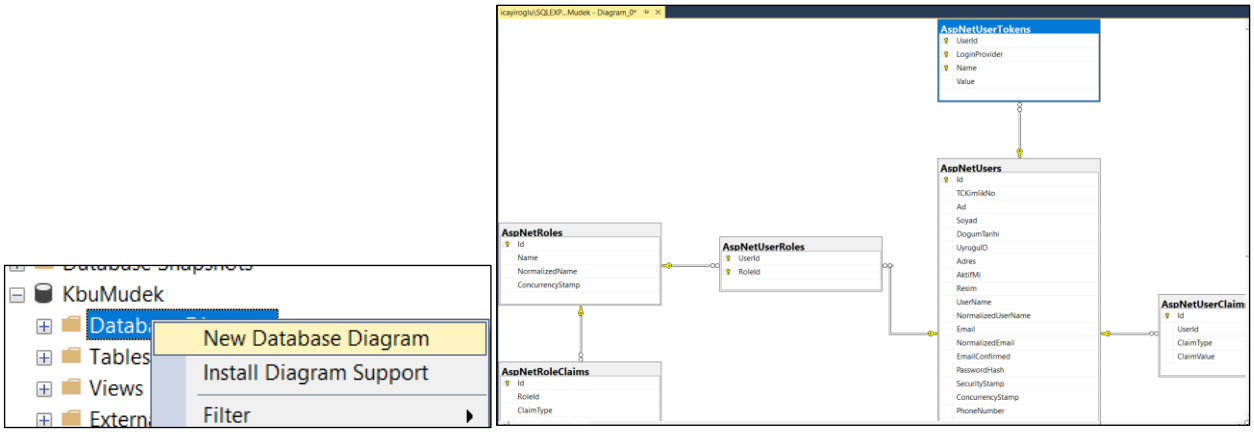
Migration klasörümüz ve dosyalarımız oluştu. Veritabanına yansıtalım. Update-Database komutunu uygulayalım ve Database üzerinde Refesh yapalım. Veritabanımızın oluşturulduğunu görüyoruz.



AspNetUsers tablomuzu design şeklinde açarsak bizim eklediğimiz alanlarında ve Id tipinin int şeklinde olduğunu görürüz.



Yeni oluşturulan Identity tabloları arasındaki ilişkiler oluşmuş mu onları bir görelim. Bunun için Veritabanında Database Diagram üzerine gelip sağ tuşla yeni bir diyağram oluşturalım. Listeden tüm tabloları tek seçip diyağram üstünde görüntüleyelim. Diyağrama baktığımızda Id bağlantılarının oluştuğunu görüyoruz.

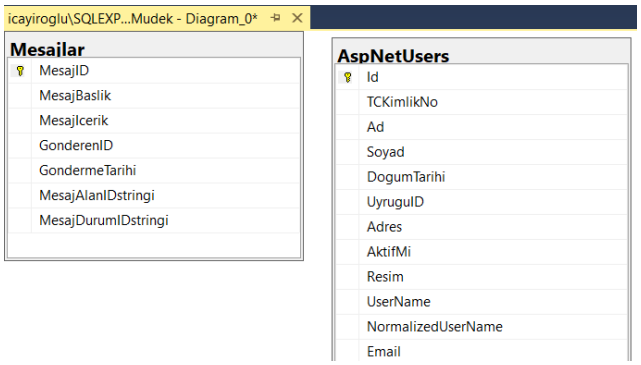


Fakat dikkat edersek burada verilen tablo isimleri hepsi Identity kütüphanesinin tablolarıdır. Bu tablolar kütüphane tarafından oluşturulduğundan Id bağlantıları kendi arasında biz müdahale etmeden oluşturulmuş oldu.

Oysa bizim kendimize ait tablolarda da kullanıcıya ait Id bilgilerini kullanacağız. Bu durumda bizim tablolar ile Identity'nin bu tabloları arasında bağlantıların oluşturulması gerekecektir. İşte şimdi bunu gerçekleştirelim.

Identity Tabloları ile Kendi Tablolarımız Arasında ID bağlantılarının Kurulması

Aşağıdaki iki tablo arasındaki ilişkiyi Veritabanı Diağram kısmında incediğimizde bu iki tablo arasında bir bağlantının olmadığını görüyoruz. Oysa mesaj çeken kişinin hangi kullanıcı olduğunu bilmek isteriz. Bu bilgiyi de GonderenID içerisinde tutacağız fakat buradaki ID bilgisinin Users tablosundaki Id bilgisi ile aynı olması gerekir. Yani bu iki tablo arasında bir bağlantının bulunması gerekecek.



LOGIN (Şifre giriş) KODLARI

```
public class AccountController : Controller
{
    private readonly UserManager<AppUser> _userManager;
    private readonly SignInManager<AppUser> _signInManager;
    private readonly Context _context; // DbContext enjekte ediliyor

    public AccountController(UserManager<AppUser> userManager, SignInManager<AppUser>
signInManager, Context context)
    {
        _userManager = userManager;
        _signInManager = signInManager;
        _context = context;
    }

   //============================== REGISTER ====================================
   //============================================================================

   [HttpGet]
   [AllowAnonymous]
   public IActionResult Register()
   {
       var model = new AppUserRegisterVm
       {
           ProfileTypes = GetProfileTypes()
       };

       return View(model);
   }
```

REGISTER (kayıt sayfası)KODLARI

```
    [AllowAnonymous]
    [HttpPost]
    public async Task<IActionResult> Register(AppUserRegisterVm appUserRegisterVm)
    {
        if (ModelState.IsValid) //Eğer sayfadan gelen modelin alanları Fluent Apilerden
hatasız olarak geçip buraya geldi ise...
        {
```

```
            //Daha önce aynı TCKimlin no ve Mail adresi ile kayıt yapılmış mı kontrol
edecek?
            var existingUser = await
_userManager.FindByEmailAsync(appUserRegisterVm.Email);
            var existingTc = await _userManager.Users.FirstOrDefaultAsync(x =>
x.IdentityNumber == appUserRegisterVm.IdentityNumber);

            bool hasError = false;

            if (existingUser != null)
            {
                ModelState.AddModelError("Email", "Bu e-posta adresi ile daha önce kayıt
yapılmış.");
                hasError = true;
            }

            if (existingTc != null)
            {
                ModelState.AddModelError("IdentityNumber", "Bu TC Kimlik Numarası ile daha
önce kayıt yapılmış.");
                hasError = true;
            }

            if (hasError)
            {
                appUserRegisterVm.ProfileTypes = GetProfileTypes();
                return View(appUserRegisterVm);
            }

            //-------------------------------------------
            Random random = new Random();
            AppUser appUser = new AppUser()
            {
                IdentityNumber = appUserRegisterVm.IdentityNumber,
                FirstName = appUserRegisterVm.FirstName,
                LastName = appUserRegisterVm.LastName,
                Email = appUserRegisterVm.Email,
                UserName = appUserRegisterVm.Email, // UserName is now set to Email
                ConfirmCode = random.Next(100000, 999999),
                ProfileTypeId = appUserRegisterVm.ProfileTypeId //
            };
            //Veritabanına User kayıt işlemi tam burada gerçekleşiyor (CreateAsync).
Şifreyi de yukarıda değil burada hash layarak kaydediyor.  Aynı zamanda şifreyi ayrı
olarak Asenkron olarak gönderiyor.
            var result = await _userManager.CreateAsync(appUser,
appUserRegisterVm.Password); //Dikkat: şifre Hash (şifrelenerek) kaydedileceğinden
yukarıda değil burada asenkron yöntemle ayrı olarak aktarılacak.

            if (result.Succeeded) //yukarıda verileri aktardı ve şifreyide hash olarak
appUser içerine aktardıysa artık mail kontrolüne geçebiliriz
            {
                //Kullanıcı oluşturulduktan hemen sonra CandidateUser rolüne ata:
                await _userManager.AddToRoleAsync(appUser, "CandidateUser");

                // 1. İlgili Constant Code'unu Veritabanından Çekme (Asenkron Olabilir)
                // appUser.ProfileTypeId, AppUser nesnesinden gelen Id'dir.
                var profileCode = _context.Constants
                                          .Where(c => c.ConstantId ==
appUser.ProfileTypeId)
                                          .Select(c => c.Code)
                                          .FirstOrDefault();

                if (profileCode != null)
                {
                    // 2. Switch ifadesini string Code üzerinden kullanma
```

```
                    switch (profileCode)
                    {
                        case "SUPERADMIN":
                            // Admin için herhangi bir şey yapmana gerek yok.
                            break;

                        case "ACADEMICPERSONNEL":
                            var academicPersonnel = new AcademicPersonnel
                            {
                                AppUserId = appUser.Id,
                                RegistryNumber = null,
                                EmploymentStartDate = null,
                                EmploymentEndDate = null,
                                CreatedAt = DateTime.Now,
                                UpdatedAt = null,
                                IsActive = true
                            };
                            _context.AcademicPersonnels.Add(academicPersonnel);
                            break;

                        case "ADMINISTRATIVEPERSONNEL":
                            var administrativePersonnel = new AdministrativePersonnel
                            {
                                AppUserId = appUser.Id,
                                RegistryNumber = null,
                                EmploymentStartDate = null,
                                CreatedAt = DateTime.Now,
                                IsActive = true
                            };

_context.AdministrativePersonnels.Add(administrativePersonnel);
                            break;

                        case "STUDENT":
                            var student = new Student
                            {
                                AppUserId = appUser.Id,
                                StudentNumber = null,
                                AcademicProgramId = null,
                                EnrollmentDate = DateTime.Now,
                                StudentStatusId = null,
                                CreatedAt = DateTime.Now,
                                IsActive = true
                            };
                            _context.Students.Add(student);
                            break;

                        case "ALUM":
                            var alum = new Alum
                            {
                                AppUserId = appUser.Id,
                                AcademicProgramId = null,
                                GraduationDate = DateTime.Now,
                                AlumStatusId = null,
                                CreatedAt = DateTime.Now,
                                IsActive = true
                            };
                            _context.Alums.Add(alum);
                            break;

                        case "STAKEHOLDER":
                            var stakeholder = new Stakeholder
                            {
                                AppUserId = appUser.Id,
                                StakeholderTypeId = null,
                                OrganizationName = null,
```

```
                                CreatedAt = DateTime.Now,
                                IsActive = true
                            };
                            _context.Stakeholders.Add(stakeholder);
                            break;

                        case "PARENT":
                            var parent = new Parent
                            {
                                AppUserId = appUser.Id,
                                StudentNumber = null, // Velinin programda kayıtlı olduğu
öğrencisi
                                OrganizationName = null,
                                CreatedAt = DateTime.Now,
                                IsActive = true
                            };
                            _context.Parents.Add(parent);
                            break;
                    }
                }

                await _context.SaveChangesAsync();


                //--CONFIRMMAIL için Linki maile gönderme -------------------------
                MimeMessage mimeMessage = new MimeMessage(); //Mail formatını düzenleyecek
nesnemizi tanımlıyoruz.
                MailboxAddress mailboxAddressFrom = new MailboxAddress("KbuMüdek",
"icayiroglu@gmail.com");  //Gönderecek mail adresinin formatını oluşturuyoruz.
                MailboxAddress mailboxAddressTo = new MailboxAddress("KbuMüdek User",
appUser.Email);        //Gidecek olan mail adresinin formatını oluşturuyoruz.

                mimeMessage.From.Add(mailboxAddressFrom);    //Gönderen adresini nesneye
tanıtıyoruz.
                mimeMessage.To.Add(mailboxAddressTo);        //Alıcı adresini nesneye
tanıtıyoruz.

                BodyBuilder bodyBuilder = new BodyBuilder();

                //Link Gönderme ile yapılacaksa-------------- DİKKAT: HOSTINGE GEÇİNCE
BURADAKİ ADRES DEĞİŞMELİ---------------
                var token = await
_userManager.GenerateEmailConfirmationTokenAsync(appUser); // ConfirmCode yerine token
oluştur
                var confirmationLink = Url.Action("ConfirmEmailResult", "Account", //
Doğrulama bağlantısı oluştur
                    new { userId = appUser.Id, token = token }, Request.Scheme);
//http://localhost:1234/Account/ConfirmEmail?userId=5&token=... linki üretilir.

                bodyBuilder.TextBody = $"KbuMudek üyeliğinizin tamamlanması için aşağıdaki
bağlantıya tıklayın:\n\n{confirmationLink}";

                mimeMessage.Body = bodyBuilder.ToMessageBody(); //mimeMessage ın Body
kısmına, bodyBuilder dan gelen message body i ekle.

                mimeMessage.Subject = "KbuMudek Üyelik Onay Linki";

                SmtpClient smtpClient = new SmtpClient(); //Mail transfer prokol örneği.
İki tane çıkıyor MailKit olanı seç.
                smtpClient.Connect("smtp.gmail.com", 587, false); //maili gönderecek
serverın adı, Türkiye çıkış port numarası, Ssl güvenlikli mi gönderilecek=hayır

                smtpClient.Authenticate("icayiroglu@gmail.com", "gvilioguwxhcoglyzku");
//mail gönderecek serverdaki hesabın adı ve şifresi. //gmail hesabından şifreyi alabilmek
için 2 adımlı doğrulamayı açmak gerekiyor.//"Uygulama şifreleri" kısmında bu ifadeyi yaz
gelir.
```

```
                smtpClient.Send(mimeMessage); //mimeMessage nesnesi içinde oluşturulmuş
olan mesajı gönder.

                smtpClient.Disconnect(true); //Mesajı gönderdiysen güvenli bir şekilde
server bağlantısını kapat

                var vm = new GenericResultVm
                {
                    Title = "Kayıt işlemi başarılıdır",
                    Description = "Üyeliğinizin tamamlanması için Emailinize gönderilen
linke tıklamanız gerekmektedir..",
                    ColorClass = "success",
                    UrlText = "Giriş",
                    ReturnUrl = "/Login" //Buton da buna bağlı olarak gözükecek

                };
                return View("RegisterSucceeded", vm);

            }
            else
            {
                foreach (var item in result.Errors)
                {
                    ModelState.AddModelError("", item.Description);
                }
            }

        }
        else
        {

            appUserRegisterVm.ProfileTypes = GetProfileTypes();

            return View(appUserRegisterVm);
        }

        appUserRegisterVm.ProfileTypes = GetProfileTypes();

        return View(appUserRegisterVm);
    }



    private List<SelectListItem> GetProfileTypes()
    {
        return _context.Constants
            .Include(c =>c.ConstantType)
            .Where(c => c.ConstantType.Name== "ProfileType" && c.IsActive)
//ConstantTypeId == 24
            .OrderBy(c => c.Order)
            .Select(c => new SelectListItem
            {
                Value = c.ConstantId.ToString(),
                Text = c.Name
            })
            .ToList();
    }
****************************************
```

REGISTER SUCCEDED (Kayıt başarı ile tamamlandı, Link kontrolü gönderildi)

```
@model GenericResultVm

@{
    Layout = null;
}
```

```
<!DOCTYPE html>
<html lang="en">

<head>
    <title>Register Succeeded</title>
    <meta content="" name="description">
    <meta content="" name="keywords">

    @* Her sayfada kullanılan sabit css ve font linkleri burada tutuluyor. *@
    @await Html.PartialAsync("_Head")

</head>

<body>

    <main>
        <div class="container">

            <section class="section register min-vh-100 d-flex flex-column align-items-
center justify-content-center py-4">
                <div class="container">
                    <div class="row justify-content-center">
                        <div class="col-lg-4 col-md-6 d-flex flex-column align-items-
center justify-content-center">

                            <div class="d-flex justify-content-center py-4">
                                <a href="index.html" class="logo d-flex align-items-center
w-auto">
                                    <img src="~/Template/assets/img/KbuMudeklogo.png"
alt="">
                                    <span class="d-none d-lg-block">KbuMüdek</span>
                                </a>
                            </div><!-- End Logo -->

                            <div class="card mb-3">

                                <div class="card-body">

                                    <div class="pt-4 pb-2">
                                        <h2 class="card-title text-center pb-0 fs-
4">@Model.Title</h2>

                                        <div class="alert alert-@Model.ColorClass">
                                            <p>@Model.Description</p>
                                         </div>

                                        @if (Model.ShowLink)
                                        {
                                            <div class="text-center mt-3">
                                                <a href="@Model.ReturnUrl" class="btn btn-
outline-@Model.ColorClass">
                                                    @Model.UrlText
                                                </a>
                                            </div>
                                        }
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>

            </section>

        </div>
```

```
    </main><!-- End #main -->

    <a href="#" class="back-to-top d-flex align-items-center justify-content-center"><i
class="bi bi-arrow-up-short"></i></a>

    @* Her sayfada kullanılan sabit java script linkleri burada tutuluyor. *@
    @await Html.PartialAsync("_Scripts")

</body>

</html>
```

CONFIRM MAIL (Mail kontrolü gerçekleşti) KODLARI

```
    //================================ CONFIRM MAIL ====================================
    //==================================================================================
    [AllowAnonymous]
    [HttpGet]
    public async Task<IActionResult> ConfirmEmailResult(int userId, string token)
    {
        var user = await _userManager.FindByIdAsync(userId.ToString());
        if (user == null)
        {
            var vm = new GenericResultVm
            {
                Title = "Kullanıcı Bulunamadı.",
                Description = "Id bilgisine bağlı kullanıcı bilgilerinize ulaşılamadı..
Tekrar kaydolmayı deneyin..",
                ColorClass = "danger",
                UrlText = "Kayıt Sayfası",
                ReturnUrl = Url.Action("Register", "Account")

            };
            return View("ConfirmEmailResult", vm);
        }

        var result = await _userManager.ConfirmEmailAsync(user, token); //Email onayı ile
ilgili VT değişikliğini burası otomatik yapar.

        if (result.Succeeded)
        {
            var vm = new GenericResultVm
            {
                Title = "E-Posta Onaylandı",
                Description = "Üyeliğiniz başarıyla tamamlandı.. Artık sisteme giriş
yapabilirsiniz..",
                ColorClass = "success",
                UrlText = "Giriş Sayfası",
                ReturnUrl = Url.Action("Login", "Account")
            };
            return View("ConfirmEmailResult", vm);
        }
        else
        {
            var vm = new GenericResultVm
            {
                Title = "Hatalı veya Süresi Dolmuş Bağlantı",
                Description = "E-posta doğrulama bağlantısı geçersiz ya da süresi dolmuş
olabilir.. Yeniden kaydolmayı deneyin..",
                ColorClass = "danger",
                UrlText = "Kayıt Sayfası",
                ReturnUrl = Url.Action("Register", "Account")
            };
            return View("ConfirmEmailResult", vm);
```

```
        }
    }
```

LOGİN İŞLEMİ (Şifre girişi)

```
    //================== LOGIN ============================================================
    //=====================================================================================

    [AllowAnonymous]
    [HttpGet]//Herkesin bu sayfaya girebilmesine müsaade ediyor. Yukarıda tüm sınıf
Autorize olmuştu.
    public IActionResult Login()
    {
        //Eğer kullanıcının daha önceden Çerez kaydı varsa (RememberMe) direk ana sayfaya
yönlendirecek.
        if (User.Identity.IsAuthenticated) //&& Request.Path == "/Users/Login"
        {
            Response.Redirect("/Home/Index");

        }

        return View();
    }

    [AllowAnonymous]
    [HttpPost]
    public async Task<IActionResult> Login(LoginVm model)
    {
        if (!ModelState.IsValid)
            return View(model);

        //var user = await _userManager.Users.FirstOrDefaultAsync(x => x.IdentityNumber ==
model.IdentityNumber);

        var user = await _userManager.FindByEmailAsync(model.Email);
        if (user == null)
        {
            ModelState.AddModelError("", "Kullanıcı bulunamadı");
            return View(model);
        }

        if (!user.EmailConfirmed)
        {
            ModelState.AddModelError("", "E-posta adresiniz henüz doğrulanmamış.");
            return View(model);
        }

        var signInManager =
HttpContext.RequestServices.GetRequiredService<SignInManager<AppUser>>();
        var result = await signInManager.PasswordSignInAsync(user.UserName,
model.Password, model.RememberMe, false);

        if (result.Succeeded)
        {
            return RedirectToAction("Index", "Home"); // veya yönlendirmek istediğin sayfa
        }

        ModelState.AddModelError("", "Şifre hatalı");
        return View(model);
    }
```

LOGOUT (Siteden Çıkış yap)

```
    //================== LOGOUT ================================
    //==========================================================

    [HttpPost]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> Logout()
    {
        await _signInManager.SignOutAsync(); // Kullanıcıyı sistemden çıkar
        return RedirectToAction("Index", "Home"); // Ana sayfaya yönlendir
    }
```

FORGOT PASSWORD (Şifremi Unuttum)

```
    //******************* FORGOT PASSWORD ********************
    //================== FORGOT PASSWORD ==================================
    [HttpGet]
    [AllowAnonymous]
    public IActionResult ForgotPassword()
    {
        return View();
    }

    //================== FORGOT PASSWORD ==================================
    [HttpPost]
    [AllowAnonymous]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> ForgotPassword(ForgotPasswordVm model)
    {
        if (!ModelState.IsValid)
            return View(model);

        var user = await _userManager.FindByEmailAsync(model.Email);
        if (user == null || !await _userManager.IsEmailConfirmedAsync(user))
        {
            // Güvenlik: Var olsa da yoksa da aynı mesaj
            return View("ForgotPasswordConfirmation");
        }

        var token = await _userManager.GeneratePasswordResetTokenAsync(user);
        var callbackUrl = Url.Action(
            "ResetPassword", "Account",
            new { userId = user.Id, token = token },
            protocol: Request.Scheme);

        // === MAİL GÖNDERME ===
        var mimeMessage = new MimeMessage();
        mimeMessage.From.Add(new MailboxAddress("KbuMüdek", "icayiroglu@gmail.com"));
        mimeMessage.To.Add(new MailboxAddress(user.FirstName + " " + user.LastName,
user.Email));
        mimeMessage.Subject = "KbuMüdek - Şifre Sıfırlama";

        var bodyBuilder = new BodyBuilder();
        bodyBuilder.HtmlBody = $@"
            <p>Merhaba {user.FirstName},</p>
            <p>Şifrenizi sıfırlamak için aşağıdaki bağlantıya tıklayın:</p>
            <p><a href='{callbackUrl}' style='color:#0d6efd;'>Şifremi Sıfırla</a></p>
            <p>Bu bağlantı 1 saat içinde geçerliliğini yitirecektir.</p>
            <p>Eğer bu isteği siz yapmadıysanız, lütfen bu e-postayı görmezden gelin.</p>
            ";
        mimeMessage.Body = bodyBuilder.ToMessageBody();

        using var client = new SmtpClient();
        await client.ConnectAsync("smtp.gmail.com", 587, false);
```

```
        //Not: Aşağıdaki şifre gmail'e giriş şifresi değildir. Smtp için gmail'in
oluşturduğu ayrı özel bir şifredir.
        await client.AuthenticateAsync("icayiroglu@gmail.com", "gvilioguwxhcoglyzku");
        await client.SendAsync(mimeMessage);
        await client.DisconnectAsync(true);

        return View("ForgotPasswordConfirmation");
    }
```

RESET PASSWORD (Şifre Yenileme)

```
    //================== RESET PASSWORD ===================================
    [HttpGet]
    [AllowAnonymous]
    public IActionResult ResetPassword(string userId, string token)
    {
        if (userId == null || token == null)
            return RedirectToAction("Login");

        var model = new ResetPasswordVm { Token = token };
        return View(model);
    }

    [HttpPost]
    [AllowAnonymous]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> ResetPassword(ResetPasswordVm model)
    {
        if (!ModelState.IsValid)
            return View(model);

        var user = await _userManager.FindByIdAsync(model.UserId);
        if (user == null)
            return RedirectToAction("ResetPasswordConfirmation");

        var result = await _userManager.ResetPasswordAsync(user, model.Token,
model.Password);
        if (result.Succeeded)
            return View("ResetPasswordConfirmation");

        foreach (var error in result.Errors)
            ModelState.AddModelError("", error.Description);

        return View(model);
    }

}
```

****

REGISTER

```
@* ***************** ACCOUNT/REGISTER ****************** *@
@* ***************** ACCOUNT/REGISTER ****************** *@
@* ***************** ACCOUNT/REGISTER ****************** *@

@model AppUserRegisterVm

@{
    Layout = null;
}

<!DOCTYPE html>
<html lang="en">

<head>
```

```
    <title>Kullanıcı Kaydı</title>
    <meta content="" name="description">
    <meta content="" name="keywords">

    @* Her sayfada kullanılan sabit css ve font linkleri burada tutuluyor. *@
    @await Html.PartialAsync("_Head")

</head>

<body>

    <main>
        <div class="container">

            <section class="section register min-vh-100 d-flex flex-column align-items-
center justify-content-center py-4">
                <div class="container">
                    <div class="row justify-content-center">
                        <div class="col-lg-4 col-md-6 d-flex flex-column align-items-
center justify-content-center">

                            <div class="d-flex justify-content-center py-4">
                                <a href="/home/index" class="logo d-flex align-items-
center w-auto">
                                    <img src="~/Template/assets/img/KbuMudeklogo.png"
alt="">
                                    <span class="d-none d-lg-block">KbuMüdek</span>
                                </a>
                            </div><!-- End Logo -->

                            <div class="card mb-3">

                                <div class="card-body">

                                    <div class="pt-4 pb-2">
                                        <h5 class="card-title text-center pb-0 fs-4">Hesap
Oluştur</h5>
                                        <p class="text-center small">Hesap oluşturmak için
kişisel bilgilerinizi girin</p>
                                    </div>

                                    <form method="post" class="row g-3 needs-validation">

                                        @* ************* TC KİMLİK NO ************* *@
                                        <div class="col-12">
                                            <label for="yourIdentityNumber" class="form-
label">TC Kimlik Numarası</label>

                                            <div class="input-group has-validation">
                                                <span class="input-group-text"
id="inputGroupPrepend"></span>

                                                <input type="tel" asp-for="IdentityNumber"
class="form-control" id="myIdentityNumber"
                                                       required
                                                       minlength="11"
                                                       maxlength="11"
                                                       pattern="\d{11}">
                                                <div class="invalid-feedback">Lütfen TC
Kimlik Numaranızı 11 hane giriniz!</div>
                                                <span asp-validation-for="IdentityNumber"
class="text-danger"></span> @* daha önceden TC kaydı varsa modelstate deki bu hata
görüntülenecek *@
                                            </div>
                                        </div>

                                        @* ************* PROFILE TYPE ************* *@
                                        <div class="col-12">
```

```
                                        <label for="ProfileTypeId" class="form-
label">Profil Tipi</label>
                                        <select asp-for="ProfileTypeId" asp-
items="Model.ProfileTypes" class="form-select" required>
                                            <option value="">-- Seçiniz --</option>
                                        </select>
                                        <span asp-validation-for="ProfileTypeId"
class="text-danger"></span>
                                    </div>


                                    @* ************* AD ************* *@
                                    <div class="col-12">
                                        <label for="yourName" class="form-
label">Adınız</label>
                                        <input type="text" asp-for="FirstName"
class="form-control" id="myFirstName" required>
                                        <div class="invalid-feedback">Lütfen adınızı
giriniz!</div>
                                    </div>

                                    @* ************* SOYAD ************* *@
                                    <div class="col-12">
                                        <label for="yourName" class="form-
label">Soyadınız</label>
                                        <input type="text" asp-for="LastName"
class="form-control" id="myLastName" required>
                                        <div class="invalid-feedback">Lütfen
soyadınızı giriniz!</div>
                                    </div>

                                    @* ************* EMAIL ************* *@
                                    <div class="col-12">
                                        <label for="yourEmail" class="form-label">Mail
Adresi</label>
                                        <input type="email" asp-for="Email" class="
form-control" id="myEmail" required>
                                        <div class="invalid-feedback">Lütfen geçerli
bir mail adresi giriniz!</div>
                                        <span asp-validation-for="Email" class="text-
danger"></span> @* daha önceden Email kaydı varsa modelstate deki bu hata görüntülenecek
*@
                                    </div>


                                    @* ************* ŞİFRE ************* *@
                                    <div class="col-12">
                                        <label for="yourPassword" class="form-
label">Şifre</label>
                                        <input type="password" asp-for="Password"
class=" form-control" id="myPassword" required>
                                        <div class="invalid-feedback">Lütfen şifrenizi
girin!</div>
                                    </div>

                                    @* ************* ŞİFRE TEKRARI ************* *@
                                    <div class="col-12">
                                        <label for="yourPassword" class="form-
label">Şifre tekrarı</label>
                                        <input type="password" asp-
for="ConfirmPassword" class=" form-control" id="myConfirmPassword" required>
                                        <div class="invalid-feedback">Lütfen şifrenizi
tekrar girin!</div>
                                        <span asp-validation-for="ConfirmPassword"
class="text-danger"></span>
                                    </div>
```

```
                                        @* VALIDATION ÖZETİ----------ASLINDA yukarıdaki
"invalit-feedback" ler validation sağlıyor. Fakat şifre detaylarının validasyonu model
içerisinde gelmeldir.  *@
                                        <div asp-validation-summary="ModelOnly"></div>


                                        @* ************* ŞARTLAR ************* *@
                                        <div class="col-12">
                                            <div class="form-check">
                                                <input class="form-check-input"
name="terms" type="checkbox" value="" id="acceptTerms" required>
                                                <label class="form-check-label"
for="acceptTerms"><a href="#">Şartlar ve koşulları</a> kabul ediyorum </label>
                                                <div class="invalid-feedback">Göndermeden
önce kabul etmeniz gerekmektedir</div>
                                            </div>
                                        </div>
                                        <div class="col-12">
                                            <button class="btn btn-primary w-100"
type="submit">Hesap oluştur</button>
                                        </div>
                                        <div class="col-12">
                                            <p class="small mb-0">Zaten bir hesabınız var
mı? <a href="/account/login">Giriş yap</a></p>
                                        </div>
                                    </form>

                                </div>
                            </div>

                            <div class="credits" style="font-size: 8pt;">
                                &copy; Copyright, Programmed by <a
href="https://ibrahimcayiroglu.com/" target="_blank">icayiroglu</a>
                            </div>

                        </div>
                    </div>
                </div>

            </section>

        </div>
    </main><!-- End #main -->

    <a href="#" class="back-to-top d-flex align-items-center justify-content-center"><i
class="bi bi-arrow-up-short"></i></a>

    @* Her sayfada kullanılan sabit java script linkleri burada tutuluyor. *@
    @await Html.PartialAsync("_Scripts")

</body>

</html>
```

****

CONFIRMEPASSWORDRESULT

```
@model GenericResultVm

@{
    Layout = null;
}
```

```
<!DOCTYPE html>
<html lang="en">

<head>
    <title>Confirm Email Error</title>
    <meta content="" name="description">
    <meta content="" name="keywords">

    @* Her sayfada kullanılan sabit css ve font linkleri burada tutuluyor. *@
    @await Html.PartialAsync("_Head")

</head>

<body>

    <main>
        <div class="container">

            <section class="section register min-vh-100 d-flex flex-column align-items-
center justify-content-center py-4">
                <div class="container">
                    <div class="row justify-content-center">
                        <div class="col-lg-4 col-md-6 d-flex flex-column align-items-
center justify-content-center">

                            <div class="d-flex justify-content-center py-4">
                                <a href="index.html" class="logo d-flex align-items-center
w-auto">
                                    <img src="~/Template/assets/img/KbuMudeklogo.png"
alt="">
                                    <span class="d-none d-lg-block">KbuMüdek</span>
                                </a>
                            </div><!-- End Logo -->

                            <div class="card mb-3">

                                <div class="card-body">

                                    <div class="pt-4 pb-2">
                                        <h2 class="card-title text-center pb-0 fs-
4">@Model.Title</h2>

                                        <div class="alert alert-@Model.ColorClass">
                                            <p>@Model.Description</p>
                                         </div>

                                        @if (Model.ShowLink)
                                        {
                                            <div class="text-center mt-3">
                                                <a href="@Model.ReturnUrl" class="btn btn-
outline-@Model.ColorClass">

                                                    @Model.UrlText
                                                </a>
                                            </div>
                                        }
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>

            </section>

        </div>
```

```
    </main><!-- End #main -->

    <a href="#" class="back-to-top d-flex align-items-center justify-content-center"><i
class="bi bi-arrow-up-short"></i></a>

    @* Her sayfada kullanılan sabit java script linkleri burada tutuluyor. *@
    @await Html.PartialAsync("_Scripts")

</body>

</html>
```

****

LOGIN

```
@model LoginVm
@{
    Layout = null;
}

<!DOCTYPE html>
<html lang="en">

<head>

    <title>Kullanıcı Girişi</title>
    <meta content="" name="description">
    <meta content="" name="keywords">

    @* Her sayfada kullanılan sabit css ve font linkleri burada tutuluyor. *@
    @await Html.PartialAsync("_Head")

</head>

<body>

    <main>
        <div class="container">

            <section class="section register min-vh-100 d-flex flex-column align-items-
center justify-content-center py-4">
                <div class="container">
                    <div class="row justify-content-center">
                        <div class="col-lg-4 col-md-6 d-flex flex-column align-items-
center justify-content-center">

                            <div class="d-flex justify-content-center py-4">
                                <a href="/home/index" class="logo d-flex align-items-
center w-auto">
                                    <img src="~/Template/assets/img/KbuMudeklogo.png"
alt="">
                                    <span class="d-none d-lg-block">KbuMüdek</span>
                                </a>
                            </div><!-- End Logo -->

                            <div class="card mb-3">

                                <div class="card-body">

                                    <div class="pt-4 pb-2">
                                        <h5 class="card-title text-center pb-0 fs-
4">Kullanıcı Girişi</h5>
                                        @* <p class="text-center small">TC Kimlik
Numaranızı ve Şifrenizi giriniz.</p> *@
```

```
                                    <p class="text-center small">E-posta adresinizi ve
şifrenizi giriniz.</p>
                                </div>

                                <form asp-action="Login" method="post" class="row g-3
needs-validation" novalidate>

                                    @* Genel hatalar burada gösterilir *@
                                    <div asp-validation-summary="All" class="text-
danger"></div>

                                    <div class="col-12">
                                        <label asp-for="Email" class="form-label">E-
posta</label>
                                        <div class="input-group has-validation">
                                            <span class="input-group-text"> </span>
                                            <input type="email" asp-for="Email"
class="form-control" required />
                                            <div class="invalid-feedback">Lütfen
geçerli bir e-posta adresi giriniz.</div>
                                        </div>
                                        <span asp-validation-for="Email" class="text-
danger"></span>
                                    </div>

                                    <div class="col-12">
                                        <label asp-for="Password" class="form-
label"></label>
                                        <input asp-for="Password" class="form-control"
required />
                                        <div class="invalid-feedback">Lütfen şifrenizi
giriniz.</div>
                                        <span asp-validation-for="Password"
class="text-danger"></span>
                                    </div>

                                    <div class="col-12">
                                        <div class="form-check">
                                            <input asp-for="RememberMe" class="form-
check-input" />
                                            <label asp-for="RememberMe" class="form-
check-label"></label>
                                        </div>
                                    </div>

                                    <div class="col-12">
                                        <button class="btn btn-primary w-100"
type="submit">Giriş Yap</button>
                                    </div>

                                    <div class="col-12">
                                        <p class="small mb-0">Hesabınız yok mu? <a
href="/Account/Register">Hesap Oluştur</a></p>
                                        <p class="small mb-0">
                                            Şifrenizi mi unuttunuz? <a
href="/Account/ForgotPassword">Şifremi Unuttum</a>
                                        </p>
                                    </div>
                                </form>

                            </div>
                        </div>

                        <div class="credits" style="font-size: 8pt;">
```

```
                                &copy; Copyright, Programmed by <a
href="https://ibrahimcayiroglu.com/" target="_blank">icayiroglu</a>
                            </div>


                        </div>
                    </div>
                </div>

            </section>

        </div>
    </main><!-- End #main -->

    <a href="#" class="back-to-top d-flex align-items-center justify-content-center"><i
class="bi bi-arrow-up-short"></i></a>


 @* Her sayfada kullanılan sabit java script linkleri burada tutuluyor. *@
    @await Html.PartialAsync("_Scripts")

</body>

</html>
```

****

FORGOTPASSWORD

```
@model ForgotPasswordVm
@{
    Layout = null;
}

<!DOCTYPE html>
<html lang="tr">
<head>
    <title>Şifremi Unuttum</title>
    @await Html.PartialAsync("_Head")
</head>
<body>

<main>
    <div class="container">
        <section class="section register min-vh-100 d-flex flex-column align-items-center
justify-content-center py-4">
            <div class="container">
                <div class="row justify-content-center">
                    <div class="col-lg-4 col-md-6 d-flex flex-column align-items-center
justify-content-center">

                        <div class="d-flex justify-content-center py-4">
                            <a href="/home/index" class="logo d-flex align-items-center w-
auto">
                                <img src="~/Template/assets/img/KbuMudeklogo.png" alt="">
                                <span class="d-none d-lg-block">KbuMüdek</span>
                            </a>
                        </div>

                        <div class="card mb-3">
                            <div class="card-body">
                                <div class="pt-4 pb-2">
                                    <h5 class="card-title text-center pb-0 fs-4">Şifremi
Unuttum</h5>
                                    <p class="text-center small">E-posta adresinizi girin,
şifre sıfırlama linki gönderilecektir</p>
```

```
                                </div>

                                <form asp-action="ForgotPassword" method="post" class="row
g-3 needs-validation" novalidate>
                                    <div asp-validation-summary="All" class="text-
danger"></div>

                                    <div class="col-12">
                                        <label asp-for="Email" class="form-label">E-
posta</label>
                                        <div class="input-group has-validation">
                                            <span class="input-group-text"></span>
                                            <input asp-for="Email" class="form-control"
required />
                                            <div class="invalid-feedback">Geçerli bir e-
posta giriniz.</div>
                                        </div>
                                        <span asp-validation-for="Email" class="text-
danger"></span>
                                    </div>

                                    <div class="col-12">
                                        <button class="btn btn-primary w-100"
type="submit">Gönder</button>
                                    </div>

                                    <div class="col-12">
                                        <p class="small mb-0"><a
href="@Url.Action("Login")">Giriş sayfasına dön</a></p>
                                    </div>
                                </form>
                            </div>
                        </div>

                        <div class="credits" style="font-size: 8pt;">
                            &copy; Copyright, Programmed by <a
href="https://ibrahimcayiroglu.com/" target="_blank">icayiroglu</a>
                        </div>
                    </div>
                </div>
            </div>
        </section>
    </div>
</main>

@await Html.PartialAsync("_Scripts")
</body>
</html>
```

****

RESET PASSWORD

```
@model ResetPasswordVm
@{
    Layout = null;
}

<!DOCTYPE html>
<html lang="tr">
<head>
    <title>Yeni Şifre Oluştur</title>
    @await Html.PartialAsync("_Head")
</head>
<body>
```

```
    <main>
        <div class="container">
            <section class="section register min-vh-100 d-flex flex-column align-items-
center justify-content-center py-4">
                <div class="container">
                    <div class="row justify-content-center">
                        <div class="col-lg-4 col-md-6 d-flex flex-column align-items-
center justify-content-center">

                            <div class="d-flex justify-content-center py-4">
                                <a href="/home/index" class="logo d-flex align-items-
center w-auto">
                                    <img src="~/Template/assets/img/KbuMudeklogo.png"
alt="">
                                    <span class="d-none d-lg-block">KbuMüdek</span>
                                </a>
                            </div>

                            <div class="card mb-3">
                                <div class="card-body">
                                    <div class="pt-4 pb-2">
                                        <h5 class="card-title text-center pb-0 fs-4">Yeni
Şifre Belirle</h5>
                                        <p class="text-center small">Güvenli bir şifre
oluşturun.</p>
                                    </div>

                                    <form asp-action="ResetPassword" method="post"
class="row g-3 needs-validation" novalidate>
                                        <input type="hidden" asp-for="UserId" />
                                        <input type="hidden" asp-for="Token" />

                                        <div asp-validation-summary="All" class="text-
danger"></div>

                                        <div class="col-12">
                                            <label asp-for="Password" class="form-
label">Yeni Şifre</label>
                                            <input asp-for="Password" class="form-control"
required />
                                            <div class="invalid-feedback">En az 6
karakterli şifre girin.</div>
                                        </div>

                                        <div class="col-12">
                                            <label asp-for="ConfirmPassword" class="form-
label">Tekrar</label>
                                            <input asp-for="ConfirmPassword" class="form-
control" required />
                                            <div class="invalid-feedback">Şifreler
eşleşmiyor.</div>
                                        </div>

                                        <div class="col-12">
                                            <button class="btn btn-primary w-100"
type="submit">Şifreyi Değiştir</button>
                                        </div>
                                    </form>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </section>
        </div>
```

```
    </main>

    @await Html.PartialAsync("_Scripts")
</body>
</html>
```

****

****

PASSWORD RESET CONFIRMATION

```
@{
    Layout = null;
}

<!DOCTYPE html>
<html lang="tr">
<head>
    <title>Şifre Değiştirildi</title>
    @await Html.PartialAsync("_Head")
</head>
<body>

    <main>
        <div class="container">
            <section class="section register min-vh-100 d-flex flex-column align-items-
center justify-content-center py-4">
                <div class="container">
                    <div class="row justify-content-center">
                        <div class="col-lg-4 col-md-6 d-flex flex-column align-items-
center justify-content-center">

                            <div class="d-flex justify-content-center py-4">
                                <a href="/home/index" class="logo d-flex align-items-
center w-auto">
                                    <img src="~/Template/assets/img/KbuMudeklogo.png"
alt="">
                                    <span class="d-none d-lg-block">KbuMüdek</span>
                                </a>
                            </div>

                            <div class="card mb-3">
                                <div class="card-body text-center py-5">
                                    <i class="bi bi-lock-fill text-success" style="font-
size: 3rem;"></i>
                                    <h5 class="card-title mt-3">Şifreniz
Değiştirildi!</h5>
                                    <p>Yeni şifrenizle giriş yapabilirsiniz.</p>
                                    <a href="@Url.Action("Login")" class="btn btn-
primary">Giriş Yap</a>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </section>
        </div>
    </main>

    @await Html.PartialAsync("_Scripts")
</body>
</html>
```

****

ROLE KULLANIMI

Rol kullanımı ile ilgii YZ de yapılan araştırmalar hakkında kısa bilgiler aşağıda verilmiştir.

https://chatgpt.com/c/67d09e9a-99b4-8013-91da-12942cdab317

https://claude.ai/chat/328714dc-e10a-495f-b26a-100fb003673c

https://x.com/i/grok?focus=1&conversation=1899560240120754298