



YOZGAT BOZOK ÜNİVERSİTESİ
MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
YAZILIM MÜHENDİSLİĞİ DERSİ
TEKNİK DOKÜMAN

Ders Konusu: Üniversite Öğrencileri İçin Eğitim Ağı Aracı

Proje Başlığı: TEKNO EĞİTİM

Grup Numarası: 10

Teslim Tarihi: 03.05.2020

Ekip Üyeleri İsimleri:

ECEM YAMAN
RÜMEYSA GEDİK
BEYZA DURNA
FURKAN BAL
MEHMET ÜZÜM

İçindekiler

A.ADMİN TEKNİK DOKÜMAN.....	1
AZURE CLOUD KULLANILMA SEBEBİ:	1
ASP.NET CORE KULLANILMA SEBEBİ:	1
ASP.NET CORE IDENTITY	1
ASP.NET CORE UYGULAMASI OLUŞTURMA.....	2
UYGULAMAYA GEREKLİ KÜTÜPHANELERİN EKLENMESİ İSKELETİN OLUŞTURULMASI	3
PROJEDE KULLANILAN KÜTÜPHANELER	3
BU KÜTÜPHANELERİ İNDİRMEK İÇİN İZLENECEK YOL;	3
ASPNET USERS TABLOSU	6
ASPNET ROLES TABLOSU	7
USER MANAGER.....	8
USER VIEW MANAGER	8
B.KULLANICI TEKNİK DOKÜMAN.....	10

A.ADMİN TEKNİK DOKÜMAN

Bu doküman technical role map içermektedir.

PROJEDE KULLANILAN TEKNOLOJİLER:

-ASP.NET CORE- MVC

-IDENTİTY API

-MICROSOFT AZURE CLOUD

ASP.NET CORE MVC olmakla birlikte kullanılan MVC türü, Entity Frame Work Codefirst yöntemidir.

Oluşturulan siten uygulamasının verilerini saklamak ve yönetmek amacıyla Microsoft Azure Cloud kullanılacaktır.

AZURE CLOUD KULLANILMA SEBEBİ:

-Azure'da bulunan ölçeklendirme özelliği sayesinde sunucuya yük veya plana göre elle veya otomatik olarak ölçeklendirme yapılabilmesi.

-Web sitesini yedekleyerek güvende kalmasının sağlanabilmesi.

- Etki alanının verimli kullanılabilmesi.

- Güvenliği güçlendirmek amacıyla SSL sertifikası .

-Uygulamanın host edilip, yayınlanması.

ASP.NET CORE KULLANILMA SEBEBİ:

ASP.NET Core, tüm ASP.NET altyapısının yeniden tasarlanarak, Web API ve MVC altyapıları ile birleştirilmesini sağlamıştır.

Web uygulamaları için API'lar oluşturulması bu şekilde daha kolay hale gelmiştir. Microsoft bu platformu düzenli olarak geliştirmektedir. ASP.NET Core teknolojisi sayesinde modern uygulamaları daha az efor ve maliyet ile, daha kısa sürede oluşturabilmeye olanak sağlanmıştır.

ASP.NET CORE IDENTITY

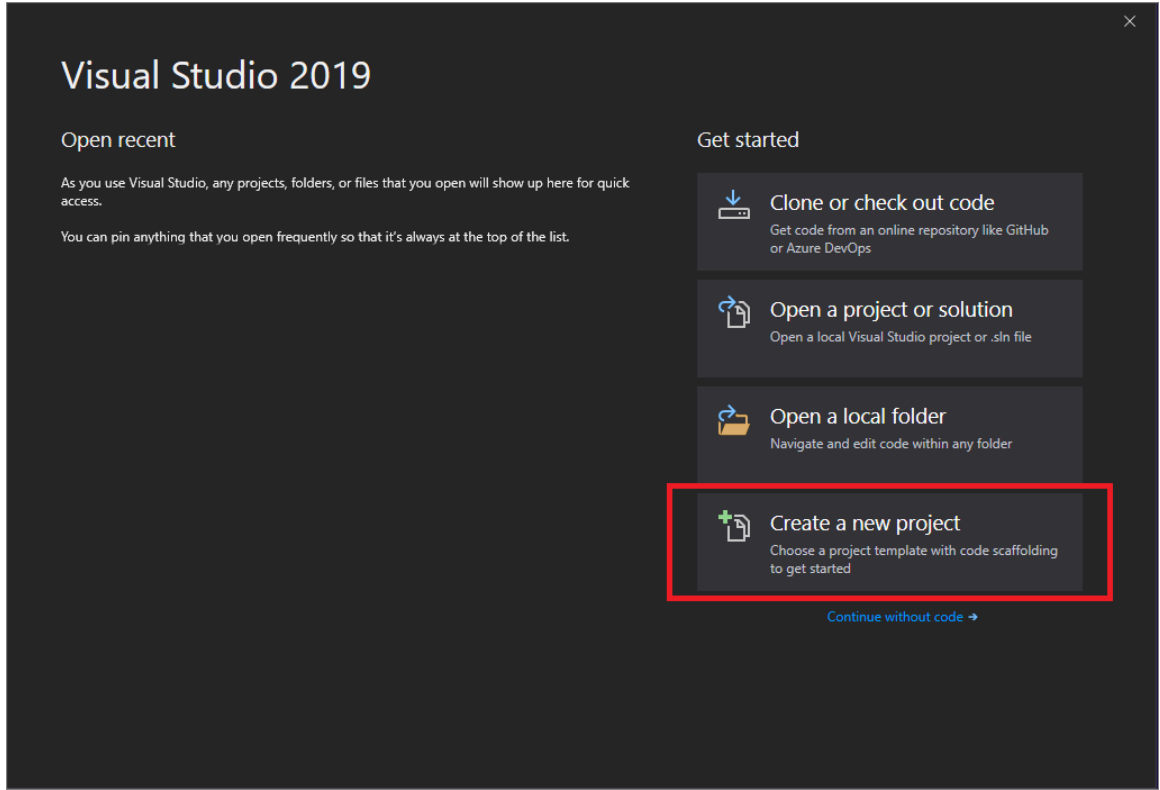
Asp.NET Core Identity; üyelerin, login(giriş), out(çıkış), yetkilendirme, token, şifre hatırlatma vs. tüm işlemleri hızlı bir şekilde gerçekleştirmemizi sağlayan ve bunların dışında önceki nesillere nazaran herhangi bir kısıtlaması olmaksızın uygulamalarımızı destekleyen

çağdaş bir sistemidir.

ASP.NET CORE UYGULAMASI OLUŞTURMA

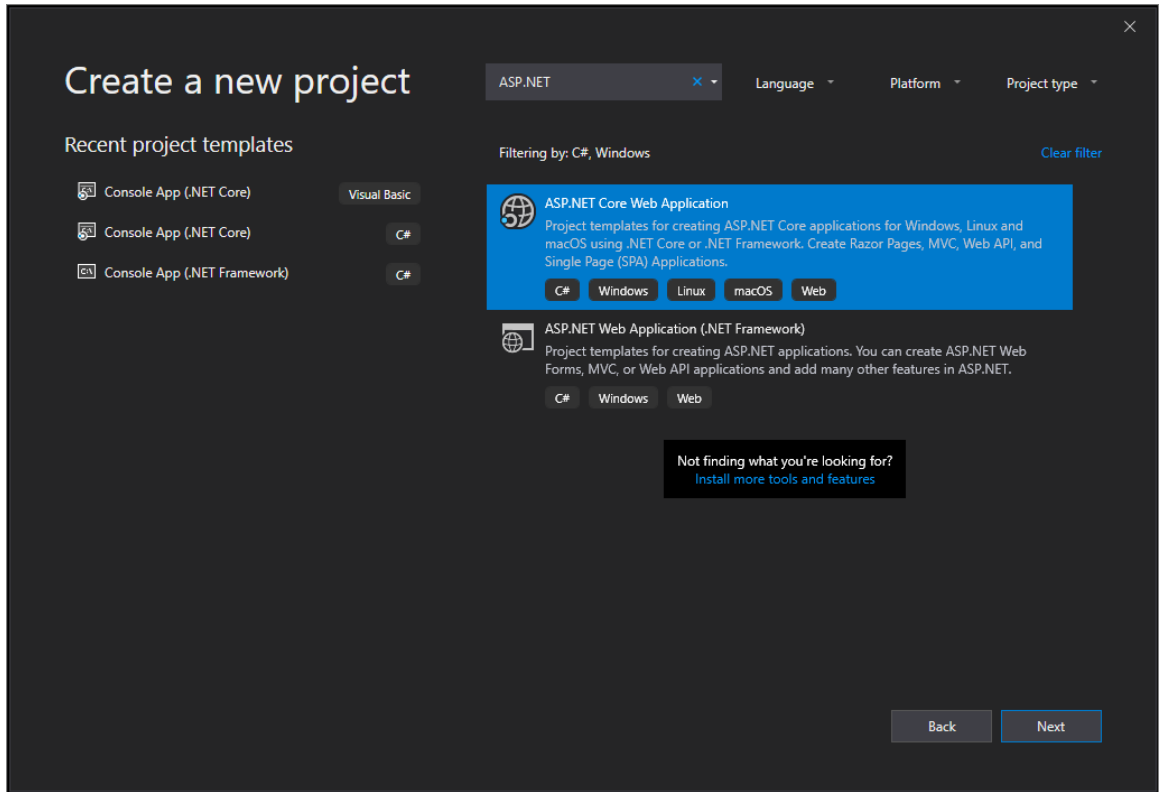
1-) Visual Studio'yu açın.

2-) Başlangıç penceresinde yeni bir **proje oluştur**'u seçin.

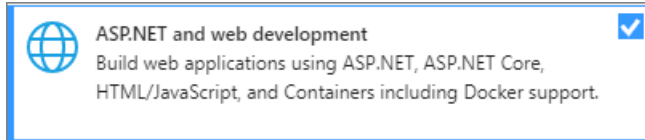


3-) Yeni proje oluştur penceresinde, arama kutusuna ASP.NET girin veya yazın. Ardından, Dil listesinden C#'yi seçin ve ardından Platform listesinden Windows'u seçin.

Dil ve platform filtrelerini uyguladıktan sonra, ASP.NET Çekirdek Web Uygulaması şablonunu seçin ve sonra İleri'yiseçin.



* **ASP.NET Çekirdek Web Uygulaması şablonu görmüyorsanız, yeni bir proje oluştur penceresinden yükleyebilirsiniz. Aradığınızı bulamıyor musunuz? iletisinde, daha fazla araç ve özellik yükle bağlantısını seçin.**



UYGULAMAYA GEREKLİ KÜTÜPHANELERİN EKLENMESİ İSKELETİN OLUŞTURULMASI

PROJEDE KULLANILAN KÜTÜPHANELER

Microsoft.AspNetCore.Identity.EntityFrameworkCore

Microsoft.EntityFrameworkCore.SqlServer

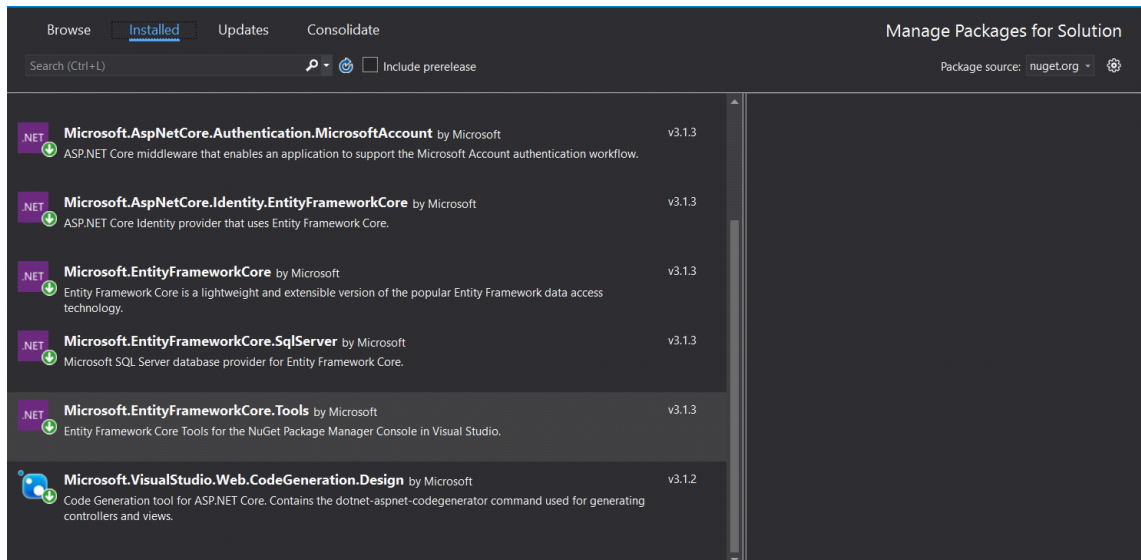
Microsoft.EntityFrameworkCore.Design

Microsoft.EntityFrameworkCore.Tools

Microsoft.AspNetCore.Identity.UI

BU KÜTÜPHANELERİ İNDİRMEK İÇİN İZLENECEK YOL;

Visual Studio > Tools > NuGet Package Manager > Package Manager Settings



Nuget paketlerini ekledikten sonra veritabanı bağlantımız için **appsettings.json** dosyamıza database connection string'i eklenir.

```
"ConnectionStrings": {  
  "DefaultConnection": "User ID =User;Password=Password;Server=Host;Port=5432;Database=Database;Inte  
}
```

Sonrasında **DbContext**'i oluşturulur. Identity ile gelen hazır tablo yapılarını kullanabilmek için DbContext'i, **IdentityDbContext** sınıfı ile türetilir.

Startup.cs içindeki **ConfigureServices** metodunun içine aşağıdaki kod bloğu eklenir.

```
1 services.AddRazorPages();  
2 services.AddDbContext<IdentitySampleDbContext>(options =>  
3     options.UseNpgsql(Configuration.GetConnectionString("DefaultConnection")));  
4 services.AddDefaultIdentity<IdentityUser>()  
5     .AddEntityFrameworkStores<IdentitySampleDbContext>();
```

Aynı class içindeki Configure metoduna eklenir.

```
1 app.UseAuthentication();
```

CodeFirst mantığıyla veritabanını oluşturmak için Microsoft PowerShell açılır .

Projenin olduğu dosya dizini kopyalanıp PowerShell'e yapıştırılır.

Select Windows PowerShell (x86)

```
PS C:\Users\RUMEYSA\Desktop\WebApp\WebApp> cd C:\Users\RUMEYSA\Desktop\WebApp\WebApp
```

Dotnet tooları indirilir.

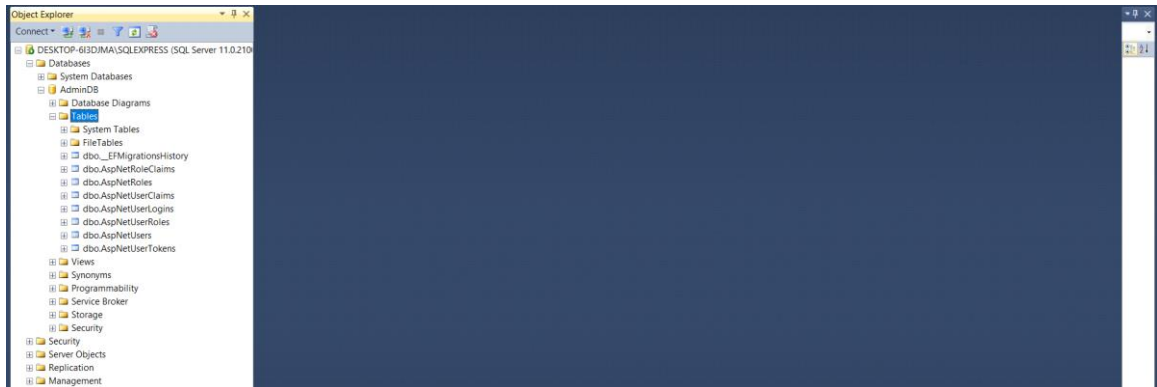
```
PS C:\Users\RUMEYSA\Desktop\WebApp\WebApp> dotnet tool install --global dotnet-ef --version 3.1.3
```

Kod ile veritabanı arasındaki metod akışı bilgisini tutan ve köprü görevini gören Migration oluşturulur.

```
PS C:\Users\RUMEYSA\Desktop\WebApp\WebApp> dotnet ef migrations add Initial
```

Sonrasında işlem başarı ile oluştuktan sonra veritabanı oluşur.

```
Build started...  
Build succeeded.  
Done.
```



İskelet Bu şekildedir.

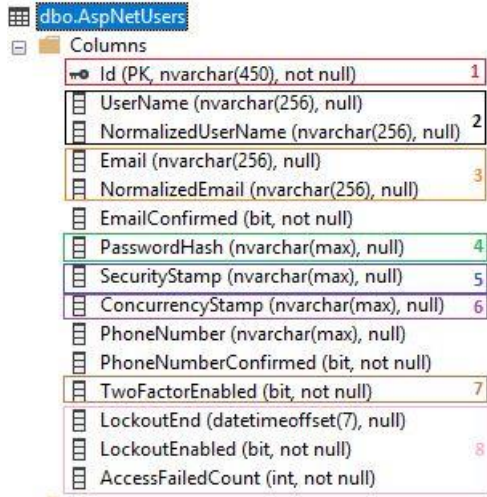
Identity ile gelen veritabanında aktif olarak developer'ın kullanıp müdahale edebileceği iki tablo vardır;

-AspNetUsers

-AspNetRoles

Diğer var olan tablolar bu iki tablodaki değişime göre şekillenmektedir.

ASPNET USERS TABLOSU



Columns	
Id (PK, nvarchar(450), not null)	1
UserName (nvarchar(256), null)	
NormalizedUserName (nvarchar(256), null)	2
Email (nvarchar(256), null)	
NormalizedEmail (nvarchar(256), null)	3
EmailConfirmed (bit, not null)	
PasswordHash (nvarchar(max), null)	4
SecurityStamp (nvarchar(max), null)	5
ConcurrencyStamp (nvarchar(max), null)	6
PhoneNumber (nvarchar(max), null)	
PhoneNumberConfirmed (bit, not null)	
TwoFactorEnabled (bit, not null)	7
LockoutEnd (datetimeoffset(7), null)	
LockoutEnabled (bit, not null)	8
AccessFailedCount (int, not null)	

Kullanıcılarla ilgili genel geçer bilgileri tutan tablodur.

1. Her bir kullanıcıyı temsil edecek olan Primary Key Id değeri. Dikkat edilirse nvarchar olarak ayarlanmıştır. Bu ayarlama güvenlik sebebi ile yapılmıştır.
2. Kullanıcı adını tutan kolondur. Dikkat edilirse “UserName” kolonunun hemen altında “NormalizedUserName” kolonunda mevcuttur. Milyonluk/milyarlık verilerde (**Big Data**) **arama** sorguları gerçekleştirilirken hız kazanmak için oluşturulmuş kolondur. Big datanın söz konusu olduğu durumlarda bir arama sorgusu “UserName” üzerinde değil “NormalizedUserName” kolonu üzerinde gerçekleştirilir. Çünkü “Normalized...” ile başlayan kolonlar indekslenmiştir. Dolayısıyla sorgularda daha hızlı çalışan bir altyapıya sahiptirler. “Normalized...” kolonlarının bir diğer özelliği tek bir formatta veri tutmasıdır. Örneğin; “UserName” kolonunda bulunan “gedik” değeri aynen “NormalizedUserName” kolonuna “GEDİK” olarak büyük harflerle kopyalanacaktır. Büyük harflerle girilmesinin sebebi tek bir format olmasının getirisi olan daha hızlı eşleştirme yapılabilir olmasıyla birlikte daha hızlı indexlenebilir olmasıdır. Bir kolonun “Normalized...” versiyonu varsa eğer o kolonun kendisi indexlenmemektedir. Dolayısıyla asıl hızlı sorgulamalar Entity Framework tarafından Identity’ye özel “Normalized...” kolonlarına odaklanılarak gerçekleştirilir. Bir önceki Identity sürümünde tüm indexlemeler direkt olarak ilgili kolon üzerinde gerçekleştiriliyordu. Dolayısıyla “Normalized...” ile başlayan kolonlar söz konusu değildi. Bunun getirisinin yanında birçok kolonun hataların yaşanması maliyeti oldukça yükseltiyordu.
3. Kullanıcının e-posta bilgisini tutan alandır. 2. numaralı kolonda olduğu gibi bu kolonunda aynı amaca hizmet eden indexlenmiş değeri olan “NormalizedEmail” kolonu mevcuttur.
4. Kullanıcıdan alınan tüm passwordler **Hash algoritmasıyla** şifrelenerek tutulmaktadır.

5. Kullanıcı ilk oluşturulduğunda(create) buraya bir build değeri atanır. Sonraki her güncelleme üzerine bu değer güncellenecektir. Dolayısıyla bizlerde bu kullanıcı üzerinde bir değişiklik olduğuna dair bilgi edinmiş olacağız. Bir nevi Data Concurrency sağlayabilmek için oluşturulmuştur.

6. İlgili veri üzerinde **Data Concurrency** sağlayabilmek için oluşturulmuştur.

7. Kullanıcının kaydı neticesinde aktivasyon yapılanmasının iki adımlı olup olmadığına dair kayıt tutar. Bu kolon genellikle 3. party üyelik sistemleriyle birlikte kullanılır. Örneğin; kullanıcı Facebook üzerinden giriş yaptığında ekstradan telefon ile onay gerektiriyorsa işte **“TwoFactorEnabled”** kolonu true/1 olarak işaretlenecektir.

8. Kullanıcı girişlerine dair yapılan hataları ve engel durumunu tutan kolonlardır. **“LockoutEnd”** kolonu kaç kez yanlış girildiğine dair, **“LockoutEnabled”** kolonu kullanıcının aktifliğine dair ve **“AccessFailedCount”** kolonu ise kaç başarısız giriş yapılmaya çalışıldığına dair bilgi tutmaktadır. Identity mimarisi bu yapıları otomatik işleyecektir. Dolayısıyla tarafımızca custom bir kod geliştirmemize lüzum görülmemektedir.

ASPNET ROLES TABLOSU



Columns
Id (PK, nvarchar(450), not null)
Name (nvarchar(256), null)
NormalizedName (nvarchar(256), null)
ConcurrencyStamp (nvarchar(max), null)

Uygulamada kullanıcılara özgü tanımlanan rolleri tutan tablodur. Şöyle tabloyu incelersek rollerin adını tutacak olan **“Name”** kolonu ve bu değerleri indexleyip big data durumlarında performans kazandıracak indexlenmiş **“NormalizedName”** kolonu mevcuttur. Ayriyetten veri tutarlılığı için de **“ConcurrencyStamp”** kolonu mevcuttur.

Üye olma ve admin yetkilendirme işlemleri **Cookie** tabanlı bir mekanizma ile gerçekleştirilmektedir.

Identity ile ilgili tüm ayarların gerçekleştiği yer **startup.cs** dosyasının **ConfigureServices** metodudur.

Identitynin defaulttaki şifre mekanizmalarını kullanmayıp custom olarak oluşan şifre mekanizmaları burada gerçekleşir.

Asıl iş **IPasswordValidator<AppUser>** classlara interface olarak geçtiğimizde bu interfacenin içerisine metod gelecek bu metod **ValidateAsync** Metodu olacak bu sayede custom

mekanizmalar yapabiliyoruz. içerisinde yararlı bilgilerde geliyor parametre olarak user manager user password gibi Hatta identity devre dışında bırakabiliyoruz fazla değişikliklerde yapabiliyoruz.

API de iki tane kullanıcı doğrulama mekanizması var:

-RequireUniqueEmail, username'in, veritabanında tek olup olmadığı kontrolü yapar.

-AllowedUserNameCharacters, kullanılmasını istediğimiz karakterlerin belirlenmesinde olarak sağlar.

Identity de kullanıcı aranırken username ile aranıyor. O yüzden kullanıcı isminin iki tane olmasına müsaade edilmez.

Emailin, kullanıcı adının tek olması **startup.cs** burada sağlanır.

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<AppIdentityDbContext>(opts =>
    {
        opts.UseSqlServer(configuration["ConnectionStrings:DefaultConnectionString"]);
    });

    services.AddIdentity<AppUser, AppRole>(opts =>
    {
        opts.User.RequireUniqueEmail = true;
        opts.User.AllowedUserNameCharacters = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-._";

        opts.Password.RequiredLength = 4;
        opts.Password.RequireNonAlphanumeric = false;
        opts.Password.RequireLowercase = false;
        opts.Password.RequireUppercase = false;
        opts.Password.RequireDigit = false;
    });

    services.AddPasswordValidator<CustomPasswordValidator>().AddEntityFrameworkStores<AppIdentityDbContext>();
}
```

USER MANAGER

Kullanıcıyla ilgili tüm işlemlerin gerçekleştirildiği sınıftır. İçerisinde birçok metod barındırıyor.

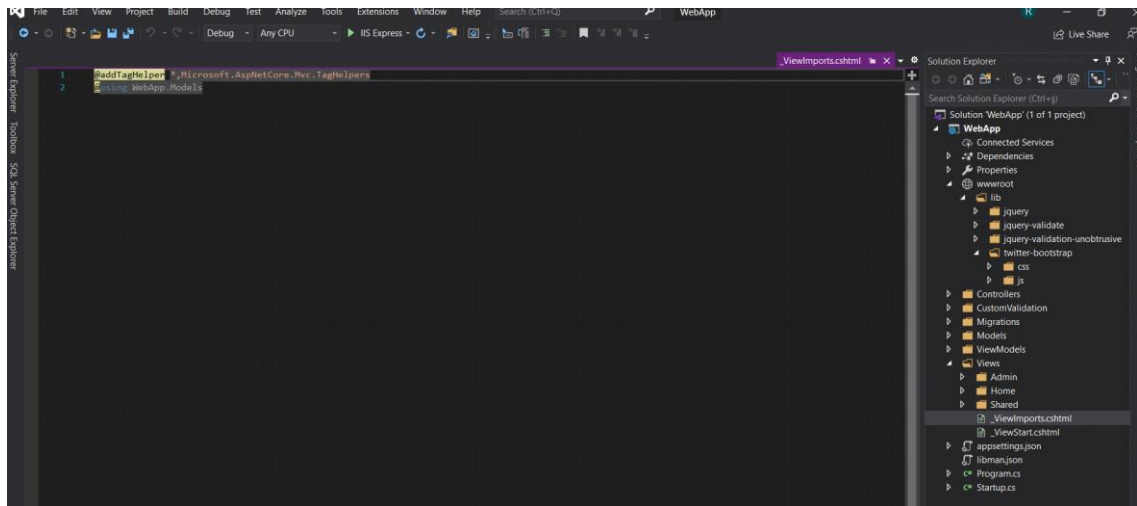
User manager'ın sadece get özelliği var dependence injection sayesinde set edilecek.

USER VIEW MANAGER

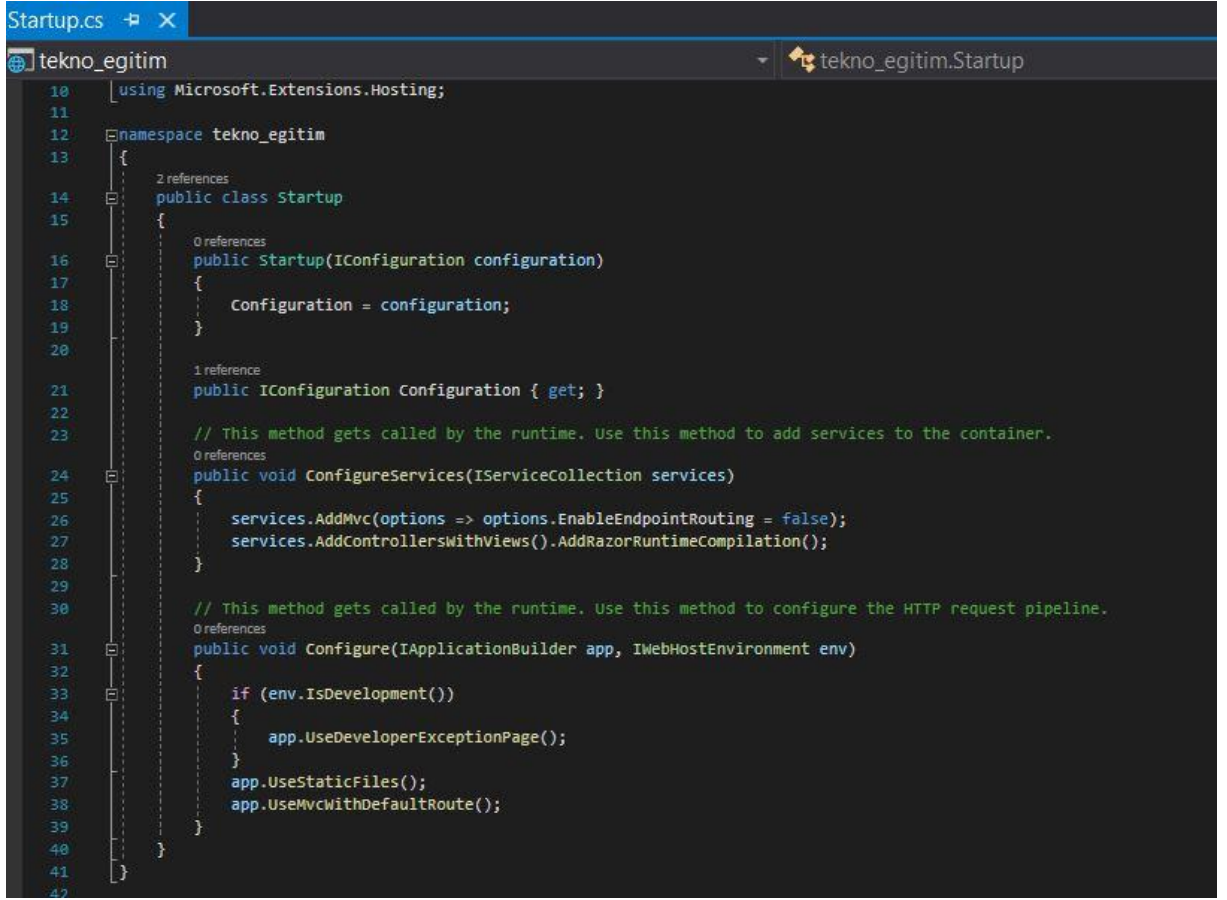
Sql'de var olan bilgilerin hepsi kullanıcıya gösterilmez. İstenmediğinden dolayıda sqlde'ki tabloda olan verileri koda dökmeye gerek yoktur. Bunun için biz AppUser ile kullanıcıya gösterilecek alan arasında köprü kuracak view modeller oluşturulur.

Kısaca custom bir sınıf olur bu da entity ile kullanıcı arasında köprü görevi görür.

Layoutlarda bulunan **asp action**lar href gibi bir **taghelper**dir. **View Import** tarafından geliyor.

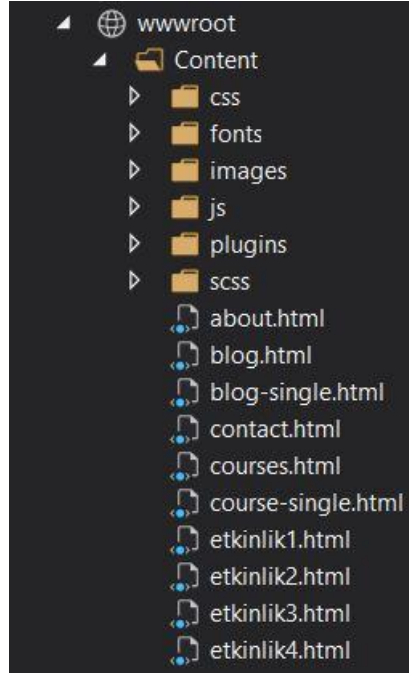


B.KULLANICI TEKNİK DOKÜMAN



```
10 using Microsoft.Extensions.Hosting;
11
12 namespace tekno_egitim
13 {
14     public class Startup
15     {
16         public Startup(IConfiguration configuration)
17         {
18             Configuration = configuration;
19         }
20
21         public IConfiguration Configuration { get; }
22
23         // This method gets called by the runtime. Use this method to add services to the container.
24         public void ConfigureServices(IServiceCollection services)
25         {
26             services.AddMvc(options => options.EnableEndpointRouting = false);
27             services.AddControllersWithViews().AddRazorRuntimeCompilation();
28         }
29
30         // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
31         public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
32         {
33             if (env.IsDevelopment())
34             {
35                 app.UseDeveloperExceptionPage();
36             }
37             app.UseStaticFiles();
38             app.UseMvcWithDefaultRoute();
39         }
40     }
41 }
```

- Startup.cs sınıfımızın içerisine bulunan ConfigureServices alanında projenin yapılandırılması gerekir. Bu yapılandırma sırasında projemize Runtime.Complitation paketi entegre edilmesi gerekir. Bu paket bize projeyi kapatmadan alanların refresh edilerek içeriğin güncellenmesini sağlar. Bu paketin eksikliği ile birlikte sayfayı yenileme işlemi projenin sürekli yeniden başlatılarak kullanılmasıyla giderilir.
- App.UseStaticFiles() metodu ile wwwroot klasörü kullanıma açılmış olur.
- App.UseWithDefaultRoute() metodu bize varsayılan olarak oluşturulan Home Controllerın içindeki Index metodunu çalıştırır. İd si opsiyoneldir.



- Projede wwroot klasörü altına yeni bir Content klasörü oluşturularak sitemizin template'i projeye dahil edilir. wwwroot alanıcss,javascript dosyaları, resimler gibi browserın ulaşması gerek tüm dosyaları içermektedir.

```
Layout.cshtml
61 </div>
62 </div>
63 <!-- navbar -->
64 <div class="navigation w-100">
65   <div class="container">
66     <nav class="navbar navbar-expand-lg navbar-light p-0">
67       <a class="navbar-brand" href="~/Content/index.html"></a>
68       <button class="navbar-toggler rounded-0" type="button" data-toggle="collapse" data-target="#navigation"
69         aria-controls="navigation" aria-expanded="false" aria-label="Toggle navigation">
70         <span class="navbar-toggler-icon"></span>
71       </button>
72     </nav>
73     <div class="collapse navbar-collapse" id="navigation">
74       <ul class="navbar-nav ml-auto text-center">
75         <li class="nav-item active">
76           <a class="nav-link" href="/Home/Index">ANASAYFA</a>
77         </li>
78         <li class="nav-item @blog">
79           <a class="nav-link" href="/Home/Blog">ÜNİVERSİTELER</a>
80         </li>
81         <li class="nav-item @events">
82           <a class="nav-link" href="/Home/Etkinlik">ETKİNLİKLER</a>
83         </li>
84         <li class="nav-item dropdown view">
85           <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-toggle="dropdown"
86             aria-haspopup="true" aria-expanded="false">
87             Enler
88           </a>
89           <div class="dropdown-menu" aria-labelledby="navbarDropdown">
90             <a class="dropdown-item" href="/Home/Blog">En çok İzlenen Videolar</a>
91             <a class="dropdown-item" href="/Home/Blog">En çok Okunan Makaleler</a>
92             <a class="dropdown-item" href="/Home/Blog">En çok Okunan Haberler</a>
93             <a class="dropdown-item" href="/Home/Blog">En çok Görüntülenen Blog Yazıları</a>
94             <a class="dropdown-item" href="/Home/Blog">En çok Görüntülenen Notlar</a>
95           </div>
96         </li>
97         <li class="nav-item @about">
98           <a class="nav-link" href="/Home/Hakkimizda">HAKKIMIZDA</a>
99         </li>
100        <li class="nav-item @contact">
101          <a class="nav-link" href="/Home/Iletisim">İLETİŞİM</a>
102        </li>
103      </ul>
104    </div>
105  </div>
```

- _Layout.cshtml dosyası View klasörünün içerisindeki Shared klasörü içerisinde bulunur. Diğer bir olması gereken _view.cshtml dosyası ile bağlantılıdır. Bu dosyalar header ve footer kısımlarımızı tutmamız için gereklidir. Fakat bu işlem yeterli olmamakla birlikte düzenlemeler yapılması gerekir. Html kodları içerisindeki dosya yolları değiştiğinden dolayı href, src ve background kodlarında dosya yollarının düzeltilmesi gerekir. Bu işlemler yapılırken header ve footer arasında RenderBody() kodunun yazılması gerekir. Bu method türetilen bir .cshtml sayfasının ilk yayınlanacağı alanı ifade eder.

Sırada template'i parçalama ve gerekli yerlerinin eklenmesi işlemleri yapılacaktır. İlk işlem Home Controller üzerinde gerçekleştirilir.

```
4 using System.Threading.Tasks;
5 using Microsoft.AspNetCore.Mvc;
6
7 namespace tekno_egitim.Controllers
8 {
9     public class HomeController : Controller
10    {
11        public IActionResult Index()
12        {
13            return View();
14        }
15        public IActionResult Hakkimizda()
16        {
17            return View();
18        }
19        public IActionResult Kurs()
20        {
21            return View();
22        }
23        public IActionResult Etkinlik()
24        {
25            return View();
26        }
27        public IActionResult Blog()
28        {
29            return View();
30        }
31        public IActionResult Iletisim()
32        {
33            return View();
34        }
35    }
36 }
37
```

- Her bir layout alanının verileri Ana sayfada görüntülenmekteydi. Biz ana sayfada sadece Header ve Footer alanı kalacak şekilde işlemlerimiz oluşturacağız.

- Layout'da bulunan tüm alanlar IActionResult metoduna bağlı olarak yazılır. Bu method Controller yapısına gelen isteklere göre işlem yapıp, kullanıcıya View ilgili isteğine göre bilgileri geri döndüren metottur. Bu metotların üzerine gelip Add View diyerek gerekli istek bilgilerini oluşturacağımız alanlarımızı yaratırız.

```

1 2
3 ViewData["title"] = "Anasayfa";
4
5
6 <!-- hero slider -->
7 <section class="hero-section overlay bg-cover" data-background="~/Content/Images/banner/banner-1.jpg">
8   <div class="container">
9     <div class="hero-slider">
10      <!-- slider item -->
11      <div class="hero-slider-item">
12        <div class="row">
13          <div class="col-md-8">
14            <h1 class="text-white" data-animation-out="fadeOutRight" data-delay-out="5" data-duration-in=".3" data-animation-in="fadeInLeft" data-delay-in=".1">Misyonumuz: PARLAK BİR GELECEK.</h1>
15            <p class="text-muted mb-4" data-animation-out="fadeOutRight" data-delay-out="5" data-duration-in=".3" data-animation-in="fadeInLeft" data-delay-in=".4">
16              Kaliteli eğitimin her yerde ve herkes için var olması için çalışacağız. Uzakta eğitim yöntemlerinin geliştirilmesi için yeni yöntemler arayacağız.
17              Öğrencinin başarılı olabileceği eğitim dünyasında ücretsiz uzaktan eğitim metoduna sürdürebilir kalmayı hedefliyoruz.
18              Öncelikli hedefimiz kişisel gelişim oluşturma niyetiyle toplumsal gelişimi hedefliyoruz.
19              Öğrencilerimizin sayısı arttıkça bireylere değil, küresel topluma hizmet etmiş oluruz.
20            </p>
21            <a href="~/Content/contact.html" class="btn btn-primary" data-animation-out="fadeOutRight" data-delay-out="5" data-duration-in=".3" data-animation-in="fadeInLeft" data-delay-in=".7">BİZE KATIL</a>
22          </div>
23        </div>
24      </div>
25      <!-- slider item -->
26      <div class="hero-slider-item">
27        <div class="row">
28          <div class="col-md-8">
29            <h2 class="text-white" data-animation-out="fadeOutRight" data-delay-out="5" data-duration-in=".3" data-animation-in="fadeInRight" data-delay-in=".1">Geleceğin güvencesi sağlam temellere dayalı bir eğitime,
30            <p class="text-muted mb-4" data-animation-out="fadeOutRight" data-delay-out="5" data-duration-in=".3" data-animation-in="fadeInRight" data-delay-in=".4">
31              MUSTAFA KEMAL ATATÜRK
32            </p>
33          </div>
34        </div>
35      </div>
36    </div>
37  </div>
38 </div>
39
40 </div>

```

- IActionResult kullanıcı ile veri iletişimini sağladığından dolayı kullanıcıya gönderilecek veri sayfasını oluşturacağımız, _Layout alanından diğer sayfalarda görünmesini istemediğimiz kod parçasını buraya entegre ettik. Bu kod yapısı kullanıcı Ana sayfa isimli alana tıkladığında, görüntülenecek olan Slider kısmını ve Banner dediğimiz alanı ekrana yansıtır.

```

69
70 <!-- page title -->
71 <section class="page-title-section overlay" data-background="images/backgrounds/page-title.jpg">
72   <div class="container">
73     <div class="row">
74       <div class="col-md-8">
75         <div align="right"><p><font face="times new roman" size="15" color="white"><b>ÜNİVERSİTELER</b></font></p></div>
76         <ul class="list-inline custom-breadcrumb">
77           <li class="list-inline-item"><a class="h2 text-primary font-secondary" href="@page-link">_____ ÜNİVERSİTESİ</a></li>
78           <li class="list-inline-item text-white h3 font-secondary @nasted"></li>
79         </ul>
80       </div>
81     </div>
82   </div>
83 </section>
84 <!-- /page title -->
85 <!-- blogs -->
86 <section class="section">
87   <div class="container">
88     <div class="row">
89       <!-- blog post -->
90       <div class="col-lg-4 col-sm-6 mb-5">
91         <article class="card rounded-0 border-bottom border-primary border-top-0 border-left-0 border-right-0 hover-shadow">
92           
93           <div class="card-body">
94             <!-- post meta -->
95             <ul class="list-inline mb-3">
96               <!-- post date -->
97               <li class="list-inline-item mr-3 ml-0">27 Nisan 2020</li>
98               <!-- author -->
99             </ul>
100             <a href="~/Content/Universiteler.html">
101               <h4 class="card-title">ÜNİVERSİTEMİZ PERSONELİNE HOBİ BAĞÇELERİ VERİLDİ</h4>
102             </a>
103             <a href="~/Content/Universiteler.html" class="btn btn-primary btn-sm">Daha Fazla</a>
104           </div>
105         </div>
106       </div>
107     </div>
108   </div>
109   <!-- blog post -->
110   <div class="col-lg-4 col-sm-6 mb-5">
111     <article class="card rounded-0 border-bottom border-primary border-top-0 border-left-0 border-right-0 hover-shadow">
112       

```

- Yukarıda verilen kısım diğer bir alanımız olan “Üniversiteler” alanını içermektedir. Kullanıcının bu kısma tıklamasıyla görüntülenecek olan İçerik ekranı, _Layout içerisindeki blogs kod parçacıklarının bu alana entegre edilmesi sonucunda oluşturulur.

```

Etkinlik.cshtml
69 <!-- page title -->
70 <section class="page-title-section overlay" data-background="/Content/images/backgrounds/page-title.jpg">
71 <div class="container">
72 <div class="row">
73 <div class="col-md-8">
74 <div class="col-md-8">
75 <div align="center"><p><font face="times new roman" size="28" color="white"><b>ETKİNLİKLER</b></font> </p> </div>
76 <ul class="list-inline custom-breadcrumb">
77 <li class="list-inline-item"><a class="h2 text-primary font-secondary" href="@page-link">ÜNİVERSİTESİ</a></li>
78 <li class="list-inline-item text-white h3 font-secondary @nasted"></li>
79 </ul>
80 </div>
81 </div>
82 </div>
83 </div>
84 </section>
85 <!-- /page title -->
86 <!-- courses -->
87 <section class="section">
88 <div class="container">
89 <div class="row">
90 <div class="col-lg-4 col-sm-6 mb-5">
91 <div class="card border-0 rounded-0 hover-shadow">
92 <div class="card-img position-relative">
93 
94 <div class="card-date"><span>7</span><br>Mayıs</div>
95 </div>
96 <div class="card-body">
97 <div class="location">
98 <p><i class="ti location-pin text-primary mr-2"></i>ETKİNLİKLER</p>
99 <a href="/Content/etkinlik.html">
100 <h4 class="card-title">10.Yıl Etkinlikleri 17 Mart - 7 Mayıs</h4>
101 </a>
102 </div>
103 </div>
104 </div>
105 </div>
106 <div class="col-lg-4 col-sm-6 mb-5">
107 <div class="card border-0 rounded-0 hover-shadow">
108 <div class="card-img position-relative">
109 
110 <div class="card-date"><span>18</span><br>Mayıs</div>
111 </div>
112 </div>

```

- Bu alana ise kullanıcının Etkinlikler alanına tıklamasıyla beraber görebileceği _layout alanımızdaki courses isimli kod parçacıkların bulunduğu kısmı ekliyoruz.

```

Hakkimizda.cshtml
67 </div>
68 </div>
69 </div>
70 <!-- page title -->
71 <section class="page-title-section overlay" data-background="/Content/images/backgrounds/page-title.jpg">
72 <div class="container">
73 <div class="row">
74 <div class="col-md-8">
75 <ul class="list-inline custom-breadcrumb">
76 <li class="list-inline-item"><a class="h2 text-primary font-secondary" href="@page-link">HAKKIMIZDA</a></li>
77 <li class="list-inline-item text-white h3 font-secondary @nasted"></li>
78 </ul>
79 </div>
80 </div>
81 </div>
82 </section>
83 <!-- /page title -->
84 <!-- about -->
85 <section class="section">
86 <div class="container">
87 <div class="row">
88 <div class="col-12">
89 <p>
90 Öğrencilerin yeni bakış açıları kazanması, mesleki becerilerini geliştirmesi ve belirli bir alanda uzmanlaşmaları için oluşturulmuş uzakt
91 Sadece akademisyenler ve öğretim görevlileri ile sınırlı kalmayıp, siz öğrencilerin de bilgi birikimini aktarabilmelerini amaç edindik. E
92 </p>
93 <p>NEDEN BİZ? </p>
94 <p>Okulunuzda bulunan akademisyenlerin desteğiyle ve ayrıca sahip olduğunuz bilgi birikiminiz ile bilgi kaynağının herkese açık olduğu uzakt
95 
96 </div>
97 </div>
98 </div>
99 </section>
100 <!-- /about -->
101 </div>

```

- Hakkımızda alanına _layout alanımızdaki about kod parçacıkları hakkımızda viewimize eklenir. Kullanıcının site hakkında bilgi alabileceği detaylı yazı ve yazı hakkında resim bu alanda görüntülenecektir.


```
iletisim.cshtml  X
91      <div class="card-body">
92          <a href="/Content/teacher-single.html">
93              <h4 class="card-title">Ecem Yaman</h4>
94          </a>
95          <p>Bilgisayar Mühendisi</p>
96          <p>Yönetici</p>
97          <ul class="list-inline">
98              <li class="list-inline-item"><a class="text-color" href="#"><i class="ti-facebook"></i></a></li>
99              <li class="list-inline-item"><a class="text-color" href="#"><i class="ti-twitter-alt"></i></a></li>
100              <li class="list-inline-item"><a class="text-color" href="#"><i class="ti-google"></i></a></li>
101              <li class="list-inline-item"><a class="text-color" href="#"><i class="ti-linkedin"></i></a></li>
102          </ul>
103      </div>
104  </div>
105  <!-- teacher -->
106  <div data-ref="mixitup-target" class="col-lg-4 col-sm-6 mb-5 bio-science">
107      <div class="card border-0 rounded-0 hover-shadow">
108          
109          <div class="card-body">
110              <a href="/Content/teacher-single.html">
111                  <h4 class="card-title">Rümeysa Gedik</h4>
112              </a>
113              <p>Bilgisayar Mühendisi</p>
114              <p>Yönetici</p>
115              <ul class="list-inline">
116                  <li class="list-inline-item"><a class="text-color" href="#"><i class="ti-facebook"></i></a></li>
117                  <li class="list-inline-item"><a class="text-color" href="#"><i class="ti-twitter-alt"></i></a></li>
118                  <li class="list-inline-item"><a class="text-color" href="#"><i class="ti-google"></i></a></li>
119                  <li class="list-inline-item"><a class="text-color" href="#"><i class="ti-linkedin"></i></a></li>
120              </ul>
121          </div>
122      </div>
123  </div>
124  <!-- teacher -->
125  <div data-ref="mixitup-target" class="col-lg-4 col-sm-6 mb-5 business-study">
126      <div class="card border-0 rounded-0 hover-shadow">
127          
128          <div class="card-body">
129              <a href="/Content/teacher-single.html">
130                  <h4 class="card-title">Beyza Durna</h4>
131              </a>
132              <p>Bilgisayar Mühendisi</p>
133              <p>Yönetici</p>
134          </div>
135      </div>
136  </div>
```

İletisim viewinde ise kullanıcıların yöneticileri görebilecekleri alanın entegrasyonu yapılır.
_Layout arasından content başlıklı kod parçası bu alana