ASP.NET CORE 8.0 MVC DERS NOTLARI-(6.HAFTA)

İçindekiler

LAYOUT VE PARTIAL SAYFALARININ OLUŞTURULMASI (SHARED KLASÖRÜ)

A. _Layout şablon kullanımı

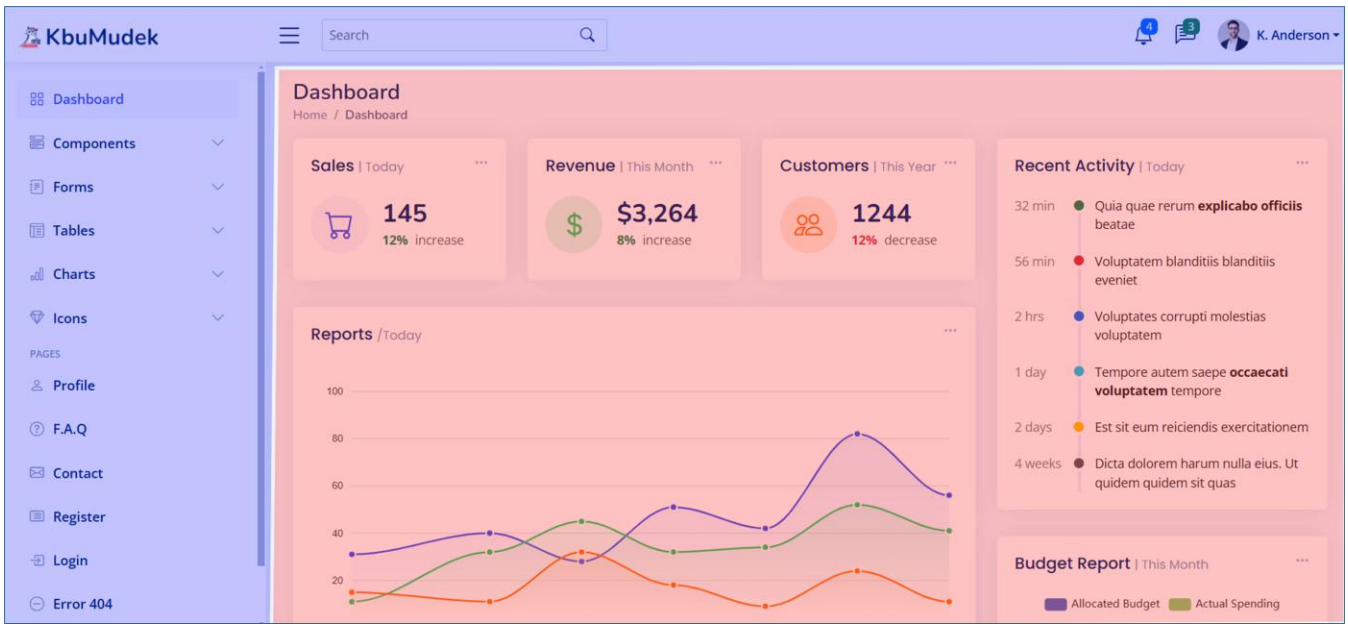
Sayfalarımızın bir çok kısımları ortaktır. Bu ortak kısımlar için tek bir **_layout.cshtml** sayfası oluşturursak bu yapıyı tüm sayfalarda kullanabiliriz. Dolayısı ile diğer sayfalarda sadece içerik kısımlarını değiştirebiliriz. Dış şablon kısmı ise ortak **Shared** klasörü içerisinde **_Layout.cshtml** dosyasında bulunacak.

Bu ders notları ile birlikte Bölümümüzün Müdek Otomasyonunu oluşturmaya çalıştığımızdan örnek format olarak kullandığımız şablonun sabit ve değişen kısımlarını inceleyelim

https://bootstrapmade.com/website-templates/
https://startbootstrap.com/
https://html5up.net/
https://colorlib.com/wp/free-bootstrap-admin-dashboard-templates/
https://www.templatemonster.com/
https://www.wrappixel.com/templates/category/bootstrap-templates/
https://www.canva.com/website-builder/templates/
https://themewagon.com/theme-price/free/
https://w3layouts.com/

Daha çok sayıda siteyi inceleyebilirsiniz.

Şablonda aşağıda gösterilen mavi bölgeler _layout sayfasında oluşsun. Kırmızı bölgeler ise içerik sayfası olan esas View sayfalarında oluşsun.



Buna göre bu yapıyı projemizin içinde oluşturalım.

Öncelikle projemizin şablon (Template) dosyalarını wwwroot altına atalım. Oraya projemize ati Css ve Js dosyaları atılmış olur. Diğer sayfa dosyalarının içeriklerini oradan alıp kendi sayfalarımızın içine atarız. Sadece css ve Js dosyalarının yollarında düzeltme yaparak dosyaları kullanabiliriz.

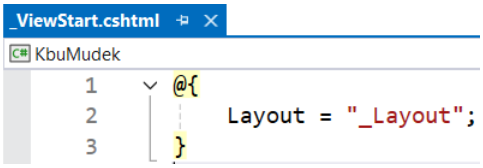
_Layout sayfasında içerik sayfasının görüntüleneceği yere aşağıdaki ifadenin konulması gerekir.

```
@RenderBody()
```

İçerik sayfasınında ilgili Layout sayfasını kullanacağını anlaması içinde sayfanın üst kısmına aşağıdaki ifade yazılmalıdır.

```
@{
    Layout = "_Layout";
}
```

Eğer her View sayfasının başına bu ifadeyi yazmak istiyorsak, yani tüm View sayfaları _Layout sayfası ile birlikte açılmasını istiyorsak bu sefer bu ifadeyi _ViewStart.cshtml dosyasının içine yazabiliriz. Böyle her View sayfası bu _layout sayfası ile birlikte çalışır.



Bu sefer de, bazı sayfaların _Layout sayfasını kullanması gerekmezse ne yapacağız. Sadece o sayfaların başına layout sayfasının iptal etmek için aşağıdaki kodu yapabiliriz yada onun layout sayfası başka bir sayfa ise onun adını yazabiliriz. Şunu unutmayalım herhangi bir View sayfası içerisinde html yapısını barındırmazsa açılmaz. Yani Layout=null; dedikten sonra o sayfanın yapısı tam bir html sayfası formatında olmalıdır. İçerisinde <head> ve <body> etiket yapıları bulunmalıdır.

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

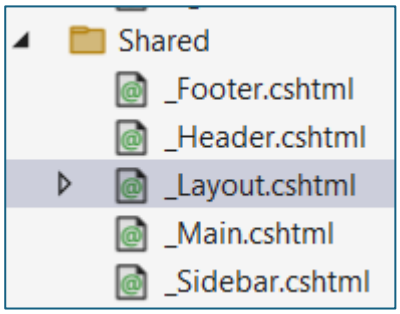
B. _Partial sayfa kullanımı

View yada Layout sayfalarımızın içindeki belli bölgeleri başka sayfalarda kullanma ihtiyacı duyabiliriz. Bu durumda bu sayfa parçalarını Shared klasörü içinde Partial (kısmi, parçalı) olarak tutarsak ihtiyacımız olan her yerde sadece bu sayfa parçasını çağırarak daha az kod yazmış oluruz.

Bunun için ilgili Partial sayfaya adres verirken aşağıdaki ifadeyi kullanırız. Partial sayfalarımız Shared klasörünün içinde olmalıdır. Adet gereği tam bir sayfa formatına olmayan parçalı sayfalar (buna Layout sayfasıda dahil) adının önüne _ alt çizgi konur. Zorunlu değildir ama âdettendir.

```
@await Html.PartialAsync("_Main")
@await Html.PartialAsync("_Footer")
```

Shared klasörümüzün içerisindeki dosyalarımız şu şekilde oldu.

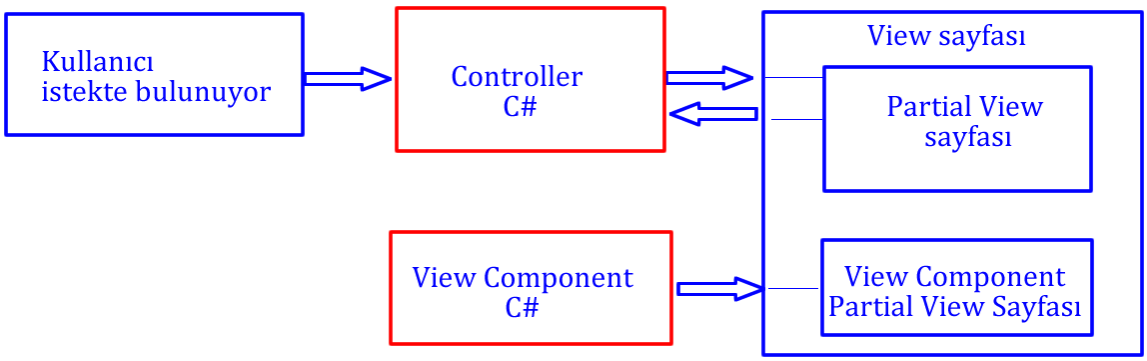


Bunlardan _Layout da aslında bir parçalı sayfadır ama Ana şablon olarak kullanıldığından (sabit kalacak ve değişecek bölgeleri belirlediğinden) _Layout olarak adlandırıldı. Bu sayfayı diğerlerinden ayıran aslında içerisinde kullanılan RenderBody() ifadesidir. Bu ifade içerik sayfalarının nereye konulacağını belirler. Diğer _Footer, _Header, _Main, _Sidebar sayfaları parçalı sayfalardır. Başka bir sayfa tarafından çağrılırlar. Örneğin burada _Header ve _Sidebar parçalı sayfaları _Layout sayfası içinden çağrılmaktadır. Diğer _Main ve _Footer parçalı sayfaları da Index sayfasından çağrılmaktadır.

C. ViewComponent Kullanımı

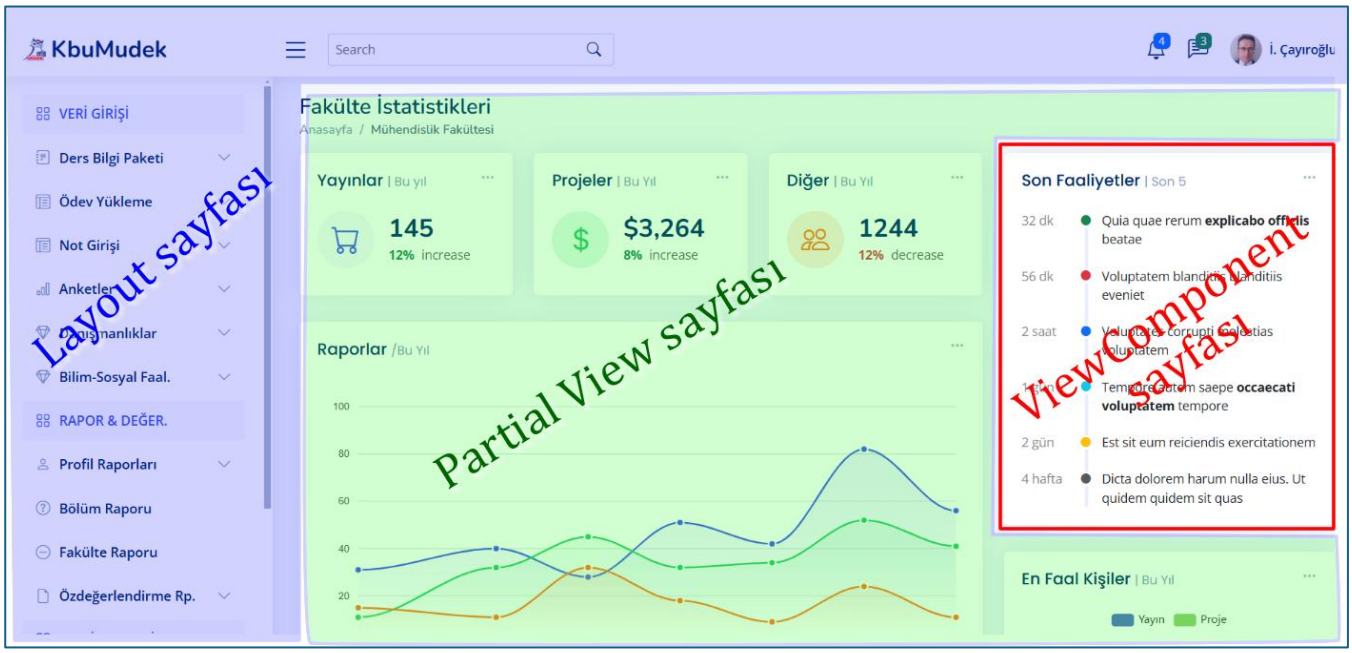
View sayfalarını yada Partial View sayfalarında bilgileri görüntülemek için Controller üzerinden model içerisinde bilgileri taşıyorduk. Tabi bu işlemleri yaparken sürekli olarak View sayfası ile Controller arasında gidip gelmemiz gerekebilir.

Fakat bazen sayfalarda bilgiler görüntülenirken tek yönlü olarak sabit bilgileri görüntülemek gerekebilir. Yani View sayfası her açıldığında sabit bir bilgiyi veritabanından çekip görüntüleyebilir. Böyle bir işlemde kullanıcının dışarıdan herhangi bir müdahalesine ihtiyaç olmamalı. Yani kullanıcı bazı parametreleri belirleyip ona göre bilgileri göstermek isterse yine Controller kullanmak gerekir. Onun yerine her seferinde sabit bilgi görüntülenirse bu kullanılabilir. Bunu temsil eden grafik anlatım çizim şu şekilde ifade edilebilir.



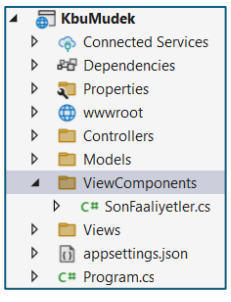
Örneğin View sayfamızda akademisyenin Yayınlarını görüntülemek isteyelim. Kullanıcı hangi yayınları görüntüleyeceğini belirleyip ona göre listeleme yaptıracaksa Controller üzerinden Veritabanından çekilecek bilgileri alıp sayfada gösterebilir. Fakat View sayfasının bir köşesinden kullanıcı bir şey belirlemeden sürekli olarak sabit en son 5 yayını görüntülemek gerekirse bu durumda controller üzerinden değil ViewComponent in C# kodları üzerinden yayınları sayfa her açılışında otomatik olarak çekebilir. Doğal olarak ViewComponent de model yapısını kullanarak bilgileri sayfaya taşıyacaktır.

Devam ettiğimiz projede Layout sayfası içinde Partial view sayfası vardı. Onun içindede ViewComponenti kullanarak en son faaliyetleri veritabanından getirelim fakat henüz kullanmadığımızdan C# içinde model yapısına bilgileri yükleyip ViewComponent içinde gösterelim.



ViewComponents yapısı MVC yapısı içinde yoktur. Asp.net core ile gelen bir kullanımdır.

Aşağıdaki gibi klasörümüzü oluşturalım. Klasörün adı önemli, hatalı yazılmamalıdır. İçerisine boş bir class yapısı oluşturalım.




```
namespace KbuMudek.ViewComponents
{
    public class SonFaaliyetler
    {
    }
}
```

Sayfanın yan tarafında ViewComponent içerisinde son faaliyetleri listelemek için öncelikle bu bilgileri taşıyacak model yapımızı Model klasörü içerisinde oluşturalım. Ardından ViewComponents klasörü içerisinde oluşturduğumuz SonFaaliyetle sınıfı içerisinde bilgileri buradan alıp View sayfasına taşıyacak metodumuzu oluşturalım.

Models>Faaliyet.cs

```
namespace KbuMudek.Models
{
    public class Faaliyet
    {
        public string Tur { get; set; }
        public string Baslik { get; set; }
        public string Yazar { get; set; }
    }
}
```

ViewComponents içerisindeki SonFaaliyetler sınıfımız kullanacağı alt yapıyı: ViewComponent kütüphanesinden almaktadır. Bu kütüphane ise `using Microsoft.AspNetCore.Mvc;` içerisinde bulunmaktadır. Bu kütüphane içerisindeki metodlardan biri olan belli bir kullanım şablonu olan IViewComponentResult metodu burada kullanıldı. Dikkat edilirse bu metod Controller içinde kullandığımız ve View sayfalarına bilgi taşıdığımız ActionResult metodlarına benzemektedir. Onun gibi result View() sayfasına bilgileri götürmektedir. Burada geçen **Invoke()** ifadesi ise kütüphaneden oluşturduğumuz IViewComponentResult metodunu aktif ediyor yani çağırıyor.

ViewComponents>SonFaaliyetler.cs

```
using KbuMudek.Models;
using Microsoft.AspNetCore.Mvc;

namespace KbuMudek.ViewComponents
{
    public class SonFaaliyetler:ViewComponent
    {
        public IViewComponentResult Invoke()
        {
            var FaaliyetListesi = new List<Faaliyet> {

                new Faaliyet { Tur="Yayın", Baslik="Aradsdr sergse stesg", Yazar="Ali
Sucubaı"},
                new Faaliyet { Tur="Konferans", Baslik="Jaşdj  sdrer kjsdfdr şsaser",
Yazar="Oya Karataş"},
                new Faaliyet { Tur="Yayın", Baslik="Arssjxc dsjrer sdferrsg", Yazar="Cem
Aksu" },
                new Faaliyet { Tur="Proje", Baslik="Şarsajg sadrrıs", Yazar="Can Su" },
                new Faaliyet { Tur="Sosyal", Baslik="jaşjser sarajs agarera  sreg",
Yazar="Mustafa Sarıtaş" }

            };
            return View(FaaliyetListesi);
        }
    }
}
```

Şimdi C# kodlarımızın olduğu yerden View sayfasına bilgileri taşıyalım. Yukarıda resmini verdiğimiz tasarımda sayfamızın sağ tarafındaki kısımda bilgileri görüntülemek istiyorduk. O kısım **_Layout.cshtml** sayfamızın içerisindeki **_main.cshtml** adı ile oluşturduğumuz (partial-kısmi sayfa) şeklinde bir bölge idi. Dolayısı ile o yere gidip oradan ViewComponent e ait html sayfamızı çağırmalıyız. Bunun için aşağıdaki kodu View sayfasında çağıracağımız yere yazıyoruz.

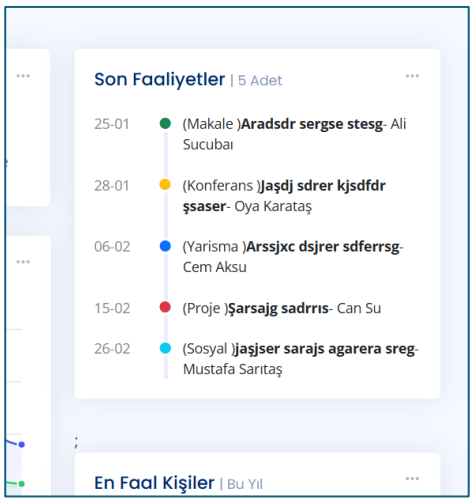
```
@await Component.InvokeAsync("SonFaaliyetler");
```

Burada “SonFaaliyetler” sınıfına ait Componente ait sayfayı açmaya çalışıyor fakat hata veriyor. Çünkü bu component’e ait bir html sayfasını Views klasörü içerisinde bulamıyor. Bu klasörü aşağıda gösterilen iki yoldan biri içerisinde arıyor bulamıyor.

An unhandled exception occurred while processing the request.

InvalidOperationException: The view 'Components/SonFaaliyetler/Default' was not found. The following locations were searched:
/Views/Home/Components/SonFaaliyetler/Default.cshtml
/Views/Shared/Components/SonFaaliyetler/Default.cshtml

Buradan Componentlere ait sayfaların belli bir format bulunan bir yere konulması gerektiğini anlıyoruz. Gösterilen ikinci yolu tercih edelim. SonFaaliyetleri görüntüleyeceğimiz Component sayfasını “Views/Shared/Components/SonFaaliyetler” şeklinde klasörleri oluşturduktan sonra içerisine default.cshtml olarak listeleme kodlarını içerisinde oluşturalım.



Views/Shared/**Components**/**Default.cshtml**

```
@model List<KbuMudek.Models.Faaliyet>

<!-- Recent Activity -->
<div class="card">
        <div class="filter">
                <a class="icon" href="#" data-bs-toggle="dropdown"><i class="bi bi-three-dots"></i></a>
                <ul class="dropdown-menu dropdown-menu-end dropdown-menu-arrow">
                        <li class="dropdown-header text-start">
                                <h6>Filtre</h6>
                        </li>

                        <li><a class="dropdown-item" href="#">Bugün</a></li>
                        <li><a class="dropdown-item" href="#">Bu Ay</a></li>
                        <li><a class="dropdown-item" href="#">Bu Yıl</a></li>
                </ul>
        </div>

<div class="card-body">
<h5 class="card-title">Son Faaliyetler <span>| 5 Adet</span></h5>

<div class="activity">

@foreach(var faaliyet in Model)
{
        <div class="activity-item d-flex">
                <div class="activite-label">@faaliyet.Tarih.ToString("dd-MM")</div>

                @if (faaliyet.Tur=="Makale")
                {
                        <i class='bi bi-circle-fill activity-badge text-success align-self-start'></i>
                }
                else if (faaliyet.Tur == "Proje")
                {
                        <i class='bi bi-circle-fill activity-badge text-danger align-self-start'></i>
                }
```

```
                else if (faaliyet.Tur == "Yarisma")
                {
                        <i class='bi bi-circle-fill activity-badge text-primary align-self-start'></i>
                }
                else if (faaliyet.Tur == "Sosyal")
                {
                        <i class='bi bi-circle-fill activity-badge text-info align-self-start'></i>
                }
                else if (faaliyet.Tur == "Konferans")
                {
                        <i class='bi bi-circle-fill activity-badge text-warning align-self-start'></i>
                }

                <div class="activity-content">
                        (@faaliyet.Tur )<a href="#" class="fw-bold text-dark">@faaliyet.Baslik</a>- @faaliyet.Yazar
                </div>
        </div>
}

</div>

</div>
</div><!-- End Recent Activity -->
```

D. _ViewStart.cshtml ve _ViewImports.cshtml dosyaları

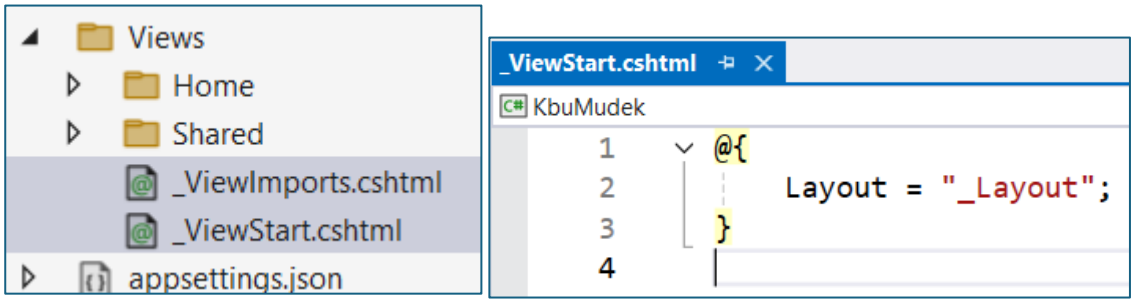
Yukarıda View sayfalarının sabit kısımlarında _Layout sayfasını kullanmıştık. Hangi View sayfası _Layout.cshtml sayfası ile birlikte çalışıyorsa onun en üstünde kullanacağı layout sayfasını aşağıdaki şekilde tanımlıyorduk.

```
@{
    Layout = "_Layout";
}
```

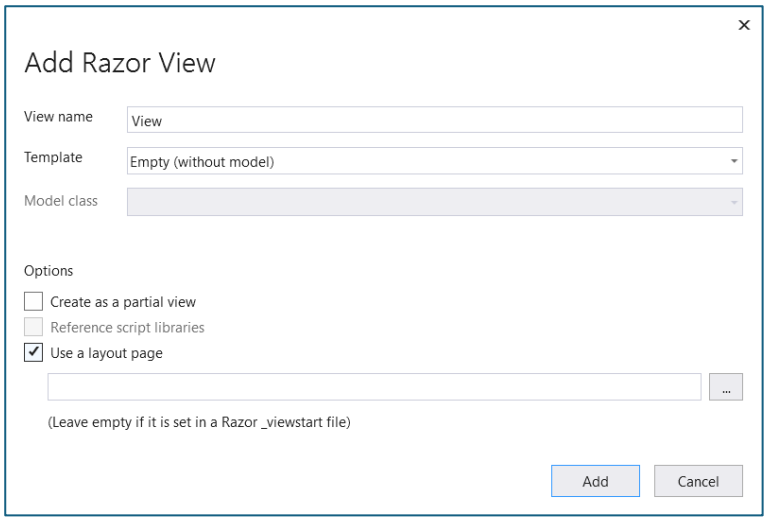
Bütün View sayfalarında bu Layout sayfasını kullanmak istediğimizde her seferinde o sayfaların üstünde bu tanımlamayı yapmak durumunda kalıyorduk. Onun yerine tüm View sayfalarında kullanılmak üzere tek bir yerden tanımlama yapmak istersek bu kodları Views hemen altında olan **_ViewStart.cshtml** dosyasını kullanabiliriz. Bu dosyanın içerisine yukarıdaki ifadeyi yazdığımızda artık her sayfaya yazmak durumunda kalmayız.

Fakat bazen bazı sayfalarda bu layout'u kullanmak istemeyebiliriz. Örneğin siteye ilk girişti şifre girişi sayfası kullandığımızda burada Layout tasarımı olmayabilir. Böyle durumlarda, Layout'un kullanılmayacağı durumlarda o sayfaların üstüne aşağıdaki kodları yazmalıyız.

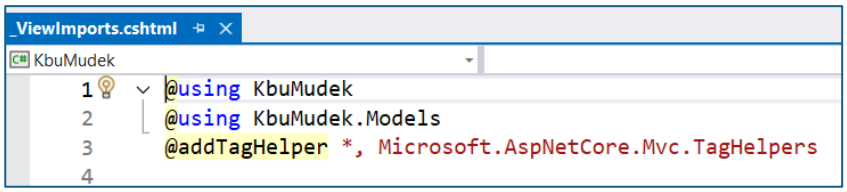
```
@{
    Layout = null;
}
```



Eğer Views içerisine herhangi bir yeni View sayfası ekliyorsak aşağıda olduğu gibi "Use Layout page" seçili olursa alttaki kısımdan herhangi bir başka layout seçmezsek o sayfa otomatik olarak varsayılan Layout sayfasına bağlanmış olur. Başka bir layout sayfası kullanacaksak (Bir sitede bir çok layout sayfası olabilir) alttaki kısımdan seçebiliriz. Eğer sayfanın kendisi zaten Layout olacaksa o zaman buradaki işareti kaldırmalıyız.



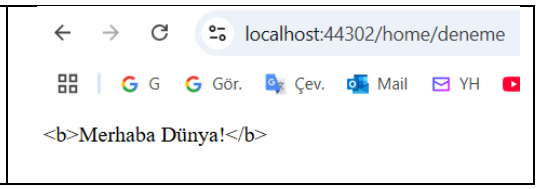
Çoğu sayfada ortak kütüphaneleri çağırıyor olabiliriz. Bu kütüphaneleri her sayfanın başında çağırmak yerine bunların hepsini tek bir yerde toplayıp tüm sayfalarda bunların kullanılmasını istiyorsak bu seferde **_ViewImport.cshtml** dosyasını bu amaç için kullanabiliriz.



E. C# kodları içinde html etiketlerinin kullanımı (@Html.Raw() Fonksiyonun kullanımı)

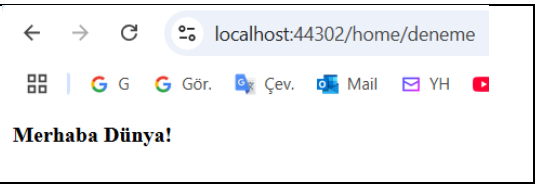
Bazen C# kodları içerisinde html etiketlerini kullanmamız gerekebilir. Örneğin aşağıdaki sayfada bir mesajı bold olarak yazdırmak istediğimizde tırnaklar içerisindeki ifadeyi düz bir metin olarak gördüğü için direk ne varsa hepsini ekrana yazdırmıştır. (Dikkat: Layout=null ifadesi, bu sayfa için varsayılan Layout sayfasının kullanılmayacağını bildiriyor).

```
@{
    Layout = null;
    var Mesaj = "<b>Merhaba Dünya!</b>";

    @Mesaj;
}
```



Aşağıdaki kodlarda ise Mesaj değişkenin içindeki ifadeleri ekrana yansıtırken html formatı ile okumasını sağladığımızdan etiketlerin içindeki ifadeleri dikkate alarak göstermiş oluyor.

```
@{
    Layout = null;
    var Mesaj = "<b>Merhaba Dünya!</b>";

    @Html.Raw(Mesaj);
}
```



ŞİFRE GİRİŞ SAYFASI (login Sayfası)

```
using Microsoft.AspNetCore.Mvc;

namespace Proje_A.Controllers
{
    public class LoginController : Controller
    {
        public IActionResult Index()
        {
            return View();
        }
    }
}
```

```
<!DOCTYPE html>
<html lang="tr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Giriş Yap</title>

  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <!-- Bootstrap Icons -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-
icons.css" rel="stylesheet">



<style>
    body {
        /* Koyu Gri/Mavi Arkaplan (Ana Sayfa ile Uyumlu) */
        background-color: #25303F;  /* #12121e;  */
        height: 100vh;
        display: flex;
        align-items: center;
        justify-content: center;
        font-family: "Segoe UI", sans-serif;
    }
    .login-card {
        /* Koyu Kart Arkaplanı */
        background-color: #1f2029;
        border-radius: 15px;
        /* Daha belirgin, koyu bir gölge */
        box-shadow: 0 8px 30px rgba(0,0,0,0.5);
        padding: 40px;
        width: 100%;
        max-width: 400px;
        /* Kartın kenarlarına hafif bir vurgu */
        border: 1px solid #333642;
    }
    .login-card h3 {
        text-align: center;
        margin-bottom: 30px;
        /* Açık metin rengi */
        color: #e0e0e0;
    }
    .form-label {
        /* Açık metin rengi */
        color: #bdbdbd;
    }
    .form-control {
        background-color: #2a2b38; /* Input arkaplanı */
```

```
        border: 1px solid #3c3e4a;
        color: #e0e0e0; /* Input metin rengi */
    }
    .form-control::placeholder {
        color: #757575; /* Placeholder rengi */
        opacity: 1;
    }
    .form-control:focus {
        box-shadow: none;
        /* Parlak Vurgu Rengi (Home sayfasındaki ilerleme çubukları ile uyumlu) */
        border-color: #00ffc3;
        background-color: #2a2b38;
        color: #e0e0e0;
    }
    .form-check-label {
        color: #bdbdbd;
    }

    /* Ana Vurgu Buton Rengi */
    .btn-primary {
        background-color: #00ffc3;
        border-color: #00ffc3;
        color: #12121e; /* Buton metni için koyu renk */
        font-weight: bold;
        transition: background-color 0.3s ease;
    }
    .btn-primary:hover, .btn-primary:focus {
        background-color: #00e6b8; /* Daha koyu/yoğun hover rengi */
        border-color: #00e6b8;
        color: #12121e;
        box-shadow: 0 0 10px rgba(0, 255, 195, 0.5); /* Hafif parlaklık efekti */
    }

    .text-muted a {
        /* Parlak Vurgu Rengi */
        color: #00ffc3;
        text-decoration: none;
        transition: color 0.3s ease;
    }
    .text-muted {
        color: #bdbdbd !important;
    }
    .text-muted a:hover {
        color: #00e6b8;
        text-decoration: underline;
    }
</style>


</head>

<body>
  <div class="login-card">
    <h3><i class="bi bi-person-circle"></i> Giriş Yap</h3>
    <form method="post" action="/Account/Login">
      <div class="mb-3">
        <label for="email" class="form-label">E-posta</label>
        <input type="email" class="form-control" id="email" name="Email" placeholder="E-
posta adresinizi girin" required>
      </div>

      <div class="mb-3">
        <label for="password" class="form-label">Şifre</label>
        <input type="password" class="form-control" id="password" name="Password"
placeholder="Şifrenizi girin" required>
      </div>
```

```
      <div class="d-flex justify-content-between align-items-center mb-3">
        <div class="form-check">
          <input class="form-check-input" type="checkbox" id="rememberMe"
name="RememberMe">
          <label class="form-check-label" for="rememberMe">Beni hatırla</label>
        </div>
        <a href="/Account/ForgotPassword" class="text-muted">Şifremi unuttum?</a>
      </div>

      <div class="d-grid">
        <button type="submit" class="btn btn-primary">
          <i class="bi bi-box-arrow-in-right"></i> Giriş Yap
        </button>
      </div>
    </form>

    <div class="text-center mt-4 text-muted">
      Hesabınız yok mu? <a href="/Account/Register">Kayıt Ol</a>
    </div>
  </div>

  <!-- Bootstrap JS -->
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></scrip
t>
</body>
</html>
```