



İNTERNET PROGRAMCILIĞI

www.hacibayram.edu.tr/uzem

Bu bölümde MVC Yapısı Proje Üzerinde Oluşturulmuş, Default Routing ve ViewBag Kullanımı uygulaması yapılmıştır.

1. MVC YAPISININ OLUŞTURULMASI

ADIM 1: Bağımsız bileşenler .net 5.0 seçilir.



ADIM 2: Proje oluşturulur.

Kullanmaya başlayın Depoyu klonla GitHub veya Azure DevOps gibi bir çevrimiçi Depoyu klonla GitHub veya Azure DevOps gibi bir çevrimiçi Depoyu klonla GitHub veya Azure DevOps gibi bir çevrimiçi Perei bir projeyi veya çözümü aç Yerel bir Visual Studio projesi veya .sln dosyası Actional Companyation Perei bir klasör aç Herhangi bir klasör içinde koda gidip düzenleyin

Yeni bir proje oluştur

Başlamak için kod iskelesi içeren bir proje şablonu seçin

Kodsuz olarak devam et \rightarrow



ADIM 3: ASP.Net Core Boş Proje Şablonu seçilir.



ADIM 4: Proje isimlendirilir.

Yeni projenizi yapılandırın		
ASP.NET Core Boş C# Linux macOS Windows Bulut Hizmet Web		
Proje adı		
MovieApp.Web		
<u> </u>		
C:\Users\gurso\source\repos		
Çözüm a <u>d</u> ı 🛈		
MovieApp		
Cozumu ve projeyi aynı <u>d</u> izine yerleştirin		
Proje "C:\Users\gurso\source\repos\MovieApp\MovieApp.Web\" içinde oluşturulacak		
	<u>G</u> eri	Sonrak <u>i</u>



ADIM 5: .NET 5.0 seçili. Tickler kapalı.

Ek bilgiler ASP.NET Core Boş C# Linux macOS Windows Bulut Hizmet Web	-		×
Verçeve U .NET 5.0 (Destek kapsamı dışında) HTTPS'yi Yapılandır () Docker'ı Etkinleştir ()			
Linux			
Ge	ri (Dluştur	

ADIM 6: Nodejs kurulumu

nodejs.org/en	
•	LEARN ABOUT DOWNLOAD
	Node.js® is an open-source, cross-platf
	Download
	20.11.1 LTS Recommended For Most Users

ADIM 7: Startup.cs dosyası içerisine View u görüntülemeyi sağlayacak komut eklenir.



∨name ∣r	espace MovieApp.Web
t 	public class Startup
↓ ↓ ↓ ↓	<pre>// This method gets called by the runtime. Use this method to add services to the // For more information on how to configure your application, visit <u>https://go.mic</u> public void ConfigureServices(IServiceCollection services)</pre>
-	<pre>services.AddControllersWithViews(); }</pre>
\ \ 	<pre>// This method gets called by the runtime. Use this method to configure the HTTP r public void Configure(IApplicationBuilder app, IWebHostEnvironment env) {</pre>
Ĭ	<pre>if (env.IsDevelopment()) {</pre>
-	<pre>app.UseDeveloperExceptionPage(); }</pre>

ADIM 8: Startup.cs de default olarak bize gelen yazı "Hello World!"



ADIM 9: Çalıştırılır.

~	3	localhos	t:2105	4 × +
\leftarrow	\rightarrow	G	0	localhost:21054
Hello V	Vorld!			

ADIM 10: Startup.cs de / ile başlayan yukarıdaki ana dizin altına /movies adında bir alt dizin ekleseydik.





ADIM 11: Çalıştırıp url yi düzenlediğimizde karşımıza alt dizin gelir.

•	S la	ocalhost	t:21054/movies × +
4	\rightarrow	G	localhost:21054/movies
Movie L	ist		

ADIM 12: Movie App te sağ tıklanarak üç yeni dosya eklenir.





ADIM 13: Dosya isimleri Models-Views-Controllers olarak düzenlenir.



ADIM 14: Daha önce yapmış olduğumuz yönlendirmeyi controller aracılığı ile yapmak için controllers dosyası içine controller "denetleyici" ekliyoruz.





ADIM 15: MVC Denetleyicisi-Boş olarak seçilir.



ADIM 16: MVC Denetleyicisi-Boş olarak seçilir ve isimlendirilir.



ovieApp.Web					
	Sıralama	ölçütü: Varsayılan		[Ara (Ctrl+E)
	₽	Sinif	C#		Tür: C#
lore	•-0	Arabirim	C#	ł	(ASP.NET MVC Çerçevesi, De adlarının "Controller" sonekii
	ര്	Razor Bileseni	C#		olmasını gerektirir)
	۲ <u>с</u> *	MVC Denetleyicisi - Boş	C#		
	4; Ĩ	Okuma/yazma eylemleri ile M	VC De C#		
	⊢ 4;	API Denetleyicisi – Boş	C#		
	4 3	Okuma/yazma eylemlerine sa	hip APIC#		
	C	Razor Sayfası - Boş	C#	-	
MoviesControll	er.cs				
					Ekle

ADIM 17: Startup.cs üzerinde daha önce tanımlamış olduğumuz kodlar silinir. Sonrasında oluşturmuş olduğumuz controller aracılığıyla MapControllerRoute metodu kullanılır.

```
app.UseRouting();
app.UseEndpoints(endpoints =>
{
    endpoints.MapControllerRoute(
        name: "movieList",
        pattern: "movies",
        defaults: new { controller = "Movies", action = "List" }
        );
}
```

ADIM 18: MoviesController.cs dosyasındaki Action kısmı string değişkene göre düzenlenir.



ADIM 19: Çalıştırılır ve uzantı /movies olarak düzenlenir.





ADIM 20: Örneğin film detayını geriye döndürecek olan Details adında ikinci bir metod tanımlansın.



ADIM 21: ilgili erişim metoduna göre Startup.cs dosyasındaki kodlar düzenlenir.





```
{
    endpoints.MapControllerRoute(
        name: "movieList",
        pattern: "movies/list",
        //MoviesController dosyasındaki action list metodu
        defaults: new { controller = "Movies", action = "List" }
        );
    endpoints.MapControllerRoute(
        name: "movieList",
        pattern: "movies/details",
        //MoviesController dosyasındaki action Details metodu
        defaults: new { controller = "Movies", action = "Details" }
        );
});
```

ADIM 22: URL düzenlenerek ilgili metod çağırılır.

•	3	ocalhos	t:21054/movies/list × +
÷	\rightarrow	G	 localhost:21054/movies/list
Film	List	tesi	

ADIM 23 : URL değiştirilir.

•	0	ocalhos	t:21054/movies/details × +
÷	\rightarrow	G	localhost:21054/movies/details
Film	Deta	ауі	





ADIM 24: Uygulamaya yeni bir Controller eklenir. Controller sağ tıkla.

ADIM 25: Boş Controller seçiyoruz.







ADIM 26: HomeController olarak isimlendiriyoruz.

ADIM 27: HomeController.cs dosyası gerekli kodları yazılır. Anasayfa ve Hakkımızda sayfaları oluşturulur.





ADIM 28: Startup.cs dosyasında tüm ilgili URL uzantıları düzenlenir.





ADIM 29: Çalıştırılır.



ADIM 30: URL düzenlenir





SORU: Tek bir Routing ile list- details- Index-About sayfalarına erişim sağlansın ve açılış sayfası HomeController altındaki Index olsun.

2. DEFAULT ROUTING TANIMLAMAK



ADIM 1: Dört farklı routing bilgisi yazmak yerine tek bir routing ile tanımlanabilir.

Startup.o	cs ⊣¤	×	MoviesController.cs	MovieApp.Web
			🝷 🛠 MovieApp.Web.St	artup
app	.UseR	out	ing();	
app {	.UseE	ndp	oints(endpoints =>	
	endp	oin na pa	ts.MapControllerRoute(me: "default", ttern: "{controller}/{a	ction}"
);		
}); }				

ADIM 2: Çalıştırıp tek routing ile erişebildiğimizi görelim. HomeController için.



```
hakkımızda
```

ADIM 3: Çalıştırıp tek routing ile erişebildiğimizi görelim. MoviesController için.



Film Detayı

ADIM 4: Herhangi bir Controller ismi belirtmeden Varsayılan Controller tanımlayarak anasayfaya erişim sağlanır. Açılış sayfası HomeController altındaki Index olacak şekilde kodlar düzenlenir.



HACI BAYRAM VELİ ÜNİVERSİTESİ UZAKTAN EĞİTİM UYGULAMA VE ARAŞTIRMA MERKEZİ



ADIM 5: Çalıştırarak kontrol edilir. Bu sayede sayfa çalıştırıldığında doğrudan Index action metodu gelir.



anasayfa

ADIM 6: Film Listesi İçerisinde bir elemana erişmek isteseydik, Kodlardaki Route bilgimiz 3 bölmeli olacak şekilde düzenlenirdi.



ADIM 7: Çalıştırılır. Route bilgisi üç bölmeli olarak tanımlandığından varsayılan açılış sayfasına erişilemez.





www.hacibayram.edu.tr/uzem

ADIM 8: URL üç bölmeli şekilde düzenlenerek tekrar çalıştırılır. HomeController için;



```
hakkımızda
```

ADIM 9: URL üç bölmeli şekilde düzenlenerek tekrar çalıştırılır. MoviesController için;



ADIM 10: Üç bölmeli Rout bilgisi olmadan da sayfalara iki bölmeli URL ile erişebilmek için kodlar düzenlenir.



ADIM 11: URL iki bölme olarak düzenlenerek çalıştırılır.





Film Detayı

SORU: Tek parametreli (bölümlü) URL ile movies e erişim sağlamak için gerekli kodları yazınız.

ADIM 12: Sadece Controller bilgisi ile URL de tek bölümlü erişim sağlamak için Movies Controller altına bir Index metodu tanımlanır. Home Controller altına Index metodu daha önce tanımlanmıştı.



ANKARA HACI BAYRAM VELI ÜNIVERSITESI UZAKTAN EĞITIM UYGULAMA VE ARAŞTIRMA MERKEZİ

vnamespace MovieApp.Web.Controllers { public class MoviesController : Controller { public string Index() { return "Film Index"; } //localhost:***/movies/list public string List() { return "Film Listesi"; } //localhost:****/movies/details public string Details() { return "Film Detay1"; } } }

ADIM 13: Çalıştırıp tek parametreli URL ile Movies e erişim kontrol edilir.



Film Index

3. CONTROLLER DA STRING DEĞER YERİNE HTML SAYFASI DÖNDÜRMEK



ADIM 1: Bir HTML sayfası döndürmek için View komutu kullanılır ve metod IActionResult olarak düzenlenir.



ADIM 2: Views /Home/Index dosyasını bulamadı. Bu uyarıyı almamak için Views altına Home Controller eklenmeli.

•	S Internal S	Server Error	×	+
\leftarrow	\rightarrow C	localhost:2	1054	

An unhandled exception

InvalidOperationException: The view ' /Views/Home/Index.cshtml /Views/Shared/Index.cshtml

Microsoft.AspNetCore.Mvc.ViewEngines.ViewEng

ADIM 3: Views sağ tıkla yeni dosya ekle Home olarak isimlendir.





ADIM 4: Home sağ tıkla Görünüm / View ekle

J	Görünüm			Oturum aç 🏼 🗙 🛛 —	o ×
@	Razor Bileşeni		-⊼ ^	7	<u>ƙ</u>
ם לם	Yeni Öğe Var Olan Öğe Yeni İskeleli Öğe	Ctrl+Shift+A Shift+Alt+A	özüm JII 7	Gezgini ⊙ ▼ ≒ 🖨 🛱 📴 - 🎤 🛋	→ ∓ ×
造 衛	Yeni Klasör Kapsayıcı Düzenleyicisi Desteği		özüm ♪ ♪	Gezgini İçinde Ara (Ctrl+ş) Connected Services Properties	<u>- م</u>
Ē *?	Docker Desteği Application Insights Telemetrisi		1	Cantrollers C# HomeController.cs	- 1
** #	Makine Öğrenmesi Modeli İstemci Tarafı Kitaplığı		4	Models	
+	Yeni Azure Web İşi Projesi Azure Web İşi Olarak Mevcut Proje		Ð	Tarayıcıda Göster (Google Chrome) Birlikte Gözat	Ctrl+Shift+W
1	Sinif New EditorConfig		_	Ekle	•
_		Č) *3	Yeni Çözüm Gezgini Görünümü Projeden Çıkart	
		E		Kes Kopyala Sil	Ctrl+X Ctrl+C Del
	Sat: 11 Krkt: 30 I	3SL CRIF		Yeniden Adlandır Tam Yolu Kopyala Klasörü Dosya Gezgini'nde Aç Terminalde Ac	F2
	244 11 144 55 1	- E	ŗ	Özellikler	Alt+Enter

ADIM 5: Boş olan seçilir.



B	Razor Görünümü - Boş
@	Razor Görünümü

ADIM 6: İsimlendirilir.

	Sıralama	ölçütü: Varsayılan 👻 🏥 📃		Ŀ
	₽	Sinif	C#	
	••0	Arabirim	C#	
	0	Razor Bileşeni	C#	
	۲ ^{C#}	MVC Denetleyicisi - Boş	C#	
	۲ ^{C#}	Okuma/yazma eylemleri ile MVC Denetleyicisi	C#	
	₽	API Denetleyicisi – Boş	C#	
	₽ ₽	Okuma/yazma eylemlerine sahip API Denetleyicisi	C#	
	@	Razor Sayfası - Boş	C#	
		Razor Görünümü - Boş	C#	
	0	Razor Düzeni	C#	
	@	Başlat Razor Görünümü	C#	
	i î	Bütünleştirilmiş Kod Bilgi Dosyası	C#	
		Kod Dosyası	C#	
		Marchine Learning Marchall (MILNETS	Ст.	-
Index.cshtml				



ADIM 7: İndex.cshtml içeriğini silinir html tab ile kodlar getirilir.



ADIM 8: Home/Index yazarak deniyoruz.



ADIM 9: HomeController içinde ActionResult metodu Index olarak düzenlenmişti. Çalıştırarak deniyoruz.



Home/Index





ADIM 10: Aynı işlem About için de yapılır. HomeController da kod bloğu düzenlenir.

ADIM 11: Dosya eklenir ve isimlendirilir.





Index.cshtml	₽ × About.cshtml*	+ ×	HomeController.cs*	Star
🖙 MovieApp.	Web		•	
1 2 3 4 5 6	<html xmlns="http://
<head><title></
<body>
Home/About
</body>
</html></th><td>//www.v</td><th><u>w3.org/1999/xhtml</u>"> ></html>			

ADIM 12: Açılan dosyanın hangisi olduğunu görmek için About içindeki yazı değiştirilir.

ADIM 13: Çalıştırılır. URL düzenlenerek açılış sayfası değiştirilir ve erişim kontrol edilir.



ADIM 14: MoviesControll içinde de aynı işlemler sürdürülür.



ARAŞTIRMA MERKEZİ

₽ Çözüm Gezgini д. X Ì, 0 • ≒ 🗐 🗗 ۵ Çözüm Gezgini İçinde Ara (Ctrl+ş) -Controllers 4 C# HomeController.cs ⊳ C# MoviesController.cs Models Views 🔺 🛅 Home About.cshtml Index.cshtml Movies appsettings.json Þ Þ C# Program.cs r C# Cözüm Gezaini Git Değisiklikleri Sınıf Görünümü

ADIM 15: Views altında yeni klasör oluşturulur Movies olarak isimlendirilir.

ADIM 16: Movies Klasörüne Sağ tık ile Görünüm eklenir.





ADIM 17: Boş Görünüm Şablonu seçilir.



ADIM 18: Index olarak isimlendirilir.

	Sıralam	a ölçütü: Varsayılan 👻 📰 📃		
	 €;_]	Sınıf	C#	
	••0	Arabirim	C#	
	@	Razor Bileşeni	C#	
	₽ ₽	MVC Denetleyicisi - Boş	C#	
	┍ ╬	Okuma/yazma eylemleri ile MVC Denetleyicisi	C#	
	, rg tagetter tagett	API Denetleyicisi – Boş	C#	
	, ₽	Okuma/yazma eylemlerine sahip API Denetleyicisi	C#	
		Razor Sayfası - Boş	C#	
	6	Razor Görünümü - Boş	C#	
	@ *	Razor Düzeni	C#	
	@ *	Başlat Razor Görünümü	C#	
	i l	Bütünleştirilmiş Kod Bilgi Dosyası	C#	
		Kod Dosyası	C#	
		KALING TILLING A KALINI (KALINET)	C#	Ŧ
Index.cshtml				

ADIM 19: Index.cshtml içindeki kodlar silinir. Html kodları tab ile getirilir. İçine açılan dosya açılışını deneyebilmek için movies/index yazılır.



Index.cshtm	ıl* ⊰	■ ×	Index.cshtml	Abou	ut.cshtml	HomeCor
🖽 MovieAp	p.Wel	b			-	
1	\mathbf{x}	<h< td=""><td>tml xmlns="<u>htt</u></td><td>p://www.w</td><td>v3.org/1999/</td><td>/xhtml"></td></h<>	tml xmlns=" <u>htt</u>	p://www.w	v3.org/1999/	/xhtml">
2		<h< td=""><td>ead><title><td>itle></td></title></td></h<> <td>ead></td> <td></td>	ead> <title><td>itle></td></title>	itle>	ead>	
3	~	<b< b="">(</b<>	ody>			
4			<pre>movies/index</pre>			
5		<td>body></td> <td></td> <td></td> <td></td>	body>			
6	</td <td>html</td> <td>></td> <td></td> <td></td> <td></td>	html	>			
	_					

ADIM 20: Movies.cshtml isminde bir dosyayı görüntüleyebilmek için MoviesController içindeki kodlar güncellenir.



ADIM 21: Dosyayı görüntüleyebilmek için Movies Klasörü altına yeni bir görünüm eklenir. Movies olarak isimlendirilir.





ADIM 22: Kodlar temizlenir html içeriği tab tuşu yardımıyla eklenir. Html içeriğine movies/movies.cshtml yazarak çalıştırılır. (Movies Controller altındaki movies.cshtml dosyasına erişim)

Movies.cshtml*	-⊧ X	Index.cshtml*	MoviesController.cs*
🖼 MovieApp.We	eb		▼
1 v< 2	html x <mark><h< mark="">e</h<></mark>	mlns=" <u>http://www</u> ad> <title><td>.w3.org/1999/xhtml"> e></td></title>	.w3.org/1999/xhtml"> e>
3 4	<bo< th=""><td>dy> movies/movies.c</td><td>shtml</td></bo<>	dy> movies/movies.c	shtml
5 6 <	/html>	ody>	

ADIM 23: Çalştırılır URL düzenlenir. Movies a erişim kontrol edilir.



 ✓ Solocalhost:21054/movies × + 	
\leftrightarrow \rightarrow C \bigcirc localhost:21054/movies	
movies/index	
 ✓ Solocalhost:21054/movies/list × + ← → C O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O	
movies/movies.cshtml	

ÇALIŞMA SORUSU

Tek bir Routing ile list- details- Index-About sayfalarına erişim sağlanan ve açılış sayfası HomeController altındaki Index olan bir proje mimarisi oluşturunuz.

Tek parametreli (bölümlü) URL ile movies e erişim sağlamak için gerekli kodları yazınız.

