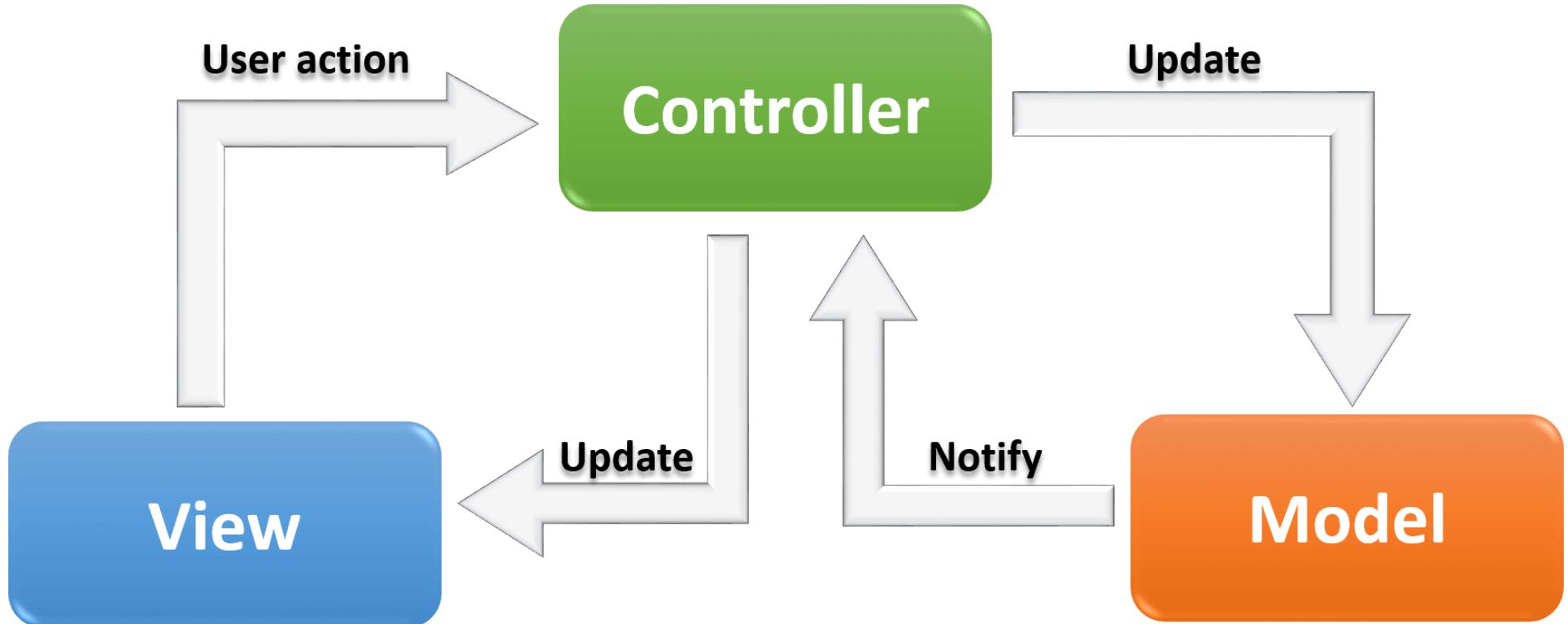
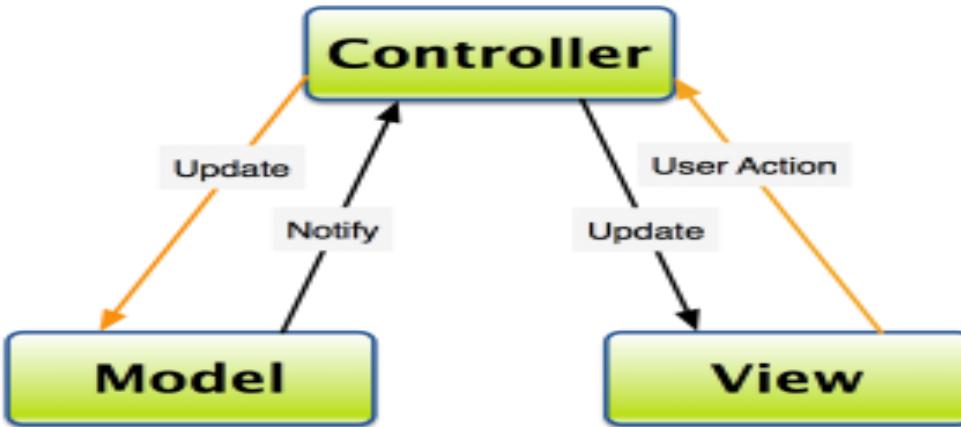


ASP.NET(C#) MVC 5





Models: Kullanacağınız class'lar bulunur. Bu katmanda database'e erişim yollarınız ve metodlarınız yer almaktadır. Bu katman data erişim katmanı(access layer) olarak Entity Framework, ADO.NET ile database üzerinde veri işlemleri yapılması için kullanılır.

View: Basitçe, uygulamanızın kullanıcılarınızın gözüyle gördüğü kısmıdır, arayüzdür. Uygulanmanın arayüz tasarımı burada tasarlanır, projenin html dosyaları burada bulunur.

Controller: View ve Model arasında ki iletişimini sağlar. Kullanıcıların View üzerinden gerçekleştirdiği gelen istekleri (request) Model'e taşır, Model'den aldığı veriyi View üzerinden kullanıcıya gösterir.

MVC'nin Çalışma Mantığı

- Şimdi bu durumu örneklendirecek olursak eğer siz Instagram da bir kullanıcının profilini görmek için istek(request) göndermeniz durumunda, router dosyasında bu istek kontrol edilir.
- Kontrol edildikten sonra controller dosyasında profile ait fonksiyon çalışır. Fonksiyonda gerekli veritabanı işleri yapılır .
- Ardından sizi o fonksiyona ait view'i yani kullanıcının profilini cevap(response) olarak geri döndürür.

View Engine Nedir? Razor Nedir?

View Engine

View'lerinizi HTML çıktısı olarak render etmek için kullanılan bir mekanizma/teknoloji'dir.

Razor View Engine

Razor söz dizimi, programcının bir HTML yapı akışını kullanabilmesine olanak tanıyan ve C# tabanlı bir şablon etiketleme söz dizimidir.

MVC3 ile birlikte Razor çıkana kadar MVC 1 ve 2'de defaulf olarak kullanılan Web Forms View Engine'di. Çalışmasında herhangi bir sorun yoktu fakat yazımı, kullanımı ve bakımı uğraştırıcı ve zahmetliydi. Razor'la artık çok güçlü bir View Engine kavuştuk.

Razor View Engine'in yazım kurallarının temelinde @ simgesi bulunmaktadır.
Asp.Net Web Form ile kodlama yapılırken HTML kodlarının arasında sunucu taraflı kodlamada <% ... %> yazımı kullanılmaktadır.

Razor için bu yapının yerine @ simgesi kullanılmaktadır. Yazım olarak Web Form'a göre oldukça kolaydır ve Asp.Net MVC'de kodlama daha hızlı olmaktadır.

Web Forms kullanımı ve Razor kullanımına örnek:

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the code for `default.aspx`, which contains ASP.NET markup and C# code-behind logic for generating a navigation menu. The code includes session handling for user authentication and links to various pages like `urunler.aspx` and `uyekayit.aspx`. The Solution Explorer on the right lists all files in the project, including other ASPX pages and supporting files like `Web.config` and `Web.Release.config`.

```
52 </div>
53 <ul class="nav navbar-nav">
54     <li class="active"><a href="#">Ana Sayfa</a></li>
55     <li class="dropdown"><a class="dropdown-toggle" data-toggle="dropdown" href="#">Kategoriler <span clas
56         <ul class="dropdown-menu">
57             <%ArrayList kategorilistesi = new Kategoridao().tumkategoriler();%
58             foreach (Kategori gelenkategori in kategorilistesi)
59                 { //for each açtım
60
61                     %>
62                     <li><a href="urunler.aspx?kateno=<%=gelenkategori.Kateno %>">
63                         <%Response.Write(gelenkategori.Kateadi); %></a></li>
64                     <%}//for iç %>
65
66                 </ul>
67             </li>
68             <% if (Session["uyeid"] == null)
69                 { //if%>
70                 <li><a href="uyekayit.aspx">Üye Ol</a></li>
71             <% } //if
72             else
73                 { //else %
74                 <li><a href="uyelikislemeleri.aspx">Üyelik İşlemleri</a></li>
75                 <li><a href="guvenlicik.aspx">Güvenli Çıkış</a></li>
76                 <%}//else bitisi %>
77             </ul>
78         </div>
79     </nav>
80
81
```

Output: Successfully installed TypeScript typings

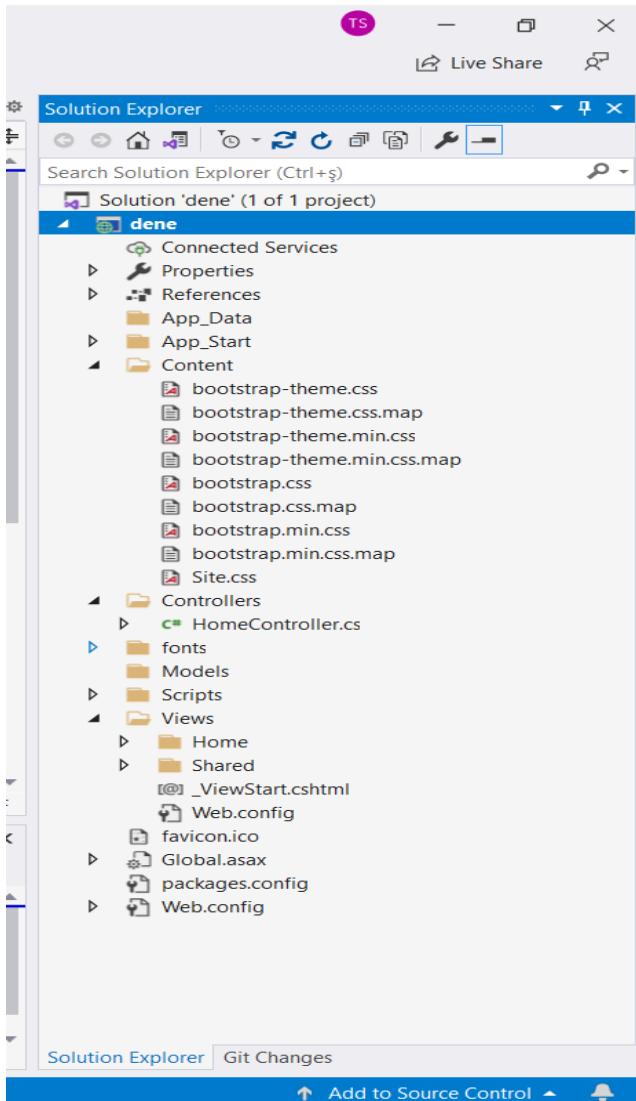
Add to Source Control

The screenshot shows the Microsoft Visual Studio IDE interface with the following details:

- Top Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Search (Ctrl+Q).
- Toolbar:** Includes icons for Undo, Redo, Cut, Copy, Paste, Find, Replace, and others.
- Solution Explorer:** Shows the project structure:
 - HomeController.cs
 - KatagoriController.cs
 - Kateliste.cshtml
 - _Layout.cshtml
 - UrulerController.cs
 - Yenikategori_Ekle.cshtml
- Code Editor:** Displays the `Kateliste.cshtml` file content, which is an MVC view for displaying category lists and actions.

```
1 @using Mvc_Eticaret.Models.Entity
2 @model List<kategoriler>
3 @{
4     ViewBag.Title = "Kateliste";
5     Layout = "~/Views/Shared/_Mainlayout.cshtml";
6 }
7
8 <table class="table table-bordered">
9     @foreach (var katgr in Model)
10    {
11        <tr>
12            <td>@katgr.kateno</td>
13            <td> @katgr.kateadi</td>
14            <td><a href="Sil/@katgr.kateno" class="btn btn-danger">Sil</a></td>
15            <td><a href="Guncelle/@katgr.kateno" class="btn btn-success">Güncelle</a></td></tr>
16        }
17        <tr><td><a href="Yenikategori_Ekle" class=" btn btn-primary">Ekle</a></td></tr>
18    </table>
```
- Status Bar:** Shows "Ready" status, file count (133 %), and other development metrics.
- Output Window:** Shows "No issues found" and "Using TypeScript 4.3 for IntelliSense".
- Bottom Bar:** Includes "Add to Source Control" and a notification icon.

Proje İçerisinde Öğelerin Anlamları Neler?



Controllers: URL isteklerini handle eden Controller class'larınızın bulunduğu klasördür.

Models: Data ve Business(iş) objelerinizi manipule(isleyen) eden ve temsil eden class'larınızın bulunduğu klasördür

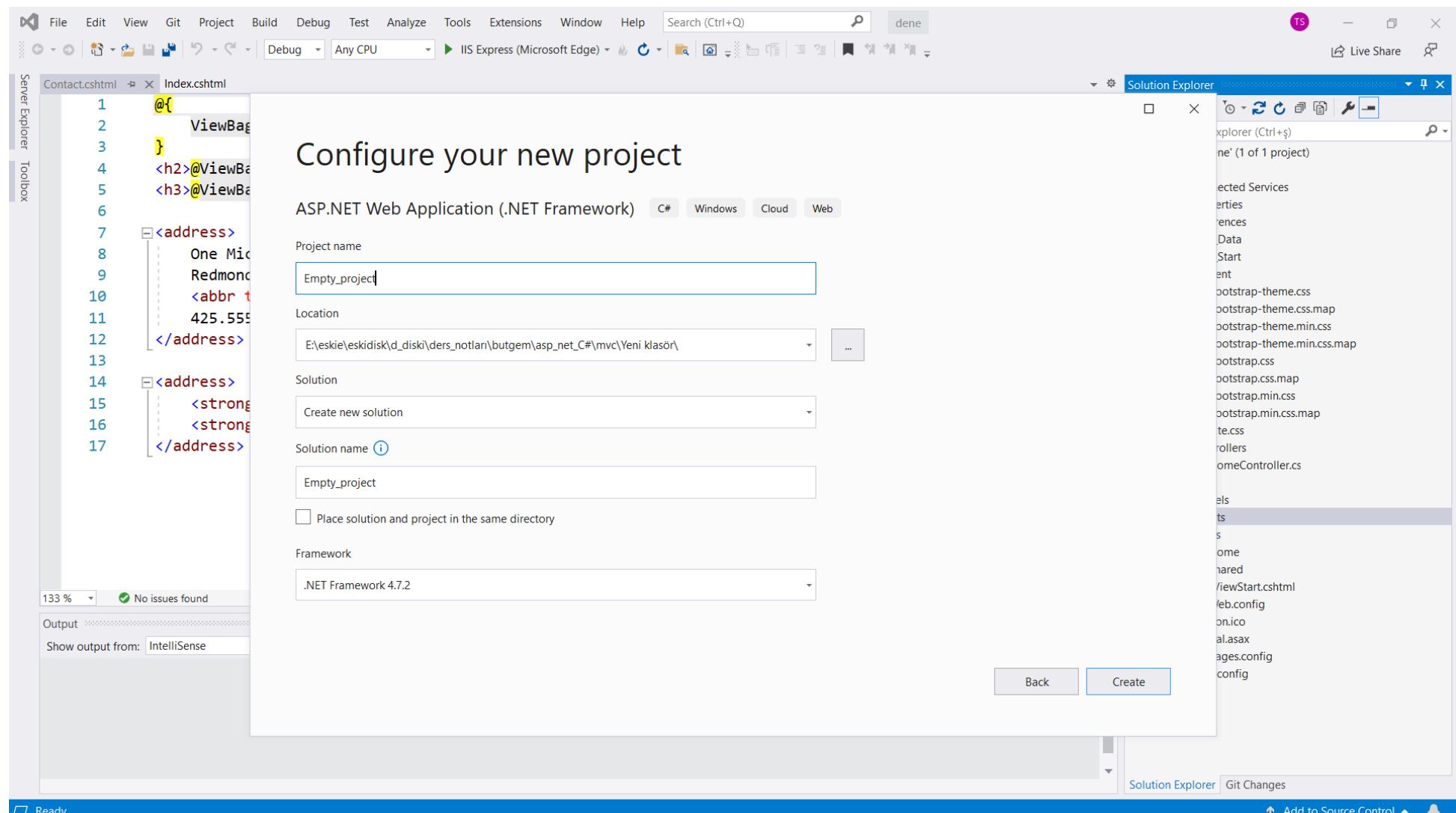
Views: HTML çıktısı olarak karşımıza çıkan View ve Partial View'leri tutar. İçerisinde Controller'in ismindé bir klasör olur ve içerisinde View'ler vardır. Ayrıca Shared klasörü adı altında layout'lar tutulur, ASP.NET teki master page sayfaların işlevi yaparlar. Views içerisindeki Web.Config default Web.Config ten farklı olarak bu viewlere doğrudan erişimi engeller yani Controller'sız çalıştırmasını engeller.

App_Data: Okuma/yazma işlemi yapacağınız dosyaların bulunduğu klasördür.

Content: Dinamik olarak değişmeyecek olan ve JavaScript olmayan, css/image dosyalarınız gibi statik dosyaları tutabileceğiniz klasördür

Scripts: Bu klasör içerisinde, Script'ler ve JavaScript kütüphaneleri tutulur.

İlk Uygulama (Empty Template)



File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) 

Live Share 

Server Explorer Toolbox

Create a new ASP.NET Web Application

 **Empty**
An empty project template for creating ASP.NET applications. This template does not have any content in it.

 **Web Forms**
A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.

 **MVC**
A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

 **Web API**
A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.

 **Single Page Application**
A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes client-side interactions using HTML5, CSS3, and JavaScript.

Authentication
No Authentication [Change](#)

Add folders & core references

Web Forms
 MVC
 Web API

Advanced

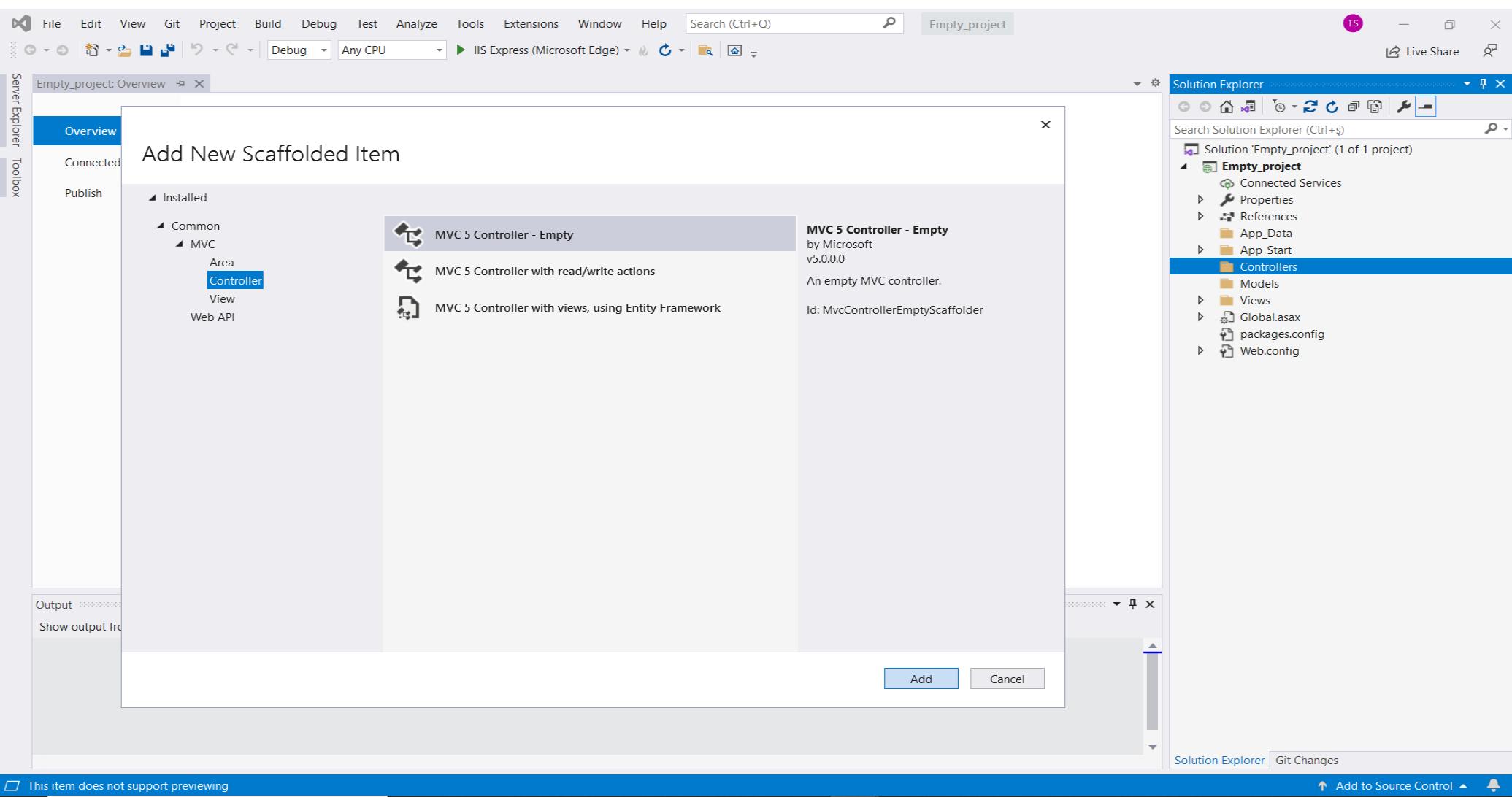
Configure for HTTPS
 Docker support
(Requires [Docker Desktop](#))
 Also create a project for unit tests
`Empty_project.Tests`

[Back](#) [Create](#)

Output Show output from: `IntelliSense`

Solution Explorer Git Changes

- Biz uygulamamı çalıştırıyoruz fakat uygulamamızın hat veriyor!!!!? Kullanıcının gördüğü dosyalar “View“’lerdir.
- View oluşturmadan önce “Controller” oluşturmak gerekiyor. Çünkü View’ler Controller den oluşturulur ve çağrıılır.
- Bir Controller olmadan View’inde kullanılması düşünülemez.
- Uygulamamızda “Controllers” klasörüne sağ tıklayıp Add daha sonra da Controller diyoruz.



File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) Empty_project

Server Explorer Toolbox

Empty_project: Overview

ASP.NET

Learn about the .NET platform, create your first application and extend it to the cloud.

Build Your App Connect To The Cloud Learn Your IDE

Get started with ASP.NET Publish your app to Azure See our productivity guide

.NET application architecture

Add Controller

Controller name: HomeController

Add Cancel

Solution Explorer

Search Solution Explorer (Ctrl+Shift+F)

Solution 'Empty_project' (1 of 1 project)

- Empty_project
 - Connected Services
 - Properties
 - References
 - App_Data
 - App_Start
 - Controllers
 - Models
 - Views
 - Global.asax
 - packages.config
 - Web.config

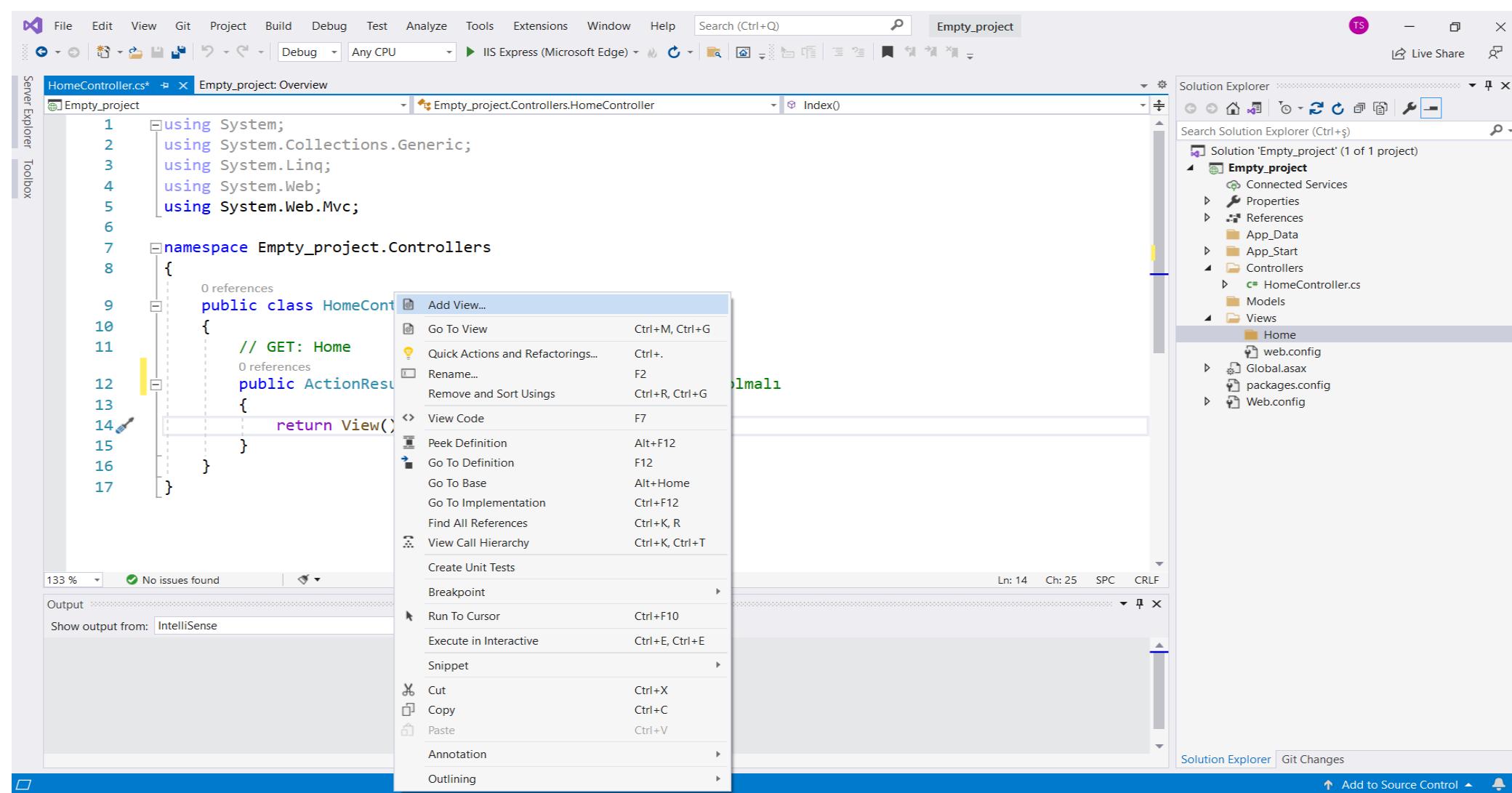
Output

Show output from: IntelliSense

This item does not support previewing

Add to Source Control

HomeController’ım içerisindeki Index adındaki ActionResult metodu içerisinde herhangi bir yer sağ tıklayıp “Add View” diyoruz.



Screenshot of Microsoft Visual Studio showing an ASP.NET MVC project named "Empty_project".

The code editor displays `HomeController.cs` with the following content:

```
4  using System.Web;
5  using System.Web.Mvc;
6
7  namespace Empty_project.Controllers
8  {
9      public class HomeController : Controller
10     {
11         // GET: Home
12         public ActionResult Index()
13         {
14             return View();
15         }
16         public ActionResult About()
17         {
18             return View();
19         }
20     }
21 }
22 }
```

An "Add View" dialog is open over the code editor, showing the following settings:

- View name: Index
- Template: Empty (without model)
- Model class: (empty)
- Options:
 - Create as a partial view
 - Reference script libraries
 - Use a layout page:
[Select dropdown]

The Solution Explorer shows the project structure:

- Solution 'Empty_project' (1 of 1 project)
 - Empty_project
 - Connected Services
 - Properties
 - References
 - App_Data
 - App_Start
 - Content
 - Site.css
 - Controllers
 - HomeController.cs
 - Models
 - Views
 - Home
 - About.cshtml
 - Index.cshtml
 - Shared
 - _Layout.cshtml
 - _ViewStart.cshtml
 - Global.asax
 - packages.config
 - Web.config

The Output window shows the following log entries:

```
iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Web.Mobile\v4.0_4.0.0.0__b03f5f7f11d56'
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Users\lastapex\AppData\Local\Temp\Temporary ASP.NET Files\vs\4da70872\ccac0642\App_We
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Users\lastapex\AppData\Local\Temp\Temporary ASP.NET Files\vs\4da70872\ccac0642\App_We
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Dynamic\v4.0_4.0.0.0__b03f5f7f11d50a3c
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\Microsoft.CSharp.resources\v4.0_4.0.0.0__tr_b6
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Users\lastapex\AppData\Local\Temp\Temporary ASP.NET Files\vs\4da70872\ccac0642\App_We
The program '[3736] iisexpress.exe' has exited with code -1 (0xffffffff).
```

The status bar at the bottom indicates "Item(s) Saved".

HomeController'ime geliyoruz ve birde About diye ActionResult tipinde bir metot ve About view oluşturuyoruz.

The screenshot shows the Microsoft Visual Studio IDE interface. In the center, the code editor displays the `HomeController.cs` file with two methods: `Index()` and `About()`. A modal dialog titled "Add View" is open over the code editor, prompting the user to create a new view named "About". The "Template" dropdown is set to "Empty (without model)". The "Model class" dropdown is empty. Under "Options", the "Create as a partial view" and "Reference script libraries" checkboxes are unchecked, while the "Use a layout page" checkbox is checked. Below this checkbox is a text input field containing the path `@{Layout = "Shared/_Layout.cshtml";}`. At the bottom of the dialog are "Add" and "Cancel" buttons. The Solution Explorer on the right side of the interface shows the project structure for "Empty_project", including the `HomeController.cs` file and its corresponding `Index.cshtml` view under the `Home` folder. The `Index.cshtml` file is currently selected. The status bar at the bottom indicates "Ready".

```
4  using System.Web;
5  using System.Web.Mvc;
6
7  namespace Empty_project.Controllers
8  {
9      public class HomeController : Controller
10     {
11         // GET: Home
12         public ActionResult Index() //index isminde bir view olmali
13         {
14             return View();
15         }
16         public ActionResult About()
17         {
18             return View();
19         }
20     }
21 }
22
```

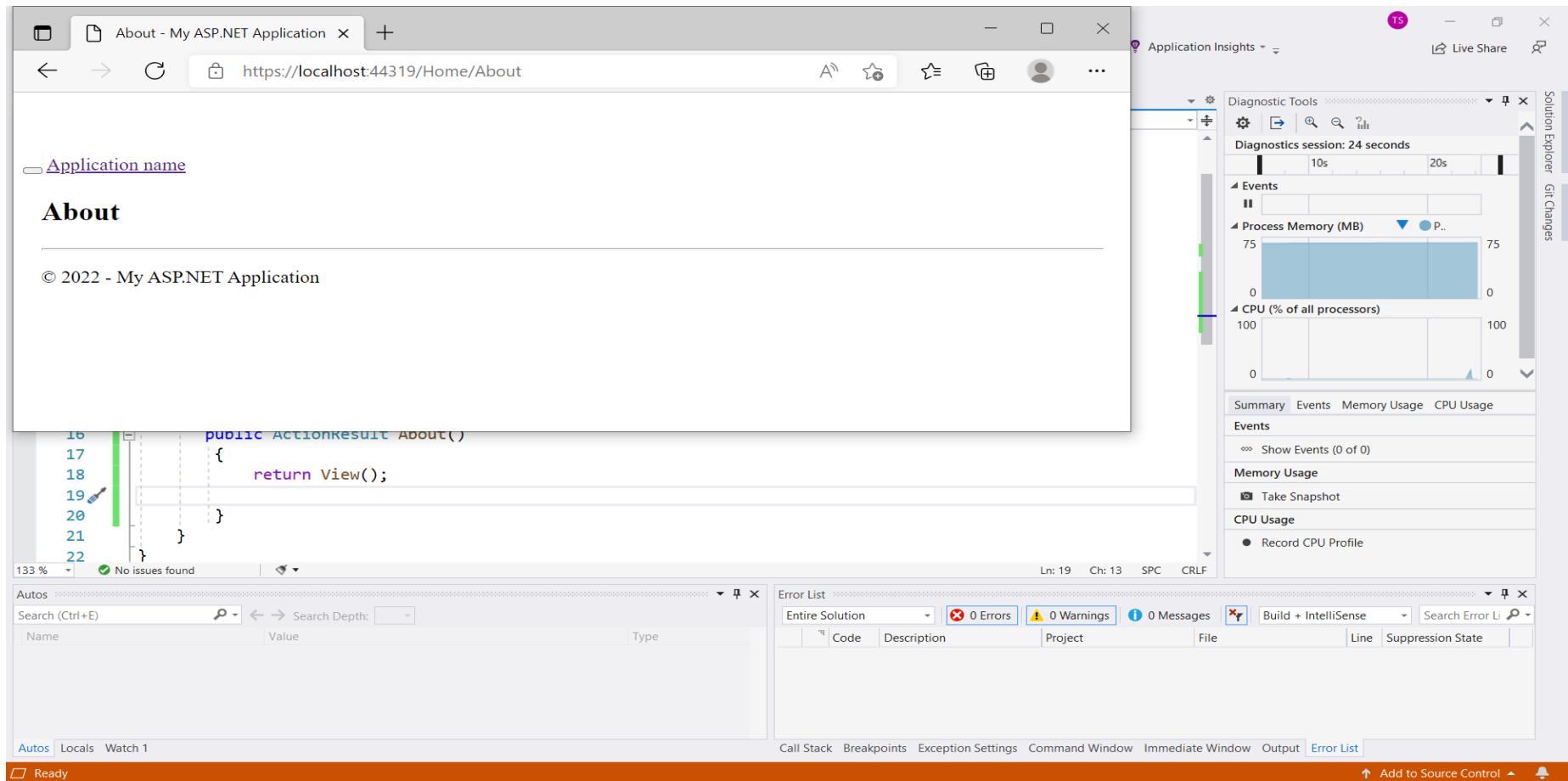
Output window:

```
iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\Temporary ASP.NET Files\vs\4da70872\ccac0642\App_Web-156e.htm', Symbols loaded.
iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\Temporary ASP.NET Files\vs\4da70872\ccac0642\App_Web-156e.htm', Symbols loaded.
iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\Temporary ASP.NET Files\vs\4da70872\ccac0642\App_Web-156e.htm', Symbols loaded.
iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\Temporary ASP.NET Files\vs\4da70872\ccac0642\App_Web-156e.htm', Symbols loaded.
iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\Temporary ASP.NET Files\vs\4da70872\ccac0642\App_Web-156e.htm', Symbols loaded.
iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132941762235036278): Loaded 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\Temporary ASP.NET Files\vs\4da70872\ccac0642\App_Web-156e.htm', Symbols loaded.
The program '[3736] iisexpress.exe' has exited with code -1 (0xffffffff).
```

About sayfasını çağrırmak için, Home/About yazdık.

Home: Oluşturduğum Controller’ım adı **About:** HomeController içerisindeki geriye View döndüren metotumun adı.

Sonuç klasik ASP.NET kullanırken sayfaları çağrıır ve öyle çalıştırırız, ASP.NET MVC de ise metotları çalıştırırız ve onlarla ,istisna dışında, aynı isimde bağlı olan View’ler görüntülenir.



Global.asax ve Route Mantığını Kavrama

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the code editor for the `Global.asax.cs` file, which contains the following code:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Web;
5  using System.Web.Mvc;
6  using System.Web.Routing;
7
8  namespace Empty_project
9  {
10    public class MvcApplication : System.Web.HttpApplication
11    {
12      protected void Application_Start()
13      {
14        AreaRegistration.RegisterAllAreas();
15        RouteConfig.RegisterRoutes(RouteTable.Routes);
16      }
17    }
18  }
19
```

The Solution Explorer on the right side shows the project structure for "Empty_project". It includes files like `RouteConfig.cs`, `HomeController.cs`, `Index.cshtml`, `About.cshtml`, `Application_Start()`, `Global.asax`, `packages.config`, and `Web.config`.

The Output window at the bottom shows the following message:

```
The thread 0x1d0c has exited with code 0 (0x0).
The thread 0x5008 has exited with code 0 (0x0).
The thread 0xdaec has exited with code 0 (0x0).
The thread 0xb58 has exited with code 0 (0x0).
The thread 0x3a80 has exited with code 0 (0x0).
The thread 0xfc8 has exited with code 0 (0x0).
The program '[10224] iisexpress.exe' has exited with code -1 (0xffffffff).
```

Static olarak tanımlanmış `RouteConfig` sınıfı içindeki “`RegisterRoutes`” adındaki metodumuzu inceleyelim.

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and a search bar. The toolbar contains icons for various operations like Open, Save, and Run. The main code editor window displays the `RouteConfig.cs` file, which defines a route configuration for a web application. The Solution Explorer on the right lists the project structure, including files like `Empty_project.csproj`, `RouteConfig.cs`, `Index.cshtml`, `HomeController.cs`, and `Global.asax.cs`. The Output window at the bottom shows debug logs indicating thread exits and the program's exit.

```
4  using System.Web;
5  using System.Web.Mvc;
6  using System.Web.Routing;
7
8  namespace Empty_project
9  {
10     public class RouteConfig
11     {
12         public static void RegisterRoutes(RouteCollection routes)
13         {
14             routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
15
16             routes.MapRoute(
17                 name: "Default",
18                 url: "{controller}/{action}/{id}",
19                 defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }
20             );
21         }
22     }
23 }
24
```

Output

```
Show output from: Debug
The thread 0x1d0c has exited with code 0 (0x0).
The thread 0x5008 has exited with code 0 (0x0).
The thread 0x4aec has exited with code 0 (0x0).
The thread 0xb58 has exited with code 0 (0x0).
The thread 0x3a80 has exited with code 0 (0x0).
The thread 0x1fc8 has exited with code 0 (0x0).
The program '[10224] iisexpress.exe' has exited with code -1 (0xffffffff).
```

Bu metot içerisinde `MapRoute`'ları mevcut. Burada Default isminde Route tanımlanmıştır. Daha sonra çalışma şekli olarak; **domain/controller adı/action adı**
Örnek: `deneme.com/home/index` Bu tanımlanan Route içerisinde default olarak Controller=Home, Action = Index olarak verilmiştir. İlk çalıştığı an siz aksini yazmazsanız, Home/Index i arayacaktır. bulamadığı içinde hata fırlatacaktır.
Kendimiz var olan Route üzerinde değişiklik yapabileceğimiz için kendimizde yeni `MapRoute`'lar ekleyebiliriz.

Razor ile Web Form Kullanım Farklılıkları Neler? Implicit Code Expression, Explicit Code Expression

Implicit Code Expression

```
@{string mesaj = "Implicit Code Expression";}
```

Razor @mesaj

Explicit Code Expression

Web Forms <%: mesaj %>

Bazı Özel Durumlar

```
@{string kisisim = "tuncaysali";}
```

<h3>@kisisim.com </h3> → string olduğu için .com metodunu arar hata olur

<h3>@(kisisim).com </h3> → (.....) içine alınarak metin olarak değerlendirilir.

Email adresi yazarken:tsali@gmail.com

Normalde yukarıkida kullanımdan hata alacağımızı düşünürüz fakat Razor bu ifadesinin email paternine sahip olduğunu anlamaktadır ve bu yüzden hata almıyoruz.

Ancak @tunsali, @BillGates hata verir.

Bu durumda @@tunsali, @@BillGates yazarız.

HTML ve C# İç İçe

Razor

```
@for (int i = 1; i < 11; i++)  
{  
<p>Sayı @i</p>  
}
```

Web Forms

```
<% for (int i=1; i < 11; i++) { %>  
<p> Sayı <%: i %></span>  
<% } %>
```

Razor Yorum satırları

@*

yorum satırları buraya yazılır

.....

*@

// tek satır yorum eklemek için yine kullanılabilir.

ViewBag

ASP.NET MVC'de Controller ile HTML arasında veri geçişlerini yapmaktan sorumludur.

Controller'da içerisinde data koyarız ve View içerisinde erişebiliriz, object veri almaktadır.

Ayrıca içerisine atayacağınız herhangi bir list, tabloya View içerisinde erişip verilerinizi kullanabilirsiniz. Bu sayede Controller içerisinde atadığınız verileri herhangi bir işlem yapmadan SelectList içerisinde görebilirsiniz

ViewBag

The image shows two side-by-side Microsoft Visual Studio IDE windows. Both windows have a dark theme and are displaying code related to a ViewBag example.

Left Window (Controller Code):

```
4  using System.Web;
5  using System.Web.Mvc;
6
7  namespace sil.Controllers
8  {
9      public class DefaultController : Controller
10     {
11         // GET: Default
12         public ActionResult Index()
13         {
14             List<int> sayilar = new List<int>();
15             for (int i=0;i<=100;i++)
16             {
17                 sayilar.Add(i);
18             }
19             ViewBag.sayilar = sayilar;
20             return View();
21         }
22     }
23 }
```

Right Window (View Code):

```
1
2  @{
3      ViewBag.Title = "Index";
4      List<int> sayilarim = ViewBag.sayilar;
5
6      foreach (int i in sayilarim)
7      {
8
9
10         <p>@i</p>
11
12     }
13     <h2>Index</h2>
14
15
16 }
```

```
tisim.cshtml          _benim_sablonum.cshtml      Index.cshtml      _Layout.cshtml      About.cshtml      HomeController.cs + X <<
}
0 references
public ActionResult Iletisim()
{
    return View();
}
0 references
public ActionResult ViewBag_Ornek()
{
    var sayilar = new List<int>();
    for (int i=0;i<=10;i++)
    {
        sayilar.Add(i);
    }
    ViewBag.sayilarim = new SelectList(sayilar);
    return View();
}
```

The screenshot shows the Visual Studio IDE interface. On the left, the Solution Explorer displays files like tisim.cshtml, _benim_sablonum.cshtml, Index.cshtml, _Layout.cshtml, About.cshtml, and HomeController.cs. The HomeController.cs file is open, showing two action methods: Iletisim() and ViewBag_Ornek(). The ViewBag_Ornek() method creates a list of integers from 0 to 10 and assigns it to the ViewBag. The viewbag_ornek.cshtml file contains an H2 tag and an @Html.DropDownList("sayilarim") helper. The browser window on the right shows the URL https://localhost:44319/Home/viewbag_ornek, displaying the title "viewbag_ornek" and the heading "viewbag_ornek". A dropdown menu is open, showing the numbers 0 through 10.

```
viewbag_ornek.cshtml  Iletisim.cshtml  _benim_sablonum.cshtml  Index.cshtml  _Layout.cshtml  About.cshtml
1
2  @{
3      ViewBag.Title = "viewbag_ornek";
4      Layout = "~/Views/Shared/_benim_sablonum.cshtml";
5  }
6
7  <h2>viewbag_ornek</h2>
8  @Html.DropDownList("sayilarim");
9
```

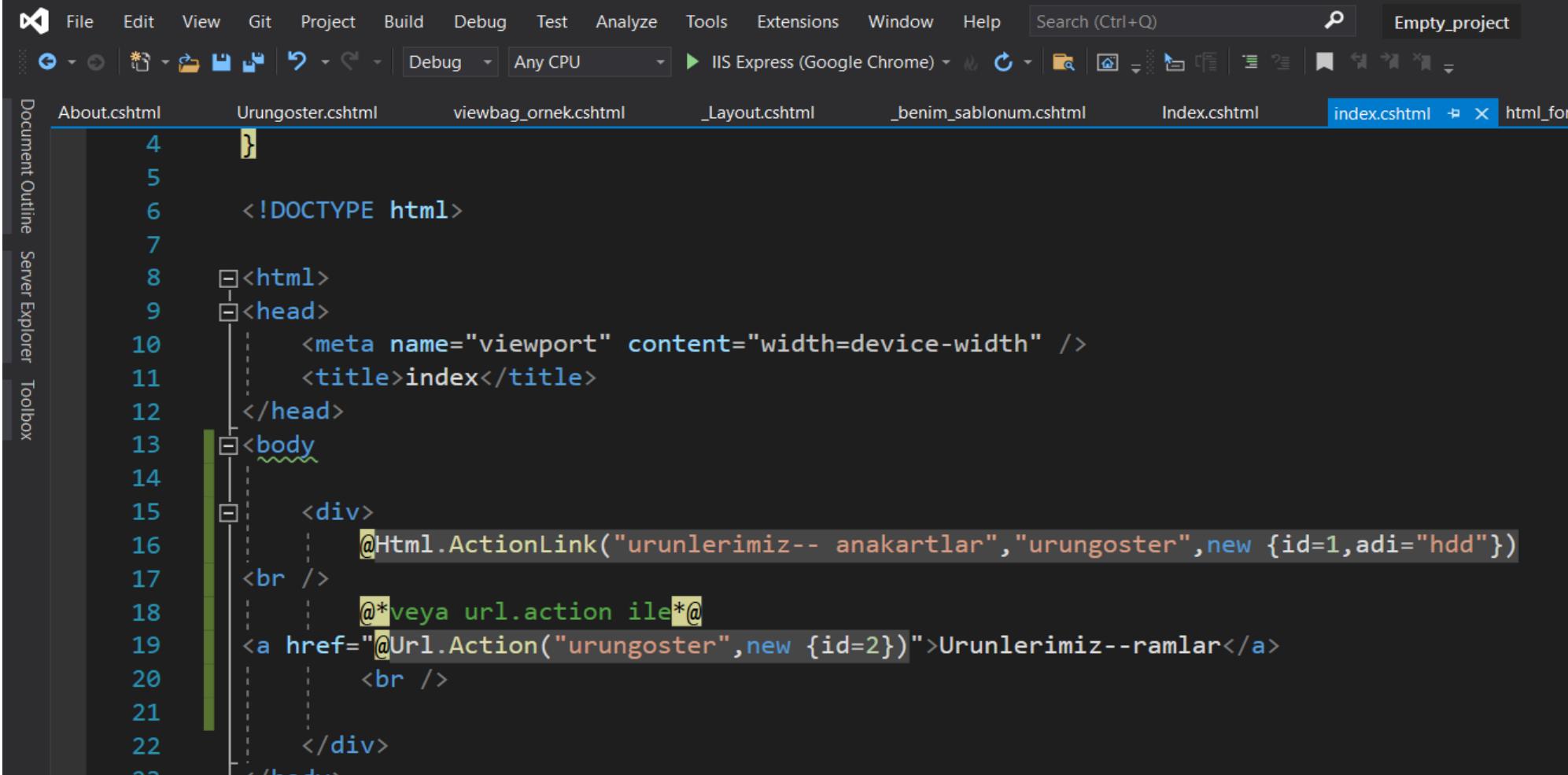
benim şablonum

viewbag_ornek

0 ▾;
0
1
2
3
4
5
6
7
8
9
10

ActionLink,Url.Action

Herhangi bir uygulama içerisinde diğer kaynaklara linkler verebilir, View'leri parçalara ayırarak bölümler halinde tekrar tekrar kullanabilir ya da diğer sayfaları kendi sayfalarımız içerisinde açabiliriz.



The screenshot shows the Microsoft Visual Studio IDE interface. The title bar reads "Empty_project". The menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help. The toolbar has various icons for file operations. The solution explorer shows files like About.cshtml, Urungoster.cshtml, viewbag_ornek.cshtml, _Layout.cshtml, _benim_sablonum.cshtml, Index.cshtml, index.cshtml, and html_for. The code editor displays the following C# Razor code:

```
4    }
5
6    <!DOCTYPE html>
7
8    <html>
9      <head>
10        <meta name="viewport" content="width=device-width" />
11        <title>index</title>
12      </head>
13      <body>
14
15        <div>
16          @Html.ActionLink("urunlerimiz-- anakartlar","urungoster",new {id=1,adi="hdd"})
17        <br />
18        @*veya url.action ile*@
19        <a href="@Url.Action("urungoster",new {id=2})">Urunlerimiz--ramlar</a>
20        <br />
21
22      </div>
23    </body>
```

ActionLink, Url.Action

The image shows two side-by-side Microsoft Visual Studio IDE windows. Both windows have a dark theme.

Left Window (Controller View):

- File menu: File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search bar: Search (Ctrl+Q).
- Toolbars: Standard toolbar with icons for file operations.
- Project Explorer: Shows files: About.cshtml, Urungoster.cshtml, viewbag_ornek.cshtml, _Layout.cshtml, _benim_sablonum.cshtml, Index.cshtml.
- Document Outline: Shows the structure of the current file.
- Server Explorer: Shows connection status.
- Toolbox: Available tools for the current project.
- Code Editor:

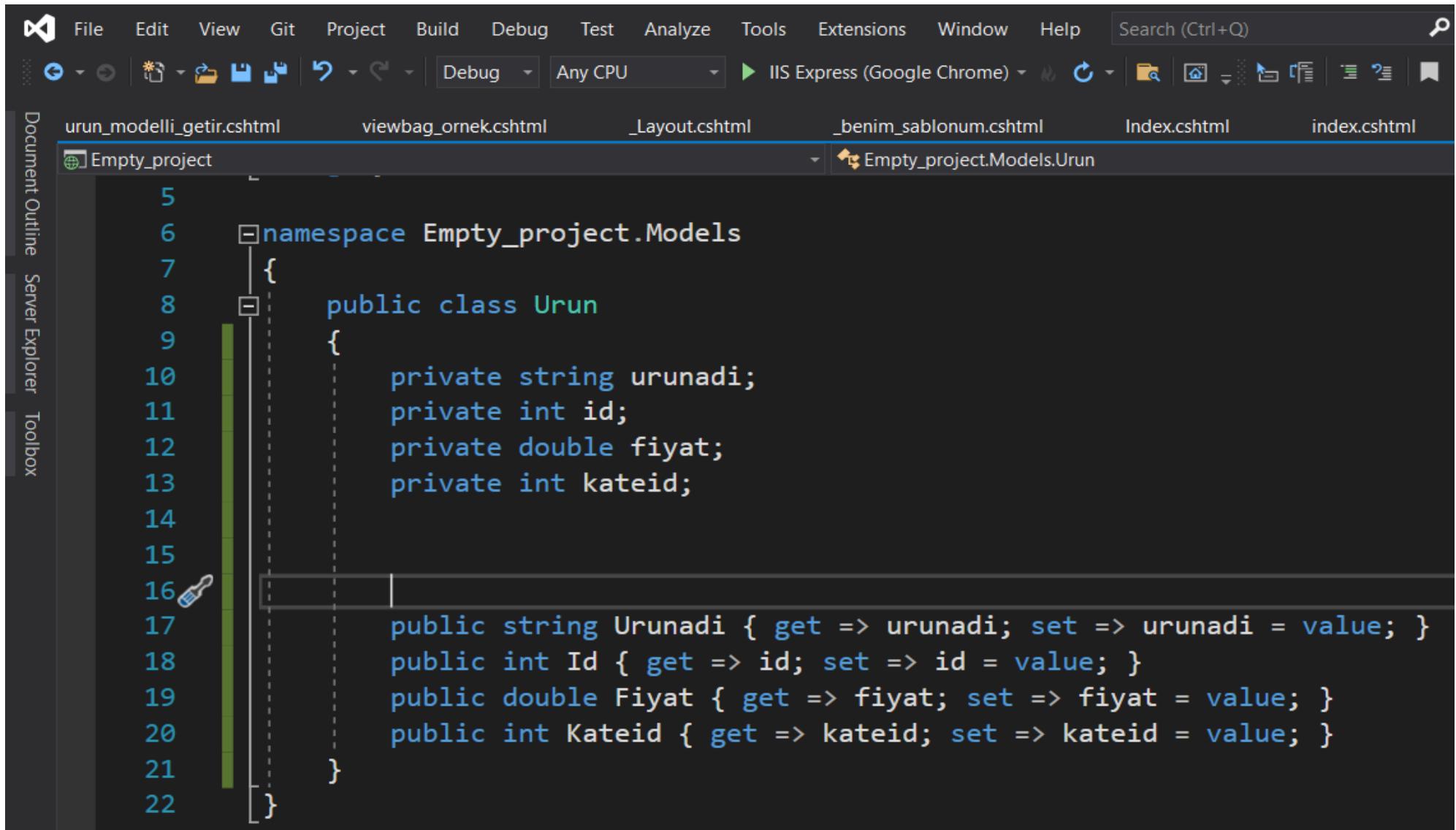
```
7  namespace Empty_project.Controllers
8  {
9      public class UrunlerController : Controller
10     {
11         // GET: Urunler
12         public ActionResult Urungoster(int id, String adi)
13         {
14             List<string> anakartlar = new List<string>();
15             anakartlar.Add("gigabyte 500");
16             anakartlar.Add("msi 300");
17             List<string> ramler = new List<string>
18             {
19                 "kşngson 25", "oem 15"
20             };
21
22             List<string> geri_gidecek;
23             if (id == 1) geri_gidecek = anakartlar;
24             else geri_gidecek = ramler;
25             ViewBag.urunler = geri_gidecek;
26             return View();
27         }
28         public ActionResult index()
29         {
30             return View();
31         }
32     }
33 }
```
- Status Bar: 105%, No issues found.

Right Window (View Code):

- File menu: File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search bar: Search (Ctrl+Q).
- Toolbars: Standard toolbar with icons for file operations.
- Project Explorer: Shows files: About.cshtml, Urungoster.cshtml, viewbag_ornek.cshtml, _Layout.cshtml, _benim_sablonum.cshtml.
- Document Outline: Shows the structure of the current file.
- Server Explorer: Shows connection status.
- Toolbox: Available tools for the current project.
- Code Editor:

```
1
2     @{
3         ViewBag.Title = "Urungoster";
4     }
5
6     <table>
7         @if (ViewBag.urunler != null)
8         {
9             foreach (string urun in ViewBag.urunler)
10             {
11                 <tr><td>@urun</td></tr>
12             }
13         }
14     </table>
15
16
17
18
19
```

ActionLink, Url.Action-MODEL OLUŞTURALIM



The screenshot shows the Microsoft Visual Studio IDE interface. The title bar includes the standard menu items: File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search (Ctrl+Q). Below the title bar is a toolbar with various icons for file operations like Open, Save, and Print.

The solution explorer on the left lists several files: urun_modelli_getir.cshtml, viewbag_ornek.cshtml, _Layout.cshtml, _benim_sablonum.cshtml, Index.cshtml, and index.cshtml. The current file being edited is Empty_project/Empty_project.Models.Urun.

The code editor displays the following C# class definition:

```
5
6 namespace Empty_project.Models
7 {
8     public class Urun
9     {
10         private string urunadi;
11         private int id;
12         private double fiyat;
13         private int kateid;
14
15
16     }
17     public string Urunadi { get => urunadi; set => urunadi = value; }
18     public int Id { get => id; set => id = value; }
19     public double Fiyat { get => fiyat; set => fiyat = value; }
20     public int Kateid { get => kateid; set => kateid = value; }
21 }
22 }
```

ActionLink, Url.Action-MODEL OLUŞTURALIM

The screenshot shows the Microsoft Visual Studio IDE interface. The title bar reads "ActionLink,Url.Action-MODEL OLUŞTURALIM". The main window displays the code for a class named "Urun_db" located in the file "Empty_project.cshtml". The code defines a class "Urun_db" with a list of products ("urunler") and initializes it with three product objects. The properties for each product are: Id, Urunadi, Fiyat, and Kateid.

```
7 public class Urun_db
8 {
9     public List<Urun> urunler;
10    public Urun_db()
11    {
12        urunler = new List<Urun>
13        {
14            new Urun
15            {
16                Id =1,
17                Urunadi="gigabyte",
18                Fiyat=500,Kateid =1
19            },
20            new Urun
21            {
22                Id =2,
23                Urunadi="msi",
24                Fiyat=400,Kateid=1
25            },
26            new Urun
27            {
28                Id =3,
29                Urunadi="kingson",
30                Fiyat=400,Kateid=2
31            },
32        };
33    }
34 }
35
36
37
38
39
40
41 }
```

105 % No issues found

Output Error List

Ln: 32 Ch: 41 SPC CRLF



Add to Source Control

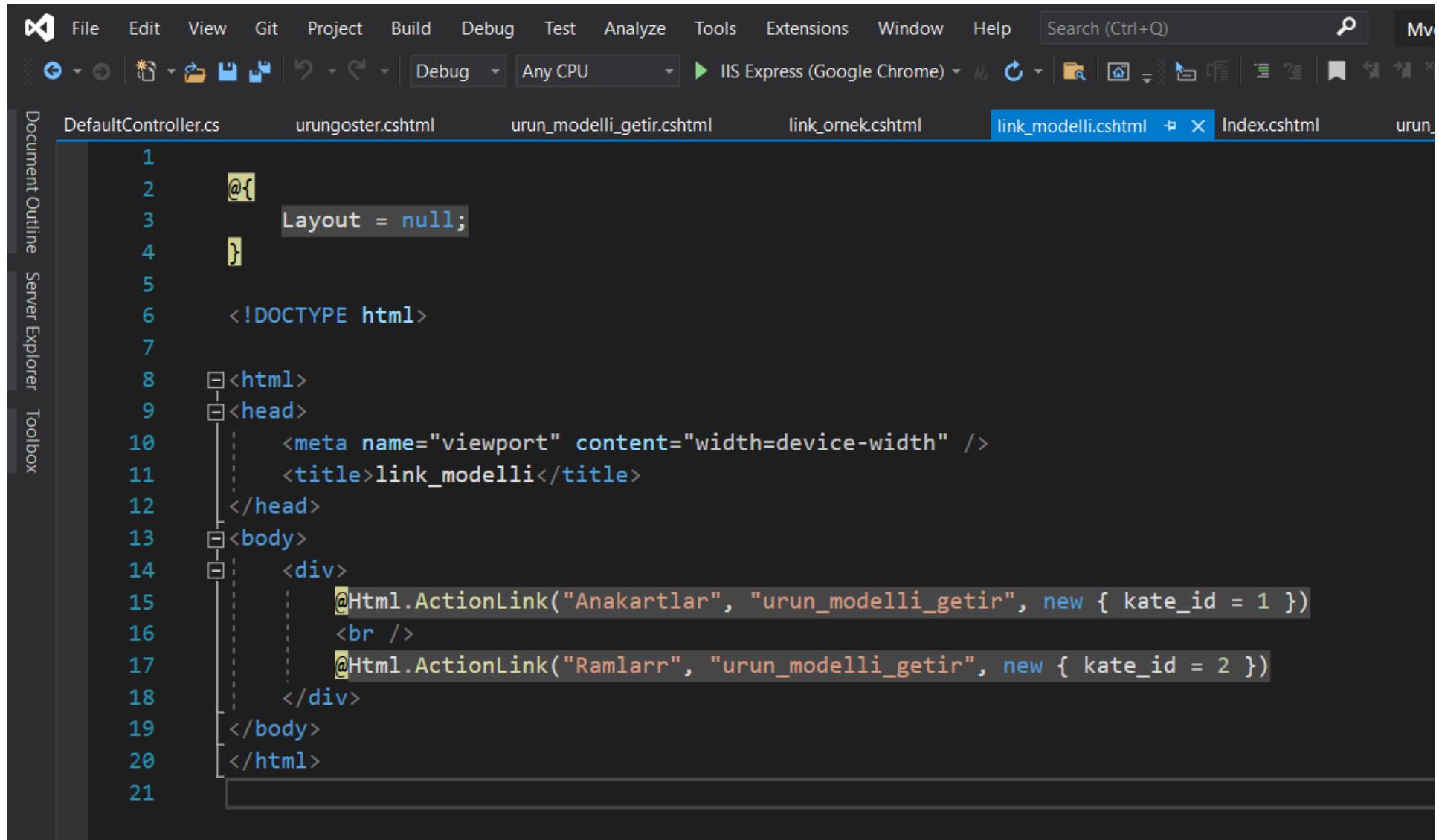


Aramak için buraya yazın



23°C 10:32
9.06.2022

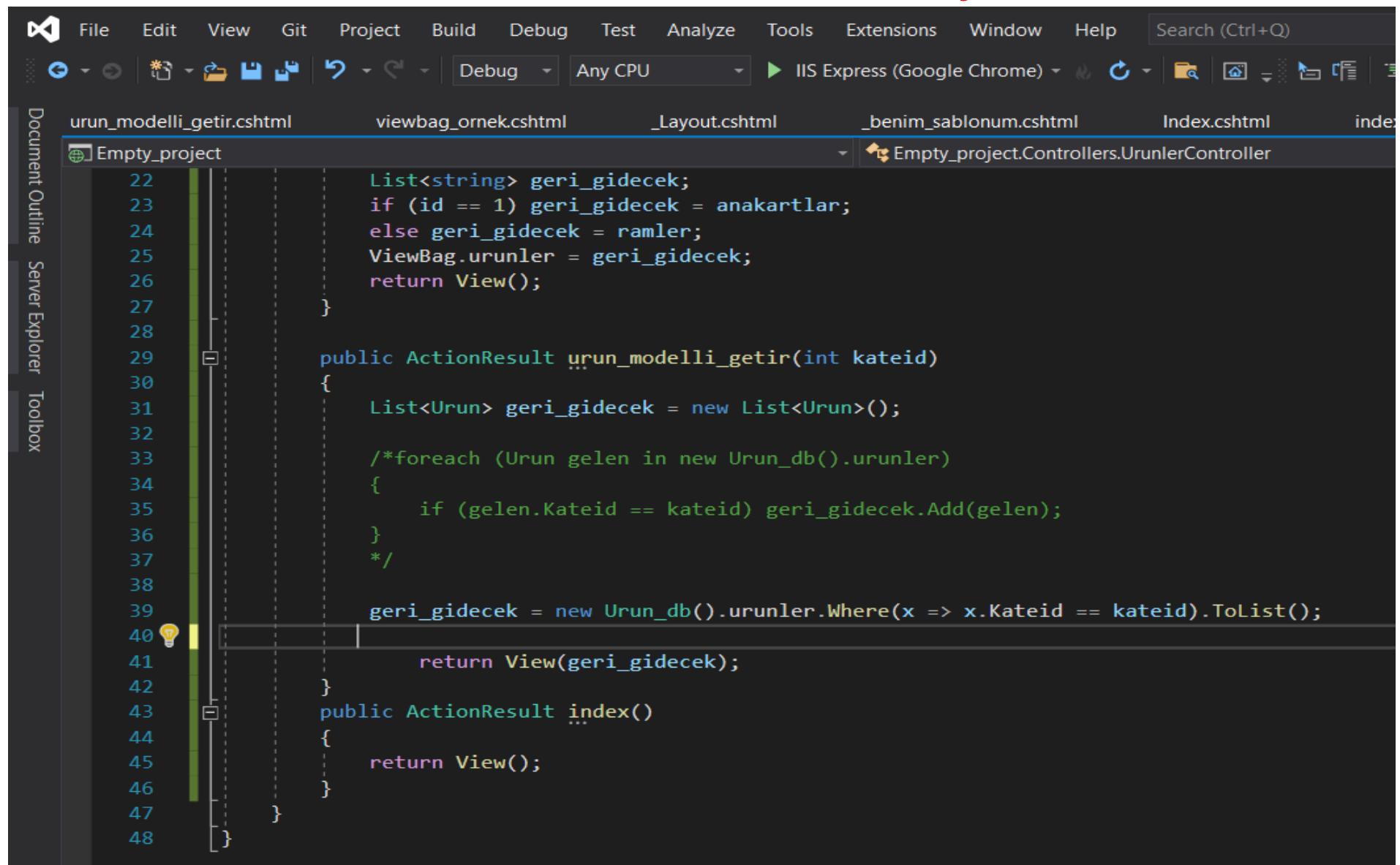
ActionLink,Url.Action-MODEL OLUŞTURALIM



The screenshot shows the Microsoft Visual Studio IDE interface. The title bar includes the standard menu options: File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and a Search bar. Below the title bar, the toolbar contains icons for file operations like Open, Save, and Build. The main window displays a code editor with several tabs at the top: DefaultController.cs, urungoster.cshtml, urun_modelli_getir.cshtml, link_ornek.cshtml, link_modelli.cshtml (which is currently selected), and Index.cshtml. The code editor itself shows the following C# and HTML code:

```
1  @{
2      Layout = null;
3  }
4
5
6  <!DOCTYPE html>
7
8  <html>
9  <head>
10     <meta name="viewport" content="width=device-width" />
11     <title>link_modelli</title>
12 </head>
13 <body>
14     <div>
15         @Html.ActionLink("Anakartlar", "urun_modelli_getir", new { kate_id = 1 })
16         <br />
17         @Html.ActionLink("Ramlarr", "urun_modelli_getir", new { kate_id = 2 })
18     </div>
19 </body>
20 </html>
21
```

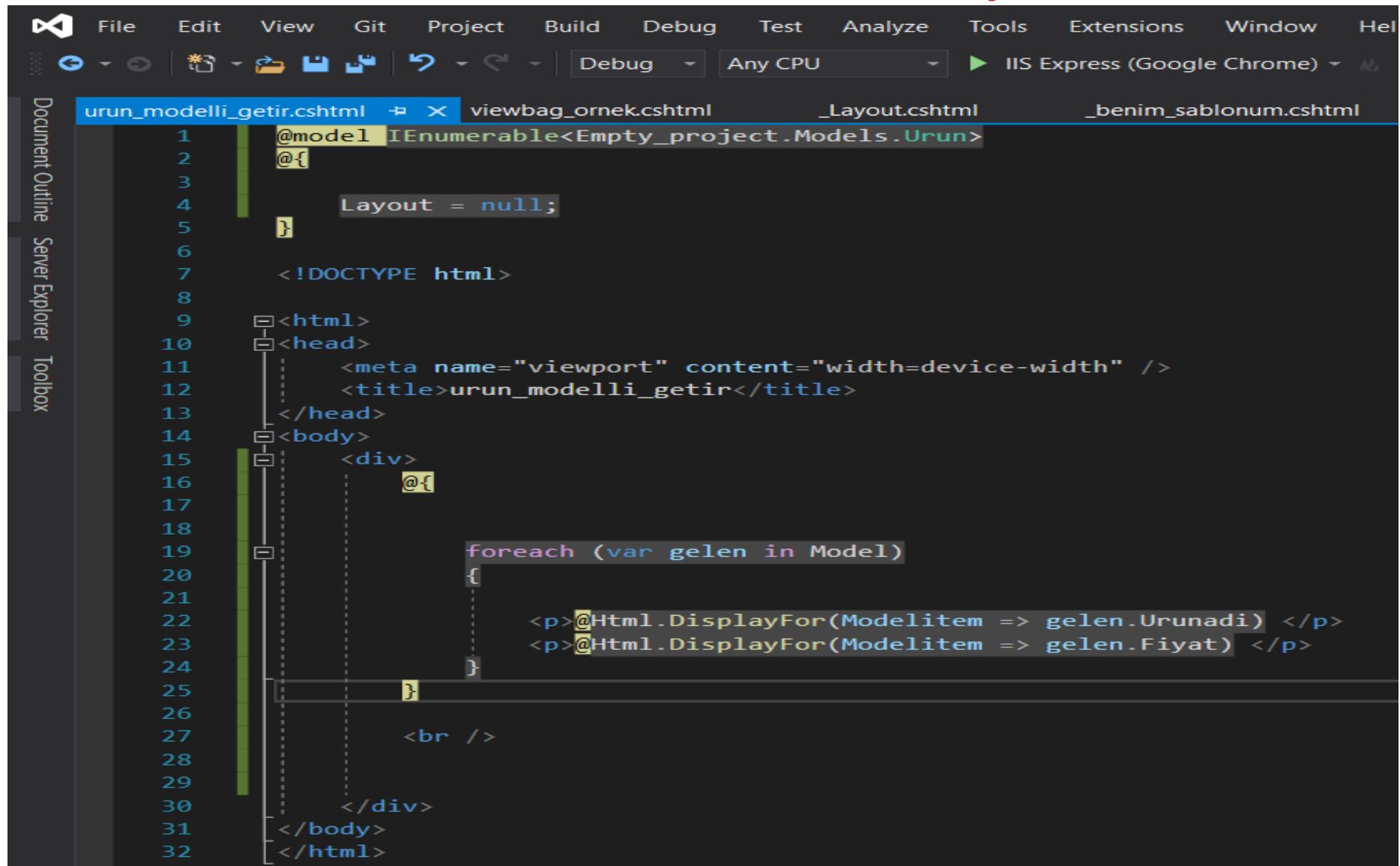
ActionLink, Url.Action-MODEL OLUŞTURALIM



The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search (Ctrl+Q). The toolbar below the menu has icons for file operations like Open, Save, and Print. The solution explorer on the left lists files: urun_modelli_getir.cshtml, viewbag_ornek.cshtml, _Layout.cshtml, _benim_sablonum.cshtml, Index.cshtml, and index. The current file is Empty_project.cshtml, which contains the following C# code:

```
22     List<string> geri_gidecek;
23     if (id == 1) geri_gidecek = anakartlar;
24     else geri_gidecek = ramler;
25     ViewBag.urunler = geri_gidecek;
26     return View();
27 }
28
29 public ActionResult urun_modelli_getir(int kateid)
30 {
31     List<Urun> geri_gidecek = new List<Urun>();
32
33     /*foreach (Urun gelen in new Urun_db().urunler)
34     {
35         if (gelen.Kateid == kateid) geri_gidecek.Add(gelen);
36     }
37 */
38
39     geri_gidecek = new Urun_db().urunler.Where(x => x.Kateid == kateid).ToList();
40
41     return View(geri_gidecek);
42 }
43 public ActionResult index()
44 {
45     return View();
46 }
47 }
48 }
```

ActionLink, Url.Action-MODEL OLUŞTURALIM



The screenshot shows the Microsoft Visual Studio IDE interface. The title bar includes 'File', 'Edit', 'View', 'Git', 'Project', 'Build', 'Debug', 'Test', 'Analyze', 'Tools', 'Extensions', 'Window', and 'Help'. Below the title bar, there are tabs for 'urun_modelli_getir.cshtml', 'viewbag_ornek.cshtml', '_Layout.cshtml', and '_benim_sablonum.cshtml'. The 'urun_modelli_getir.cshtml' tab is active. The code editor displays the following C# Razor view code:

```
1 @model IEnumerable<Empty_project.Models.Urun>
2 @{
3     Layout = null;
4 }
5
6 <!DOCTYPE html>
7
8
9 <html>
10 <head>
11     <meta name="viewport" content="width=device-width" />
12     <title>urun_modelli_getir</title>
13 </head>
14 <body>
15     <div>
16         @{
17             foreach (var gelen in Model)
18             {
19                 <p>@Html.DisplayFor(Modelitem => gelen.Urunadi) </p>
20                 <p>@Html.DisplayFor(Modelitem => gelen.Fiyat) </p>
21             }
22         }
23         <br />
24     </div>
25 </body>
26 </html>
```

Şablon Oluşturuken Kendi Stil Dosyamız nasıl Ekleyeceğiz?

- <link href="~/Content/sablonstil.css" rel="stylesheet" /> veya
- @Styles.Render("~/Content/css") çağrırlabiliriz ancak bunun için Appstart altındaki BundleConfig dosyasında

```
bundles.Add(new StyleBundle("~/Content/css").Include(  
    "~/Content/bootstrap.css",  
    "~/Content/site.css",  
    "~/Content/sablonstil.css"));
```

Eklemeliyiz.

Yeni BundleConfig Ekleme

- Tools > NugetPackageManager > Package Manager Console
Install-Package Microsoft.AspNet.Web.Optimization ile eklentiyi yükliyoruz.
- **MVC şablonlu yeni proje açıyoz** .Ordan App_start altındaki BundleConfig.cs alıp kendi projemize kopyalıyoruz.
namespace kendi projemizle değiştiriyoruz..
- **Global.aspx** içerisinde
using System.Web.Optimization;
BundleConfig.RegisterBundles(BundleTable.Bundles);
- **Views/web.Config içindeki namespaces**
< add namespace="System.Web.Optimization" />
- **.cshtml sayfanızda css dosyalarının olması gereken yere**
@Styles.Render("~/Content/css") yazıyoruz
@Html.ActionLink("Anakartlar", "urungoster", new { id = 1 }, new { @class = "btn btn-danger" })

HTML.Form ELEMANLARI

- `@Html.TextBox(string name, object value)`

Kullanımı: `@Html.TextBox("txtEmail", "Email Giriniz...")`

Html Çıktısı:<input id="txtEmail" name="txtEmail" type="text" value="Email Giriniz" />

- `@Html.TextArea("txtMesaj", "Lütfen mesajınızı yazınız.")`

Html Çıktısı:<textarea cols="20" id="txtMesaj" name="txtMesaj" rows="2"> Lütfen mesaj; mesajınızı yazınız.</textarea>

- `Html.CheckBox`

CheckBox Helper'i, 2 input element render ettiği için benzersizdir.

`@Html.CheckBox(string name, bool isChecked)`

Kullanımı: `@Html.CheckBox("cbEhliyetVarmi",false)`

Html Çıktısı:

```
<input id="cbEhliyetVarmi" name="cbEhliyetVarmi" type="checkbox" value="true" /><input  
name="cbEhliyetVarmi" type="hidden" value="false" />
```

Tarayıcının checkbox'ın sadece selected olma durumunda değeri submit etmesini sağlamak için iki input element render eder

HTML.Form ELEMANLARI

- **@Html.RadioButton**

Html.RadioButton'lar, Genellikle tek bir sonuç için muhtemel seçenekleri sağlayan gruplanmış kontrollerdir. Radiobutton'ları grüplamak için, her birisine aynı ismi(name) vermelisiniz.

Kullanımı: @Html.RadioButton("pet", "Köpek",true)

 @Html.RadioButton("pet", "Kedi")

Html Çıktısı:

```
<input checked="checked" id="pet" name="pet" type="radio" value="Köpek" />
<input id="pet" name="pet" type="radio" value="Kedi" />
```

HTML.Form ELEMANLARI

- **@Html.DropDownList(string name, IEnumerable selectList, string optionLabel, IDictionary<string, object> htmlAttributes)**
- **@Html.ListBox(string name, IEnumerable selectList, string optionLabel, IDictionary<string, object> htmlAttributes)**

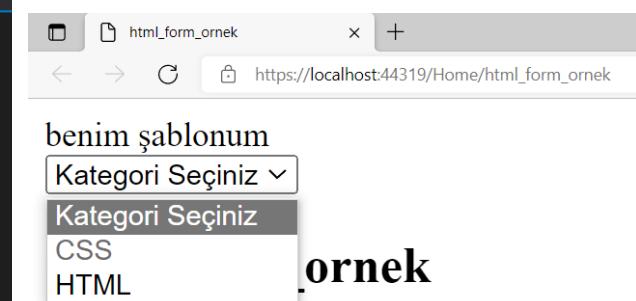
string name = "Kontrolün ismini"

IEnumerable = "Listenin tanımlandığı parametredir"

string optionLabel = "Default değeridir -Bir seçim yapınız-"

IDictionary<string, object> = "DropDownListimizde HTML kodlarını kullanacağımız

```
1  @{
2      ViewBag.Title = "html_form_ornek";
3      Layout = "~/Views/Shared/_benim_sablonum.cshtml";
4  }
5  @Html.DropDownList("kategoriler",
6      new List<SelectListItem>
7      {
8          new SelectListItem
9          { Text="CSS", Value="1",
10             //Selected = true, //seçili olarak gelsin
11             Disabled = true //disable olarak gelsin istersek kullanıyoruz
12         },
13         new SelectListItem
14         { Text="HTML", Value="2" },
15     }, "Kategori Seçiniz", new { @class = "form-control"})
16
17
18 <h2>html_form_ornek</h2>
19
20
```



HTML.Form ELEMANLARI

- **@Html.Label(string expression,string text)**

İk parametre olarak For Attribute'ine atacağınız değerdir. Bir radiobutton'a,checkbox 'a For diyerek expression verebilir ve kullanabilirsiniz.

Kullanımı:

```
@Html.Label("cbEhliyetVarmi", "Ehliyetiniz Varmı?") @Html.CheckBox("cbEhliyetVarmi")
```

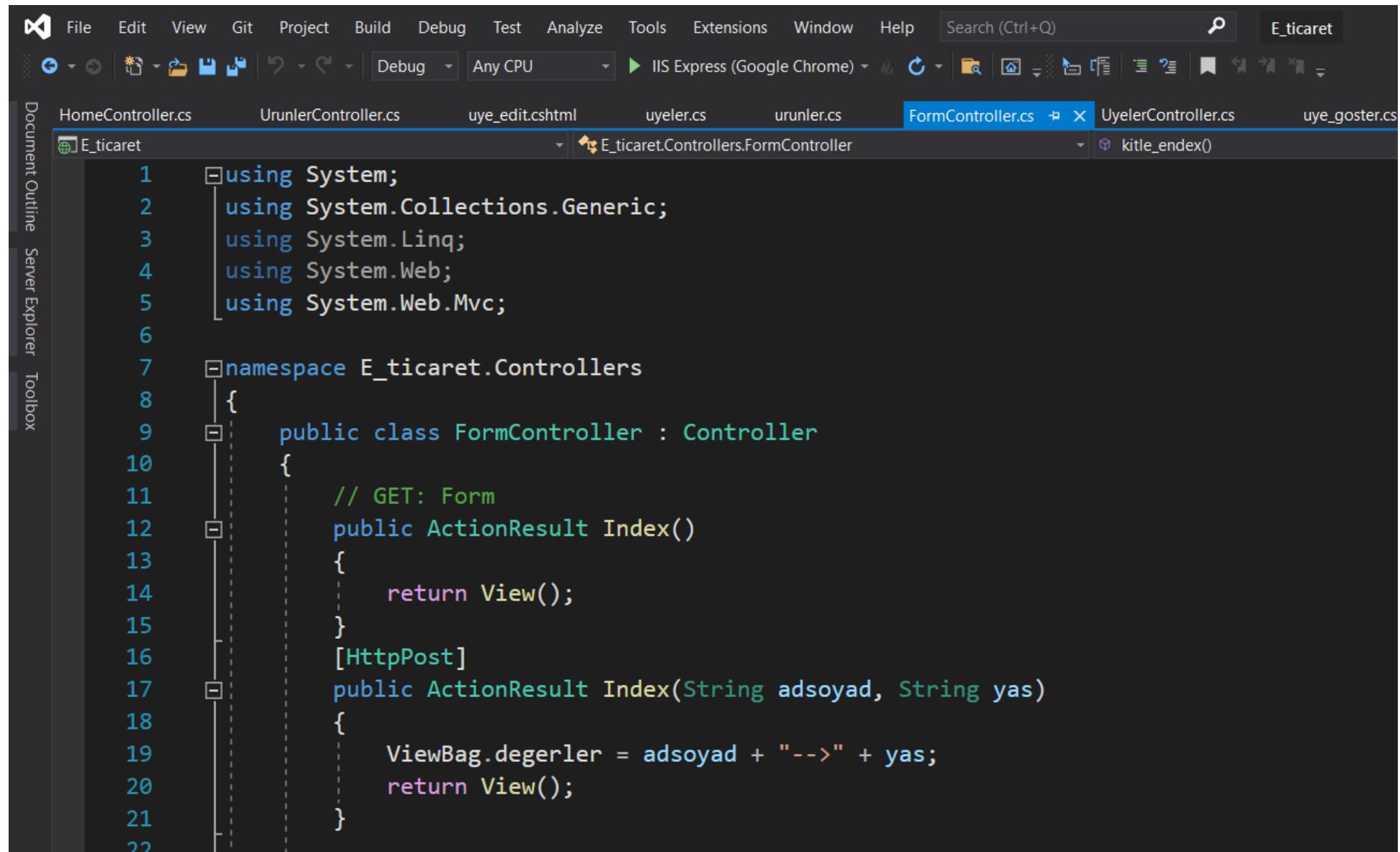
- **@Html.Password**

Password helper'i, password alanı render eder. TextBox helper'ina benzer fakat post edilen değeri içerisinde tutmaz ve isminde anlaşıldığı gibi password maskeli olarak giriş yapılır.

Kullanımı:

```
@Html.Password("txtSifre")
```

HTML.Form ELEMLANLARI



The screenshot shows the Microsoft Visual Studio IDE interface. The title bar reads "E_ticaret". The menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help. The toolbar has various icons for file operations like Open, Save, Print, etc. The status bar indicates "IIS Express (Google Chrome)" and "Any CPU". The code editor window displays the following C# controller code:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Web;
5  using System.Web.Mvc;
6
7  namespace E_ticaret.Controllers
8  {
9      public class FormController : Controller
10     {
11         // GET: Form
12         public ActionResult Index()
13         {
14             return View();
15         }
16         [HttpPost]
17         public ActionResult Index(String adsoyad, String yas)
18         {
19             ViewBag.degerler = adsoyad + "-->" + yas;
20             return View();
21         }
22     }
}
```

HTML.Form ELEMLANLARI

```
1
2     @{
3         Layout = null;
4     }
5
6     <!DOCTYPE html>
7
8     <html>
9         <head>
10            <meta name="viewport" content="width=device-width" />
11            <title>Index</title>
12        </head>
13        <body>
14            @ViewBag.degerler
15            <div>
16                @using (Html.BeginForm()) {
17                    <table>
18                        <tr><td>Adsoyad:@Html.TextBox("adsoyad")</td></tr>
19                        <tr><td>Yaşınız:@Html.TextBox("yas")</td></tr>
20                        <tr><td><input id="Submit1" type="submit" value="submit" /></td></tr>
21                    </table>
22                }
23            </div>
24        </body>
25    </html>
26
```

HTML.Form ELEMANLARI

The screenshot shows the Microsoft Visual Studio IDE interface. The title bar reads "HTML.Form ELEMANLARI". The menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search (Ctrl+Q). The toolbar has various icons for file operations like Open, Save, and Build. The solution explorer on the left shows "DefaultController.cs", "urungoster.cshtml", "urun_modelli_getir.cshtml", "link_ornek.cshtml", and "link_modelli.cshtml". The main code editor window displays C# code for a controller named "FormController". The code defines two actions: "dort_islem" and "dort_islem(int sayi1,int sayi2,int islem)". The first action returns a view. The second action performs arithmetic operations based on the value of "islem" (1 for addition, 2 for subtraction, 3 for multiplication, 4 for division) and sets the result in ViewBag.sonuc. A status bar at the bottom indicates "126 % No issues found".

```
22
23     public ActionResult dort_islem()
24     {
25         return View();
26     }
27
28     [HttpPost]
29
30     public ActionResult dort_islem(int sayi1,int sayi2,int islem)
31     {
32         double sonuc = 0;
33         switch (islem)
34         {
35             case 1:
36                 sonuc = sayi1 + sayi2;
37                 break;
38             case 2:
39                 sonuc = sayi1 - sayi2;
310                 break;
311             case 3:
312                 sonuc = sayi1 * sayi2;
313                 break;
314             case 4:
315                 sonuc = sayi1 / sayi2;
316                 break;
317         }
318         ViewBag.sonuc = sonuc;
319         return View();
320     }
321 }
```

HTML.Form ELEMANLARI

The screenshot shows the Microsoft Visual Studio IDE interface. The title bar includes the project name "Mvc_ders". The menu bar contains File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and a Search bar. Below the menu is a toolbar with various icons. The code editor displays a C# MVC view file named "dort_islem.cshtml". The code uses @Html helpers to generate form elements like TextBoxes and RadioButtons. The code editor has syntax highlighting and a vertical scroll bar. To the left, there's a Document Outline and Server Explorer window.

```
7
8     <html>
9         <head>
10            <meta name="viewport" content="width=device-width" />
11            <title>dort_islem</title>
12        </head>
13        <body>
14            <div>
15                @ViewBag.sonuc
16                @using (Html.BeginForm())
17                {
18                    Sayi1:@Html.TextBox("sayi1","",new {@class="form-control",placeholder="sayi giriniz"})
19                    Sayi2:@Html.TextBox("sayi2","",new { @class = "form-control"})
20
21                    Toplam:@Html.RadioButton("islem", "1")
22                    Çıkar:@Html.RadioButton("islem", "2")
23                    Çarp:@Html.RadioButton("islem", "3")
24                    Böl:@Html.RadioButton("islem", "4")
25
26                    <p>
27                        <input id="Submit1" type="submit" value="Gönder" />
28
29                }
30            </div>
31        </body>
32        </html>
```

HTML.Form ELEMLANLARI

The screenshot shows a Visual Studio IDE window with the following details:

- Menu Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Full Screen.
- Toolbar:** Standard toolbar icons.
- Toolbox:** Standard toolbox icons.
- Code Editor:** The code editor displays C# code for a controller named `FormController.cs`. The code includes methods for calculating BMI and generating a list of kilogram values.

```
22  public ActionResult kitle_endex()
23  {
24      doldur();
25      return View();
26  }
27
28  [HttpPost]
29  public ActionResult kitle_endex(int kilo,double boy)
30  {
31      doldur();
32      String durum;
33      var kitle = kilo / (boy * boy);
34      if (kitle >= 0 && kitle < 19) durum = "zayıf";
35      else if (kitle >= 20 && kitle < 25) durum = "normal";
36      else if (kitle >= 26 && kitle < 30) durum = "kilolu";
37      else durum = "obez";
38      ViewBag.durum = durum;
39      return View();
40  }
41  void doldur()
42  {
43      List<SelectListItem> kilolar = new List<SelectListItem>();
44      for (var i = 10; i <= 200; i++)
45      {
46          SelectListItem kilo = new SelectListItem()
47          {
48              Text = i.ToString(),
49              Value = i.ToString()
50          };
51          kilolar.Add(kilo);
52      }
53      ViewBag.kilolar = kilolar;
54  }
55
56
57
58
59 }
```

HTML.Form ELEMLANLARI

```
File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Full Screen
Index.cshtml HomeController.cs UrunlerController.cs uye_edit.cshtml uyeler.cs FormController.cs UyelerController.cs kitle_endex.cshtml X urunler.cs uye_goster.cshtml _Layout.cshtml
1
2
3     Layout = null;
4
5
6 >DOCTYPE html>
7
8 <ml>
9 <ad>
10    <meta name="viewport" content="width=device-width" />
11    <title>kitle_endex</title>
12 <ead>
13 <dy>
14   <div>
15     @using (Html.BeginForm())
16     {
17       <table>
18         <tr><td>Kilonuz:<@Html.DropDownList("kilo", (List<SelectListItem>)ViewBag.kilolar, new { onchange = "this.form.submit()" })</td></tr>
19         <tr><td>Boyunu:<@Html.TextBox("boy")</td></tr>
20         <tr><td>sonuç:<@ViewBag.durum</td></tr>
21         <tr><td><input id="Submit1" type="submit" value="submit" /></td></tr>
22       </table>
23     }
24
25   </div>
26 <dy>
27 <ml>
```

HTML.Form ELEMANLARI

urun_kaydet x +

localhost:44319/Urunler/urun_islemleri

Urun id:
5

Urun adı:
silgi

fiyat:
500

Kategori:Anakart: Ram:

kaydet sil ara edit tümü

Ürün no	Ürün Adı	fiyat	kategori
1	gigabyte	500	1
2	msi	400	1
3	kingson	400	2
5	silgi	500	1

HTML.Form ELEMLANLARI

The screenshot shows a Visual Studio code editor with the following file structure:

- i.cshtml*
- Urun_db.cs
- Urun.cs
- UrunlerController.cs
- BundleConfig.cs

The i.cshtml file contains the following code:

```
@model IEnumerable<Empty_project.Models.Urun>
@{
    Layout = null;
}
@Styles.Render("~/Content/css")
<!DOCTYPE html>
<html>
    <head>
        <meta name="viewport" content="width=device-width" />
        <title>urun_kaydet</title>
    </head>
    <body>
        <div>
            @using (Html.BeginForm())
            {
                <p>Urun id:@Html.TextBox("id", "", new { @class = "form-control" })</p>
                <p>Urun adı:@Html.TextBox("urunadi", "", new { @class = "form-control" })</p>
                <p>fiyat:@Html.TextBox("fiyat", "", "", new { @class = "form-control" })</p>
                <p>Kategori:Anakart:@Html.RadioButton("kateid", "1", true) Ram:@Html.RadioButton("kateid", "2", true)</p>

                <input name="islem" type="submit" value="kaydet" />
                <input name="islem" type="submit" value="sil" />
                <input name="islem" type="submit" value="ara" />
                <input name="islem" type="submit" value="edit" />
                <input name="islem" type="submit" value="tümü" />
            }

        </div>
        <table class="table table-bordered">
            <tr><th>Ürün no</th> <th>Ürün Adı</th><th>fiyat</th><th>kategori</th></tr>
            @foreach (var gelen_urun in Model)
            {
                <tr>
                    <td>@Html.DisplayFor(ModelItem => gelen_urun.Id) </td>
                    <td>@Html.DisplayFor(ModelItem => gelen_urun.Urunadi) </td>
                    <td>
```

HTML.Form ELEMLARI

The screenshot shows a Microsoft Visual Studio interface with the following details:

- Menu Bar:** Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Full Screen.
- Toolbox:** Urun_db.cs, Urun.cs, UrulerController.cs*, BundleConfig.cs.
- Code Editor:** The active file is UrulerController.cs*. It contains C# code for a controller named UrulerController. The code handles HTTP GET and POST requests to manage a list of products (Urun). It uses Session to store the product list and performs actions like adding, deleting, and updating products based on the request parameters.
- Status Bar:** Shows 'Issues found' and navigation icons.

```
[HttpGet]
public ActionResult urun_islemleri()
{
    List<Urun> urulerim = new Urun_db().uruler;
    return View(urulerim);
}
[HttpPost]
public ActionResult urun_islemleri(Urun urunum, string islem)
{
    List<Urun> urulerim;
    List<Urun> arama_listesi=null;
    if (Session["urulerim"] != null) urulerim = (List<Urun>)Session["urulerim"];
    else urulerim = new Urun_db().uruler;

    //string islem = Request.Form["islem"];
    if (islem.Equals("kaydet"))
    {
        urulerim.Add(urunum);
    }
    else if (islem.Equals("sil"))
    {
        Urun sil_urun = urulerim.Where(x => x.Id == urunum.Id).First();
        urulerim.Remove(sil_urun);
    }
    else if (islem.Equals("ara"))
    {
        arama_listesi = urulerim.Where(x => x.Id == urunum.Id).ToList();
    }
    else if (islem.Equals("edit"))
    {
        int z = urulerim.FindIndex(x => x.Id == urunum.Id);
        urulerim[z] = urunum;
    }
    Session["urulerim"] = urulerim;
    if (islem.Equals("ara")) return View(arama_listesi);
    else return View(urulerim);
}
```

Entity Framework ile Veritabanı Modelleme

○ Entity Framework nedir?

Entity Framework 2008 yılından itibaren Microsoft tarafından geliştirilen ORM aracıdır.

○ ORM nedir?

ORM veya **Object to Relational Mapping** temel olarak veritabanında yer alan tablo ve alanları nesne olarak kullanmamıza imkan veren bir yazılım mimarisidir.

○ Neden kullanılır?

.NET veya herhangi bir programlama dili ile veritabanı uygulamaları yaparken ilk olarak veritabanı bağlantısı yapılır.

Daha sonra SQL komutları ile veritabanındaki veriler alınır.

Alınan veriler programlama diline uygun veri yapılarında saklanarak işlem yapılır.

Verilerin programlama diline uygun veri yapılarına dönüştürülmesi sırasında beklenmedik hatalar, sorunlar oluşabilir.

Ayrıca karmaşık veritabanı sorguları geliştirmeyi daha da zor hale getirir.

ORM araçlarının temel kullanım nedeni bu ve bunun gibi sorunları ortadan kaldırmaktır.

Entity Framework ile Veritabanı Modelleme

○ Avantajları

- OOP olarak kod geliştirmemize olanak tanır.
- Hiçbir SQL bilgisi olmayan bir kimse veritabanı işlemlerini **EF** ile gerçekleştirebilir.
- Herhangi bir veritabanına bağımlılık yoktur. Oracle, MS SQL ile kullanılabilir.
- Code First sayesinde projenizi veritabanınızı taşıma gereği duymadan istediğiniz yerde oluşturabilirsiniz. Bu da projelerinize büyük bir esneklik sağlamaktadır.
- Yazılım geliştirme zamanını azaltır.
- Yazılım geliştirme maliyetini azaltır.

○ Dezavantajları

- En büyük sorunumuz performans. Ado.Net gibi hızlı bir performansı yoktur. Tabi bu yavaş olduğu anlamına gelmez.
- Veritabanından veri alış-verişi yapılacak zaman kontrol bizde değil Entity Frameworktedir. Yani arka planda veritabanı işlemleri için kendisi sorgular oluşturmaktadır. Basit bir veri işlemi için karmaşık bir sorgu gönderebilmektedir. Syntax'ı yeni kullanacak kişiler için karmaşık gelebilmektedir ancak zamanla alışacaksınız.
- Schema'da herhangi bir değişiklik yapıldığı zaman EF çalışmayacaktır. Sizin bu Schema'yı solution'da güncellemeniz gerekmektedir.

Entity Framework ile Veritabanı Modelleme

- Entity Framework kurulumu

Entity Framework ORM aracı bir Nuget paketi olduğundan Visual studio geliştirme ortamında projeye sağ tıklayıp **Manage NuGet Packages** seçeneği ile kurulum yapılabilir.

Ayrıca **Package Manager Console** alanına aşağıdaki komut yazılarakta indirilebilir.
Install-Package EntityFramework

Model oluşturma

Entity Framework ile model oluşturmak için Database First, Model First ve Code First yaklaşımları kullanılır.

- Database First

Bu yapıda daha önceden hazırlanmış veritabanı tablolarına uygun nesneler Visual Studio veya komut satırı ile otomatik olarak oluşturulur.

Bu işlem için Visual Studio IDE ile projeye **ADO.NET Entity Data Model** ve **EF Designer from database** seçeneği ile veritabanı bağlantısı yapılarak işlem yapılacak tabloların eklenmesi yeterli olacaktır.

Entity Framework ile Veritabanı Modelleme

○ Model First

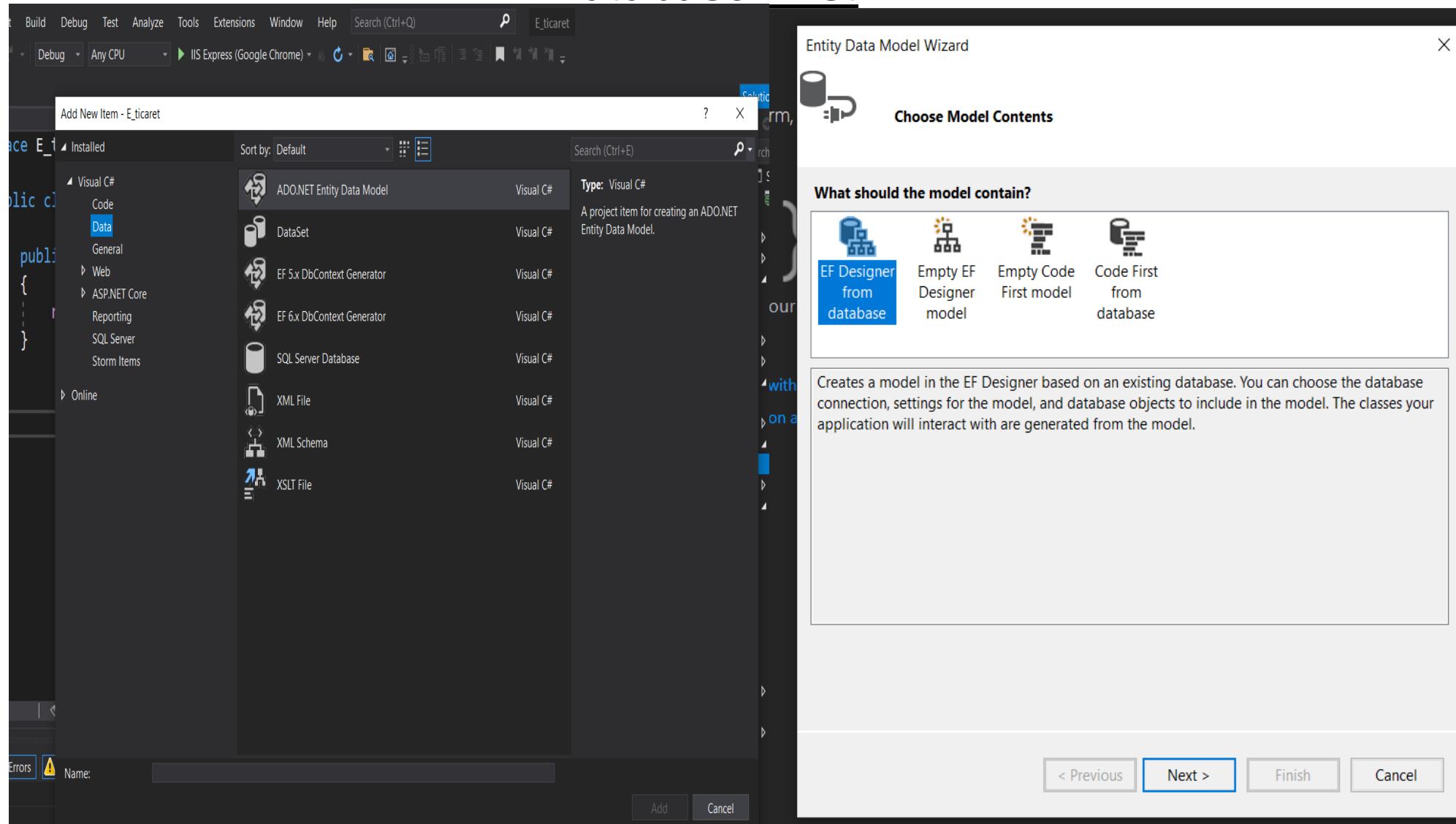
Bu yapıda Database First yapısına benzer şekilde **ADO.NET Entity Data Model** ve **EF Designer from model** seçeneği ile oluşturulur.,
Oluşturulan modele görsel olarak tablolar, tablo alanları eklenerek model oluşturulur.

○ Code First

Yukarıda yer alan Database First ve Model First ile aslında yapılan görsel arayüz ile veritabanı veya model uygun kodların oluşturulmasıdır.
Code First yaklaşımında ilk olarak tabloya karşılık gelen sınıf ve özellikler yazılır.

Entity Framework ile Veritabanı Modelleme

Database First



Entity Framework ile Veritabanı Modelleme

Database First

The screenshot shows two windows of the Entity Data Model Wizard running in parallel.

Left Window: Choose Your Data Connection

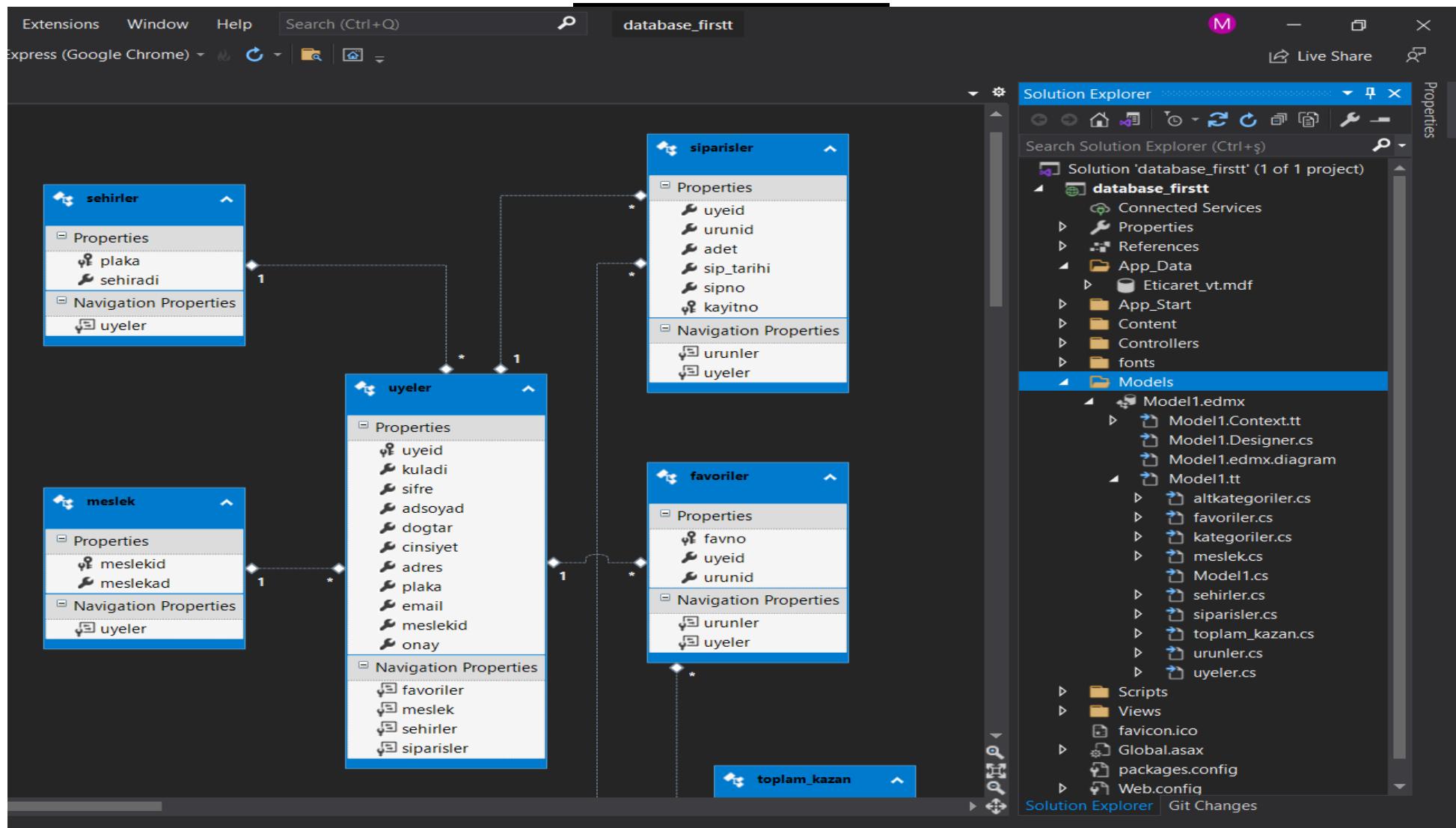
- Section:** Choose Your Data Connection
- Label:** Which data connection should your application use to connect to the database?
- Input:** Eticaret_vt.mdf (selected)
- Buttons:** New Connection...
- Note:** This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk.
- Options:**
 - No, exclude sensitive data from the connection string. I will set it in my application code.
 - Yes, include the sensitive data in the connection string.
- Section:** Connection string:
A large text area containing the connection string metadata=res://*/Model1.csdl|res://*/Model1.ssdl|res://*/Model1.msl;provider=System.Data.SqlClient;provider connection string="data source=(LocalDB)\MSSQLLocalDB;attachdbfilename=|DataDirectory|\Eticaret_vt.mdf;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework"
- Checkboxes:**
 - Save connection settings in Web.Config as:
- Input:** Eticaret_vtEntities
- Buttons:** < Previous, Next >, Finish, Cancel

Right Window: Choose Your Database Objects and Settings

- Section:** Choose Your Database Objects and Settings
- Section:** Which database objects do you want to include in your model?
 - Tables
 - dbo
 - active
 - altkategoriler
 - favoriler
 - kategoriler
 - meslek
 - sehirler
 - siparisler
 - toplam_kazan
 - urunler
 - uyeiler
 - Views
 - Stored Procedures and Functions
 - Pluralize or singularize generated object names
 - Include foreign key columns in the model
 - Import selected stored procedures and functions into the entity model- Section:** Model Namespace:
Eticaret_vtModel
- Buttons:** < Previous, Next >, Finish, Cancel

Entity Framework ile Veritabanı Modelleme

Database First



Entity Framework ile Veritabanı Modelleme

Model First

The screenshot shows the Entity Data Model Wizard and the Entity Explorer in Visual Studio.

Entity Data Model Wizard: This window is titled "Entity Data Model Wizard" and "Choose Model Contents". It asks "What should the model contain?" with four options: "EF Designer from database", "Empty EF Designer model" (which is selected), "Empty Code First model", and "Code First from database". Below the options is a description: "Creates an empty model in the EF Designer as a starting point for visually designing your model. Later, you can generate a database from your model. The classes your application will interact with are generated from the model." At the bottom are buttons for "< Previous", "Next >", "Finish", and "Cancel".

Entity Explorer: This window is titled "Model1.edmx [Diagram1]*" and "model_first: Overview". It shows a context menu with the following items: "Add New" (with "Diagram", "Zoom", "Grid", "Scalar Property Format", "Select All"), "Validate", "Update Model from Database...", "Generate Database from Model...", "Add Code Generation Item...", "Mapping Details", "Model Browser", and "Properties".

Entity Framework ile Veritabanı Modelleme

Model First

The screenshot shows the Entity Framework Model First interface. On the left, a modal dialog titled "Add Entity" is open, allowing the creation of a new entity type named "ogrenciler". The "Key Property" section is configured with a primary key named "ogno" of type Int32. On the right, the "ogrenciler" entity is selected in the model browser, and a context menu is open, displaying options like "Add New", "Refactor", "Rename", and "Entity Key". A secondary menu on the right lists "Scalar Property", "Navigation Property", and "Complex Property".

Add Entity

Properties

Entity name: ogrenciler

Base type: (None)

Entity Set: ogrenciler

Key Property

Create key property

Property name: ogno

Property type: Int32

OK Cancel

ogrenciler

Properties

ogno

Navigation

Add New

Refactor

Rename

Convert to Enum

Cut Ctrl+X

Copy Ctrl+C

Paste Ctrl+V

Delete from Model Del

Entity Key

Move Properties

Validate

Update Model from Database...

Generate Database from Model...

Add Code Generation Item...

Table Mapping

Stored Procedure Mapping

Show in Model Browser

Properties Alt+Enter

Scalar Property

Navigation Property

Complex Property

Entity Framework ile Veritabanı Modelleme

Model First

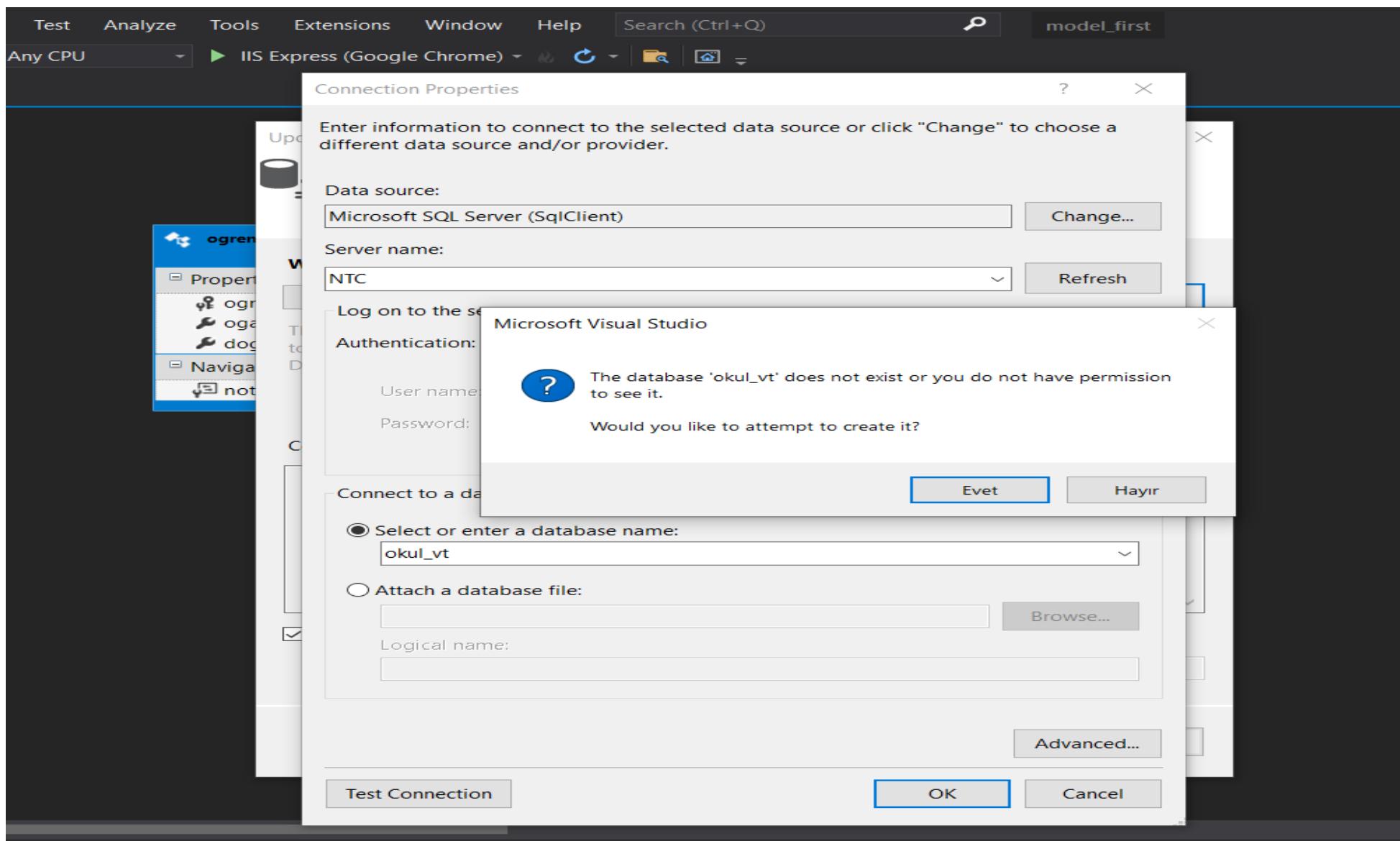
The screenshot shows the Entity Framework Model First interface in Visual Studio. On the left, the Solution Explorer displays two entities: 'ogrenciler' and 'notlar'. The 'Properties' window is open for the 'ogrenciler' entity, specifically for the 'dogtar' navigation property. The 'Type' dropdown is set to 'String'. A context menu is open over the 'ogrenciler' entity, with 'Entity...' selected. Other options in the menu include 'Association...', 'Inheritance...', 'Complex Type...', 'Enum Type...', and 'Function Import...'. The 'notlar' entity is also visible in the background.

Entity Framework ile Veritabanı Modelleme

Model First

The screenshot shows the Entity Framework Model First interface. On the left, the 'ogrenciler' entity is selected, displaying its properties (ogno, adsoyad, dogtar) and navigation properties (notlar). A context menu is open at the bottom right, with 'Generate Database from Model...' highlighted. In the center, the 'Add Association' dialog is open, showing the configuration for a new association named 'ogrencilernotlar'. The 'End' section for 'ogrenciler' has 'Entity: ogrenciler', 'Multiplicity: 1 (One)', and 'Navigation Property: notlar'. The 'End' section for 'notlar' has 'Entity: notlar', 'Multiplicity: * (Many)', and 'Navigation Property: ogrenciler'. A checkbox 'Add foreign key properties to the 'notlar' Entity' is checked. Below the dialog, a tooltip provides information about the generated navigation properties. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

Entity Framework ile Veritabanı Modelleme



Entity Framework ile Veritabanı Modelleme

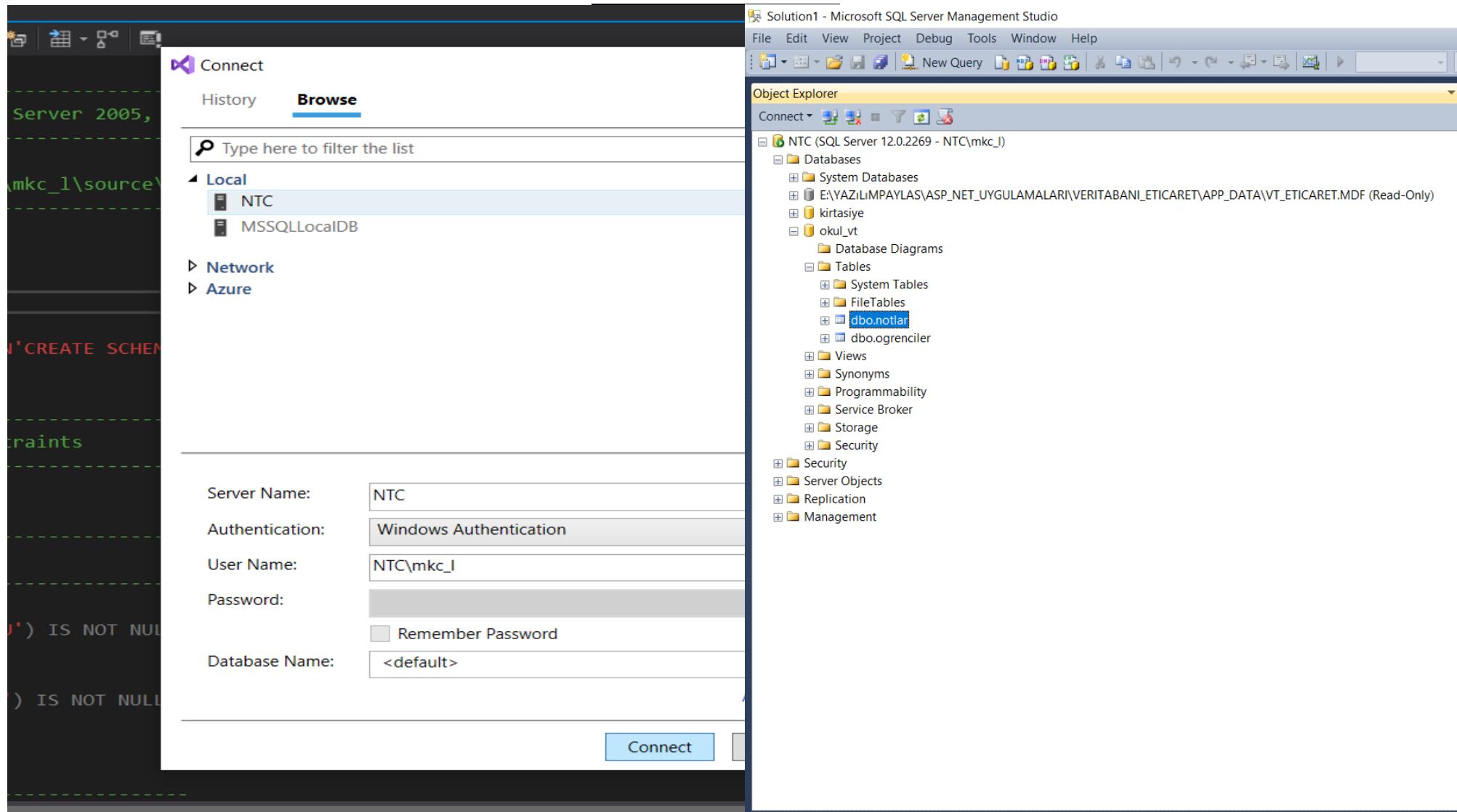
Model First

The screenshot shows the Visual Studio IDE interface. On the left, the 'Generate Database Wizard' window is open, displaying the 'Summary and Settings' page. It includes a 'Save DDL As:' dropdown set to 'Models\Model1.edmx.sql'. Below it is the 'DDL' tab containing the Entity Designer DDL Script. The script is a SQL script generated from the EDMX file, starting with `SET QUOTED_IDENTIFIER OFF;` and `GO`. It creates a schema named 'dbo' if it doesn't exist, drops existing FOREIGN KEY constraints, tables, and then drops the 'ogrenciler' and 'notlarSet' tables if they exist. On the right, the 'Model1.edmx [Diagram1]*' designer view is shown, with the Entity Designer DDL Script for SQL Server 2005, 2008, 2012 and Azure. A context menu is open over the script, showing options like 'Execute', 'Execute With Debugger', and 'Parse'. The status bar at the bottom indicates 'IIS Express (Google Chrome)'.

```
-- Entity Designer DDL Script for SQL Server 2005, 2008, 2012 and Azure  
-- Date Created: 06/16/2022 11:25:10  
-- Generated from EDMX file: C:\Users\mkc_1\source\repos\WebApplication1\WebApplication1\Models\Model1.edmx  
  
SET QUOTED_IDENTIFIER OFF;  
GO  
USE [okull_vt];  
GO  
IF SCHEMA_ID(N'dbo') IS NULL EXECUTE(N'CREATE SCHEMA [dbo]');  
GO  
  
-- Dropping existing FOREIGN KEY constraints  
  
-- Dropping existing tables  
  
IF OBJECT_ID(N'[dbo].[ogrenciler]', 'U') IS NOT NULL  
DROP TABLE [dbo].[ogrenciler];  
GO  
IF OBJECT_ID(N'[dbo].[notlarSet]', 'U') IS NOT NULL  
DROP TABLE [dbo].[notlarSet];  
GO
```

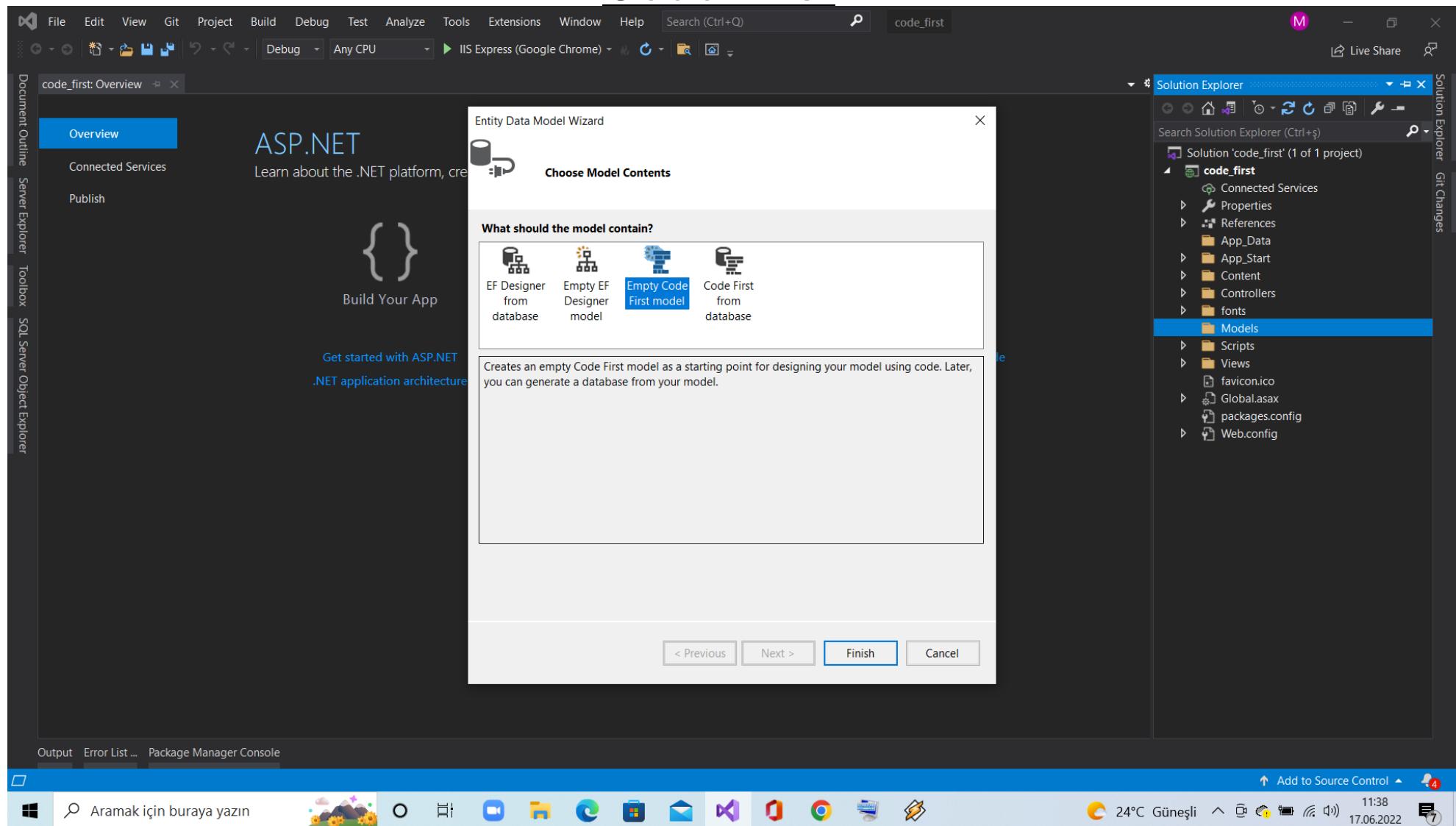
Entity Framework ile Veritabanı Modelleme

Model First



Entity Framework ile Veritabanı Modelleme

Code First



Entity Framework ile Veritabanı Modelleme

Code First

```
13
14     [Key]
15     public int ogno { set; get; }
16     [Required(ErrorMessage ="ad soyad girmek zorludur")]
17     [StringLength(30,ErrorMessage ="30 karakterden fazla olamaz")]
18     public string ad_soyad { set; get; }
19     // [Column(TypeName = "smalldatetime")]
20     [DisplayFormat(ApplyFormatInEditMode = true, DataFormatString = "{0:dd/MM/yyyy}")]
21     [Required(ErrorMessage="tarih giriniz")]
22     public DateTime dogtar { set; get; }
23     [Index(IsUnique = true)]
24     public long tckimlik { set; get; }
25     public string adres { set; get; }
26
27         public virtual ICollection<Not> notlar { set; get; }
28
29 }
```

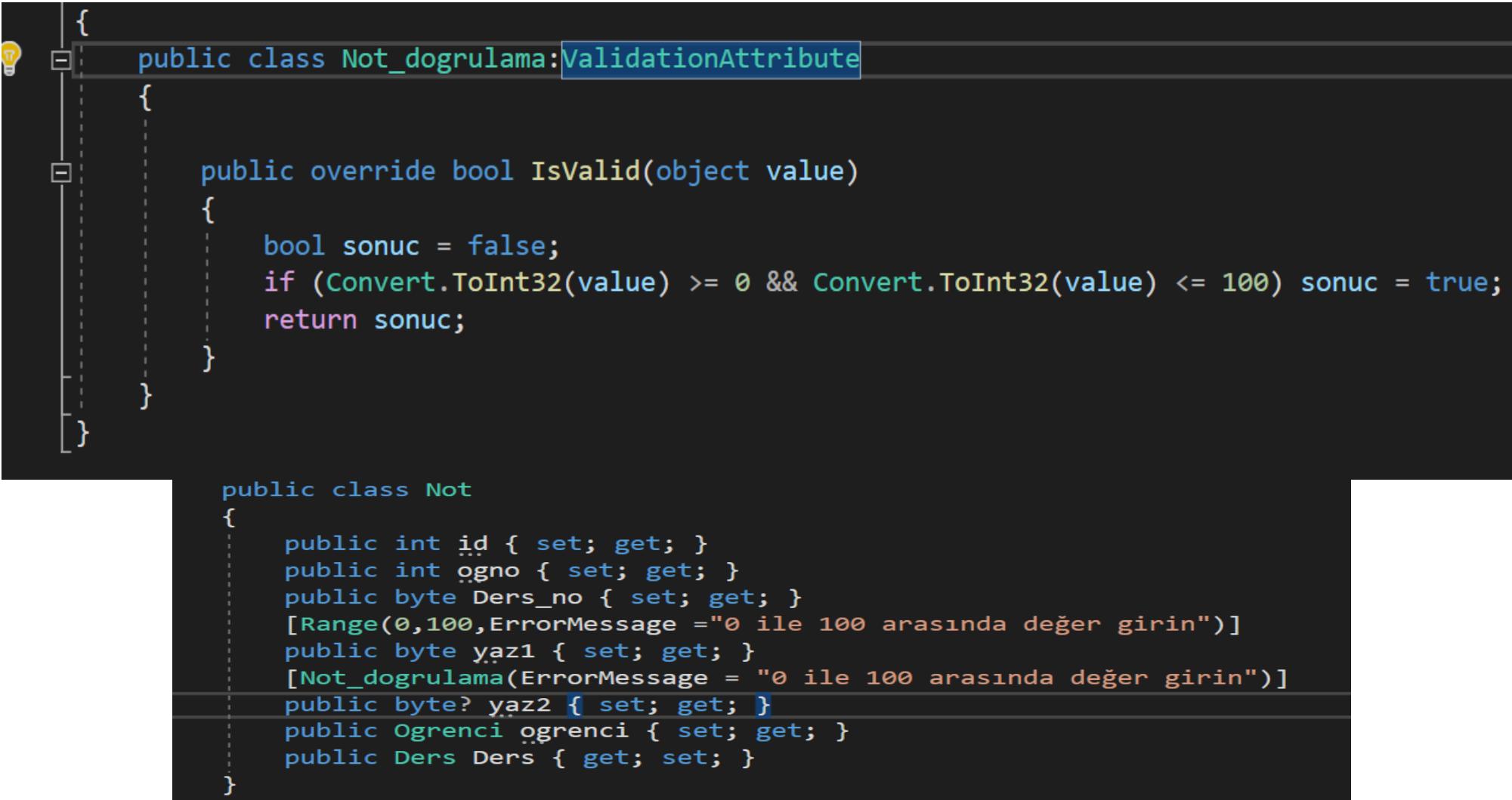
Doğrulama Kontrolleri

- using System.ComponentModel.DataAnnotations; kütüphanesi eklenmelidir

Required	Özellikin gerekli bir alan olduğunu belirtir.
StringLength	Dize alanı için maksimum uzunluk tanımlar
Range	Sayısal bir alan için maksimum ve minimum değer tanımlar
RegularExpression	Alan değerinin belirtilen Normal İfade ile eşleşmesi gerektiğini belirtir.
CreditCard	Belirtilen alanın bir kredi kartı numarası olduğunu belirtir.
CustomValidation	Alanı doğrulamak için belirtilen özel doğrulama yöntemi
EmailAddress	E-posta adresi formatıyla doğrulanır
FileExtension	Dosya uzantısıyla doğrulanır
MaxLength	Bir dize alanı için maksimum uzunluğu belirtir.
MinLength	Dize alanı için minimum uzunluk belirtir
Phone	Alanın telefon numaraları için normal ifadeyi kullanan bir telefon numarası olduğunu belirtir.

Doğrulama Kontrolleri

- Eğer kendimiz Custom validator yazmak istersek Model içinde bir class oluşturur ve validationAttribute den türetiriz.



```
public class Not_dogrulama : ValidationAttribute
{
    public override bool IsValid(object value)
    {
        bool sonuc = false;
        if (Convert.ToInt32(value) >= 0 && Convert.ToInt32(value) <= 100) sonuc = true;
        return sonuc;
    }
}

public class Not
{
    public int id { set; get; }
    public int ogno { set; get; }
    public byte Ders_no { set; get; }
    [Range(0,100,ErrorMessage ="0 ile 100 arasında değer girin")]
    public byte yazı1 { set; get; }
    [Not_dogrulama(ErrorMessage = "0 ile 100 arasında değer girin")]
    public byte? yazı2 { set; get; }
    public Ogrenci ogrenci { set; get; }
    public Ders Ders { get; set; }
}
```

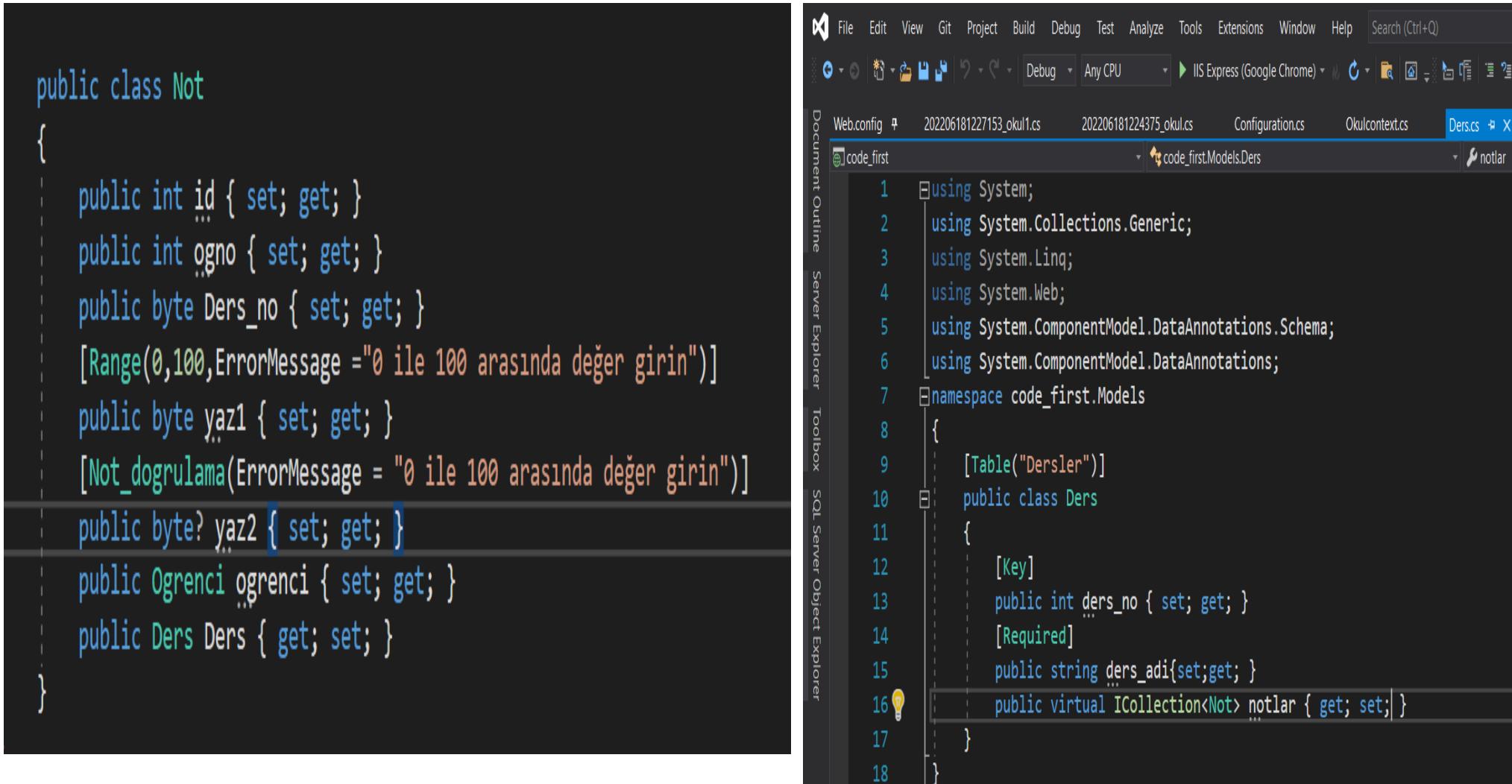
Doğrulama Kontrolleri(Örnek mesela uye tablomuz)

- Daha sonra model içerisinde ilgili sınıfımıza gideriz. ve alanlardan önce doğrulama tanımlamaları yapılır.

```
sehirler.cs      uye_kaydet.cshtml      uye.cs*  ✎ X Index.cshtml      HomeController.cs      login_giris.cshtml      _Layout.cshtml
|   this.siparisler = new HashSet<siparisler>();
|
|
public int uyeid { get; set; }
[Required(ErrorMessage="Kullanıcı Adı Zorunludur")]
public string kuladi { get; set; }
[Required(ErrorMessage ="Şifre zorunludur")]
[MinLength(8,ErrorMessage="şifre en az 8 karakter olamalı")]
public string sifre { get; set; }
[Required(ErrorMessage ="Ad soyad Zorunludur")]
public string adsoyad { get; set; }
[Required(ErrorMessage ="tarih girilmeli")]
[DataType(DataType.Date,ErrorMessage ="tarih bilgisi giriniz")]
public System.DateTime dogtar { get; set; }
[Required(ErrorMessage ="cinsiyet seçilmeli")]
public bool? cinsiyet { get; set; }
[Required(ErrorMessage ="Adres Seçiniz")]
public string adres { get; set; }
//plaka doğrulaması sehirler classında çünkü compositon var
public byte plaka { get; set; }
[EmailAddress(ErrorMessage ="Doğru email giriniz")]
public string email { get; set; }
//meslekid doğrulaması meslek classında çünkü compositon var
public short meslekid { get; set; }
public Nullable<bool> onay { get; set; }
```

Entity Framework ile Veritabanı Modelleme

Code First



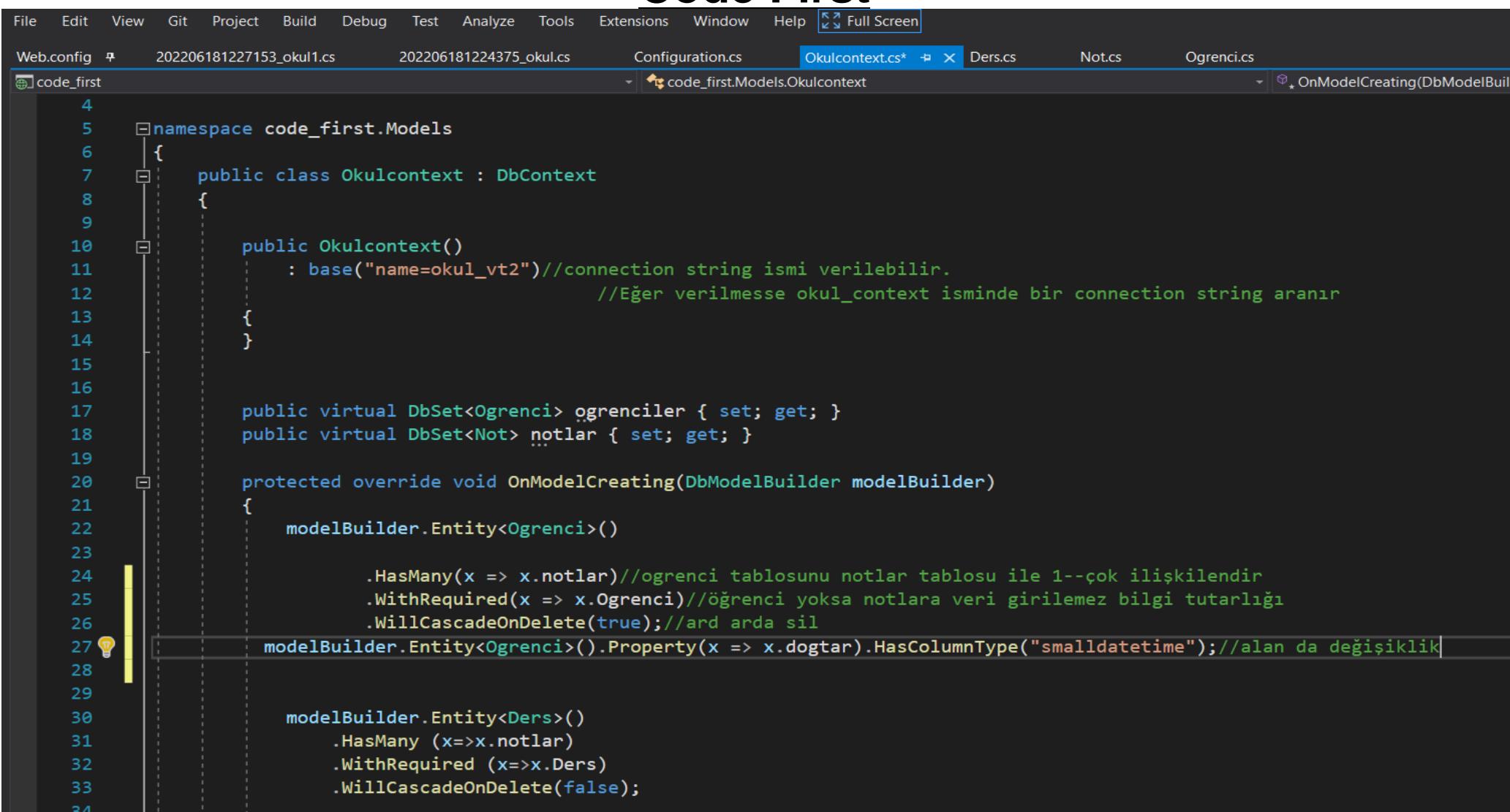
The image shows two side-by-side screenshots of the Visual Studio IDE. The left screenshot displays a code editor with C# code for a 'Not' class, which has properties for id, ogno, Ders_no, yaz1, yaz2, ogrenci, and Ders. The right screenshot shows the 'Ders.cs' file in the Solution Explorer, which contains the generated code for the 'Ders' class, including its properties and a navigation property named 'notlar'.

```
public class Not
{
    public int id { set; get; }
    public int ogno { set; get; }
    public byte Ders_no { set; get; }
    [Range(0,100,ErrorMessage ="0 ile 100 arasında değer girin")]
    public byte yaz1 { set; get; }
    [Not_dogrulama(ErrorMessage = "0 ile 100 arasında değer girin")]
    public byte? yaz2 { set; get; }
    public Ogrenci ogrenci { set; get; }
    public Ders Ders { get; set; }
}
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Web;
5  using System.ComponentModel.DataAnnotations.Schema;
6  using System.ComponentModel.DataAnnotations;
7  namespace code_first.Models
8  {
9      [Table("Dersler")]
10     public class Ders
11     {
12         [Key]
13         public int ders_no { set; get; }
14         [Required]
15         public string ders_adi{set;get; }
16         public virtual ICollection<Not> notlar { get; set; }
17     }
18 }
```

Entity Framework ile Veritabanı Modelleme

Code First



```
File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Full Screen
Web.config 202206181227153_okul1.cs 202206181224375_okul.cs Configuration.cs Okulcontext.cs* Ders.cs Not.cs Ogrenci.cs
code_first
4
5     namespace code_first.Models
6     {
7         public class Okulcontext : DbContext
8         {
9
10             public Okulcontext()
11                 : base("name=okul_vt2") //connection string ismi verilebilir.
12                     //Eğer verilmemesse okul_context isminde bir connection string aranır
13             {
14             }
15
16
17             public virtual DbSet<Ogrenci> ogrenciler { set; get; }
18             public virtual DbSet<Not> notlar { set; get; }
19
20             protected override void OnModelCreating(DbModelBuilder modelBuilder)
21             {
22                 modelBuilder.Entity<Ogrenci>()
23
24                     .HasMany(x => x.notlar) //ogrenci tablosunu notlar tablosu ile 1--çok ilişkilendir
25                     .WithRequired(x => x.Ogrenci) //öğrenci yoksa notlara veri girilemez bilgi tutarlığı
26                     .WillCascadeOnDelete(true); //ard arda sil
27                 modelBuilder.Entity<Ogrenci>().Property(x => x.dogtar).HasColumnType("smalldatetime"); //alan da değişiklik
28
29
30                 modelBuilder.Entity<Ders>()
31                     .HasMany (x=>x.notlar)
32                     .WithRequired (x=>x.Ders)
33                     .WillCascadeOnDelete(false);
34 }
```

Entity Framework ile Veritabanı Modelleme

Code First

Tasarımızı tanımladıktan sonra

- Web config içerisinde <connectionStrings> içerisine

```
<add name="okul_vt" connectionString="data source=NTC;  
AttachDbFilename=|DataDirectory|\okul_vt.mdf;integrated security=True;  
MultipleActiveResultSets=True;App=EntityFramework" providerName="System.Data.SqlClient" />
```

Veya

```
<add name="okul_vt2" connectionString="data source=NTC;initial catalog=okul_vt;integrated  
security=True;MultipleActiveResultSets=True;App=EntityFramework"  
providerName="System.Data.SqlClient" />
```

ekliyoruz

- Package Manager Console açıyoruz.
- Enable-Migrations → .migrations klasörü config oluştur
- Add-Migration okul → okul.cs(Create cümleleri oluştur)
- Update-database ile veritabanımız oluşur.

Alan eklemek veya güncellemek istersek

- Add-Migration okul2 ekliyoruz
- Update-database

Veritabanı Üzerinde “CRUD” İşlemleri

○ Read İşlemi

The screenshot shows the Microsoft Visual Studio IDE interface. On the left, the Solution Explorer displays files like Web.config, OgrenciController.cs, 202206181350574_okul11.cs, and Configuration.cs. The OgrenciController.cs file is open in the main editor, showing C# code for an MVC controller. A context menu is open over the Index() action method, with the 'Add View' option selected. A 'Add View' dialog box is displayed, prompting for a 'View name' (set to 'Index'), 'Template' (set to 'List'), 'Model class' (set to 'Ogrenci (code_first.Models)'), and 'Data context class' (set to 'Okulcontext (code_first.Models)'). Other options like 'Create as a partial view' and 'Reference script libraries' are checked. At the bottom right of the dialog are 'Add' and 'Cancel' buttons.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Web;
5  using System.Web.Mvc;
6  using code_first.Models;
7  namespace code_first.Controllers
8  {
9      public class OgrenciController : Controller
10     {
11         Okulcontext db = new Okulcontext();
12
13         // GET: Ogrenci
14         public ActionResult Index()
15         {
16             var ogrenciler = db.ogrenciler.ToList();
17             return View(ogrenciler);
18         }
19
20         public ActionResult Create()
21         {
22             return View();
23         }
24     }
25 }
```

Veritabanı Üzerinde “CRUD” İşlemleri

○ Read İşlemi

```
File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Full Screen
```

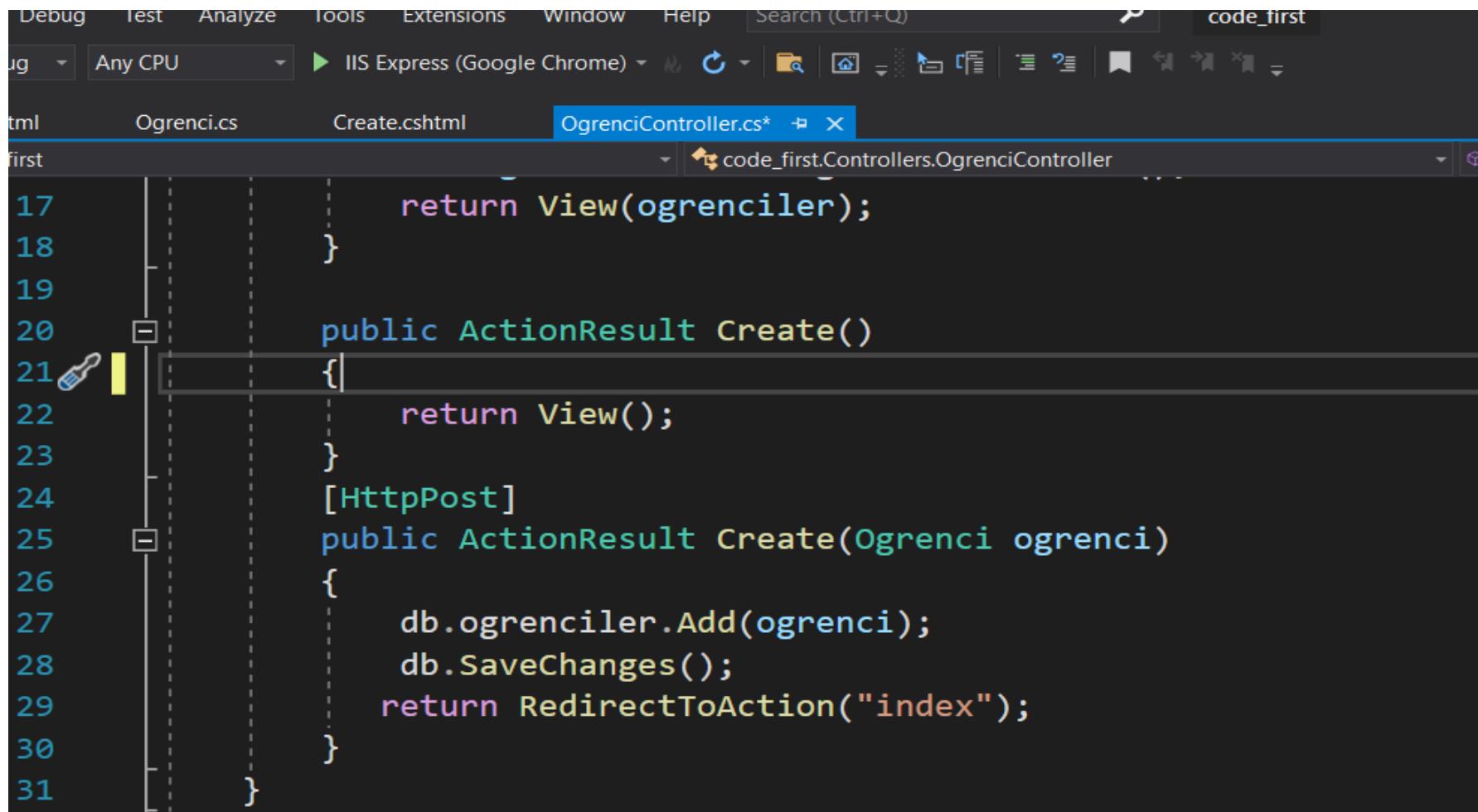
```
Web.config OgrenciController.cs 202206181350574_oku11.cs Configuration.cs Okul
```

```
1 @model IEnumerable<code_first.Models.Ogrenci>
2
3 @{
4     Layout = null;
5 }
6 <!DOCTYPE html>
7 <html>
8     <head>
9         <meta name="viewport" content="width=device-width" />
10        <title>Index</title>
11    </head>
12    <body>
13        <p>
14            @Html.ActionLink("Create New", "Create")
15        </p>
16        <table class="table">
17            <tr>
18                <th>
19                    @Html.DisplayNameFor(model => model.ad_soyad)
20                </th>
21                <th>
22                    @Html.DisplayNameFor(model => model.dogtar)
23                </th>
24                <th>
25                    @Html.DisplayNameFor(model => model.tckimlik)
26                </th>
27                <th>
28                    @Html.DisplayNameFor(model => model.adres)
29                </th>
30                <th></th>
31            </tr>
32            @foreach (var item in Model) {
33                <tr>
34                    <td>
35                        @Html.DisplayFor(modelItem => item.ad_soyad)
36                    </td>
37                    <td>
38                        @Html.DisplayFor(modelItem => item.dogtar)
39                    </td>
40                    <td>
41                        @Html.DisplayFor(modelItem => item.tckimlik)
42                    </td>
43                    <td>
44                        @Html.DisplayFor(modelItem => item.adres)
45                    </td>
46                    <td>
47                        @Html.ActionLink("Edit", "Edit", new { id=item.ogno }) ||
48                        @Html.ActionLink("Delete", "Delete", new { id=item.ogno })
49                    </td>
50                </tr>
51            }
52        </table>
53    </body>
54 </html>
```

```
Web.config OgrenciController.cs 202206181350574_oku11.cs Configuration.cs Okulcontext.cs
```

Veritabanı Üzerinde “CRUD” İşlemleri

○ Create İşlemi



The screenshot shows the Microsoft Visual Studio IDE interface. The title bar includes "Debug", "Test", "Analyze", "Tools", "Extensions", "Window", "Help", and a search bar. Below the title bar, there are dropdown menus for "Any CPU" and "IIS Express (Google Chrome)". The main window displays the code for the "Create" action in the "OgrenciController.cs" file. The code uses Entity Framework's "code first" approach to manage the database.

```
16     public ActionResult Index()
17     {
18         return View(ogrenciler);
19     }
20
21     public ActionResult Create()
22     {
23         return View();
24     }
25
26     [HttpPost]
27     public ActionResult Create(Ogrenci ogrenci)
28     {
29         db.ogrenciler.Add(ogrenci);
30         db.SaveChanges();
31         return RedirectToAction("index");
32     }
33 }
```

Veritabanı Üzerinde “CRUD” İşlemleri

○ Create İşlemi(Kendimiz Form Hazırlayabiliriz)

The screenshot shows the Visual Studio IDE with two windows. On the left, the code editor displays the `Create.cshtml` file, which contains the following HTML and C# code:

```
@{  
    Layout = null;  
}  
  
<!DOCTYPE html>  
  
<html>  
<head>  
    <meta name="viewport" content="width=device-width" />  
    <title>Create</title>  
</head>  
<body>  
    <div>  
        <table>  
            @using (Html.BeginForm())  
            {  
                <tr><td>Ad Soyad:</td><td>@Html.TextBox("ad_soyad")</td></tr>  
                <tr><td>Doğum tarihi:</td><td>@Html.TextBox("dogtar")</td></tr>  
                <tr><td>Tc:</td><td>@Html.TextBox("tckimlik")</td></tr>  
                <tr><td>adres:</td><td>@Html.TextArea("adres", "", 10, 20, new { })</td></tr>  
                <tr><td><input id="Submit1" type="submit" value="Kaydet" /></td></tr>  
            }  
        </table>  
    </div>  
</body>
```

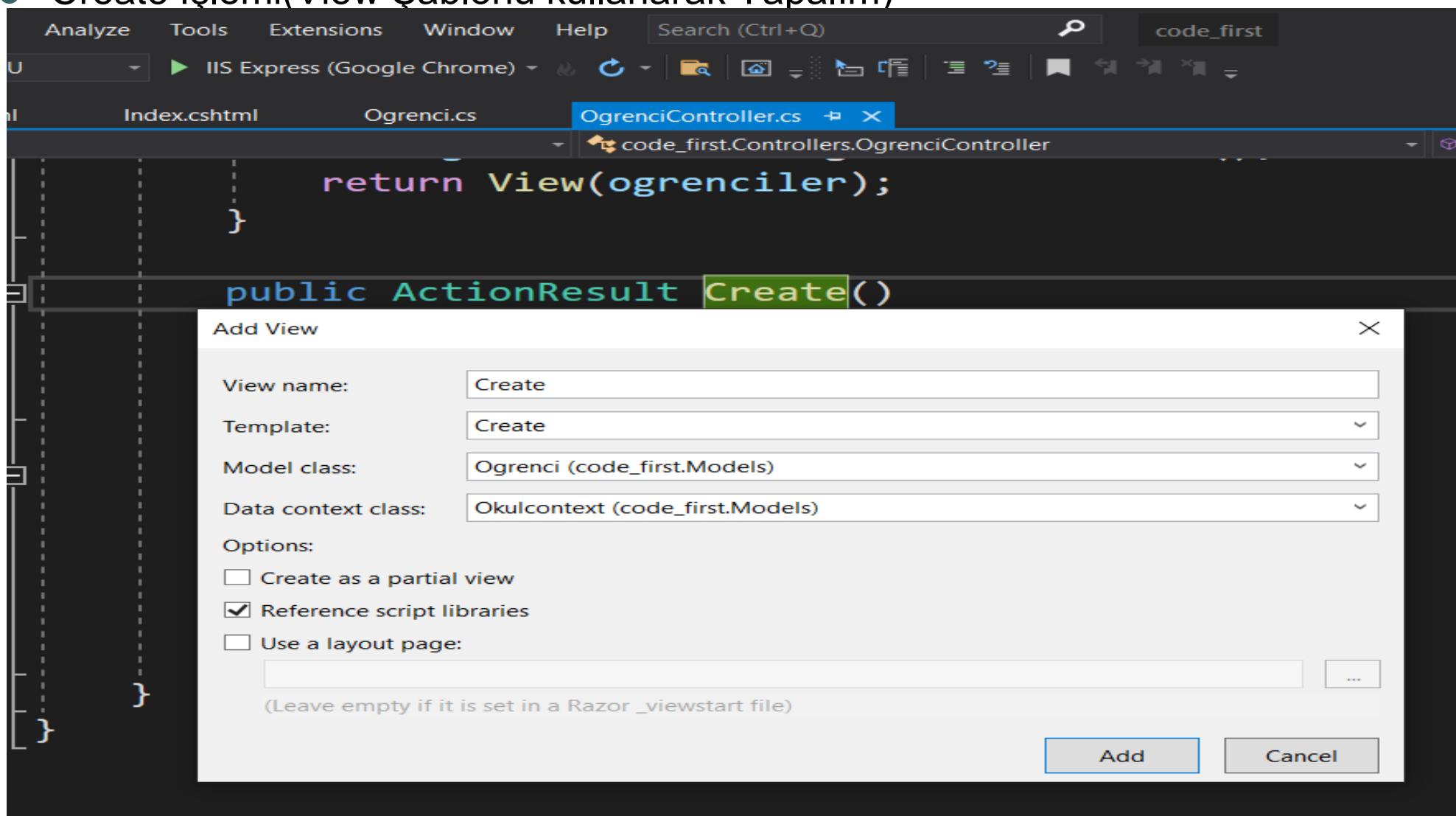
On the right, a browser window shows the resulting form. The URL is `localhost:44365/Ogrenci/Create`. The form fields are populated with the following values:

- Ad Soyad: fatma
- Doğum tarihi: 08/06/2000
- Tc: 136523
- adres: bursa

A **Kaydet** button is visible at the bottom of the form.

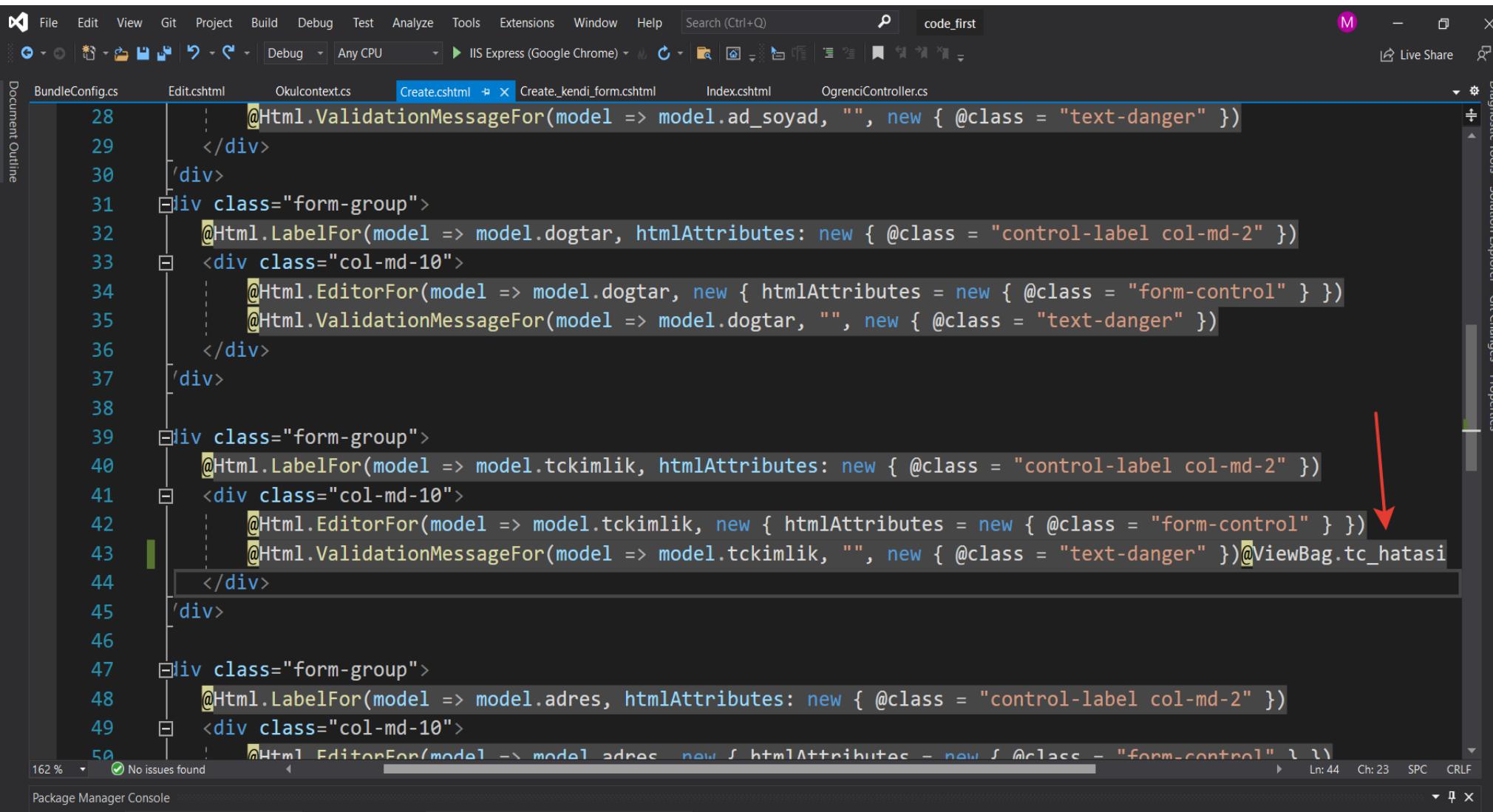
Veritabanı Üzerinde “CRUD” İşlemleri

- Create İşlemi(View Şablonu kullanarak Yapalım)



Veritabanı Üzerinde “CRUD” İşlemleri

○ Create İşlemi(View Şablonu kullanarak Yapalım)

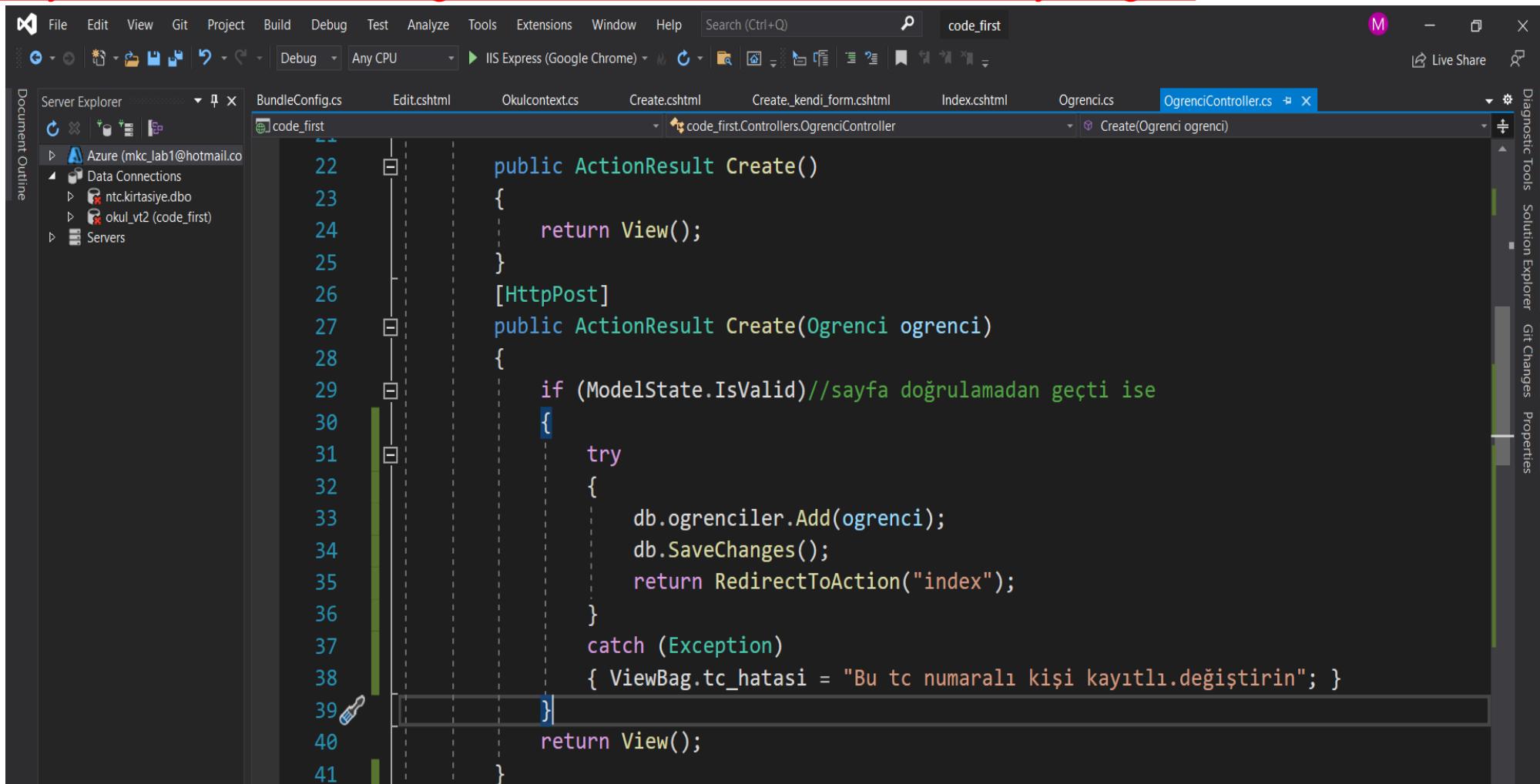


```
File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) code_first
M - Live Share Diagnostic tools Solution Explorer Git Changes Properties
BundleConfig.cs Edit.cshtml Okulcontext.cs Create.cshtml Create._kendi_form.cshtml Index.cshtml OgrenciController.cs
28     |     @Html.ValidationMessageFor(model => model.ad_soyad, "", new { @class = "text-danger" })
29     |   </div>
30   'div>
31   |<div class="form-group">
32     |   @Html.LabelFor(model => model.dogtar, htmlAttributes: new { @class = "control-label col-md-2" })
33   |<div class="col-md-10">
34     |       @Html.EditorFor(model => model.dogtar, new { htmlAttributes = new { @class = "form-control" } })
35     |       @Html.ValidationMessageFor(model => model.dogtar, "", new { @class = "text-danger" })
36   |</div>
37 '</div>
38
39 |<div class="form-group">
40   |   @Html.LabelFor(model => model.tckimlik, htmlAttributes: new { @class = "control-label col-md-2" })
41 |<div class="col-md-10">
42   |       @Html.EditorFor(model => model.tckimlik, new { htmlAttributes = new { @class = "form-control" } })
43   |       @Html.ValidationMessageFor(model => model.tckimlik, "", new { @class = "text-danger" })@ViewBag.tc_hatasi
44 |</div>
45 '</div>
46
47 |<div class="form-group">
48   |   @Html.LabelFor(model => model.adres, htmlAttributes: new { @class = "control-label col-md-2" })
49 |<div class="col-md-10">
50   |       @Html.EditorFor(model => model.adres, new { htmlAttributes = new { @class = "form-control" } })
```

Veritabanı Üzerinde “CRUD” İşlemleri

- Create İşlemi(View Şablonu kullanarak Yapalım)

Kayıt Action nımızda doğrulama kontrolü kodlarını ekleyeceğiz..



The screenshot shows the Microsoft Visual Studio IDE interface. The title bar says "code_first". The menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and a search bar. The toolbar has various icons for file operations. The Solution Explorer on the left shows a project named "code_first" with files like BundleConfig.cs, Edit.cshtml, Okulcontext.cs, Create.cshtml, Create_kendi_form.cshtml, Index.cshtml, Ogrenci.cs, and OgrenciController.cs. The OgrenciController.cs file is open in the code editor, showing the following C# code:

```
public ActionResult Create()
{
    return View();
}

[HttpPost]
public ActionResult Create(Ogrenci ogrenci)
{
    if (ModelState.IsValid) //sayfa doğrulamadan geçti ise
    {
        try
        {
            db.ogrenciler.Add(ogrenci);
            db.SaveChanges();
            return RedirectToAction("index");
        }
        catch (Exception)
        {
            ViewBag.tc_hatası = "Bu tc numaralı kişi kayıtlı.değiştirin"; 
        }
    }
    return View();
}
```

Veritabanı Üzerinde “CRUD” İşlemleri

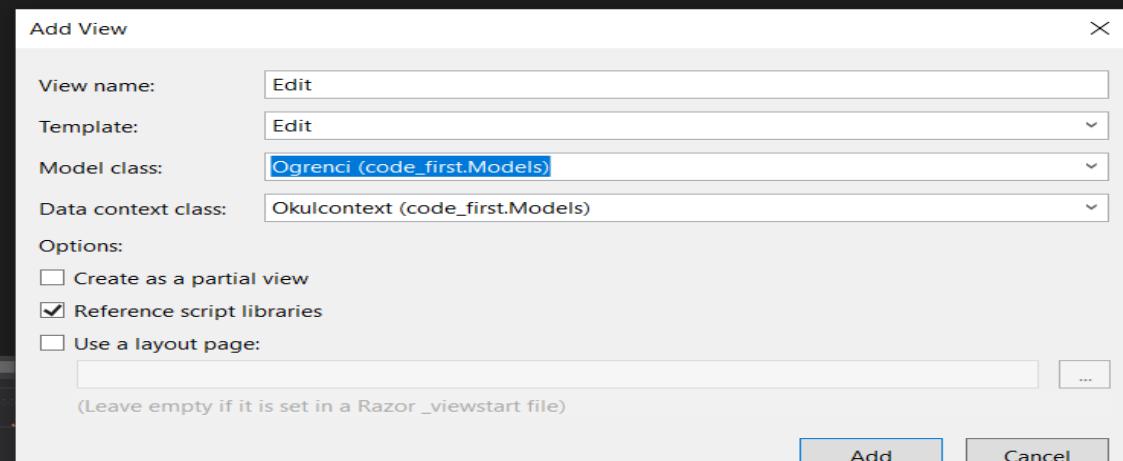
- Delete İşlemi

```
}

public ActionResult Delete(int id)//id tablodaki key olarak düşünür her zaman mvc
{
    var ogrenci = db.ogrenciler.Find(id);
    db.ogrenciler.Remove(ogrenci);
    db.SaveChanges();
    return RedirectToAction("index");
```

Veritabanı Üzerinde “CRUD” İşlemleri

○ Update İşlemi



The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes Analyze, Tools, Extensions, Window, Help, and a search bar. The title bar shows "IIS Express (Google Chrome)". The code editor displays C# code for an MVC application. The current file is "OgrenciController.cs". The code contains two methods: "Delete" and "Edit". The "Edit" method retrieves an "Ogrenci" object from the database using its ID and returns a view of the object.

```
public ActionResult Delete(int id) //id tablodaki key olarak düşünür her zaman mvc
{
    var ogrenci = db.ogrenciler.Find(id);
    db.ogrenciler.Remove(ogrenci);
    db.SaveChanges();
    return RedirectToAction("index");
}

public ActionResult Edit(int id)
{
    var ogrenci = db.ogrenciler.Find(id);
    return View (ogrenci);
}
```

A modal dialog titled "Add View" is open in the foreground. It has fields for "View name:" (set to "Edit"), "Template:" (set to "Edit"), "Model class:" (set to "Ogrenci (code_first.Models)"), and "Data context class:" (set to "Okulcontext (code_first.Models)"). Under "Options:", there are three checkboxes: "Create as a partial view" (unchecked), "Reference script libraries" (checked), and "Use a layout page:" (unchecked). A note at the bottom says "(Leave empty if it is set in a Razor _viewstart file)". At the bottom right of the dialog are "Add" and "Cancel" buttons.

Veritabanı Üzerinde “CRUD” İşlemleri

○ Update İşlemi

The screenshot shows the Microsoft Visual Studio IDE interface. The title bar includes 'Build', 'Debug', 'Test', 'Analyze', 'Tools', 'Extensions', 'Window', 'Help', 'Search (Ctrl+Q)', and 'code_first'. The toolbar has icons for file operations like Open, Save, Print, and Find. The menu bar shows 'Debug' (selected), 'Any CPU', 'IIS Express (Google Chrome)', and 'Live Share'. The code editor window displays 'OgrenciController.cs' under 'code_first.Controllers.OgrenciController'. The code implements an 'Edit' action method:

```
50
51     public ActionResult Edit(int id)
52     {
53         var ogrenci = db.ogrenciler.Find(id);
54         return View(ogrenci);
55     }
56     [HttpPost]
57     public ActionResult Edit(Ogrenci ogrenci)
58     {
59         if (ModelState.IsValid) //sayfa doğrulamadan geçtiyse
60         {
61             try
62             {
63                 db.Entry(ogrenci).State = EntityState.Modified;
64                 db.SaveChanges();
65                 return RedirectToAction("index");
66             }
67             catch (Exception)
68             { ViewBag.tc_hatasi = "Bu tc numaralı kişi kayıtlı.değiştirin"; }
69         }
70         return View();
71     }
72 }
73 }
74 }
```

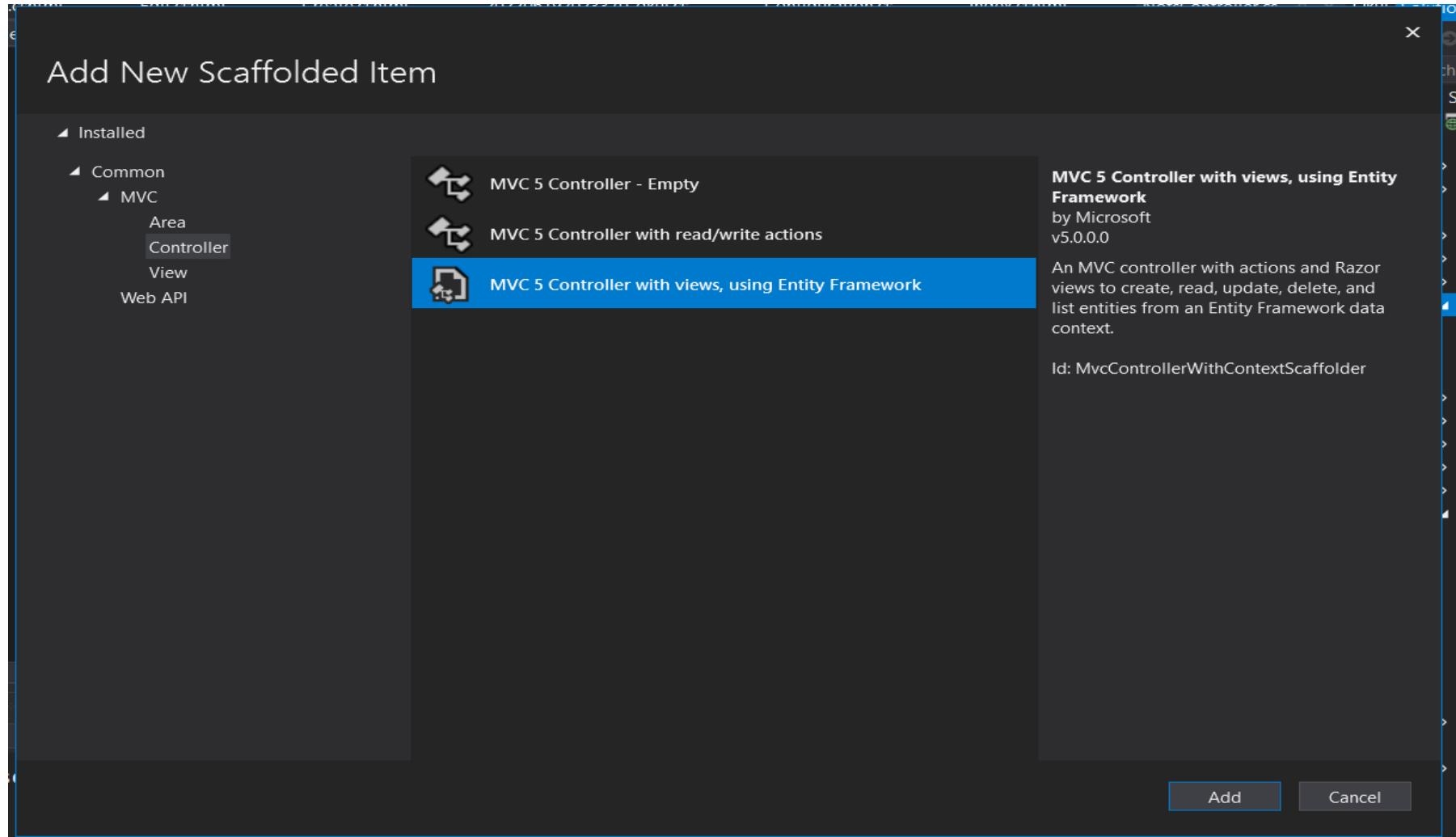
The 'Properties' tab is visible on the right side of the interface.

Veritabanı Üzerinde “CRUD” İşlemleri

- Dersler Tablosu için Gerekli Action ve View leri siz hazırlayınız.

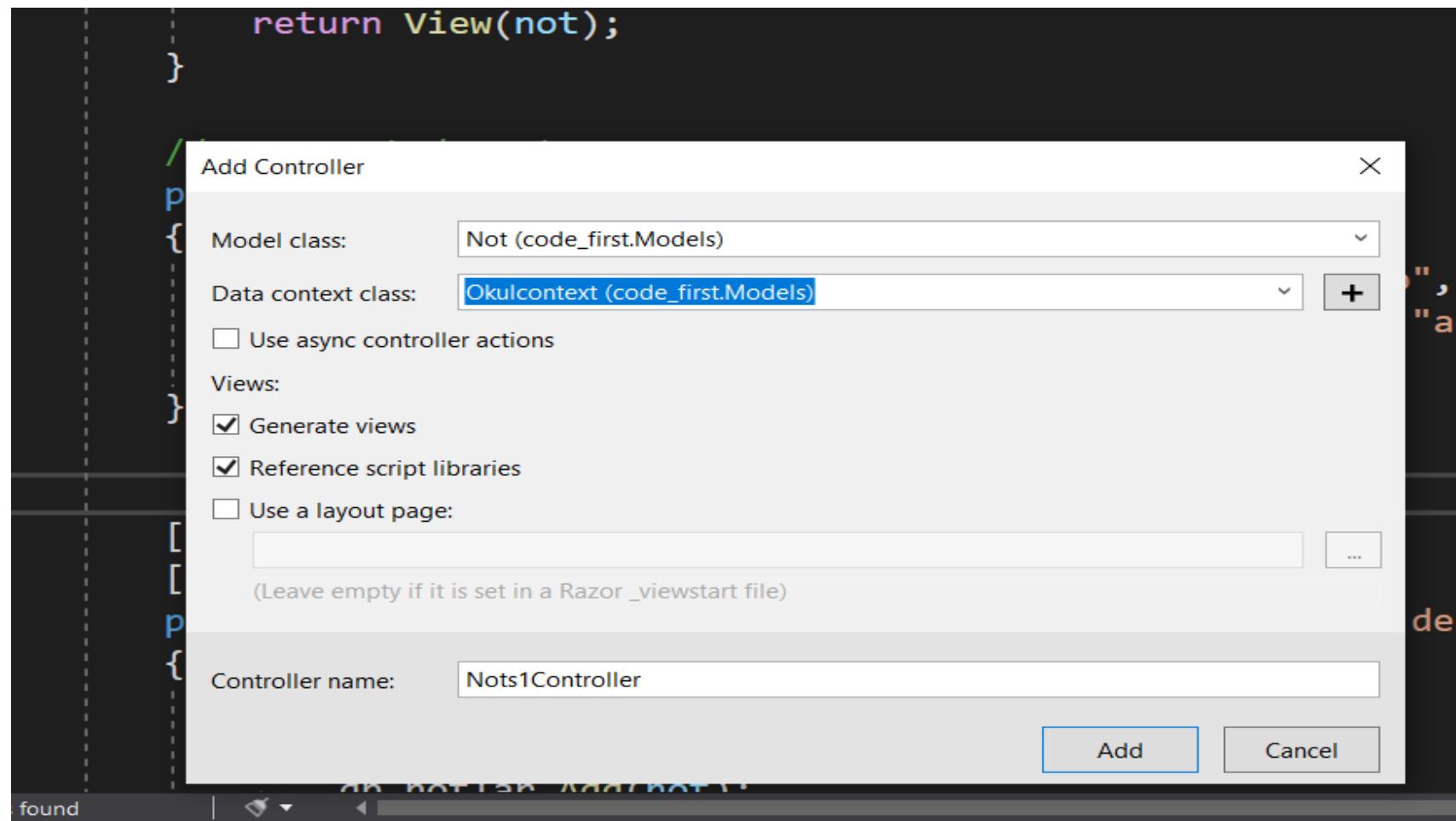
Veritabanı Üzerinde “CRUD” İşlemleri

Birden Fazla Tablodan Veri Çekme ve sihirbazla Crud Action Ve Viewlar Oluşturma (Notlar Tablosu)



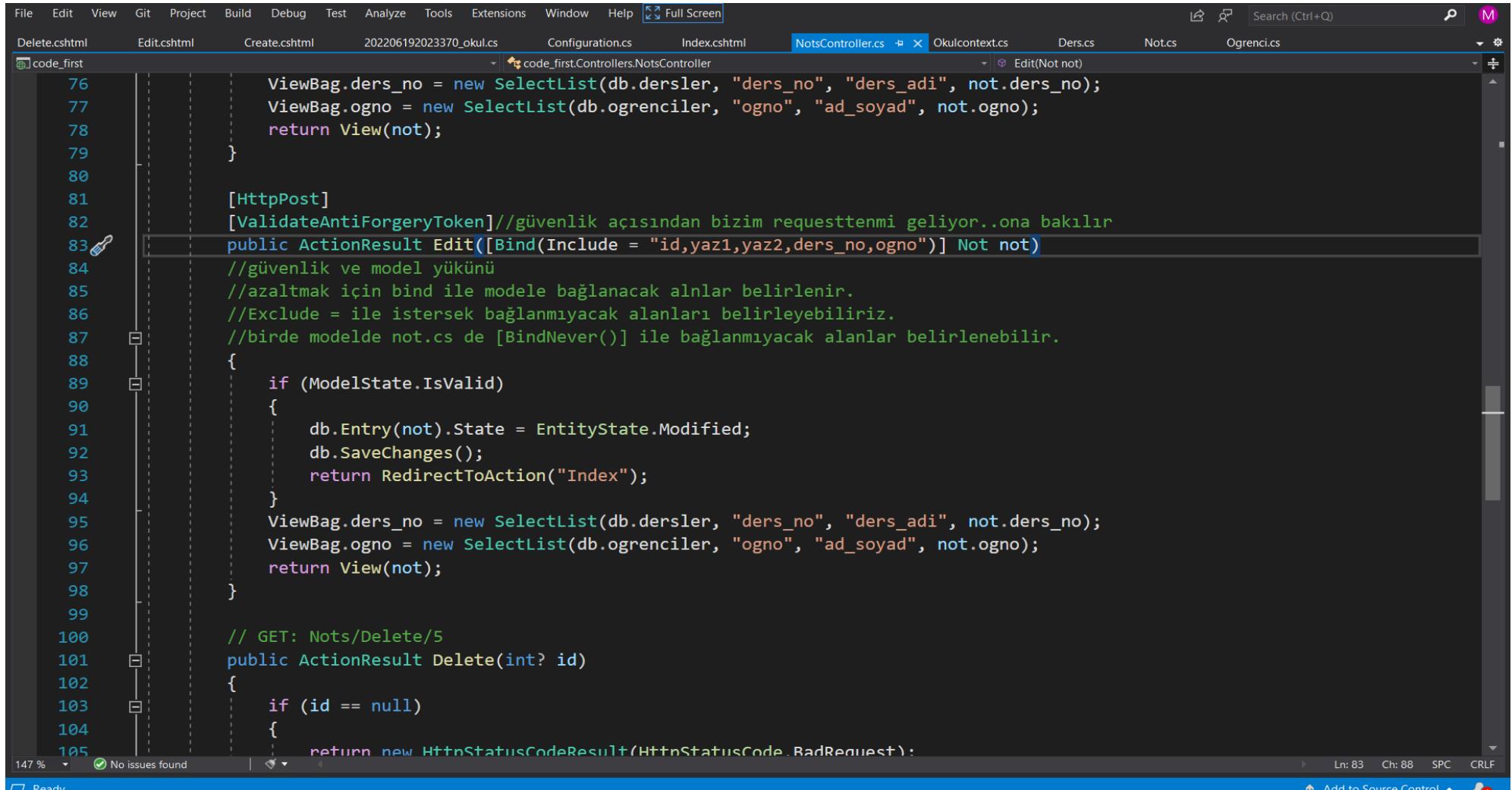
Veritabanı Üzerinde “CRUD” İşlemleri

Birden Fazla Tablodan Veri Çekme ve sihirbazla Crud Action Ve Viewlar Oluşturma (Notlar Tablosu)



Veritabanı Üzerinde “CRUD” İşlemleri

Birden Fazla Tablodan Veri Çekme ve sihirbazla Crud Action Ve Viewlar Oluşturma (Notlar Tablosu)



The screenshot shows the Microsoft Visual Studio IDE interface. The title bar includes standard menu items like File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Full Screen. Below the title bar, several tabs are visible: Delete.cshtml, Edit.cshtml, Create.cshtml, 202206192023370_okul.cs, Configuration.cs, Index.cshtml, NotsController.cs (selected), Okulcontext.cs, Ders.cs, Not.cs, and Ogrenci.cs. The main code editor area displays C# code for a controller named NotsController.cs. The code implements CRUD operations for a 'Not' entity, which is associated with 'Ders' and 'Ogrenci' entities. It uses ViewBag to pass dropdown lists for selecting course and student IDs. The code includes annotations for HttpPost, ValidateAntiForgeryToken, and Bind. A cursor is positioned over the 'Edit' action method. The status bar at the bottom shows '147 %' zoom, 'No issues found', and other development details.

```
76     ViewBag.ders_no = new SelectList(db.dersler, "ders_no", "ders_adi", not.ders_no);
77     ViewBag.ogno = new SelectList(db.ogrenciler, "ogno", "ad_soyad", not.ogno);
78     return View(not);
79 }
80
81 [HttpPost]
82 [ValidateAntiForgeryToken] //güvenlik açısından bizim requestten mi geliyor.. ona bakılır
83 public ActionResult Edit([Bind(Include = "id,yaz1,yaz2,ders_no,ogno")] Not not)
84     //güvenlik ve model yükünü
85     //azaltmak için bind ile modele bağlanacak alnlar belirlenir.
86     //Exclude = ile istersek bağlanmamışacak alanları belirleyebiliriz.
87     //birde modelde not.cs de [BindNever()] ile bağlanmamışacak alanlar belirlenebilir.
88 {
89     if (ModelState.IsValid)
90     {
91         db.Entry(not).State = EntityState.Modified;
92         db.SaveChanges();
93         return RedirectToAction("Index");
94     }
95     ViewBag.ders_no = new SelectList(db.dersler, "ders_no", "ders_adi", not.ders_no);
96     ViewBag.ogno = new SelectList(db.ogrenciler, "ogno", "ad_soyad", not.ogno);
97     return View(not);
98 }
99
100 // GET: Nots/Delete/5
101 public ActionResult Delete(int? id)
102 {
103     if (id == null)
104     {
105         return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
```

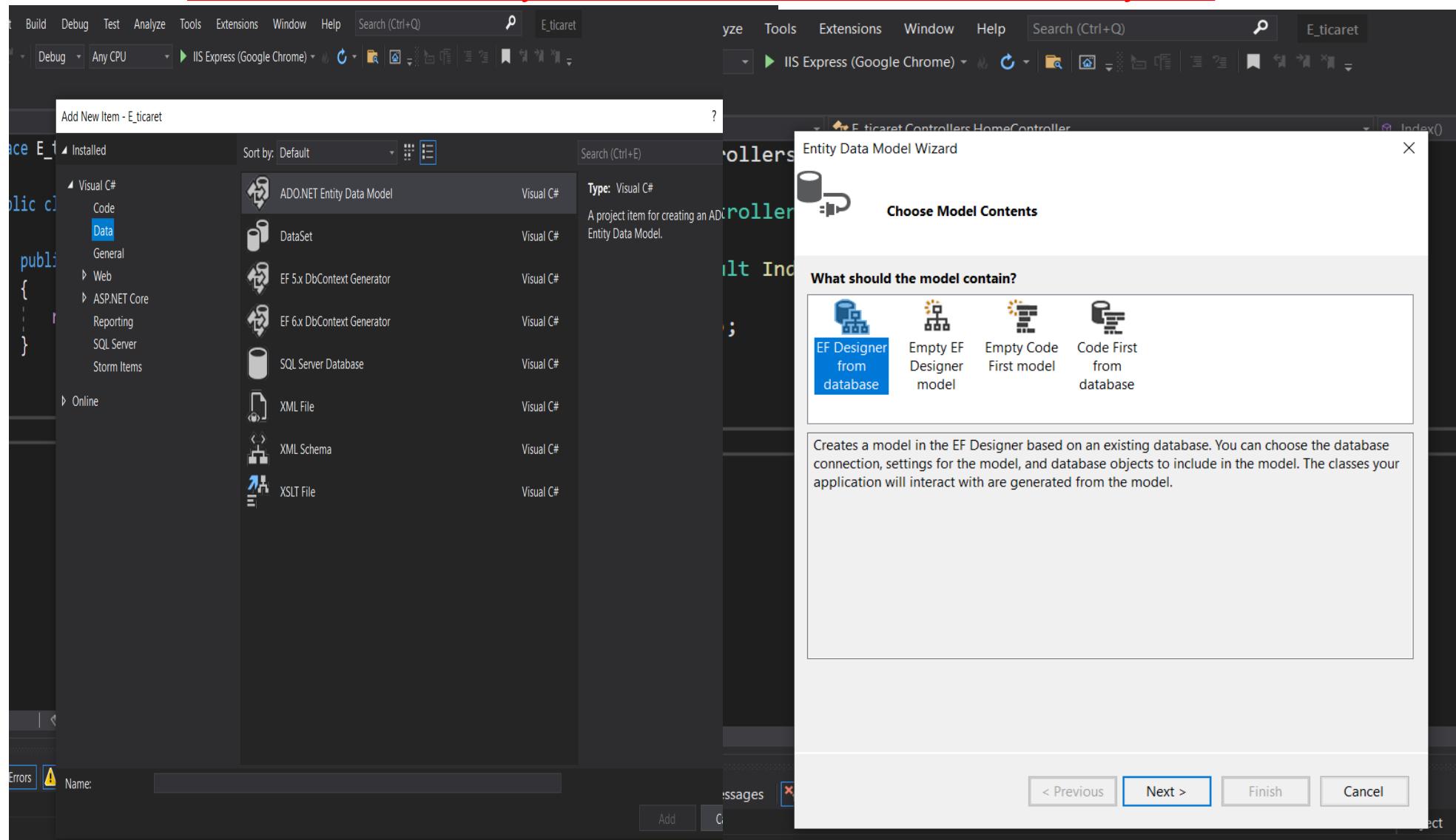
İstediğimiz Öğrenciye Ait Notları Raporlayalım

```
137     public ActionResult ogrenciye_gore_not_getir()
138     {
139         var notlar = db.notlar.Include(n => n.Ders).Include(n => n.ogrenci);
140         ViewBag.ogno = new SelectList(db.ogrenciler, "ogno", "ad_soyad");
141         return View(notlar);
142     }
143
144     [HttpPost]
145     public ActionResult ogrenciye_gore_not_getir(int? ogno)
146     {
147         Ienumerable<Not> notlar;
148         if (ogno == null)
149             notlar = db.notlar.Include(n => n.Ders).Include(n => n.ogrenci);
150         else
151             notlar = db.notlar.Include(n => n.Ders).Include(n => n.ogrenci).Where(x => x.ogno == ogno);
152         ViewBag.ogno = new SelectList(db.ogrenciler, "ogno", "ad_soyad");
153         return View(notlar);
154     }
155 }
```

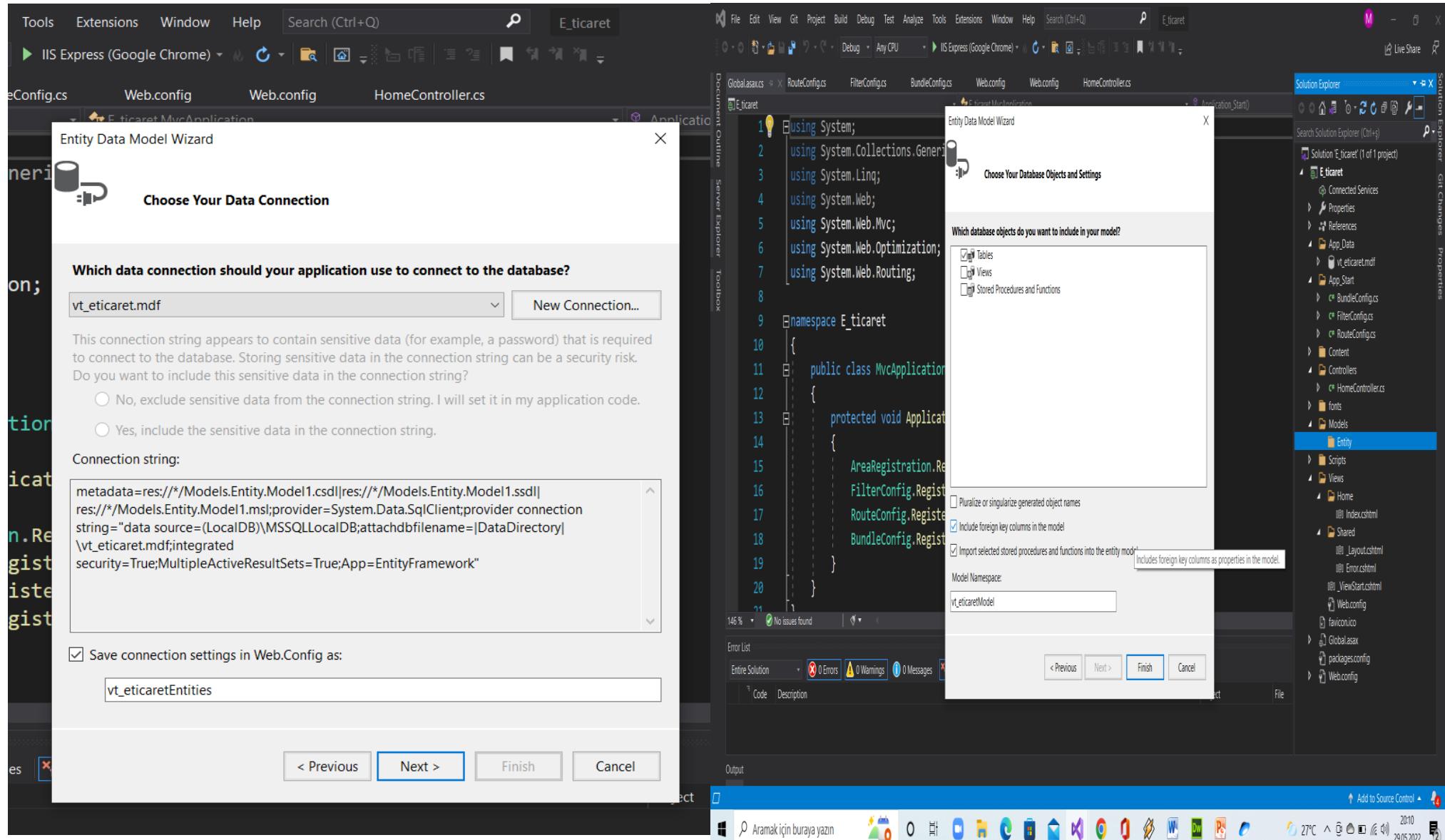
İstediğimiz Öğrenciye Ait Notları Raporlayalım

```
oller.cs Create.cshtml Okulcontext.cs Ogrenci.cs Ders.cs Create_kendi_form.cshtml OgrenciController.cs Index.cshtml Ogrenci_notu_getir.cshtml* ✎ X Tc_dogrula.cs Index.cshtml
13 </head>
14 <body>
15     @using (Html.BeginForm())
16     {
17         <p>@Html.DropDownList("ogno", null, "Seçiniz", new { onchange="this.form.submit()" }) </p>
18     }
19     <table class="table">
20         <tr>
21             <th>
22                 @Html.DisplayNameFor(model => model.Ders.ders_adi)
23             </th>
24             <th>...</th>
25             <th>...</th>
26             <th>...</th>
27             <th></th>
28         </tr>
29         @foreach (var item in Model) {
30             <tr>
31                 <td>
32                     @Html.DisplayFor(modelItem => item.Ders.ders_adi)
33                 </td>
34                 <td>
35                     @Html.DisplayFor(modelItem => item.Ogrenci.ad_soyad)
36                 </td>
37                 <td>
38                     @Html.DisplayFor(modelItem => item.yaz1)
39                 </td>
40                 <td>
41                     @Html.DisplayFor(modelItem => item.yaz2)
42                 </td>
43                 <td>
44                     @Html.DisplayFor(modelItem => item.yaz3)
45                 </td>
46             </tr>
47         }
48     </table>
49 
```

Veritabanımız Entity Framework ile Model olarak Ekleyelim



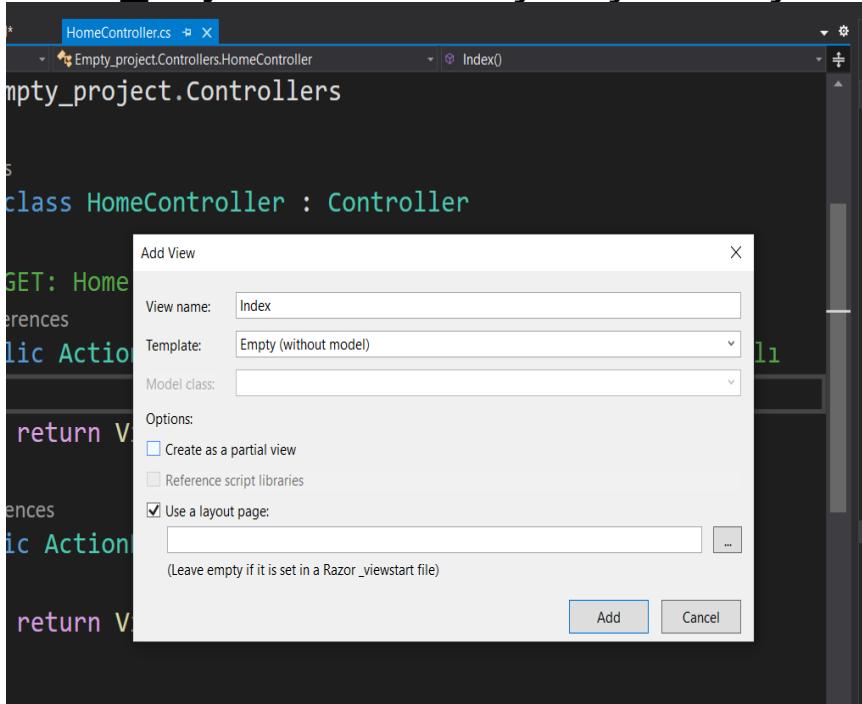
Veritabanımız Entity Framework ile Model olarak Ekleyelim



Layout(Master Page/Template)

Razor'da Layout'lar, tutarlı bir görünüm(Şablon) sağlamaya yardımcı olmanın yanında bir çok View arasında görüntülenir, kullanılırlar. Web Form'lar ASP.NET sayfaları ile çalıştiysanız, Layout'lar, Master Page'ler ile aynı amaca hizmet ederler.

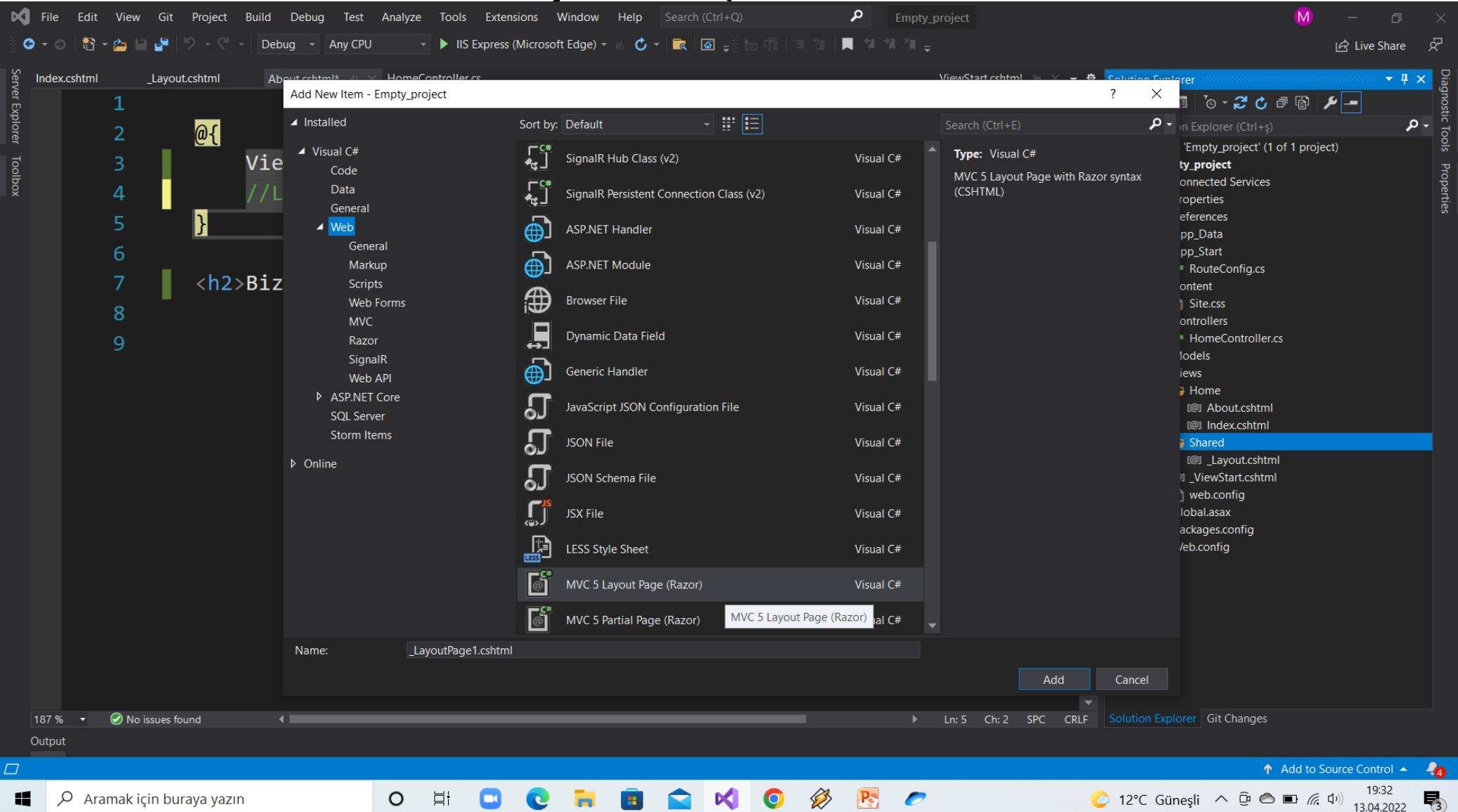
Yeni bir View oluşturduysanız ve layout(chekbox) seçimi yapıldıysa default Layout.cshtml seçilmiş olur. Açılan kutudan farklı bir layout ile değiştirebilir.



```
1  @{
2      Layout = null;
3  }
4
5  <!DOCTYPE html>
6
7  <html>
8      <head>
9          <meta name="viewport" content="width=device-width" />
10         <title>Index</title>
11     </head>
12     <body>
13         <div>
14             <a href="@Url.Action("about")">Hakkımızda</a>
15         </div>
16     </body>
17 </html>
18
19
```

Oluştururken layout seçilmesse

Layout Oluşturma



The screenshot shows the Microsoft Visual Studio IDE interface. On the left, the code editor displays `_Layout.cshtml` and `Index.cshtml`. The `_Layout.cshtml` file contains the following code:

```
1 <!DOCTYPE html>
2
3 <html>
4   <head>
5     <meta name="viewport" content="width=device-width" />
6     <title>@ViewBag.Title</title>
7   </head>
8   <body>
9     <div>
10       @RenderBody()
11     </div>
12   </body>
13 </html>
```

Annotations in yellow text explain the code:

- A yellow arrow points from the `@ViewBag.Title` placeholder in the `_Layout.cshtml` head section to the `Index.cshtml` file in the Solution Explorer. The text next to it says: "Sayfa başlığı değişkeni içeriği viewlerde atanan değer".
- A yellow arrow points from the `@RenderBody()` placeholder in the `_Layout.cshtml` body section to the `Index.cshtml` file in the Solution Explorer. The text next to it says: "Layout kullanan View'lerinizdeki içeriklerin görüntüleceği bölümdür".

The Solution Explorer on the right shows the project structure:

- Solution 'Empty_project' (1 of 1 project)
 - Empty_project
 - Connected Services
 - Properties
 - References
 - App_Data
 - App_Start
 - RouteConfig.cs
 - Content
 - Site.css
 - Controllers
 - HomeController.cs
 - Models
 - Views
 - Home
 - About.cshtml
 - Index.cshtml
 - Shared
 - _benim_sablonum.cshtml
 - _Layout.cshtml
 - _ViewStart.cshtml
 - Global.asax
 - packages.config
 - Web.config

Layout İçerisinde Bir Çok Section(bölüm) Nasıl Kullanılır

Farklı viewlerde farklı görünümler oluşturmamızı sağlar

```
4 <head>
5   <meta name="viewport" content="width=device-width" />
6   <title>@ViewBag.Title</title>
7 </head>
8 <body>
9   benim şablonum
10  <header>
11    @RenderSection("header")
12  </header>
13
14  <div>
15    @RenderBody()
16  </div>
17  <footer>
18    @RenderSection("footer",false)
19  </footer>
20 </body>
21 </html>
```

False ise bu section viewlerde kullanmak zorunlu değil

```
1
2  @{
3    ViewBag.Title = "İletişim";
4    Layout = "~/Views/Shared/_benim_sablonum.cshtml";
5  }
6  @section header
7  {
8    <h2> iletişim sayfası başlık bölümü</h2>
9  }
10 <h2>İletişim</h2>
11
12 @section footer
13 {
14    <h2> iletişim sayfası alt başlık bölümü</h2>
15 }
```

Layout Hazırlayalım

Index - Eticaret Sitem

localhost:44325/Home/Index

Ana Sayfa Kategoriler Üye ol İletişim

Index

Alt bilgi

Anakart
ram
hd
monitör
ses kartı
web cam
hoparlör
dv-cd yazıcı
mouse
3dyazıcı
y
bb
xxx
mmmmmm

misafir

Kullanıcı Adı

Şifre

Gönder

https://localhost:44325/urunler/index/4

Aramak için buraya yazın

24°C 20:55 28.06.2022

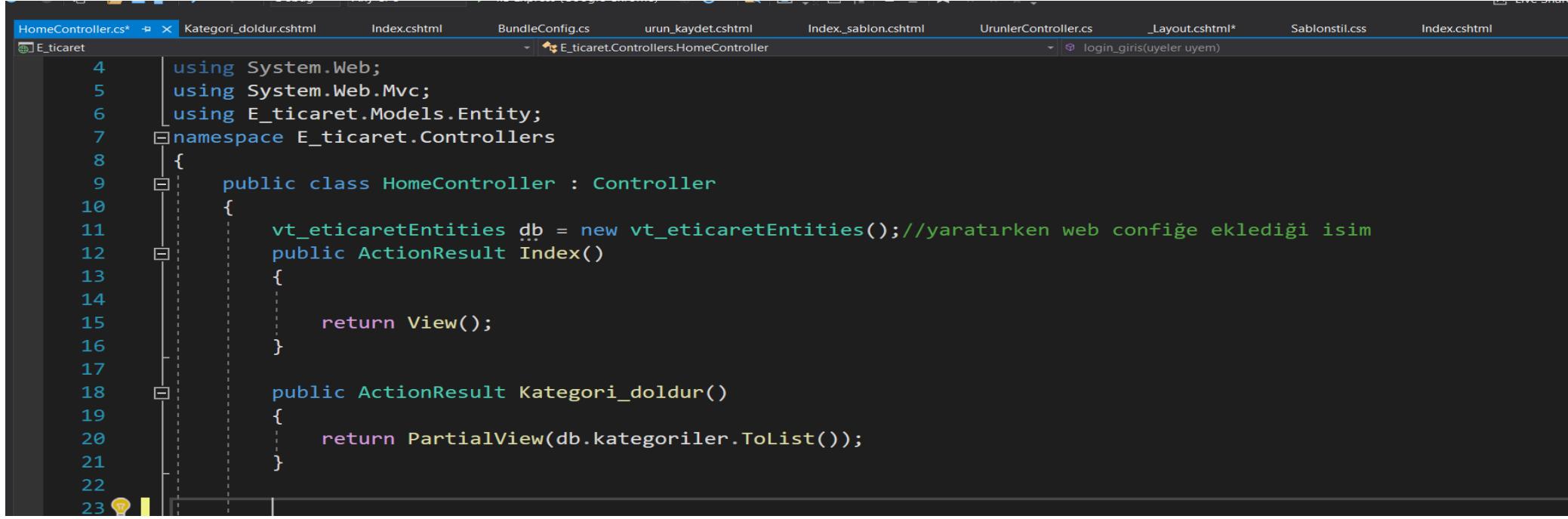
Layout Hazırlayalım

- Üst menümüzü hazırlayalım Bootstrap nav -bar kullanacağız.

```
16 </head>
17 <body>
18     <table class="tablo">
19         <tr>
20             <td colspan="3" class="ustmenu">
21                 <ul class="nav nav-pills">
22                     <li class="nav-item">
23                         <a class="nav-link active" aria-current="page" href="@Url.Action("index", "home")">Ana Sayfa</a>
24                     </li>
25                     <li class="nav-item dropdown">
26                         <a class="nav-link dropdown-toggle" data-bs-toggle="dropdown" href="#" role="button" aria-expanded="false">Kategoriler</a>
27                         <ul class="dropdown-menu">
28                             @{Html.RenderAction("Kategori_doldur", "home");}
29                         </ul>
30                     </li>
31                     <li class="nav-item">
32                         @if (Session["uyem"] == null)//üye giriş yapılmışsa
33                         {
34                             <a class="nav-link" href="@Url.Action("uye_kaydet", "uyeler")">Üye ol</a>
35                         }
36                         else
37                         {
38                             <a class="nav-link" href="@Url.Action("uye_edit", "uyeler", new { uyeid = ((uyeler)Session["uyem"]).uyeid })" >Edit</a>
39                         }
40                     </li>
41
42                     <li class="nav-item">
43                         <a class="nav-link disabled">İletişim</a>
44                     </li>

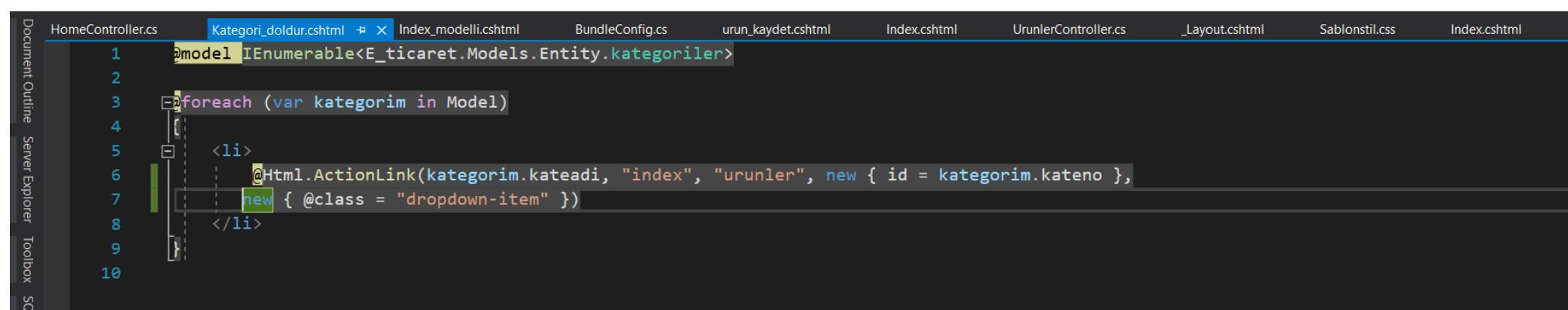
```

Layout İçerisinde PartialView kullanma



The screenshot shows the Visual Studio IDE with the HomeController.cs file open. The code defines a HomeController class that inherits from Controller. It contains two actions: Index() and Kategori_doldur(). The Kategori_doldur() action returns a PartialView result, passing a list of kategoriler entities from the database.

```
HomeController.cs
4  using System.Web;
5  using System.Web.Mvc;
6  using E_ticaret.Models.Entity;
7  namespace E_ticaret.Controllers
8  {
9      public class HomeController : Controller
10     {
11         vt_eticaretEntities db = new vt_eticaretEntities(); //yaratırırken web config'e eklediği isim
12         public ActionResult Index()
13         {
14             return View();
15         }
16
17         public ActionResult Kategori_doldur()
18         {
19             return PartialView(db.kategoriler.ToList());
20         }
21     }
22 }
23
```



The screenshot shows the Kategori_doldur.cshtml partial view. It uses a foreach loop to iterate through the Model (a list of kategoriler). For each item, it generates an li element containing an ActionLink that points to the index action of the Urunler controller, passing the id of the current category.

```
Kategori_doldur.cshtml
1 @model IEnumerable<E_ticaret.Models.Entity.kategoriler>
2
3 @foreach (var kategorim in Model)
4 {
5     <li>
6         @Html.ActionLink(kategorim.kateadi, "index", "urunler", new { id = kategorim.kateno },
7                         new { @class = "dropdown-item" })
8     </li>
9 }
```

Layout Hazırlayalım

- Sağdaki login template <https://codepen.io/hammadhz/pen/PoOjovM> indirelim

The screenshot shows the Microsoft Visual Studio IDE interface with the following details:

- Menu Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Full Screen.
- Toolbar:** Search (Ctrl+Q), M.
- Project Explorer:** dbo.uyeler [Data]
- Code Editor:** The active file is _Layout.cshtml*. It contains the following C# and Razor code:

```
43 <div class="div1">
44     @{
45         var kul_kontrol = "";
46         string ad_soyad = "Misafir";
47         if (Session["uyem"] != null)
48         {
49             ad_soyad = ((uyeler)Session["uyem"]).adsoyad;
50         }
51         if (Session["kulkontrol"] != null)
52         { kul_kontrol = Session["kulkontrol"].ToString(); }

53     }
54     @Html.Label(ad_soyad)
55     @Html.Label(kul_kontrol)

56     @using (Html.BeginForm("login_giris", "Home"))
57     {
58         <div class="input-container">
59             <i class="fa fa-envelope icon"></i>
60             @*<input type="email" placeholder="Email" name="email" class="input-field" required>*@
61             @Html.TextBox("kuladi","", new { @class= "input-field",placeholder="Kullanıcı Adı", required="true" })
62         </div>
63         <div class="input-container">
64             <i class="fa fa-key icon"></i>
65             @*<input type="password" placeholder="Password" name="password" class="input-field" required>*@
66             @Html.TextBox("sifre","", new { @class = "input-field", placeholder = "$ifre", required="true" })
67         </div>
68         <div><center><input type="submit" class="btn" value="Giriş"></center></div>
69     }
70 }
```
- Status Bar:** 159 %, No issues found, Ready, Add to Source Control, Ln: 56 Ch: 46 SPC CRLF.

Layout Hazırlayalım

- Sağdaki login template <https://codepen.io/hammadhz/pen/PoOjovM> indirelim

The image shows two side-by-side code editors in Visual Studio. The left editor displays the `HomeController.cs` file, which contains C# code for configuring a static file bundle. The right editor displays the `Login.css` file, which contains CSS styles for a login page.

HomeController.cs (Left Editor)

```
1 2
2 3    @using E_ticaret.Models.Entity
3
4
5    <!DOCTYPE html>
6    <html>
7      <head>
8        <meta charset="utf-8" />
9        <meta name="viewport" content="width=device-width, initial-scale=1.0">
10       <title>@ViewBag.Title - Eticaret Sitem</title>
11       <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
12       @Styles.Render("~/Content/css")
13
14
15       bundles.Add(new StyleBundle("~/Content/css").Include(
16
17         "~/Content/bootstrap.css",
18
19         "~/Content/site.css",
20
21         "~/Content/sablonstil.css",
22
23         "~/Content/bootstrap.min.css",
24
25         "~/Content/Login.css",
26
27         "~/Content/datepicker.css"
28
29     ));
30
31
32
33
34
35
36
37
38
39
40
41
42 }
```

Login.css (Right Editor)

```
1 * {
2   box-sizing: border-
3   padding: 1px;
4   font-family: Arial,
5
6
7   #heading1 {
8     text-align: center;
9     padding: 30px;
10
11
12   img {
13     display: block;
14     margin-left: auto;
15     margin-right: auto;
16     width: 50%;
17
18
19   .myForm {
20     margin: auto;
21     margin-top: 2px;
22
23
24   .input-container {
25     display: flex;
26     width: 100%;
27     margin-bottom: 15px;
28
29 }
```

Layout Hazırlayalım

- Sağdaki login template <https://codepen.io/hammadhz/pen/PoOjovM> indirelim

```
19     public ActionResult kategori_doldur()
20     {
21         return PartialView(db.kategoriler.ToList());
22     }
23
24     public ActionResult login_giris(uyeler uyem)
25     {
26         Session["kulkontrol"] = null;
27         var uye = db.uyeler.FirstOrDefault(x => x.kuladi == uyem.kuladi && x.sifre == uyem.sifre);
28
29         if(uye==null)
30             Session["kulkontrol"] = "Yanlış kullanıcı adı veya Şifre";
31         else
32             Session["uyem"] = uye;
33             return RedirectToAction("index");
34     }
35 }
```