

AUTO GENERATED CONTROLLERS & VIEWS

Dr. Öğr. Üyesi Fatma AKALIN

MVC projelerinde su ana kadar controller ve viewleri kendimiz oluşturuyorduk. Ama MVC projelerinde bu kontrolleri otomatik olarak oluşturabilmemiz mümkün.

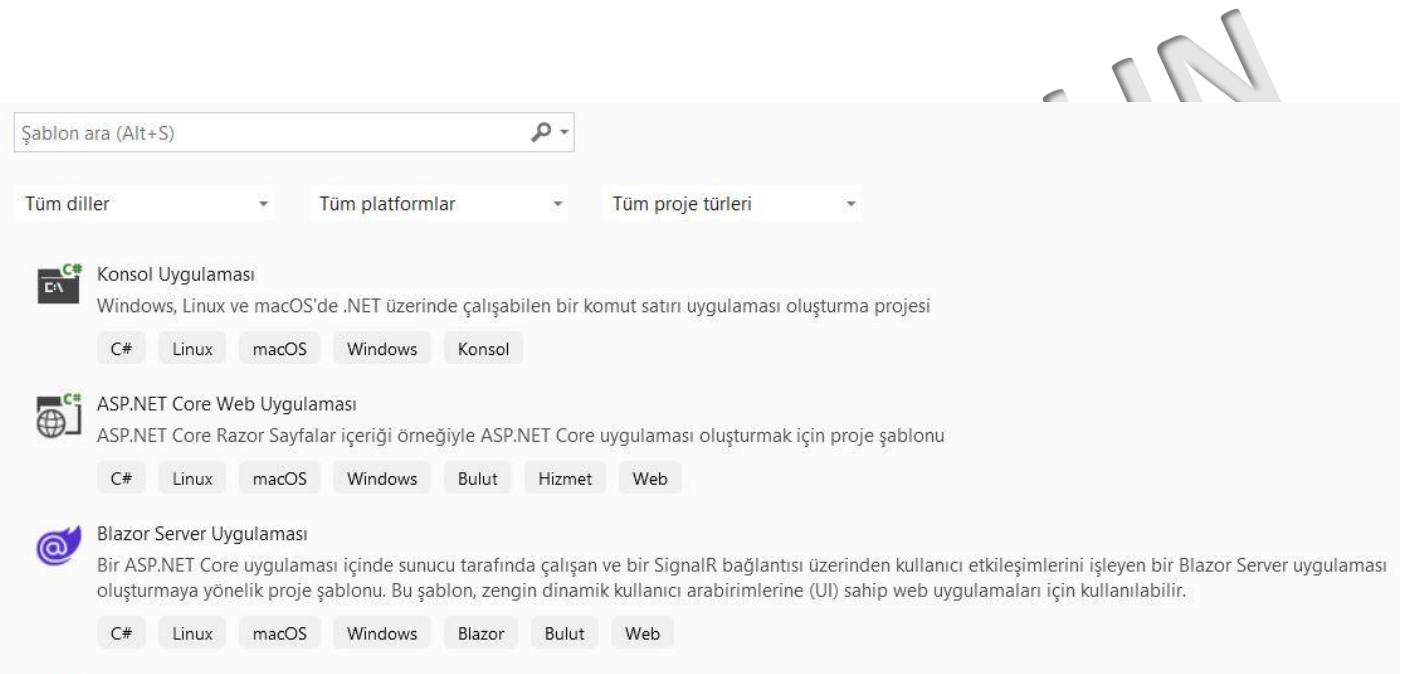
Aslında viewlar; insert, update ,delete ve select gibi her projede yer alan standart sayfalar. Bunu her tablo için oluşturabiliyor. 10 tablo içeren bir databaseniz varsa ve sizinде 10 ekran için insert update ,delete ve select sayfalarını hızlıca çıkarmak gibi bir hedefiniz mevcutsa auto generated controller ve viewlar ile bunu yapabilirsiniz. Projeye başlayalım...

Dr. Öğr. Üyesi
Mustafa KARAKOÇ

Yeni bir proje oluşturun

Son proje şablonları

-  ASP.NET Web Uygulaması (.NET Framework) C#
-  Windows Forms Uygulaması (.NET Framework) C#
-  ASP.NET Core Web Uygulaması (Model-Görünüm-Denetleyici) C#
-  Konsol Uygulaması C++



The screenshot shows the 'Create New Project' dialog in Visual Studio. At the top, there's a search bar labeled 'Şablon ara (Alt+S)' and three dropdown menus: 'Tüm diller', 'Tüm platformlar', and 'Tüm proje türleri'. Below these are four project template cards:

- Konsol Uygulaması**: A command-line application template. It includes a 'C#' button and tags for 'Linux', 'macOS', 'Windows', and 'Konsol'.
- ASP.NET Core Web Uygulaması**: An ASP.NET Core web application template. It includes buttons for 'C#', 'Linux', 'macOS', 'Windows', 'Bulut', 'Hizmet', and 'Web'.
- Blazor Server Uygulaması**: A Blazor server-side application template. It includes buttons for 'C#', 'Linux', 'macOS', 'Windows', 'Blazor', 'Bulut', and 'Web'.

Dr. Öğr.

Yeni projenizi yapılandırın

ASP.NET Web Uygulaması (.NET Framework)

Proje adı

Konum

Çözüm adı i

 Çözümü ve projeyi aynı dizine yerleştir

Altyapı

Proje "D:\aspnet\1_ADERSLER\denemeonu

Yeni ASP.NET Web Uygulaması oluştur



Boş

ASP.NET uygulamaları oluşturmak için boş bir proje şablonu. Bu şablonda içerik yoktur.



Web Forms

ASP.NET Web Forms uygulamaları oluşturmak için bir proje şablonu. ASP.NET Web Forms, bilinen sürükle ve bırak yöntemiyle olay denetimli modeli kullanarak dinamik web siteleri oluşturmanızı sağlar. Tasarım yüzeyi ile yüzlerce denetim ve bileşen, veri erişimi olan gelişmiş, güçlü, kullanıcı arabirimi denetimli web sitelerini hızlı bir şekilde oluşturmanızı sağlar.



MVC

ASP.NET MVC uygulamaları oluşturmaya yarayan bir proje şablonu. ASP.NET MVC, Model-View-Controller mimarisini kullanarak uygulamalar oluşturmanıza olanak sağlar. ASP.NET MVC en son standartları kullanan uygulamalar oluşturmak için hızlı, test güdümlü geliştirmeye olanak sağlayan birçok özellik içerir.



Web API

Tarayıcılar ve mobil cihazlar dahil olmak üzere çok çeşitli istemcilere erişebilen RESTful HTTP hizmetleri oluşturmaya yarayan bir proje şablonudur.



Tek Sayfalı Uygulama

ASP.NET Web API kullanarak JavaScript temelli zengin istemci tarafı HTML5 uygulamaları oluşturmak üzere kullanılan proje şablonudur. Tek Sayfalı Uygulamalar, HTML5, CSS3 ve JavaScript'in kullanıldığı istemci tarafı etkileşimler içeren zengin bir kullanıcı deneyimi sağlar.

Kimlik Doğrulama

Klasörleri ve çekirdek başvurularını ekle

- Web Forms
- MVC
- Web API'si

Gelişmiş

- HTTPS'yi Yapılandır
- Docker desteği
(Docker Desktop gereklidir)
- Ayrıca birim testleri için bir proje oluştur
denemeonuncuders.Tests

Home Page - ASP.NET Uygulamam x +

localhost:44374

PAMUKKALE ÜNİVE... Gmail YouTube Tumor Detection U... ROI-Based Processi... Cross-validated, bin... Using MATLAB with... GEO DataSet Browser GEO Accession vie... » | Tüm Yer İşaretleri

Uygulama adı Giriş Hakkında Kişi

ASP.NET

ASP.NET is a free web framework for building great Web sites and Web applications using HTML, CSS and JavaScript.

[Learn more »](#)

Getting started

ASP.NET MVC gives you a powerful, patterns-based way to build dynamic websites that enables a clean separation of concerns and gives you full control over markup for enjoyable, agile development.

[Learn more »](#)

Get more libraries

NuGet is a free Visual Studio extension that makes it easy to add, remove, and update libraries and tools in Visual Studio projects.

[Learn more »](#)

Web Hosting

You can easily find a web hosting company that offers the right mix of features and price for your applications.

[Learn more »](#)

© 2023 - ASP.NET Uygulamam

Microsoft Visual Studio

Seçili öğeler kalıcı olarak silinecek.

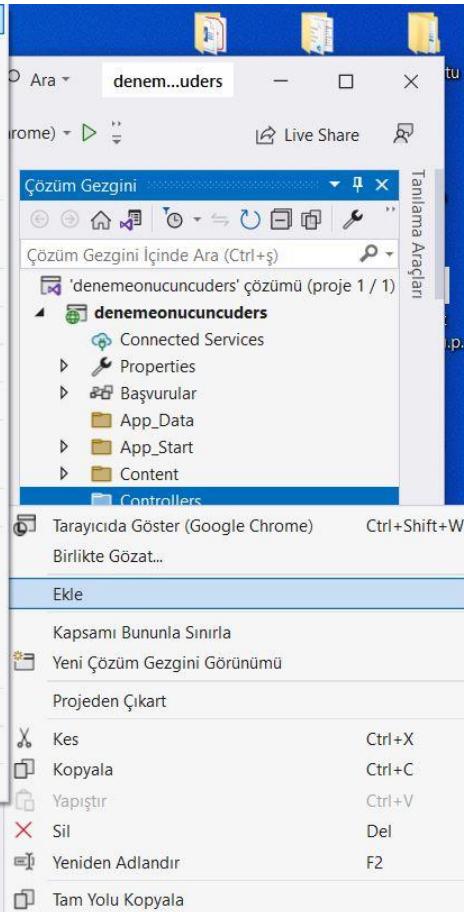
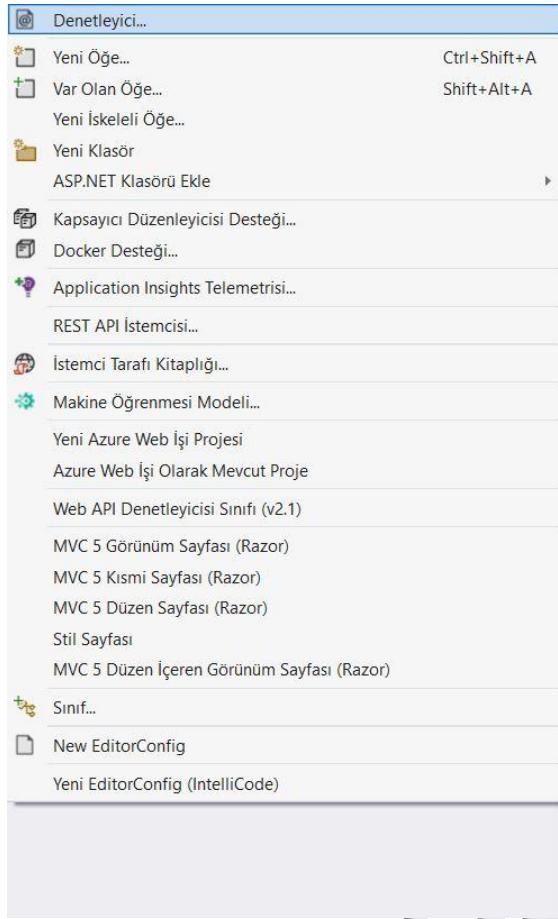
Tamam Iptal

Çözüm Gezgini

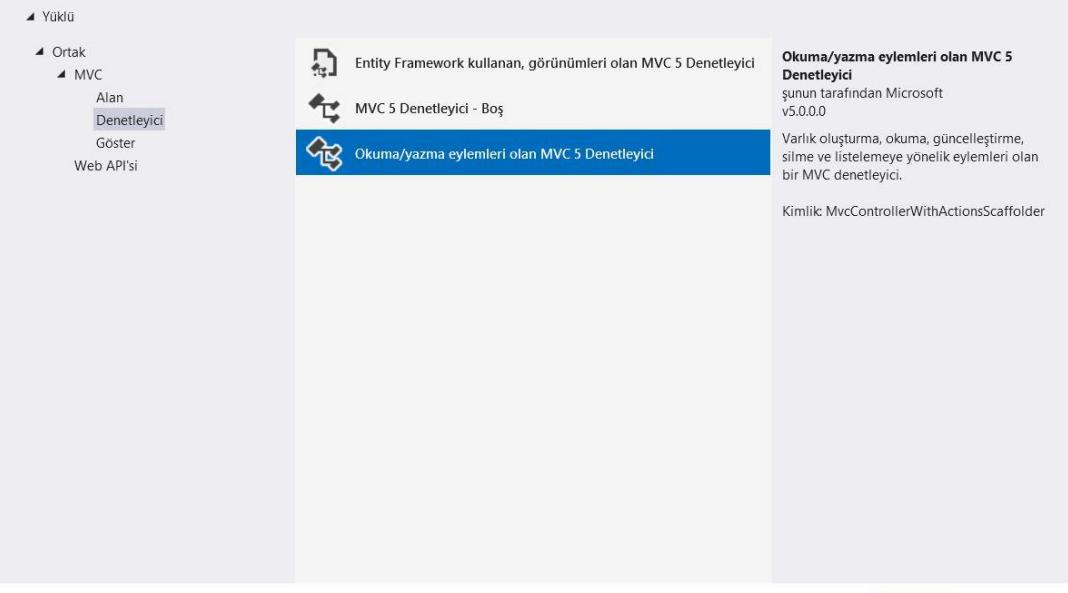
Çözüm Gezgini İçinde Ara (Ctrl+Ş)

'denemeonuncucuders' çözümü (proje 1 / 1)

- denemeonuncucuders
 - Connected Services
 - Properties
 - Başvurular
 - App_Data
 - App_Start
 - Content
 - Controllers
 - Models
 - Scripts
 - Views
 - Shared
 - _ViewStart.cshtml
 - Web.config
 - favicon.ico
 - Global.asax
 - packages.config
 - Web.config



Yeni İskeli Öğe Ekle



Denetleyici Ekle

Denetleyici adı:

HomeController

Ekle

İptal

Önceki slayttaki **Okuma yazma eylemleri olan MVC5 Denetleyicisi** türünde bir controller eklediğimizde bize farklı actionların yer aldığı bir controller dizisi geliyor. Oluşturulan actionları incelediğimizde,

Verileri listelediğimiz index action sayfası var.

Herhangi bir veriye ilişkin detayları gösteren Details action sayfası var.

Veri eklemek için gerçekleştirilen bir işlemi gösteren Create actionu var.

Düzenlemek için Edit actionu var.

Silmek için Deleteactionu var.

(metot post actionu ile birlikte geliyor...)

Yani böyle bir controller türünü eklediğimizde bize farklı actionların yer aldığı controller dizisi geliyor.

Örneğin, Create veri eklemek için kullanılan bir action. Parametre olarak FormCollection türünden dönüyor.

Hata durumunda yönlendirilme sağlanıyor.

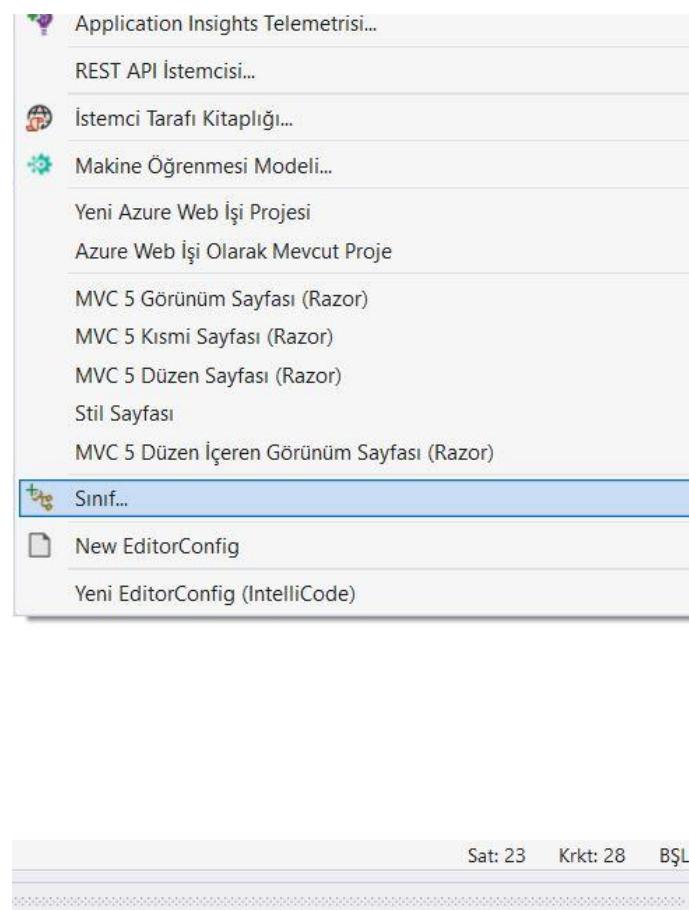
Sadece şunu belirtmek isterim ki, FormCollection türünden bize dönüyor. Ancak biz burda bir Modelimiz varsa onu ekleriz.

Şimdi Models klasöründe bir sınıf ekleyelim.

```
// GET: Home/Create
[HttpGet]
public ActionResult Create()
{
    return View();
}

// POST: Home/Create
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Create(FormCollection collection)
{
    try
    {
        // TODO: Add insert logic here

        return RedirectToAction("Index");
    }
    catch
    {
        return View();
    }
}
```



Yeni Öğe Ekle - denemeonucuncuders

Sıralama ölçütü: Varsayılan

Ara (Ctrl+E)

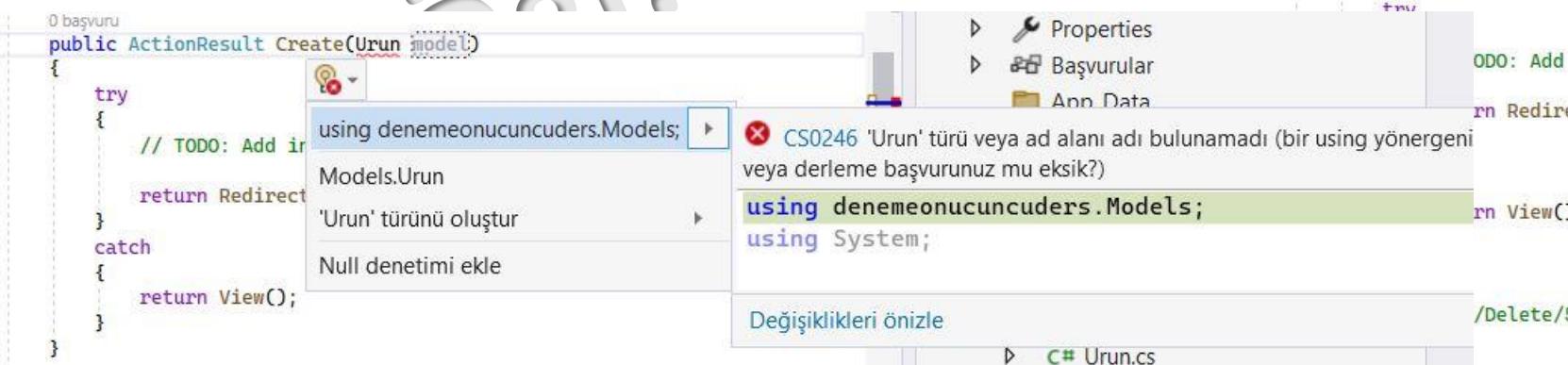
Tür: C#
Boş bir sınıf bildirimi

Icon	Adı	Tür
C# Class icon	Sınıf	C#
C# Class icon	Class for U-SQL	C#
Interface icon	Arabirim	C#
Entity Model icon	ADO.NET Entity Data Model	C#
Code Analysis icon	Bütünleştirilmiş Kod Bilgi Dosyası	C#
Code Analysis icon	Code Analysis Kural Kümesi	C#
Text icon	Çalışma Zamanı Metin Şablonu	C#
Editorconfig icon	editorconfig Dosyası (.NET)	C#
Editorconfig icon	editorconfig Dosyası (varsayılan)	C#
Entity Model icon	EF 5.x DbContext Generator	C#
Entity Model icon	EF 6.x DbContext Generator	C#
JavaScript icon	Hata Ayıklama Görselleştiricisi	C#
JavaScript icon	JavaScript JSON Yapılandırma Dosyası	C#
Source Control icon	Kaynaklar Dosyası	C#

Ad: Uruncs

Ekle Iptal

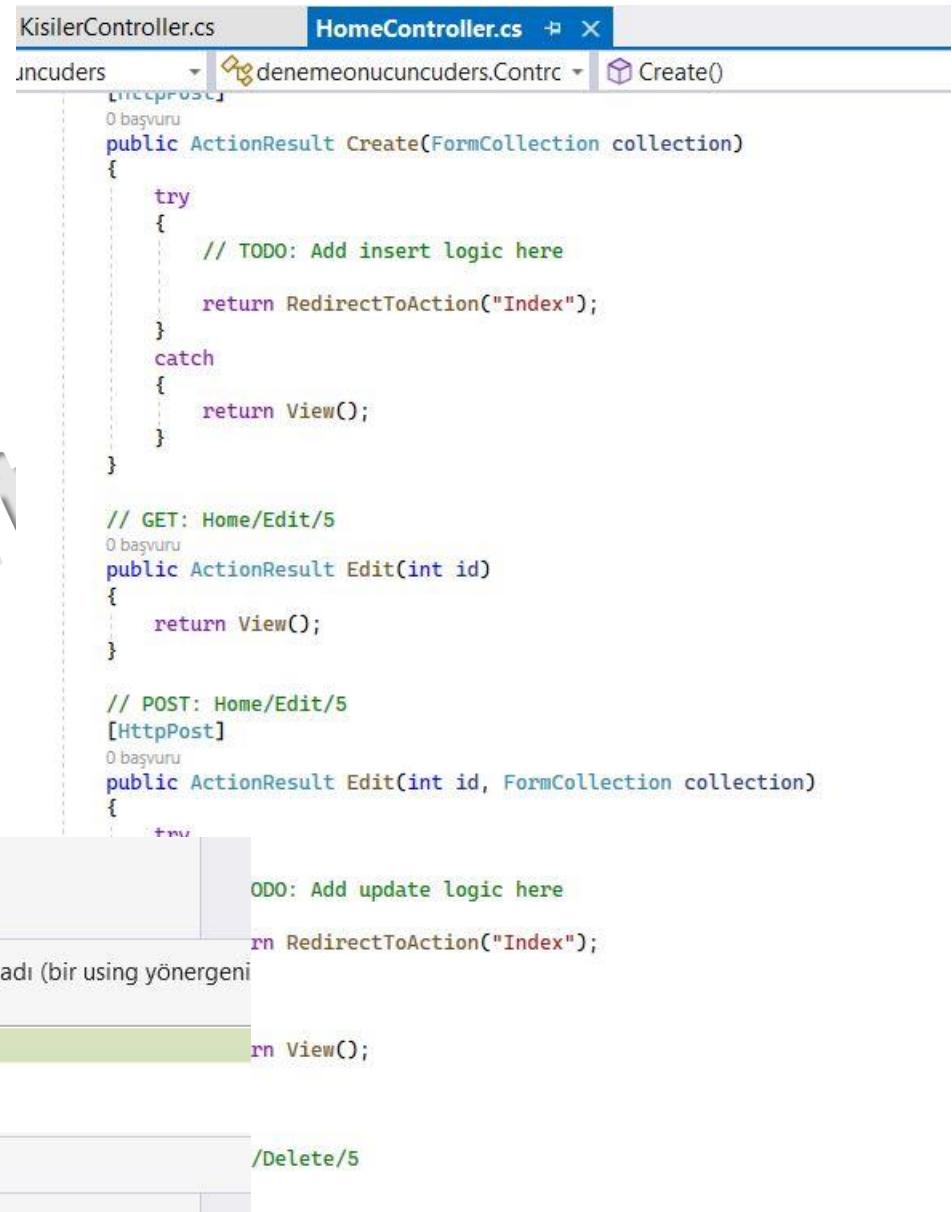
Artık modelimiz olduğu için önceki derslerde öğrendiğimiz üzere modelden gideceğiz. Fakat using yönergemizi eklemeyi unutmayalım!!!



A screenshot of Microsoft Visual Studio showing the 'Urun.cs' file. The cursor is over the word 'Urun' in the line 'using denemeonucuders.Models;'. A tooltip displays the error message 'CS0246 'Urun' türü veya ad alanı adı bulunamadı (bir using yönergeni veya derleme başvurunuz mu eksik?)'. Below the tooltip, the 'using' directive is completed as 'using denemeonucuders.Models;'.

```
0 başvuru
public ActionResult Create(Urun model)
{
    try
    {
        // TODO: Add insert logic here

        return RedirectToAction("Index");
    }
    catch
    {
        Null denetimi ekle
    }
}
```



A screenshot of Microsoft Visual Studio showing the 'HomeController.cs' file. It contains three action methods: 'Create()', 'Edit(int id)', and 'Edit(int id, FormCollection collection)'. The 'Create()' method has a note 'TODO: Add insert logic here'. The 'Edit(int id)' method has a note 'TODO: Add update logic here'. The 'Edit(int id, FormCollection collection)' method has a note 'TODO: Add update logic here'.

```
KisilerController.cs   HomeController.cs + X
denemeonucuders.Controllers.Create()
[HttpPost]
0 başvuru
public ActionResult Create(FormCollection collection)
{
    try
    {
        // TODO: Add insert logic here

        return RedirectToAction("Index");
    }
    catch
    {
        return View();
    }
}

// GET: Home/Edit/5
0 başvuru
public ActionResult Edit(int id)
{
    return View();
}

// POST: Home/Edit/5
[HttpPost]
0 başvuru
public ActionResult Edit(int id, FormCollection collection)
{
    try
    {
        // TODO: Add update logic here

        return RedirectToAction("Index");
    }
    catch
    {
        return View();
    }
}
```

```
KisilerController.cs      HomeController.cs  ✘ ×  
unciders      denemeonuncuders.Controllers.HomeController  
Create()  
  
[HttpPost]  
0 basvuru  
public ActionResult Create(FormCollection collection)  
{  
    try  
    {  
        // TODO: Add insert logic here  
  
        return RedirectToAction("Index");  
    }  
    catch  
    {  
        return View();  
    }  
  
    // GET: Home/Edit/5  
0 basvuru  
public ActionResult Edit(int id)  
{  
    return View();  
}  
  
// POST: Home/Edit/5  
[HttpPost]  
0 basvuru  
public ActionResult Edit(int id, FormCollection collection)  
{  
    try  
    {  
        // TODO: Add update logic here  
  
        return RedirectToAction("Index");  
    }  
    catch  
    {  
        return View();  
    }  
  
    // GET: Home/Delete/5
```

```
KisilerController.cs      HomeController.cs  ✘ ×  
denemeonuncuders.Controllers.HomeController  
Create()  
  
0 basvuru  
public ActionResult Create()  
{  
    return View();  
}  
  
// POST: Home/Create  
[HttpPost]  
0 basvuru  
public ActionResult Create(Urun model)  
{  
    try  
    {  
        // TODO: Add insert logic here  
  
        return RedirectToAction("Index");  
    }  
    catch  
    {  
        return View();  
    }  
  
    // GET: Home/Edit/5  
0 basvuru  
public ActionResult Edit(int id)  
{  
    return View();  
}  
  
// POST: Home/Edit/5  
[HttpPost]  
0 basvuru  
public ActionResult Edit(Urun model)//id parametresini de yanına verebilirsin ama direkt modelde verebilirsin  
{  
    try  
    {  
        // TODO: Add update logic here  
  
        return RedirectToAction("Index");  
    }  
    catch
```

Parametre olarak FormCollection türünden bir nesne değilde Urun modelinden bir nesne alalım önceki derslerde öğrendiğimiz gibi

```
0 başvuru
public ActionResult Delete(int id)
{
    return View();
}

// POST: Home/Delete/5
[HttpPost]
0 başvuru
public ActionResult Delete(int id, FormCollection collection)
{
    try
    {
        // TODO: Add delete logic here

        return RedirectToAction("Index");
    }
    catch
    {
        return View();
    }
}
```

Dr. 

```
KisilerController.cs      HomeController.cs  X
denemeonuncuders.Controllers.HomeController
0 başvuru
        return View();
    }

    // GET: Home/Delete/5
0 başvuru
public ActionResult Delete(int id)
{
    return View();
}

    // POST: Home/Delete/5
[HttpPost, ActionName("Delete")]
0 başvuru
public ActionResult DeleteConfirm(int id)//delete'in post işleminde hatırlarsak model bile gelmiyor
//direk id olarak parametre verinde yukarıdaki metot ile aynı imzalı olduğu için hata verecek
//Metodun ismini değiştireceğim ama yine Delete ismi ile çağrılmak istedığımı belirtmek koşulu ile
{
    try
    {
        // TODO: Add delete logic here

        return RedirectToAction("Index");
    }
    catch
    {
        return View();
    }
}
```

Artık işlemlerinizi kodlayabilirsiniz.

UYARI

```
// GET: Home/Delete/5
0 başvuru
public ActionResult Delete(int id)
{
    return View();
}

// POST: Home/Delete/5
[HttpPost]
0 başvuru
public ActionResult Delete(int id)//delete'in post işleminde hatırlarsak model bile gelmiyor
//direk id olarak parametre verinde yukarıdaki metot ile aynı imzalı olduğu için hata verecek
//Metodun ismini değiştireceğim ama yine Delete ismi ile çağrırmak istediğimi belirtmek koşulu ile
{
    try
    {
        // TODO: Add delete logic here
    }
}
```

Dr. Öğr. Üy

```
// GET: Home/Delete/5
0 başvuru
public ActionResult Delete(int id)
{
    return View();
}

// POST: Home/Delete/5
[HttpPost, ActionName("Delete")]
0 başvuru
public ActionResult DeleteConfirm(int id)//delete'in post işleminde hatırlarsak model bile gelmiyor
//direk id olarak parametre verinde yukarıdaki metot ile aynı imzalı olduğu için hata verecek
//Metodun ismini değiştireceğim ama yine Delete ismi ile çağrırmak istediğimi belirtmek koşulu ile
{
    try
    {
        // TODO: Add delete logic here
    }
}
```

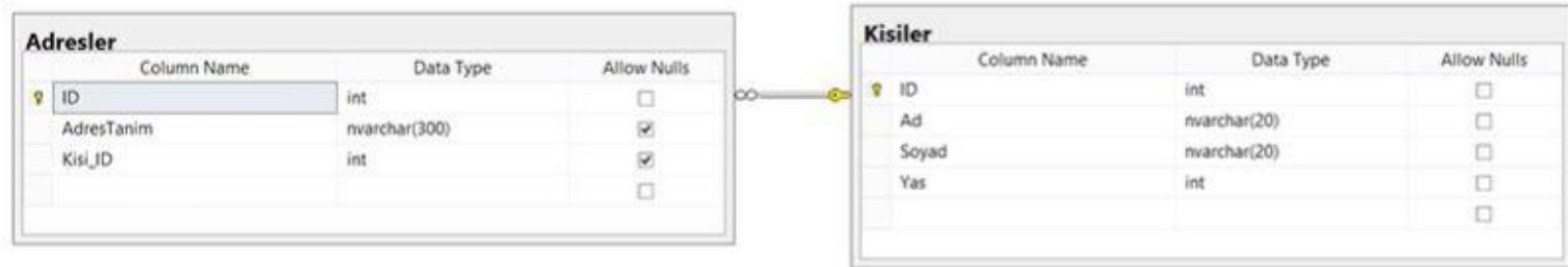
Bu kısımda hata görüldüğü üzere kalktı...

Yapacağımız projede öncesinde oluşturduğumuz aşağıdaki database yapısını kullanıyor olacağız.

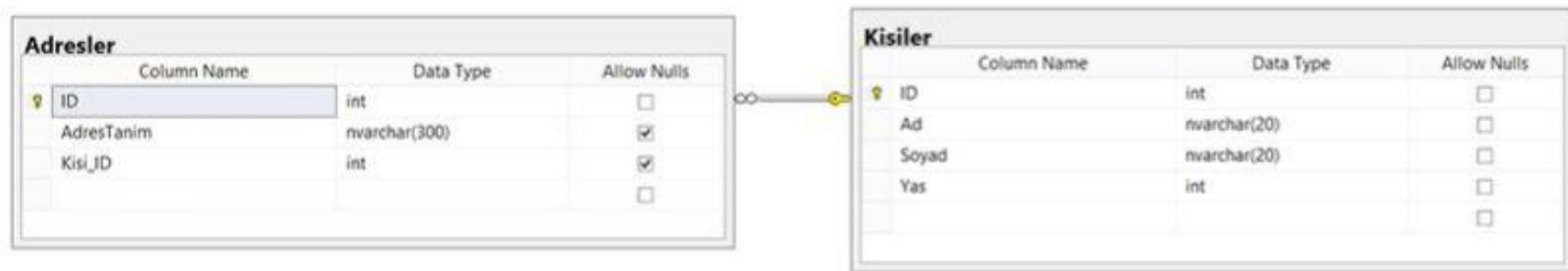
Bu tablonun insert, update ve delete işlemlerini hızlı bir şekilde oluşturabiliriz.

Şimdi Controller eklemeden önce ilgili databasemi buraya entity framework ile eklemem gerek. Süreç aşağıdadır.

Projeye ekle->new item diyerek C# modülüne geleceğiz ve aşağıda görüldüğü üzere ilgili model yapısını seçeceğiz.



Burada, bir önceki projede oluşturmuş olduğum hazır veritabanını kullanıyorum ancak aşağıdaki veritabanına benzer veritabanının nasıl oluşturulacağını bu projenin altında mevcut projemizin bitiminde göstereceğim..



Çözüm Gezgini Stiller Şekil Efe

Derle
Tekrar Derle
Temizle
Görüntüle
Çözümleme ve Kod Temizleme
Yayımla...
Application Insights'ı Yapılandır...

Genel Bakış
Kapsamı Bununla Sınırla
Yeni Çözüm Gezgini Görünümü

Ekle

NuGet Paketlerini Yönet...

İstemci Tarifi Kitaplıklarını Yönet...

Başlangıç Projelerini Yapılandır...

Başlangıç Projesi Olarak Ayarla

Hata Ayıklama

Proje Etkileşimli Pencerede Başlat

Yeni Öğe... Ctrl+Shift+A

Var Olan Öğe... Shift+Alt+A

Yeni İşkeleli Öğe...

Yeni Klasör

ASP.NET Klasörü Ekle

Kapsayıcı Düzenleyicisi Desteği...

Docker Desteği...

Application Insights Telemetrisi...

REST API İstemcisi...

İstemci Tarifi Kitaplığı...

Makine Öğrenmesi Modeli...

Yeni Azure Web İşi Projesi

Azure Web İşi Olarak Mevcut Proje

Başvuru...

Hizmet Başvurusu...

Bağılı Hizmet

Çözümleyici...

HTML Sayfası

Yeni Öğe Ekle - denemeonucuncuders

Sıralama ölçütü: Varsayılan

Yüklenenler

C# Sinif C#

C# Class for U-SQL C#

Arabirim C#

ADO.NET Entity Data Model C#

Bütünlendirilmiş Kod Bilgi Dosyası C#

Code Analysis Kural Kümesi C#

Çalışma Zamanı Metin Şablonu C#

editorconfig Dosyası (.NET) C#

editorconfig Dosyası (varsayılan) C#

EF 5.x DbContext Generator C#

EF 6.x DbContext Generator C#

Hata Ayıklama Görselleştiricisi C#

JavaScript JSON Yapılandırma Dosyası C#

Kaynaklar Dosyası C#

Tür: C#
A project item for creating an ADO.NET Entity Data Model.

Ad: TestDBModel

Sıkıştırılmış Görünümü Göster

Ekle İptal

Dr. Öğr. Üyesi VALIN

Choose Model Contents

What should the model contain?

 EF Designer from database
 Empty EF Designer model
 Empty Code First model
 Code First from database

Creates a model in the EF Designer based on an existing database. You can choose the database connection, settings for the model, and database objects to include in the model. The classes your application will interact with are generated from the model.

< Geri **İleri >** Son İptal

Veri kaynağı:
Microsoft SQL Server (SqlClient)

Sunucu adı:
DESKTOP-71D31J5

Sunucuda oturum aç

Kimlik Doğrulama: Windows Kimlik Doğrulaması

Kullanıcı adı:
Parola:
 Parolamı kaydet

Veritabanına bağlan

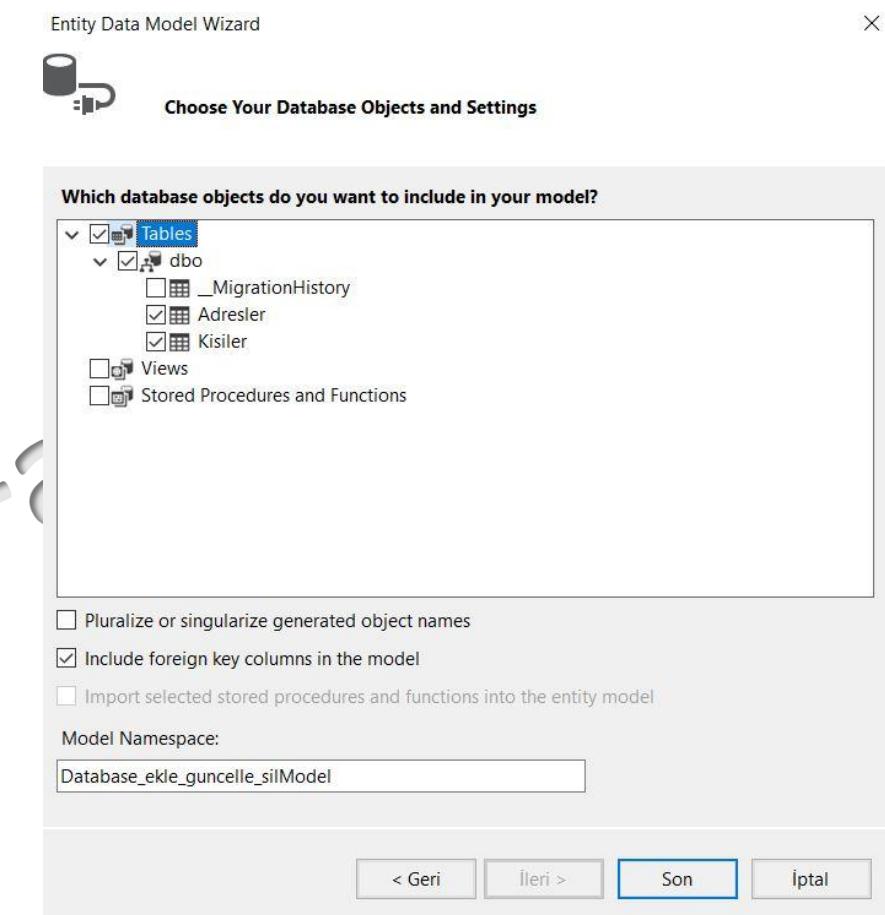
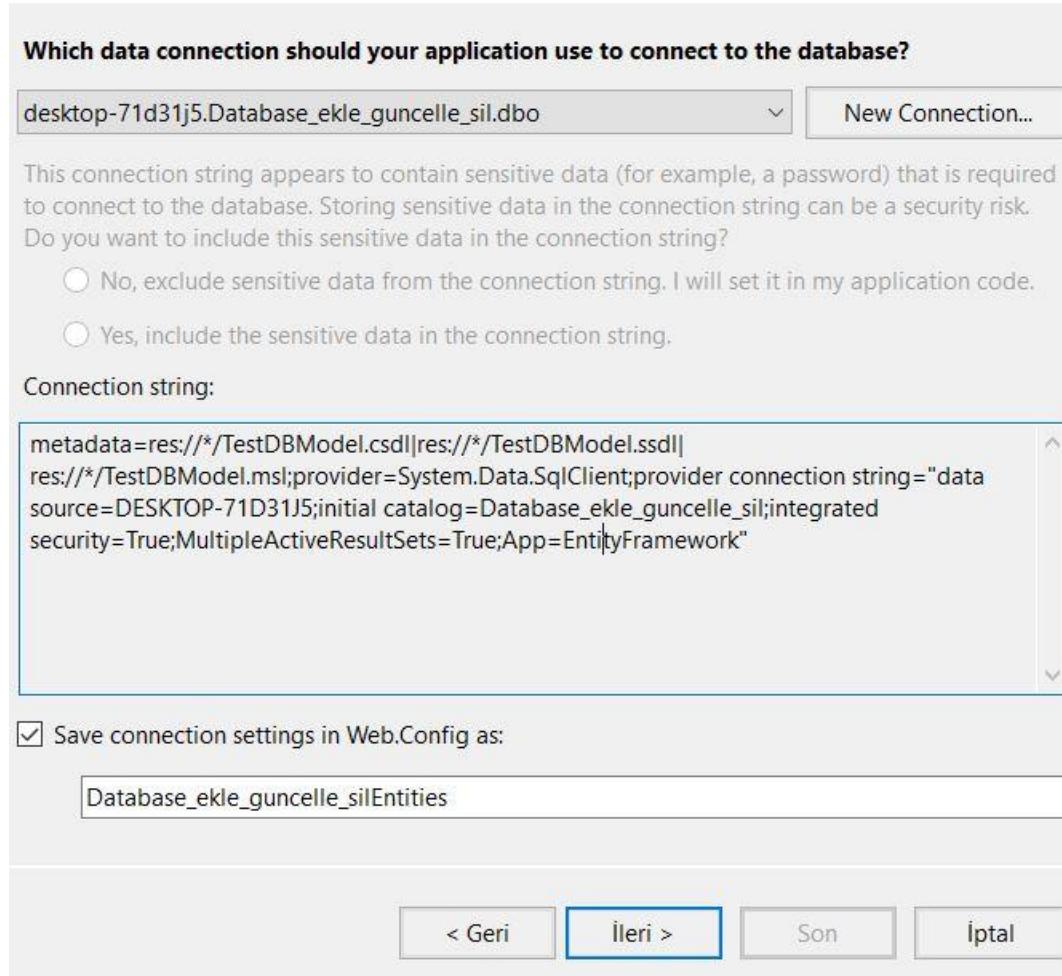
Veritabanı adını seçin veya girin:
Database_ekle_guncelle_sil

Veritabanı dosyası ilişir:
 Gözat...

Mantıksal ad:

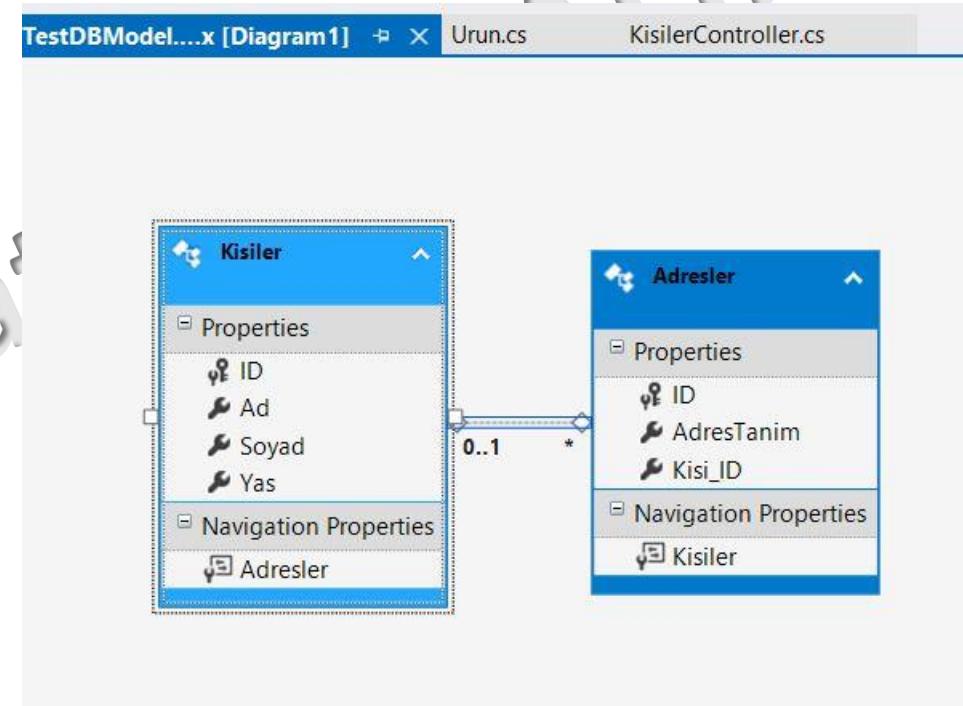
Gelişmiş...

Tamam

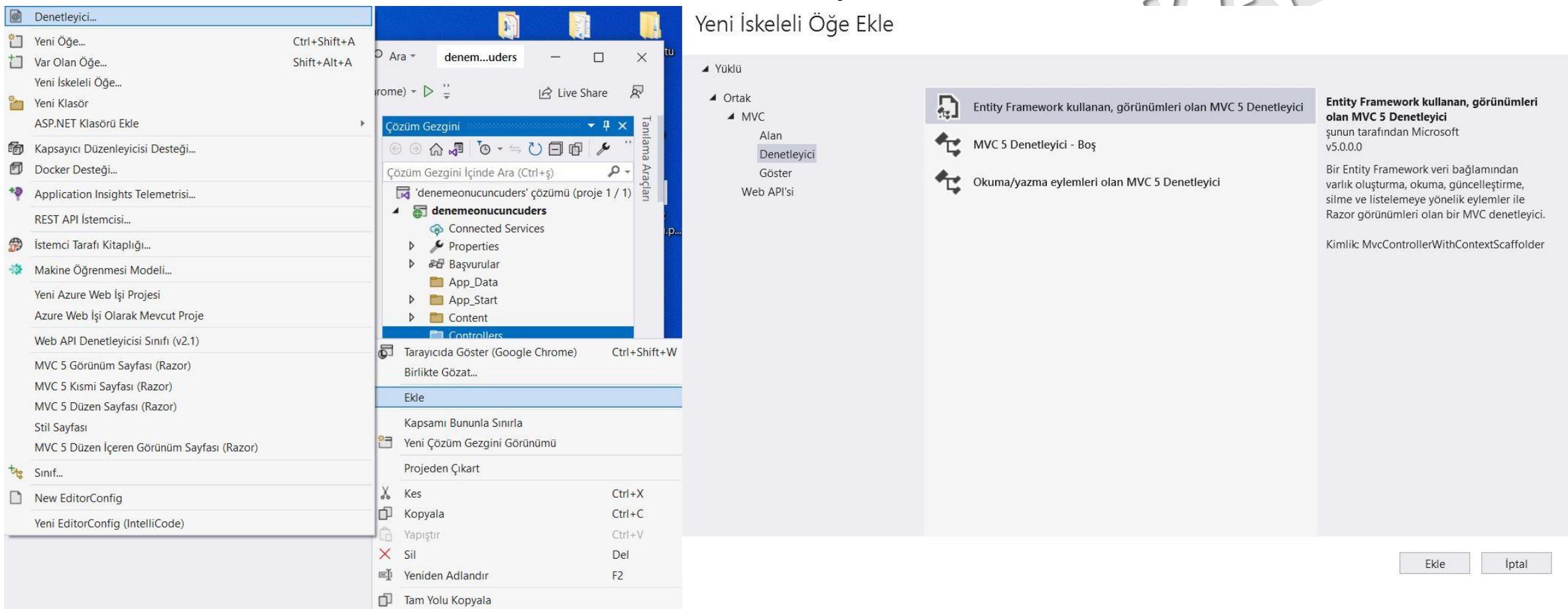


Elimiz ile seçerek yaptığımız bu işlem vasıtasıyla oluşturduğumuz bu yapı iki tablonun class karşılığının otomatik üretilmesini Finish dediğimde SQL'e gidip bazı sorgular çalıştırıp öğrenir. Tabloların veri tiplerini adlarını, kolonlarını, boş geçilip geçilmeyeceği, pf, fkHepsini öğrendikten sonra entity framework bize bir id diyagram açar ve ardından kişileri ve adresleri aradaki ilişki ile birlikte oluşturur. Buradaki her bir tablo bir class oluyor. Entitiy framewrok dili leri yani gerekli dll lerde eklendikten sonra Kaydet dediğimde de her şey hazır. Artık databasedeki tablolarımı classlaşmış oldum.

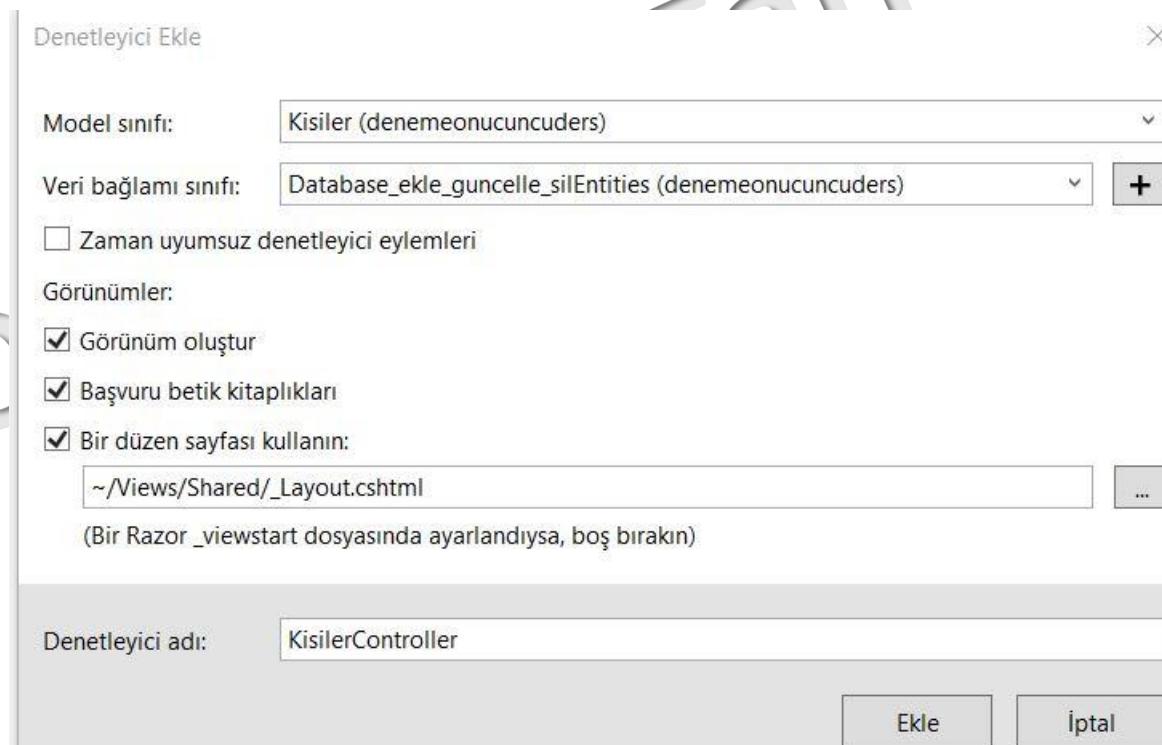
Projemizi derleyelim yapacağımız işlemlerden önce.



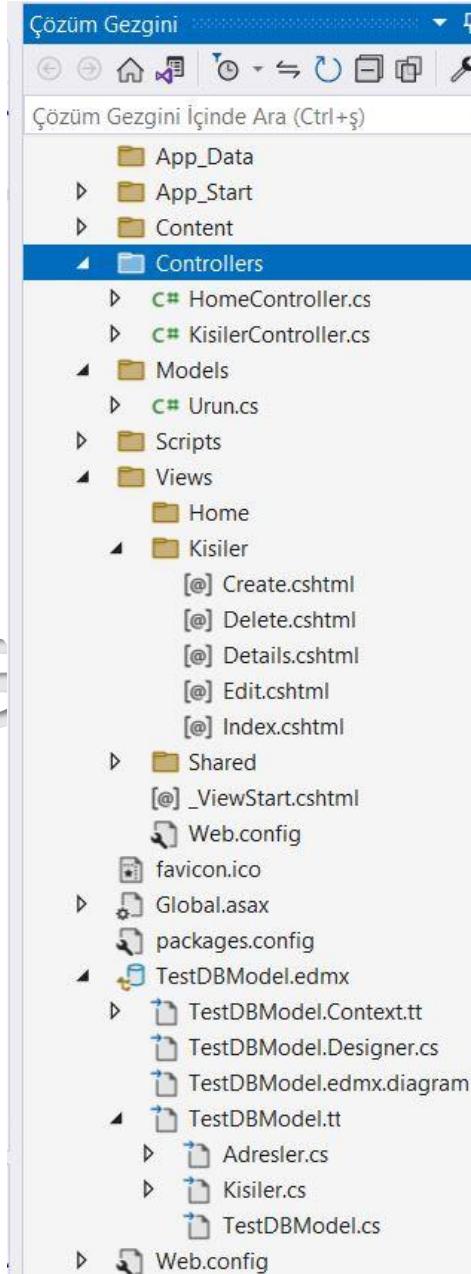
Şimdi KisilerController'ımı oluşturacağım. Bana viewları ile birlikte entity framework kullanarak Controller oluştur dersek aşağıdaki yolu izlemeliyiz.



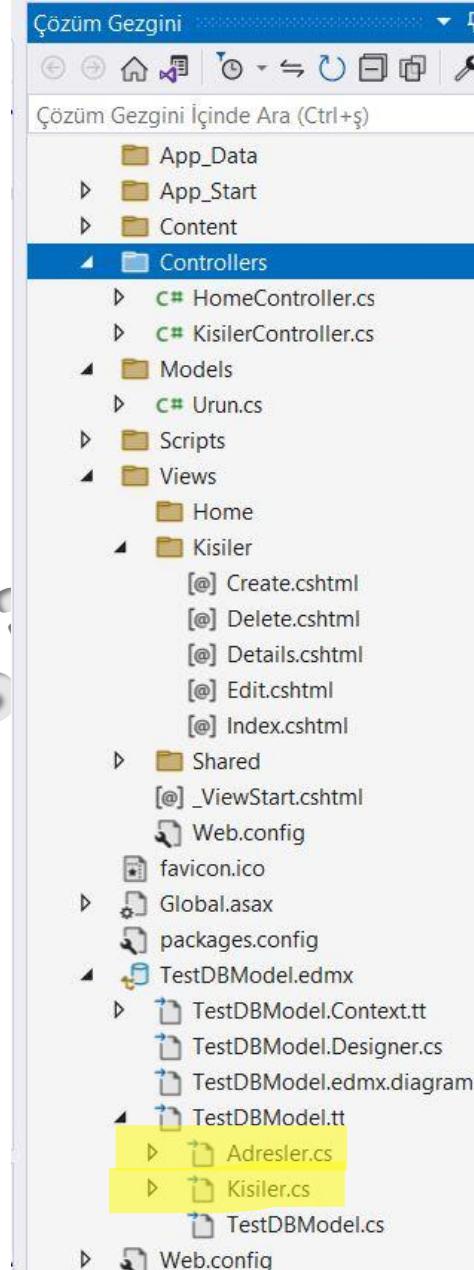
Böylece hem controller hem viewlar hem de entity frameworkteki o yapı sayesinde kodlarımız da yazılıyor olacak. Ekle dediğimde hangi modeli kulanacağınızı belirtmeniz gerek. Tüm sayfaları bir anda oluşturuyorsunuz. Önce kişiler tablomun karşılığı viewları oluşturacağım. Ardından diğer girdileri giriyorum. Ekle'ya bastığım gibi controller ve viewlar üretiliyor.



Script library'leri eklediğim için dizinden görülür ki script dosyaları eklenmiştir ve KisilerController da gelmiştir. Ek olarak Views>Kisiler klasörünün altında Create, Delete, Details, Edit ve Index cshtml sayfaları da oluşturulmuştur.



Modellerimin altında Kisiler ve Adresler yok diye sorabilirsiniz çünkü biz EntitFramework'ün içerisine modelleri ve kişileri ekledik. Entity framework kullandığımız için bu objenin altında gözükecektir.



KisilerController’ı inceleyelim.

Entity Framework ile database’ye erişebileceğimiz bir nesne oluşumu private değişken olarak bizlere sundu.

```
private Database_ekle_guncelle_silEntities db = new  
Database_ekle_guncelle_silEntities();
```

Index Actionun içine baktığımızda database nesnesinin Kisiler tablosunu view’ın içine gönderiyor. Bu doğrultuda Index view’ın içerisinde incelediğimizde

En üstte modelimizin projeye eklendiğini standart bootstrap classları ile bir yapı inşa edilmiştir. Ardından foreach ile Model’imizin içinde dönerek her bir item’ı(property’i) yerleştirmiştir.

NOTLAR

View sayfasında, Insert işlemi içinde create new dediğimizde verileri alacağımız bir form karşımıza gelir.

Create view, Kisiler'i sayfamıza eklemek için kullandığımız bir özelliktir. Burada begin form ile tüm formun bize gönderebileceği bir using bloğu açılır ve bu formun içindeki verilerin bize geri dönmesi sağlanır.

Sayfada `@Html.AntiForgeryToken()` ifadesi saldırıları önlemek için yazılmıştır. Olayın arka perdesi ise aslında bir token oluşturulması yani her istekte uzun bir anahtar oluşturulması temeline dayanıyor. Bu anahtar bizim serverimizdaki anahtar ile eşleşmezse server gelen istekleri red ediyor. Böylece birilerinin sayfamıza gelmeden bize post işlemi (veri gönderme) işlemi yapması engellenmiş oluyor.

`@Html.ValidationSummary(true, "", new { @class = "text-danger" })` Oluşan hataları tepede gösteriyor

(`int?` `id`) -> -> id nullable olabilir anlamına gelmektedir. .

`[HttpPost]`

`[ValidateAntiForgeryToken]`, post metodу kapsamında create sayfasında saldırı engelleyici token oluşturarak kontrol edilmesini sağlıyor .Kontrol edildikten sonra post işlemi çalışıyor.

Metoda da kişiler nesnemiz model olarak gönderilmiş durumdadır. Modelin veriyi hatalı bir şekilde alma durumuna karşın isvalid özelliği kullanılarak yönlendirme sağlanıyor.

Bir sonraki derstte html helpers metodlarından ve root valueslardan bahsedeceğiz. Yaptığımız projeyi anlayabilmemiz açısından ufak bir detay vereceğim. Akabinde bir sonraki derstte detaylı öğreneceğiz. Düzenleme,detaylar ve silme işlemlerinde

Her bir kayıttta bir satır oluşturduğu için HTML, her kayıtın düzenlenecek satırı anlamsız için düzenlenecek/datayı - gösterilecek/silinecek kodun ID'sini root values ile ilgili Edit/Details/Delete actionuna gönderiyor

```
@Html.ActionLink("Edit", "Edit", new { id=item.ID }) |  
@Html.ActionLink("Details", "Details", new { id=item.ID }) |  
@Html.ActionLink("Delete", "Delete", new { id=item.ID })
```

Uygulama adı Giriş Hakkında Kişi

Index

Create New

Ad	Soyad	Yas	
Peytonn	Carroll	45	Edit Details Delete
Carlos	Kenny	80	Edit Details Delete
Faith	Roberts	70	Edit Details Delete
Lillian	Peters	45	Edit Details Delete
Julian	Buckley	15	Edit Details Delete
Kaylee	Summers	24	Edit Details Delete
Julia	Mohamed	42	Edit Details Delete
Aria	Burrows	57	Edit Details Delete
Taylor	Sanders	43	Edit Details Delete
Jason	Shepherd	30	Edit Details Delete
Fatma	Akalin	22	Edit Details Delete

© 2023 - ASP.NET Uygulamam

esit

localhost:44374/Kisiler/Edit/4

PAMUKKALE ÜNİVE... Gmail YouTube Tumor Detectio

Uygulama adı Giriş Hakkında Kişi

Edit

Kisiler

```
<a href="/Kisiler/Edit/1">Edit</a> == $0  
" | "  
<a href="/Kisiler/Details/1">Details</a>  
" | "  
<a href="/Kisiler/Delete/1">Delete</a>  
</td>  
</tr>
```

localhost:44374/Kisiler/Edit/4

PAMUKKALE ÜNİVE... Gmail YouTube Tumor Detectio

Uygulama adı Giriş Hakkında Kişi

Edit

Kisiler

Ad
Lillian

Soyad
Peters

Yas
45

Save

Back to List

Tüm bu işlemleri bizim için otomatik yazıyor EF'nin bu control ve view oluşturma seçeneği ile. BUNU kullanarak tüm tablolarınızın insert update ve delete işlemlerini çok hızlı bir şekilde yapabilirsiniz Böylece üzerine yapacağınız eklemeler için güzel bir taslak olmuş olur.

Bununla birlikte layout kullanmadan bir tasarım nasıl olur? Ek olarak bunu inceleyelim..

Diğer aşamalardan farklı olarak seçimlerimizi aşağıdaki gibi yapmalıyız.(Aynı çerçevede farklı isimli bir database kullandım bu örnekte.)

The screenshot shows two windows side-by-side. On the left is the 'Denetleyici Ekle' (Add Data Source) dialog from the EntityDataSource configuration tool. It lists 'Model sınıfı: Kisiler (WebApplication1)' and 'Veri bağlamı sınıfı: database_onikiEntities (WebApplication1)'. Under 'Görünüler:' there are checked boxes for 'Görünüm oluştur' and 'Başvuru betik kitaplıklarını'. Below these are options for 'Zaman uyumsuz denetleyici eylemleri' and 'Bir düzen sayfası kullanın'. A note at the bottom says '(Bir Razor _viewstart dosyasında ayarlandıysa, boş bırakın)'. In the center is the 'Entity Data Model Wizard - Choose Your Data Connection' window. It has a title bar 'Bağlantı Özellikleri' and a message: 'Seçili veri kaynağna bağlanmak için bilgi girin ya da "Değiştir" seçeneğini tıklatarak farklı bir veri kaynağı ve/veya sağlayıcısı seçin.' It shows 'Veri kaynağı: Microsoft SQL Server (SqlClient)', 'Sunucu adı: DESKTOP-71D31J5', 'Kimlik Doğrulama: Windows Kimlik Doğrulaması', and 'Veritabanına bağlan' section with 'Veritabanı adını seçin veya girin: database_oniki'. On the right, the 'Choose Your Data Connection' window shows the selected connection 'desktop-71d31j5.database_oniki.dbo'. It includes a note about sensitive data and two radio button options: 'No, exclude sensitive data from the connection string. I will set it in my application code.' and 'Yes, include the sensitive data in the connection string.'. Below this is a 'Connection string:' section with the metadata string:

```
metadata=res://*/TestDBModel.csdl|res://*/TestDBModel.ssdl|res://*/TestDBModel.msl;provider=System.Data.SqlClient;provider connection string="data source=DESKTOP-71D31J5;initial catalog=database_oniki;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework"
```

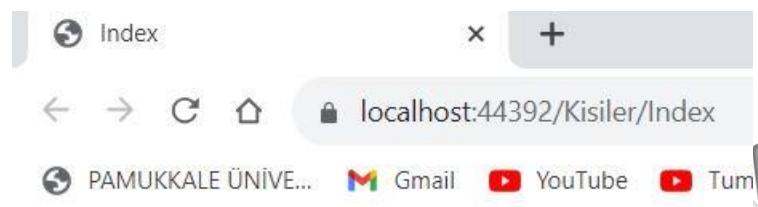
. A checked checkbox 'Save connection settings in Web.Config as:' is followed by a dropdown menu set to 'database_onikiEntities'. At the bottom are buttons for 'Gelişmiş...', 'Bağlantıyı Sıra', 'Tamam', and 'İptal'.

Index

[Create New](#)

Ad	Soyad	Yas	
Peytonn	Carroll	45	Edit Details Delete
Carlos	Kenny	80	Edit Details Delete
Faith	Roberts	70	Edit Details Delete
Lillian	Peters	45	Edit Details Delete
Julian	Buckley	15	Edit Details Delete
Kaylee	Summers	24	Edit Details Delete
Julia	Mohamed	42	Edit Details Delete
Aria	Burrows	57	Edit Details Delete
Taylor	Sanders	43	Edit Details Delete
Jason	Shepherd	30	Edit Details Delete
Fatma	Akalin	22	Edit Details Delete

© 2023 - ASP.NET Uygulamam



Ad	Soyad	Yas	
Nathann Silva	86	Edit Details Delete	
Charles Rossi	79	Edit Details Delete	
Gabriel Rossi	54	Edit Details Delete	
Genesis Watson	35	Edit Details Delete	
Kayden Riddick	48	Edit Details Delete	
Madelyn Bradshaw	70	Edit Details Delete	
Owen Summers	74	Edit Details Delete	
Emily Seymour	85	Edit Details Delete	
Jocelyn Little	31	Edit Details Delete	
Aubrey Rossi	76	Edit Details Delete	

localhost:44374/Kisiler/Edit/4

PAMUKKALE ÜNİVE... Gmail YouTube Tumor Detectio

Uygulama adı Giriş Hakkında Kişi

Edit

Kisiler

Ad	<input type="text" value="Lillian"/>
Soyad	<input type="text" value="Peters"/>
Yas	<input type="text" value="45"/>
<input type="button" value="Save"/>	

[Back to List](#)

localhost:44392/Kisiler/Edit/4

PAMUKKALE ÜNİVE... Gmail YouTube Tum

Kisiler

Ad	<input type="text" value="Nathann"/>
Soyad	<input type="text" value="Silva"/>
Yas	<input type="text" value="86"/>
<input type="button" value="Save"/>	

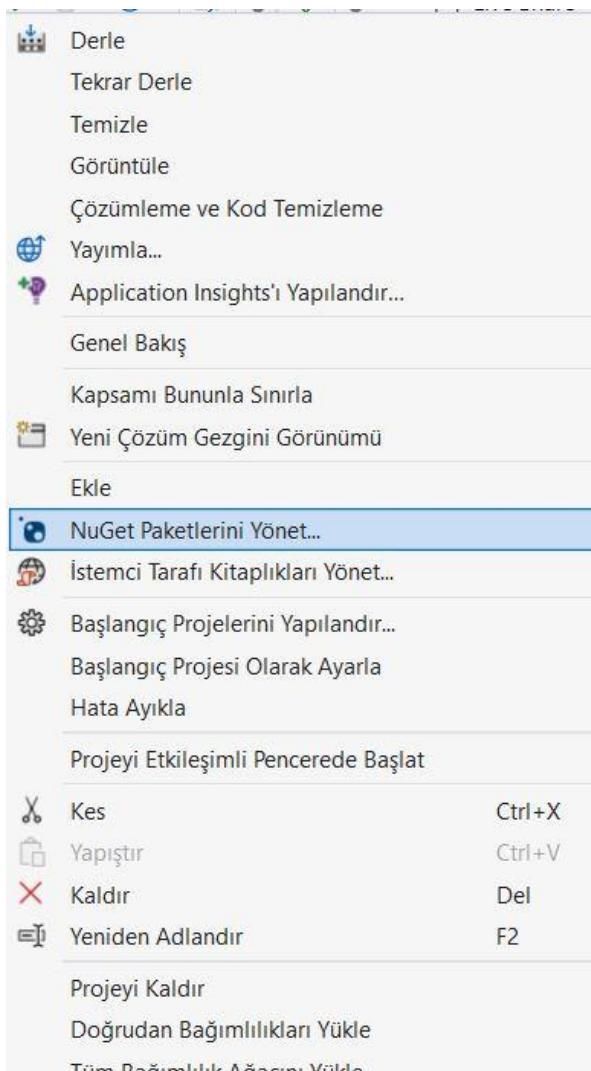
[Back to List](#)

AKALIN

Layout olmaksızın bir örnek tasarladığımızda görülmüyor ki görsel açıdan bizi tatmin edici bir sayfa üretilememektedir.

Peki layout kullanmaz isek görselliği düzeltebilir miyiz?

İlk olarak aşağıdaki kütüphaneleri yükleyelim...



The screenshot shows the NuGet Package Manager interface with several packages being downloaded:

- bootstrap**: Version 5.3.1, The Bootstrap framework in CSS. Includes JavaScript.
- Modernizr**: Version 2.8.3, Modernizr is a small and simple JavaScript library that detects features and adds support where needed.
- jQuery**: Version 3.7.0, jQuery is a new kind of JavaScript Library.
- jQuery.Validation**: Version 1.19.5, jQuery Validation plugin by Jörn Zaefferer.
- EntityFramework**: Version 6.4.4, Entity Framework 6 (EF6) is a tried and tested object-relational mapper for .NET with many years of feature development.
- Microsoft.EntityFrameworkCore**: Version 7.0.10, Microsoft.EntityFrameworkCore is a part of the Entity Framework Core library.

5 ayrı dosyada belirtilen kütüphanelerini yüklerseniz ilgili probleme çözüm sağlarsınız



```
Delete.cshtml  ✎ × Details.cshtml Edit.cshtml Index.cshtml Create.cshtml
1  @model WebApplication1.Kisiler
2
3  @{
4      Layout = null;
5
6
7  <!DOCTYPE html>
8
9  <html>
10 <head>
11     <link href="~/Content/bootstrap.min.css" rel="stylesheet" />
12
13     <script src="~/Scripts/jquery-3.7.1.min.js"></script>
14     <script src="~/Scripts/bootstrap.min.js"></script>
15     <script src="~/Scripts/modernizr-2.8.3.js"></script>
16     <meta name="viewport" content="width=device-width" />
17     <title>Delete</title>
18
19 </head>
<body>
```

KAYNAKLAR

<https://learn.microsoft.com/tr-tr/aspnet/mvc/>

Veysel Uğur Kızmaz, ASP.NET MVC5, Kodlab Yayınları

<https://github.com/muratbaseren/udemy-yazilimcilarin-yukselisi-aspnet-my-evernote-sample>