Ders 3: Katmanların Oluşturulması

1.katmanımız zaten var olan katmandır(MvcProjeKampi)

Solution Explorer penceresinde solution kısmına sağ tık🡪Add🡪New Project🡪Class Library(.NET Framework) 🡪EntityLayer (2.katman oluşturuldu)

Solution Explorer penceresinde solution kısmına sağ tık🡪Add 🡪New Project🡪Class Library(.NET Framework) 🡪DataAccessLayer (3.katman oluşturuldu)

Solution Explorer penceresinde solution kısmına sağ tık🡪Add 🡪New Project🡪Class Library(.NET Framework) 🡪BusinessLayer (4.katman oluşturuldu)

Ders 4 : Entity Katmanı Sınıflarının Oluşturulması

Tablolar(Heading-Content-Writer-Contact-About)

EntityLayer a sağ tık🡪Add 🡪New Folder🡪Concrete(Somut)

Concrete sağ tık🡪Add🡪New Class🡪Sırasıyla tablolara ait class lar eklenir bu yöntemle

Her sınıfa erişebilmek için class Heading i public etmemiz gerekiyor

Daha sonra tablo alanlarını belirleyen property leri tanımlayalım.Heading için tanımlı alanlar aşağıdaki gibidir.

namespace EntityLayer.Concrete

{

public class Heading

{

public int HeadingID { get; set; } //prop yaz iki defa tab a bas

public string HeadingName { get; set; }

public DateTime HeadingDate { get; set; }

}

}

Diğer class lar da aynı şekillerde tanımlanacaktır

Content Class ı

namespace EntityLayer.Concrete

{

public class Content

{

public int ContentID { get; set; }

public string ContentValue { get; set; }

public DateTime ContentDate { get; set; }

//ContentYazar ve ContentBaşlık gibi yapılar ilişki içinde tanımlanacaktır

}

}

Writer Class ı

namespace EntityLayer.Concrete

{

public class Writer

{

public int WriterID { get; set; }

public string WriterName { get; set; }

public string WriterSurName { get; set; }

public string WriterImage { get; set; }

public string WriterMail { get; set; }

public string WriterPassword { get; set; }

}

}

Ders 5 : Category About Contact Sınıfları

Category Class ı

namespace EntityLayer.Concrete

{

public class Category

{

public int CategoryID { get; set; }

public string CategoryName { get; set; }

public string CategoryDescription { get; set; }

public bool CategoryStatus { get; set; } //ilgili kategori aktif mi pasif mi

}

}

About Class ı

namespace EntityLayer.Concrete

{

public class About

{

public int AboutID { get; set; }

public string AboutDetails1 { get; set; }

public string AboutDetails2 { get; set; }

public string AboutImage1 { get; set; }

public string AboutImage2 { get; set; }

}

}

Contact Class ı

namespace EntityLayer.Concrete

{

public class Contact

{

public int ContactID { get; set; }

public string UserName { get; set; }

public string UserMail { get; set; }

public string Subject { get; set; }

public string Message { get; set; }

}

}

Ders 6: İlişkilerin Oluşturulması 1

Category ile Heading arasındaki ilişkinin kurulması

Heading Class ı

namespace EntityLayer.Concrete

{

public class Heading

{

public int HeadingID { get; set; } //prop yaz iki defa tab a bas

public string HeadingName { get; set; }

public DateTime HeadingDate { get; set; }

//Category ile Heading arasındaki ilişki

public int CategoryID { get; set; }//Heading sınıfında CategoryID colonu oldu

public virtual Category Category { get; set; }

}

}

Category Classı

namespace EntityLayer.Concrete

{

public class Category

{

public int CategoryID { get; set; }

public string CategoryName { get; set; }

public string CategoryDescription { get; set; }

public bool CategoryStatus { get; set; } //ilgili kategori aktif mi pasif mi

//Category ile Heading arasındaki ilişki

public ICollection<Heading> Headings { get; set; }

}

}

Content ile Heading arasındaki ilişkinin kurulması

Heading Class ı

namespace EntityLayer.Concrete

{

public class Heading

{

public int HeadingID { get; set; } //prop yaz iki defa tab a bas

public string HeadingName { get; set; }

public DateTime HeadingDate { get; set; }

//Category ile Heading arasındaki ilişki

public int CategoryID { get; set; }//Heading sınıfında CategoryID colonu oldu

public virtual Category Category { get; set; }

public ICollection<Content> Contents { get; set; }

}

}

Content Class ı

namespace EntityLayer.Concrete

{

public class Content

{

public int ContentID { get; set; }

public string ContentValue { get; set; }

public DateTime ContentDate { get; set; }

//ContentYazar ve ContentBaşlık gibi yapılar ilişki içinde tanımlanacaktır

public int HeadingID { get; set; }

public virtual Heading Heading { get; set; }

}

}

Ders 7 : İlişkilerin Oluşturulması 2

Heading ile Writer arasındaki ilişkinin kurulması

Heading Class ı

namespace EntityLayer.Concrete

{

public class Heading

{

public int HeadingID { get; set; } //prop yaz iki defa tab a bas

public string HeadingName { get; set; }

public DateTime HeadingDate { get; set; }

//Category ile Heading arasındaki ilişki

public int CategoryID { get; set; }//Heading sınıfında CategoryID colonu oldu

public virtual Category Category { get; set; }

public int WriterID { get; set; }

public virtual Writer Writer { get; set; }

public ICollection<Content> Contents { get; set; }

}

}

Writer Class ı

namespace EntityLayer.Concrete

{

public class Writer

{

public int WriterID { get; set; }

public string WriterName { get; set; }

public string WriterSurName { get; set; }

public string WriterImage { get; set; }

public string WriterMail { get; set; }

public string WriterPassword { get; set; }

public ICollection<Heading> Headings { get; set; }

}

}

Writer ile Content arasındaki ilişkinin kurulması

Writer Class ı

namespace EntityLayer.Concrete

{

public class Writer

{

public int WriterID { get; set; }

public string WriterName { get; set; }

public string WriterSurName { get; set; }

public string WriterImage { get; set; }

public string WriterMail { get; set; }

public string WriterPassword { get; set; }

public ICollection<Heading> Headings { get; set; }

public ICollection<Content> Content { get; set; }

}

}

Content Class ı

namespace EntityLayer.Concrete

{

public class Content

{

public int ContentID { get; set; }

public string ContentValue { get; set; }

public DateTime ContentDate { get; set; }

//ContentYazar ve ContentBaşlık gibi yapılar ilişki içinde tanımlanacaktır

public int HeadingID { get; set; }

public virtual Heading Heading { get; set; }

public int WriterID { get; set; }

public virtual Writer Writer { get; set; }

}

}

Ders 8 : Key ve String Length Attributleri

Her sınıftaki id ifadesi key olarak tanımlanmalıdır

Bunu kullanmanın amacı kısıtlama ve düzenleme içindir

Solution Explorer🡪EntityLayer sağ tık🡪Manage Nuget Packages🡪Browse🡪Entity Framework 6 🡪Install

Artık sınıflarımıza Key ve StringLength ekleyebiliriz

About Class ı

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EntityLayer.Concrete

{

public class About

{

[Key]//Otomatik anahtar tanımlama

public int AboutID { get; set; }

[StringLength(1000)]//Karakter sınırı

public string AboutDetails1 { get; set; }

[StringLength(1000)]

public string AboutDetails2 { get; set; }

[StringLength(100)]

public string AboutImage1 { get; set; }

[StringLength(100)]

public string AboutImage2 { get; set; }

}

}

Diğer sınıflar içinde benzer tanımlamalar kodların içinde yapılmıştır

Ders 9 : Context Sınıfı

Artık DataAccessLayer katmanında işlemleri gerçekleştireceğiz.Bu katman içine EntityLayer de olduğu gibi Concrete isminde bir klasör tanımlayalım.

Concrete üzerinde sağ tık🡪 Add 🡪 New Class🡪Context.cs yi oluşturalım.

Class Context i public hale getirelim.

EntityLayer içerisinde bulunan sınıflara erişebilmek için DataAccessLayer üzerinde;

Solution Explorer🡪DataAccessLayer sağ tık🡪Manage Nuget Packages🡪Browse🡪Entity Framework 6 🡪Install diyoruz

Daha sonra DataAccessLayer içerisinde bulunan References sağ tık🡪Add Reference🡪Projects🡪EntityLayer aktif edilir.

Context Class ı

using EntityLayer.Concrete;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace DataAccessLayer.Concrete

{

//DbContext aktif etmek için hatanın üzerine gelip

//using.System.Data.Entity seçilir

public class Context : DbContext

{

//Sınıfları dahil edebilmek için sağ tık

//using EntityLayer.Concrete eklenir

//sql e yansıyacak tablolar

public DbSet<About> Abouts { get; set; }

//About sınıf ismi iken Abouts tablo ismi olacak

public DbSet<Category> Categories { get; set; }

public DbSet<Contact> Contacts { get; set; }

public DbSet<Content> Contents { get; set; }

public DbSet<Heading> Headings { get; set; }

public DbSet<Writer> Writers { get; set; }

}

}

Ders 10 : Web Config Yapılandırması ve Bağlantı Ayarı

MVCProjeKampi Katmanında bulunan Web.config dosyası açılır

</system.web>// bu komutun altına alttaki komutlar yazılır

<connectionStrings>

<add name="Context" connectionString="data source=DESKTOP-8GIU2US\SQLEXPRESS;

initial catalog=DbMvcKamp; integrated security=true;"

providerName="System.Data.SqlClient"/>

</connectionStrings>

Ders 11 : Migration ve Veri Tabanının Oluşturulması

View Menüsü🡪Other Windows🡪Package Manager Console açılır

Açılan kısımda PM> enable-migrations komutu yazılır(DataAccesLayer ı seç) enter a bas

Çıkan pencerede (Configuration.cs) de

Aşagıdaki false kısmı true olarak değiştir

public Configuration()

{

AutomaticMigrationsEnabled = true;s

}

Package manager console a PM> update-database yazılır

SQL i açın databases altında DbMvcKamp database i gelmiştir.

Database Diagram ı açarak tabloları ekleyin

Ders 12 : Add Migration İşlemi

Writer ile Content arasında ilişkiyi devre dışı bırakmış idik.Tekrar bu sınıflara gelerek bazı düzenlemeler yapalım

Content Class ı Edit Kısmı

public int? WriterID { get; set; }

//int? ifadesi yazar id olmayabilir anlamına geliyor

public virtual Writer Writer { get; set; }

Writer Class ı Edit Kısmı

public ICollection<Content> Content { get; set; }

daha sonra package manager console a PM> add-migration mig1 yazılıp yeni migration DataAccessLayeri içinde Migrations içinde oluştu.

Daha sonra package manager console içerisinde tekrar dan update-database denilir

Content tablosuna sql den bakınca WriterID gelmiştir.

Yeni diagram kurulup ilişki kurulabilir.

Ders 13 : Örnek SQL Veri Girişleri