

docs (Solution Folder and Physical Folder)

LICENSE (Solution Folder and Physical Folder)

src (Solution Folder and Physical Folder)

tests (Solution Folder and Physical Folder)

docs

ReadMe.docs

src

Core (Solution Folder)

Framework (Solution Folder and Physical Folder)

Infrastructure (Solution Folder)

Persistence (Solution Folder)

Presentation (Solution Folder)

Core

Application (Class Library)

Constants (Class Library)

Domain (Class Library)

Resources (Class Library)

**ViewModels** / Dtos (Class Library)

Framework

Dtat (Class Library)

Dtat.Seedwork.Abstractions (Class Library)

Dtat.StateMachine (Class Library)

Dtat.StateMachine.Abstractions (Class Library)

Infrastructure

Infrastructure (Class Library)

Persistence

Persistence (Class Library)

Presentation

Api (ASP.NET Core Web API)

**Server** (**ASP.NET Core Razor Pages** / MVC)

**Project Dependencies:**

Dtat

Dtat.Seedwork.Abstractions

Dtat.StateMachine.Abstractions

Dtat.StateMachine

Dtat.StateMachine.Abstractions

Resources

Constants

Domain

Dtat

Constants

Resources

Dtat.Seedwork.Abstractions

Application

Domain

Infrastructure

Domain

ViewModels

Domain

Constants

Resources

Persistence

Dtat

Domain

Constants

NuGets:

Microsoft.EntityFrameworkCore.Tools

Microsoft.EntityFrameworkCore.Proxies

Microsoft.EntityFrameworkCore.Sqlite

Microsoft.EntityFrameworkCore.SqlServer

Server

Dtat

Domain

Constants

Resources

ViewModels

Persistence

NuGets:

System.Drawing.Common

Microsoft.EntityFrameworkCore.Design

Microsoft.AspNetCore.Authentication.Google

**نحوه تعریف یک Enum**

فرض کنید که می‌خواهیم یک Enum به نام Gender ایجاد نماییم، از آنجایی که این Enum مربوط به کاربر و Identity می‌باشد، در پوشه Features و در پوشه Identity، پوشه‌ای به نام Enums ایجاد کرده و یک Enum به نام GenderEnum در داخل آن ایجاد می‌کنیم:

public enum GenderEnum : int

{

Unspecified = 0,

Male = 1,

Female = 2,

}

برای این‌که روی عناوین Enum و فعال و غیرفعال کردن آن‌ها کنترل داشته باشیم، معادل آن، یک کلاس و به طبع آن، یک جدول در بانک اطلاعاتی ایجاد می‌کنیم.

نام کلاس را Gender نام‌گذاری می‌کنیم.

چون می‌خواهیم پروژه ما چند زبانه باشد، در کلاس Gender دو Property به نام‌های CultureId و Culture تعریف می‌کنیم.

در کلاس Gender، یک Property به نام Code از جنس GenderEnum تعریف می‌کنیم.

در کلاس Gender، یک Property به نام Title از جنس string، برای ذخیره‌سازی عنوان هر Enum تعریف می‌کنیم.

**ایجاد روابط بین مدل‌ها**

در این مثال خاص، یک رابطه یک به چند بین Culture و Gender‌ داریم و یک رابطه یک به چند بین Gender و UserProfile. این روابط را ایجاد می‌کنیم:

در کلاس Culture:

public Culture(…) : base()

{

Genders =

new System.Collections.Generic.List<Identity.Gender>();

}

public virtual System.Collections.Generic.IList

<Identity.Gender> Genders { get; private set; }

در کلاس UserProfile:

public System.Guid GenderId { get; set; }

public virtual Gender? Gender { get; set; }

در کلاس Gender:

public Gender(…) : base()

{

UserProfiles =

new System.Collections.Generic.List<UserProfile>();

}

public virtual System.Collections.Generic.IList

<UserProfile> UserProfiles { get; private set; }

**ایجاد و ویرایش کلاس‌های Configuration**

حال در پروژه Persistence، و در داخل پوشه Configurations، و سپس در پوشه Features و بعد در پوشه Identity، اقدام به ایجاد کلاس GenderConfiguration می‌کنیم:

builder

.HasIndex(current => new { current.CultureId, current.Code })

.Is Unique(unique: true)

;

builder

.HasIndex(current => new { current.CultureId, current.Title })

.IsUnique(unique: true)

;

builder

.HasMany(current => current.UserProfiles)

.WithOne(other => other.Gender)

.IsRequired(required: true)

.HasForeignKey(other => other.GenderId)

.OnDelete(deleteBehavior:

Microsoft.EntityFrameworkCore.DeleteBehavior.NoAction)

;

در کلاس Culture Configuration تغییرات ذیل را اعمال می‌کنیم:

builder

.HasMany(current => current.Genders)

.WithOne(other => other.Culture)

.IsRequired(required: true)

.HasForeignKey(other => other.CultureId)

.OnDelete(deleteBehavior:

Microsoft.EntityFrameworkCore.DeleteBehavior.NoAction)

;

**ایجاد Property در DatabaseContext**

در کلاس DatabaseContext یک Property به نام Genders و به شکل ذیل ایجاد می‌کنیم:

public Microsoft.EntityFrameworkCore.DbSet

<Domain.Features.Identity.Gender> Genders { get; set; }

**ایجاد Seed Data**

در پروژه Persistence و در پوشه Extensions و در داخل کلاس ModelBuilderExtensions اقدام به ایجاد Seed Data برای Gender می‌کنیم:

**تست پروژه**

یک‌بار پروژه را اجرا می‌کنیم که از ایجاد جداول و اطلاعات پایه اطمینان حاصل کنیم.

احتمالا به خطا برخورد می‌کنیم! چرا که برای UserProfile و در داخل Constructor آن، پس از اعمال تغییرات، GenderId را نیز به عنوان پارامتر الزامی مشخص کرده‌ایم، لذا در زمان Seed Data برای UserProfile با مشکل مواجه می‌شویم.

این مشکل را برطرف کرده و مجددا پروژه را اجرا می‌کنیم.

استلایدهای آماده تستی:

<https://www.tutorialrepublic.com/examples/images/slide1.png>

<https://www.tutorialrepublic.com/examples/images/slide2.png>

<https://www.tutorialrepublic.com/examples/images/slide3.png>

<https://smartslider3.com/wp-content/uploads/2018/11/WordPresscarousel-840x441.png>