**نسخه: ۱.۶**

**راهنمای اشکال/نمادها در این آموزش**

Solution Folder : ()

Project : [[]]

Simple Folder : []

**مرحله اول**

در مسیر:

(src) → (Core) → [[Domain]] → [SeedWork]

کلاس(هایی) به نام(های) ذیل، ایجاد می‌کنیم:

**TableEntity**

**مرحله دوم**

در مسیر:

(src) → (Core) → [[Domain]] → [Features] → [Common] → [Enums]

یک Enum (هایی) به نام(های) ذیل، ایجاد می‌کنیم:

**BaseTableEnum**

* Gender = 1000,
* Language = 2000,
* Religion = 3000,
* LanguageLevel = 4000,
* MaritalStatus = 5000,
* SocialNetwork = 6000,
* EducationDegree = 7000,
* MilitaryServiceStatus = 8000,

**BaseTableTypeEnum**

* Enum = 1
* NotEnum = 0

**مرحله سوم**

در مسیر:

(src) → (Core) → [[Domain]] → [Features] → [Common]

کلاس(هایی) به نام(های) ذیل، ایجاد می‌کنیم:

**BaseTable**

**LocalizedBaseTable**

ارتباط یک به چند، بین این دو کلاس را برقرار می‌کنیم، یعنی برای این دو کلاس، Navigation Property های لازم را درج (ایجاد) می‌کنیم.

**مرحله چهارم**

در مسیر:

(src) → (Persistence) → [[Persistence]]

و در کلاس DatabaseContext، پراپرتی‌های مربوط به کلاس‌های Entity خود را ایجاد می‌کنیم:

DbSet<Domain.Features.Common.BaseTable> **BaseTables** { get; set; }

DbSet<Domain.Features.Common.LocalizedBaseTable> **LocalizedBaseTables** { get; set; }

**مرحله پنجم**

در مسیر:

(src) → (Persistence) → [[Persistence]] → [Features] → [Common] → [Configurations]

کلاس(هایی) به نام(های) ذیل، ایجاد می‌کنیم:

**BaseTableConfiguration**

**LocalizedBaseTableConfiguration**

**مرحله ششم**

در مسیر:

(src) → (Persistence) → [[Persistence]] → [Extensions]

و در کلاس ModelBuilderExtensions، و در تابع اصلی ()Seed، تابع ذیل را فراخوانی می‌کنیم، تا اطلاعات مربوط به Seed Data جداول در زمان ایجاد بانک اطلاعاتی، ایجاد گردند:

**SeedBaseTables**(modelBuilder: modelBuilder);

**مرحله هفتم**

در مسیر:

(src) → (Core) → [[Domain]] → [Features] → [Common]

کلاس(هایی) به نام(های) ذیل، ایجاد می‌کنیم:

**BaseTableItem**

**LocalizedBaseTableItem**

ارتباط یک به چند، بین این دو کلاس را برقرار می‌کنیم، یعنی برای این دو کلاس، Navigation Property های لازم را درج (ایجاد) می‌کنیم.

ارتباط یک به چند، بین کلاس BaseTable و BaseTableItem را نیز برقرار می‌کنیم، یعنی برای این دو کلاس، Navigation Property های لازم را درج (ایجاد) می‌کنیم.

**مرحله هشتم**

در مسیر:

(src) → (Persistence) → [[Persistence]]

و در کلاس DatabaseContext، پراپرتی‌های مربوط به کلاس‌های Entity خود را ایجاد می‌کنیم:

DbSet<Domain.Features.Common.BaseTableItem> **BaseTableItems** { get; set; }

DbSet<Domain.Features.Common.LocalizedBaseTableItem> **LocalizedBaseTableItems** { get; set; }

**مرحله نهم**

در مسیر:

(src) → (Persistence) → [[Persistence]] → [Features] → [Common] → [Configurations]

کلاس(هایی) به نام(های) ذیل، ایجاد می‌کنیم:

**BaseTableItemConfiguration**

**LocalizedBaseTableItemConfiguration**

با توجه به رابطه یک به چند بین BaseTable و BaseTableItem، فایل ذیل را نیز باید تکمیل کرده و این ارتباط را ثبت نماییم:

**BaseTableConfiguration**

**مرحله دهم**

در مسیر:

(src) → (Persistence) → [[Persistence]] → [Extensions]

و در کلاس ModelBuilderExtensions، و در تابع اصلی ()Seed، تابع ذیل را برای تکمیل آیتم‌های اطلاعات پایه تکمیل می‌کنیم:

**SeedBaseTables**(modelBuilder: modelBuilder);

**نمونه:**

**BaseTable LocalizedBaseTable**

Marriage وضعیت‌های تاهل

Marriages

**BaseTableItem LocalizedBaseTableItem**

Single مجرد

Single

Married متاهل

Married

Divorced مطلقه

Divorced

**نمونه‌ای از ایجاد**

**ارتباط جداول پایه با جداول دیگر**

می‌خواهیم **سه** ارتباط یک به چند، از جدول BaseTableItem به جدول User ایجاد نماییم:

**مرحله اول**

در مسیر:

(src) → (Core) → [[Domain]] → [Features] → [Common]

و در کلاس(های) ذیل:

BaseTableItem

و در مسیر:

(src) → (Core) → [[Domain]] → [Features] → [Identity]

و در کلاس(های) ذیل:

User

پراپرتی‌های (Navigation Property) لازم را ایجاد و مقداردهی می‌کنیم:

**User:**

#region public System.Guid? MaritalStatusId { get; set; }

/// <summary>

/// وضعیت تاهل

/// </summary>

[System.ComponentModel.DataAnnotations.Display

(ResourceType = typeof(Resources.DataDictionary),

Name = nameof(Resources.DataDictionary.MaritalStatus))]

**public System.Guid? MaritalStatusId { get; set; }**

#endregion /public System.Guid? MaritalStatusId { get; set; }

#region public virtual Common.BaseTableItem? MaritalStatus { get; private set; }

/// <summary>

/// وضعیت تاهل

/// </summary>

[System.ComponentModel.DataAnnotations.Display

(ResourceType = typeof(Resources.DataDictionary),

Name = nameof(Resources.DataDictionary.MaritalStatus))]

**public virtual Common.BaseTableItem? MaritalStatus { get; private set; }**

#endregion /public virtual Common.BaseTableItem? MaritalStatus { get; private set; }

#region public System.Guid? LastEducationDegreeId { get; set; }

/// <summary>

/// آخرین مدرک تحصیلی

/// </summary>

[System.ComponentModel.DataAnnotations.Display

(ResourceType = typeof(Resources.DataDictionary),

Name = nameof(Resources.DataDictionary.LastEducationDegree))]

**public System.Guid? LastEducationDegreeId { get; set; }**

#endregion /public System.Guid? LastEducationDegreeId { get; set; }

#region public virtual Common.BaseTableItem? LastEducationDegree { get; private set; }

/// <summary>

/// آخرین مدرک تحصیلی

/// </summary>

[System.ComponentModel.DataAnnotations.Display

(ResourceType = typeof(Resources.DataDictionary),

Name = nameof(Resources.DataDictionary.LastEducationDegree))]

**public virtual Common.BaseTableItem? LastEducationDegree { get; private set; }**

#endregion /public virtual Common.BaseTableItem? LastEducationDegree { get; private set; }

#region public System.Guid? MilitaryServcieStatusId { get; set; }

/// <summary>

/// وضعیت نظام وظیفه

/// </summary>

[System.ComponentModel.DataAnnotations.Display

(ResourceType = typeof(Resources.DataDictionary),

Name = nameof(Resources.DataDictionary.MilitaryServcieStatus))]

**public System.Guid? MilitaryServcieStatusId { get; set; }**

#endregion /public System.Guid? MilitaryServcieStatusId { get; set; }

#region public virtual Common.BaseTableItem? MilitaryServcieStatus { get; private set; }

/// <summary>

/// وضعیت نظام وظیفه

/// </summary>

[System.ComponentModel.DataAnnotations.Display

(ResourceType = typeof(Resources.DataDictionary),

Name = nameof(Resources.DataDictionary.MilitaryServcieStatus))]

**public virtual Common.BaseTableItem? MilitaryServcieStatus { get; private set; }**

#endregion /public virtual Common.BaseTableItem? MilitaryServcieStatus { get; private set; }

**BaseTableItem:**

public virtual System.Collections.Generic.IList<Identity.User> **Users\_MaritalStatus** { get; private set; }

public virtual System.Collections.Generic.IList<Identity.User> **Users\_LastEducationDegree** { get; private set; }

public virtual System.Collections.Generic.IList<Identity.User> **Users\_MilitaryServcieStatus** { get; private set; }

**نکته:** باید دقت داشته باشیم که این سه فیلد (Collection) را در Constructor، ایجاد (new) نماییم!

**نکته:** باید دقت داشته باشیم که برای همه Navigation Property ها، چه تکی‌ها و چه لیستی‌ها، Set شان را private تعریف نماییم!

مرحله دوم

در مسیر:

(src) → (Persistence) → [[Persistence]] → [Features] → [Common] → [Configurations]

کلاس(های) ذیل را ویرایش می‌کنیم:

**BaseTableItemConfiguration**

builder

.HasMany(current => current.Users\_MaritalStatus)

.WithOne(other => other.MaritalStatus)

**// Note: Is Not Required!**

.IsRequired(required: false)

.HasForeignKey(other => other.MaritalStatusId)

.OnDelete(deleteBehavior:

Microsoft.EntityFrameworkCore.DeleteBehavior.NoAction)

;

builder

.HasMany(current => current.Users\_LastEducationDegree)

.WithOne(other => other.LastEducationDegree)

**// Note: Is Not Required!**

.IsRequired(required: false)

.HasForeignKey(other => other.LastEducationDegreeId)

.OnDelete(deleteBehavior:

Microsoft.EntityFrameworkCore.DeleteBehavior.NoAction)

;

builder

.HasMany(current => current.Users\_MilitaryServcieStatus)

.WithOne(other => other.MilitaryServcieStatus)

**// Note: Is Not Required!**

.IsRequired(required: false)

.HasForeignKey(other => other.MilitaryServcieStatusId)

.OnDelete(deleteBehavior:

Microsoft.EntityFrameworkCore.DeleteBehavior.NoAction)

;

**مرحله سوم**

در مسیر:

(src) → (Persistence) → [[ViewModels]] → [Pages] → [Account]

کلاس(های) ذیل را ویرایش می‌کنیم:

**UpdateProfileViewModel**

#region public System.Guid? MaritalStatusId { get; set; }

/// <summary>

/// وضعیت تاهل

/// </summary>

[System.ComponentModel.DataAnnotations.Display

(ResourceType = typeof(Resources.DataDictionary),

Name = nameof(Resources.DataDictionary.MaritalStatus))]

**[System.ComponentModel.DataAnnotations.Required**

(AllowEmptyStrings = false,

ErrorMessageResourceType = typeof(Resources.Messages.Validations),

ErrorMessageResourceName = nameof(Resources.Messages.Validations.Required))]

**public System.Guid? MaritalStatusId { get; set; }**

#endregion /public System.Guid? MaritalStatusId { get; set; }

#region public System.Guid? LastEducationDegreeId { get; set; }

/// <summary>

/// آخرین مدرک تحصیلی

/// </summary>

[System.ComponentModel.DataAnnotations.Display

(ResourceType = typeof(Resources.DataDictionary),

Name = nameof(Resources.DataDictionary.LastEducationDegree))]

**[System.ComponentModel.DataAnnotations.Required**

(AllowEmptyStrings = false,

ErrorMessageResourceType = typeof(Resources.Messages.Validations),

ErrorMessageResourceName = nameof(Resources.Messages.Validations.Required))]

**public System.Guid? LastEducationDegreeId { get; set; }**

#endregion /public System.Guid? LastEducationDegreeId { get; set; }

#region public System.Guid? MilitaryServcieStatusId { get; set; }

/// <summary>

/// وضعیت نظام وظیفه

/// </summary>

[System.ComponentModel.DataAnnotations.Display

(ResourceType = typeof(Resources.DataDictionary),

Name = nameof(Resources.DataDictionary.MilitaryServcieStatus))]

**[System.ComponentModel.DataAnnotations.Required**

(AllowEmptyStrings = false,

ErrorMessageResourceType = typeof(Resources.Messages.Validations),

ErrorMessageResourceName = nameof(Resources.Messages.Validations.Required))]

**public System.Guid? MilitaryServcieStatusId { get; set; }**

#endregion /public System.Guid? MilitaryServcieStatusId { get; set; }

**مرحله چهارم**

در مسیر:

(src) → (Presentation) → [[Server]] → [Infrastructure]

فایل(های) ذیل را ویرایش می‌کنیم:

**SelectLists.cs**

#region GetActiveItemsAsync()

public static async System.Threading.Tasks.Task

<Microsoft.AspNetCore.Mvc.Rendering.SelectList> GetActiveItemsAsync(Persistence.DatabaseContext

databaseContext, **Domain.Features.Common.Enums.BaseTableEnum baseTableEnum**, object? selectedValue = null)

{

var currentUICultureLcid = Domain.Features

.Common.CultureEnumHelper.GetCurrentUICultureLcid();

var list =

await

databaseContext.BaseTableItems

.Where(current => current.IsActive)

**.Where(current => current.BaseTable != null**

**&& current.BaseTable.Code == baseTableEnum)**

.Select(current => new ViewModels.Shared.IdNameViewModel<System.Guid?>

{

Id = current.Id,

KeyName = current.KeyName,

Ordering = current.Ordering,

#pragma warning disable CS8602

Title =

(current.LocalizedBaseTableItems == null

||

current.LocalizedBaseTableItems.Count == 0)

?

null

:

current.LocalizedBaseTableItems.FirstOrDefault

(other => other.Culture != null && other.Culture.Lcid == currentUICultureLcid).Title,

#pragma warning restore CS8602

})

.ToListAsync()

;

var **orderedList** =

list

.OrderBy(current => current.Ordering)

.ThenBy(current => current.Name)

.ToList()

;

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

var emptyItem =

new ViewModels.Shared.IdNameViewModel

<System.Guid?>(**title: Resources.DataDictionary.SelectAnItem**);

orderedList.Insert(index: 0, item: emptyItem);

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

var result =

new Microsoft.AspNetCore.Mvc.Rendering

.SelectList(items: **orderedList**, selectedValue: **selectedValue**,

dataTextField: ViewModels.Shared.IdNameViewModel<int>.DataTextField,

dataValueField: ViewModels.Shared.IdNameViewModel<int>.DataValueField);

return result;

}

#endregion /GetActiveItemsAsync()

**مرحله پنجم**

در مسیر:

(src) → (Presentation) → [[Server]] → [Pages] → [Account]

فایل(های) ذیل را ویرایش می‌کنیم:

**UpdateProfile.cshtml.cs**

public Microsoft.AspNetCore.Mvc.Rendering.SelectList? **MaritalStatusesSelectList** { get; set; }

public Microsoft.AspNetCore.Mvc.Rendering.SelectList? **EducationDegreesSelectList** { get; set; }

public Microsoft.AspNetCore.Mvc.Rendering.SelectList? **MilitaryServcieStatusesSelectList** { get; set; }

private async System.Threading.Tasks.Task UpdateSelectListsAsync(

System.Guid? maritalStatusId,

System.Guid? lastEducationDegreeId,

System.Guid? militaryServcieStatusId)

{

**MaritalStatusesSelectList** =

await

Infrastructure.SelectLists.GetActiveItemsAsync

(databaseContext: DatabaseContext, baseTableEnum:

Domain.Features.Common.Enums.BaseTableEnum.MaritalStatus,

selectedValue: maritalStatusId);

**EducationDegreesSelectList** =

await

Infrastructure.SelectLists.GetActiveItemsAsync

(databaseContext: DatabaseContext, baseTableEnum:

Domain.Features.Common.Enums.BaseTableEnum.EducationDegree,

selectedValue: lastEducationDegreeId);

**MilitaryServcieStatusesSelectList** =

await

Infrastructure.SelectLists.GetActiveItemsAsync

(databaseContext: DatabaseContext, baseTableEnum:

Domain.Features.Common.Enums.BaseTableEnum.MilitaryServcieStatus,

selectedValue: militaryServcieStatusId);

}

**مرحله ششم**

در مسیر:

(src) → (Presentation) → [[Server]] → [Pages] → [Account]

فایل(های) ذیل را ویرایش می‌کنیم:

**UpdateProfile.cshtml**

<dtat-full-select asp-for="**ViewModel.MaritalStatusId**" asp-items="@(**Model.MaritalStatusesSelectList**)" />

<dtat-full-select asp-for="**ViewModel.LastEducationDegreeId**" asp-items="@(**Model.EducationDegreesSelectList**)" />

<dtat-full-select asp-for="**ViewModel.MilitaryServcieStatusId**" asp-items="@(**Model.MilitaryServcieStatusesSelectList**)" />

**ایجاد صفحات مربوط به BaseTable**

**مرحله اول**

در مسیر:

(src) → (Core) → [[ViewModels]] → [Pages] → [Features] → [Common] → [Admin]

پوشه‌ای به نام BaseTables ایجاد می‌کنیم.

**مرحله دوم**

در پوشه ایجاد شده، کلاس‌های ذیل را ایجاد می‌کنیم:

**UpdateViewModel**

**IndexItemViewModel**

**DetailsOrDeleteViewModel**

**مرحله سوم**

در مسیر:

(src) → (Presentation) → [[Server]] → [Pages] → [Features] → [Common] → [Admin]

پوشه‌ای به نام BaseTables ایجاد می‌کنیم.

**مرحله چهارم**

در پوشه ایجاد شده، فایل‌های ذیل را ایجاد می‌کنیم:

**Index.cshtml**

**Index.cshtml.cs**

**Update.cshtml**

**Update.cshtml.cs**

**Details.cshtml**

**Details.cshtml.cs**