Table of Contents

[Basics 2](#_Toc119191825)

[Current Resources with Changes 2](#_Toc119191826)

[New Resources 2](#_Toc119191827)

[/Core/DB 2](#_Toc119191828)

[UnitOfWork 2](#_Toc119191829)

[IUnitOfWork.cs 2](#_Toc119191830)

[UnitOfWork.cs 2](#_Toc119191831)

[DataContext.cs 3](#_Toc119191832)

[/Core/Extensions 4](#_Toc119191833)

[ServiceExtensions.cs 4](#_Toc119191834)

[RegisterRepos Extension Method 4](#_Toc119191835)

[/Core/Repositories 4](#_Toc119191836)

[All Repositories 4](#_Toc119191837)

[UsersRepository 4](#_Toc119191838)

[IUsersRepository.cs 4](#_Toc119191839)

[UsersRepository.cs 4](#_Toc119191840)

[MessageRepository 5](#_Toc119191841)

[/Core/BusinessLogic 5](#_Toc119191842)

[LikesBusinesLogic.cs 5](#_Toc119191843)

[MessageBusinessLogic.cs 5](#_Toc119191844)

[SignalRBusinessLogic.cs 5](#_Toc119191845)

[UsersBusinessLogic.cs 5](#_Toc119191846)

[/Core/SignalR 5](#_Toc119191847)

[MessageHub.cs 5](#_Toc119191848)

[/Core/Dto 5](#_Toc119191849)

[MessageDto.cs 5](#_Toc119191850)

[/Core/Dto/AutoMapper 6](#_Toc119191851)

[AutoMapperProgile.cs 6](#_Toc119191852)

# Basics

|  |  |
| --- | --- |
| WorkingFolder | Copy the content of “Site-06-SignalR” in “Site-07-UnitOfWork FinishingTouces” and issue   * dotnet build |
| Cloudinary Setting | appsetting.json is not checked in so make sure to bring in this from “Site-01-Basics” |

# Current Resources with Changes

1. /Core/Extensions/ServiceExtensions
2. /Core/Repositories
3. /Core/BusinessLogic
4. /Core/ActionFilters/LogUserActivity
5. /Core/SignalR/MessageHub
6. /Core/Dto/MessageDto
7. /Core/Dato/AutoMapper
8. /Core/DB/DataContext

# New Resources

1. /Core/DB/UnitOfWork

# /Core/DB

## UnitOfWork

Create a new folder inside DB folder and name it UnitOfWork. Then create an interface and the concrete class

### IUnitOfWork.cs

using System.Threading.Tasks;

using MSC.Api.Core.Repositories;

namespace MSC.Api.Core.DB.UnitOfWork;

public interface IUnitOfWork

{

    IUsersRepository UsersRepo { get; }

    ILikesRepository LikesRepo { get; }

    IMessageRepository MessageRepo { get; }

    ISignalRRepository SignalRRepo { get; }

    Task<bool> Complete();

    bool HasChanges();

}

### UnitOfWork.cs

using System.Threading.Tasks;

using AutoMapper;

using MSC.Api.Core.Repositories;

namespace MSC.Api.Core.DB.UnitOfWork;

public class UnitOfWork : IUnitOfWork

{

    private readonly DataContext \_context;

    private readonly IMapper \_mapper;

    public UnitOfWork(DataContext context, IMapper mapper)

    {

        \_context = context;

        \_mapper = mapper;

    }

    public IUsersRepository UsersRepo => new UsersRepository(\_context, \_mapper);

    public ILikesRepository LikesRepo => new LikesRepository(\_context);

    public IMessageRepository MessageRepo => new MessageRepository(\_context, \_mapper);

    public ISignalRRepository SignalRRepo => new SignalRRepository(\_context, \_mapper);

    public async Task<bool> Complete()

    {

        return await \_context.SaveChangesAsync() > 0;

    }

    public bool HasChanges()

    {

        return \_context.ChangeTracker.HasChanges();

    }

}

## DataContext.cs

Add the following class inside DataContext.cs. This is a fix from GitHub EF Core issue

/// <summary>

/// Will convert the date times to UTC

/// </summary>

public static class UtcDateAnnotation

{

    private const String IsUtcAnnotation = "IsUtc";

    private static readonly ValueConverter<DateTime, DateTime> UtcConverter =

      new ValueConverter<DateTime, DateTime>(v => v, v => DateTime.SpecifyKind(v, DateTimeKind.Utc));

    private static readonly ValueConverter<DateTime?, DateTime?> UtcNullableConverter =

      new ValueConverter<DateTime?, DateTime?>(v => v, v => v == null ? v : DateTime.SpecifyKind(v.Value, DateTimeKind.Utc));

    public static PropertyBuilder<TProperty> IsUtc<TProperty>(this PropertyBuilder<TProperty> builder, Boolean isUtc = true) =>

      builder.HasAnnotation(IsUtcAnnotation, isUtc);

    public static Boolean IsUtc(this IMutableProperty property) =>

      ((Boolean?)property.FindAnnotation(IsUtcAnnotation)?.Value) ?? true;

    /// <summary>

    /// Make sure this is called after configuring all your entities.

    /// </summary>

    public static void ApplyUtcDateTimeConverter(this ModelBuilder builder)

    {

        foreach (var entityType in builder.Model.GetEntityTypes())

        {

            foreach (var property in entityType.GetProperties())

            {

                if (!property.IsUtc())

                {

                    continue;

                }

                if (property.ClrType == typeof(DateTime))

                {

                    property.SetValueConverter(UtcConverter);

                }

                if (property.ClrType == typeof(DateTime?))

                {

                    property.SetValueConverter(UtcNullableConverter);

                }

            }

        }

    }

}

And then include it at the end in OnModelCreating

    protected override void OnModelCreating(ModelBuilder builder)

    {

        base.OnModelCreating(builder);

        //dont foget to add migrations

        //dotnet ef migrations add MessageEntityAdded -o Core/DB/Migrations

        //and then either issue command "dotnet ef database update"

        //or do dontnet run. For this check program.cs "CUSTOM: Seed Data Start" section

        //Due to use of Identity

        CreateUserRole(builder);

        CreateUserLike(builder);

        CreateMessage(builder);

        //keep this at the bottom. UTC date fix per EF github

        builder.ApplyUtcDateTimeConverter();

    }

# /Core/Extensions

## ServiceExtensions.cs

### RegisterRepos Extension Method

Comment the repositories registered and add IUnitOfWork

        //repositories

        //using unit of work so individual repos commented

        /\*

        services.AddScoped<IUsersRepository, UsersRepository>();

        services.AddScoped<ILikesRepository, LikesRepository>();

        services.AddScoped<IMessageRepository, MessageRepository>();

        services.AddScoped<ISignalRRepository, SignalRRepository>();

        \*/

        services.AddScoped<IUnitOfWork, UnitOfWork>();

# /Core/Repositories

## All Repositories

Go through all the repositories and their interface and comment out the SaveAllSync Method

## UsersRepository

Both in the interface and the class change the “RegisterAsync” method to be non Async

### IUsersRepository.cs

    void Register(AppUser appUser);

### UsersRepository.cs

    public void Register(AppUser appUser)

    {

        if (appUser == null)

            throw new ValidationException("Invalid user");

        \_context.Users.Add(appUser);

    }

Also go to method “GetMessageThread” and remove following line from it

await \_context.SaveChangesAsync();

## MessageRepository

Method GetMessageThread is now returning MessageDto and the query is optimized to use ProjectTo

Method GetMessageForUser has been changed to use the ProjectTo towards the top to optimize the query.

# /Core/BusinessLogic

## LikesBusinesLogic.cs

Remove likeRepository and UsersRepository and instead use UnitOfWork. Also, update the calls in different methods accordingly.

## MessageBusinessLogic.cs

Remove MessageRepository and UsersRepository and instead use UnitOfWork. Also, update the calls in different methods accordingly.

## SignalRBusinessLogic.cs

Remove SignalRRepository and all its usage and instead use UnitOfWork

## UsersBusinessLogic.cs

Remove UsersRepository and instead use UnitOfWork and update all methods accordingly

For method LogUserActivityAsync change the LastActive to UtcNow

Then go to GetMessageThread and add \_uow.Complete to it since we removed it from the Repository method.

    public async Task<IEnumerable<MessageDto>> GetMessageThread(int currentUserId, int receipientId)

    {

        var messages = await \_uow.MessageRepo.GetMessageThread(currentUserId, receipientId);

        if (messages == null)

            return null;

        if (\_uow.HasChanges())

            await \_uow.Complete();

        var messagesDto = \_mapper.Map<IEnumerable<MessageDto>>(messages);

        return messagesDto;

    }

# /Core/SignalR

## MessageHub.cs

Remove use of UsersRepository and in its place use UnitOfWork

# /Core/Dto

## MessageDto.cs

Add two new properties and also make these JsonIgnore so that these do not get sent back

    [JsonIgnore]

    public bool ReceipientDeleted { get; set; }

    [JsonIgnore]

    public bool SenderDeleted { get; set; }

# /Core/Dto/AutoMapper

## AutoMapperProgile.cs

Remove the fix for the utc date. Check Data Context for the utc handling

        //not using any more since DBContext has a function to fix the utc date issue

        //We'll have the Z at the end of the dates meaning utc

        //Map\_DateTime\_Add\_Z\_ForUtc();