

# Лабораторна робота №4-6

## Тема. Створення RESTful API

**Мета.** Ознайомитися з процесом створення клієнт-серверної архітектури. Навчитися створювати RESTful API для взаємодії між клієнтом і сервером. Розвинути вміння проектувати та реалізовувати ендпоїнти для типових CRUD-операцій.

## ЗАВДАННЯ

**Завдання 1 - Створити ASP.NET Core Minimal API.**

**Завдання 2 - Створити ASP.NET Core Web API.**

**Завдання 3 - Спроєктувати архітектури Controller – Service – Repository.**

**Завдання 4 - Додати MongoDB та реалізація CRUD через Repository/Service. Підключити MongoDB замість in-memory списків і реалізувати CRUD-операції на рівні архітектури Controller–Service–Repository.**

**Завдання 5 - Впровадити Dependency Injection (DI) для всіх шарів і налаштувати AutoMapper для автоматичного мапінгу моделей.**

**Завдання 6 – Authentication & Authorization**

## Хід роботи:

### Завдання 1

Розроблено проект за допомогою C# ASP.Net Core.

Проект реалізує CRUD операції для власного API на тему: програма для керування підписками, надано три види ресурсів:

PeopleItems – модель, яка представляє користувача з властивостями: Id – унікальний ідентифікатор, Name – ім'я користувача та Email – електронна адреса користувача,

SubscritionItems – модель, яка представляє об'єкт підписки з

Змін.	Аркуш	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04		
Розроб.	Башук О. Ю.				Літ.	Арк.	Актуалів
Перевір.	Жереб Д. В.				H	1	62
Н. Контр.					ХПК		
Затверд.							

Клієнт-серверна архітектура ПЗ. Створення RESTful API

властивостями: Id – унікальний ідентифікатор, OwnerId – Ідентифікатор власника підписки та Service – сервіс на якому зроблено підписку,

MessageItems – модель, яка представляє об'єкт повідомлення про підписку з властивостями: Id – унікальний ідентифікатор, Title – повідомлення, OwnerId – Ідентифікатор власника підписки та Service – сервіс на якому зроблено підписку, SubId – ідентифікатор підписки.

Лістинг:

Створено каталог Models із такими файлами:

People.cs:

```
namespace CRUDAPI.Models
{
    /// <summary>
    /// Модель що представляє користувачів програми для управління під
    /// писками
    /// </summary>
    public class PeopleItems
    {
        /// <summary>
        /// Унікальний ідентифікатор
        /// </summary>
        public int Id { get; set; }
        /// <summary>
        /// Ім'я або логін користувача
        /// </summary>
        public string Name { get; set; }
        /// <summary>
        /// Поштова адреса користувача
        /// </summary>
        public string Email { get; set; }
    }
}
```

Subscription.cs:

```
namespace CRUDAPI.Models
{
    /// <summary>
    /// Модель, яка описує підписку, яку може мати користувач
    /// </summary>
    public class SubscriptionItems
    {
        /// <summary>
```

Змін.	Арк.	№ докум.	Підпис	Дата

```

/// Унікальний ідентифікатор підписки
/// </summary>
public int Id { get; set; }
/// <summary>
/// Ідентифікатор власника підписки
/// </summary>
public int OwnerId { get; set; }
/// <summary>
/// Сервіс для кого виконана підписка
/// </summary>
public string Service { get; set; }
}
}

```

Message.cs:

```

namespace CRUDAPI.Models
{
    /// <summary>
    /// Модель реалізує сповіщення, кі отримує користувач про підписку
    /// </summary>
    public class MessageItems
    {
        /// <summary>
        /// Унікальний ідентифікатор повідомлення
        /// </summary>
        public int Id { get; set; }
        /// <summary>
        /// Заголовок повідомлення
        /// </summary>
        public string Title { get; set; }
        /// <summary>
        /// Кому надсилається повідомлення
        /// </summary>
        public int OwnerId { get; set; }
        /// <summary>
        /// Індентифікатор підписки
        /// </summary>
        public int SubId { get; set; }
    }
}

```

З цьому каталогі також розміщено файл для валідації даних:

```

using CRUDAPI.Data;
using System.Text.RegularExpressions;
namespace CRUDAPI.Models
{
    public static class Valid

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

{
    const string compareField = @"^[\w-]+@[\\w-]+\.\w+";
    public static bool StringField(string field, int minLen, string nameField,
out string error)
    {
        if (string.IsNullOrEmpty(field))
        {
            error = $"Текстове поле {nameField} пусте";
            return false;
        }
        else if (field.Length < minLen)
        {
            error = $"Занадто коротке поле {nameField}";
            return false;
        }
        error = string.Empty;
        return true;
    }
    public static bool Email(string email, out string error)
    {
        if (!Regex.IsMatch(email, compareField))
        {
            error = "Невірна поштова адреса";
            return false;
        }
        error = string.Empty;
        return true;
    }
    public static bool CheckOwner(int id)
    {
        var owner = Elements.peopleItems.FirstOrDefault(A => A.Id == id);
        if (owner is null)
            return false;
        return true;
    }
    public static bool Check_Subscribe_Owner(int Subid, int OwnerId, out
string error)
    {
        var subscription = Elements.subscriptionItems.FirstOrDefault(A =>
A.Id == Subid);
        if (subscription is null)
        {
            error = "Не знайдено вказаної підписки";
            return false;
        }
        else if (subscription.OwnerId != OwnerId)

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        {
            error = "Неправильний власник підписки";
            return false;
        }
        error = string.Empty;
        return true;
    }
}
}

```

Створено каталог Data, в якому містяться списки із елементів класів моделей:

```
using CRUDAPI.Models;
```

```

namespace CRUDAPI.Data
{
    public static class Elements
    {
        public static List<PeopleItems> peopleItems = new List<PeopleItems>()
        {
            new PeopleItems() {Id = 1, Name = "Kate", Email = "Kate@Email.com" },
            new PeopleItems() {Id = 2, Name = "John", Email = "John@Email.com" },
            new PeopleItems() {Id = 3, Name = "Jane", Email = "Jane@Email.com" },
            new PeopleItems() {Id = 4, Name = "Steve", Email = "Steve@Email.com" },
            new PeopleItems() {Id = 5, Name = "Alex", Email = "Alex@Email.com" };
        };
        public static List<SubscriptionItems> subsriptionItems = new List<SubscriptionItems>()
        {
            new SubscriptionItems() {Id = 1, OwnerId = 1, Service = "Netflix" },
            new SubscriptionItems() {Id = 2, OwnerId = 1, Service = "Xbox" },
            new SubscriptionItems() {Id = 3, OwnerId = 1, Service = "Amazon" },
            new SubscriptionItems() {Id = 4, OwnerId = 2, Service = "Steam" },
            new SubscriptionItems() {Id = 5, OwnerId = 2, Service = "Google" },
            new SubscriptionItems() {Id = 6, OwnerId = 2, Service = "Amazon" },
            new SubscriptionItems() {Id = 7, OwnerId = 3, Service = "YouTube" },
            new SubscriptionItems() {Id = 8, OwnerId = 3, Service = "Netflix" },
            new SubscriptionItems() {Id = 9, OwnerId = 4, Service = "YouTube" },
            new SubscriptionItems() {Id = 10, OwnerId = 4, Service = "Netflix" },
            new SubscriptionItems() {Id = 11, OwnerId = 4, Service = "Megogo" },
            new SubscriptionItems() {Id = 12, OwnerId = 4, Service = "AppleFilms" },
            new SubscriptionItems() {Id = 13, OwnerId = 5, Service = "AppleSerice" },
        };
    }
}
```

Змін.	Арк.	№ докум.	Підпис	Дата	Арк.
					5

```

        new SubscriptionItems(){Id = 14, OwnerId = 5, Service = "Xbox"},  

        new SubscriptionItems(){Id = 15, OwnerId = 5, Service = "AppleFilms"},  

    };  

    public static List<MessageItems> messageItems = new List<MessageItems>()  

    {  

        new MessageItems(){Id = 1,Title = "До оплати три дні", OwnerId =  

1, SubId = 1 },  

        new MessageItems(){Id = 2,Title = "До оплати два дні", OwnerId =  

1, SubId = 1 },  

        new MessageItems(){Id = 3,Title = "До оплати один день", OwnerI  

d = 1, SubId = 1 },  

        new MessageItems(){Id = 4,Title = "Підписка прострочена", Own  

erId = 2, SubId = 5 },  

        new MessageItems(){Id = 5,Title = "Підписку скасовано", OwnerI  

d = 3, SubId = 7 },  

        new MessageItems(){Id = 6,Title = "Підписку просрочена", Owner  

Id = 4, SubId = 9 },  

        new MessageItems(){Id = 7,Title = "Підпску продовжено", OwnerI  

d = 4, SubId = 9 },  

        new MessageItems(){Id = 8,Title = "Підписка прострочена", Owne  

rId = 4, SubId = 11 },  

        new MessageItems(){Id = 9,Title = "Підписку скасовано", OwnerId  

= 4, SubId = 11 },  

        new MessageItems(){Id = 10,Title = "До оплаї підписки тиждень",  

OwnerId = 5, SubId = 15 },  

    };  

}

```

Створено каталог EndPoints із класами для реалізації endpoints моделей:

PeopleEndPoints.cs:

```

using CRUDAPI.Models;
using CRUDAPI.Data;

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

namespace CRUDAPI.EndPoints
{
    public static class PeopleEndpoints
    {
        ///<summary>
        ///Реєструє всі маршрути до ресурсу People
        ///</summary>
        ///<param name="app">Екземпляр<see cref="WebApplication"/>. </para
m>
        public static void mapPeople(this WebApplication app)
        {
            app.MapGet("/peoples/{id:int}", (int id) =>
            {
                var peopleItem = Elements.peopleItems.FirstOrDefault(A => A.Id
                == id);
                return peopleItem is null ? Results.NotFound() : Results.Ok(peopl
eItem);
            })
            .WithSummary("Отримати корисувача за Id")
            .Produces<PeopleItems>(StatusCodes.Status200OK)
            .Produces(StatusCodes.Status404NotFound)
            .WithName("GETPeople")
            .WithDescription("Отримати користувача за Id")
            .WithOpenApi();
            app.MapGet("/peoples", (string? name, string? email) =>
            {
                IEnumerable<PeopleItems> items = Elements.peopleItems;
                if (name is not null)
                    items = items.Where(A => A.Name == name);
                if (email is not null)
                    items = items.Where(A => A.Email == email);
                return items.Count() != 0 ? Results.Ok(items) : Results.NotFound(
);
            })
            .WithSummary("Отримати всіх корисувачів або конкретизувати з
а допомогою фільтрів")
            .Produces<PeopleItems>(StatusCodes.Status200OK)
            .Produces(StatusCodes.Status404NotFound)
            .WithName("GETPeoples")
            .WithDescription("Отримання всіх користувачів або конкретного з
а фільтром")
            .WithOpenApi();
            app.MapPost("/peoples", (PeopleItems peopleItem) =>
            {
                if (!Valid.StringField(peopleItem.Name, 5, "Name", out var error1))

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        return Results.BadRequest(error1);
    if(!Valid.Email(peopleItem.Email,out var error2))
        return Results.BadRequest(error2);
    peopleItem.Id = Elements.peopleItems.Max(A => A.Id) + 1;
    Elements.peopleItems.Add(peopleItem);
    return Results.Created($"/peoples/{peopleItem.Id}", peopleItem);
}
.WithSummary("Надіслати конкретного користувача в список")
.Produces<PeopleItems>(StatusCodes.Status201Created)
.Produces(StatusCodes.Status400BadRequest)
.WithName("POSTPeople")
.WithDescription("Відправити конкретного користувача")
.WithOpenApi();
app.MapPut("/peoples/{id:int}", (int id, PeopleItems updatePeople) =>
{
    var people = Elements.peopleItems.Find(A => A.Id == id);
    if (people is null)
        return Results.NotFound();
    if (!Valid.StringField(updatePeople.Name, 5, "Name", out var error1
)))
        return Results.BadRequest(error1);
    if (!Valid.Email(updatePeople.Email, out var error2))
        return Results.BadRequest(error2);
    people.Name = updatePeople.Name;
    people.Email = updatePeople.Email;
    return Results.Ok(people);
}
.WithSummary("Оновити повністю конкретного користувача")
.Produces<PeopleItems>(StatusCodes.Status200OK)
.Produces(StatusCodes.Status404NotFound)
.Produces(StatusCodes.Status400BadRequest)
.WithName("PUTPeople")
.WithDescription("Оновити повністю конкретного користувача")
.WithOpenApi();
app.MapPatch("/peoples/{id:int}", (int id, PeopleItems updatePeople) =
> {
    var people = Elements.peopleItems.Find(A => A.Id == id);
    if (people is null)
        return Results.NotFound();
    if (updatePeople.Name is not null)
    {
        if (!Valid.StringField(updatePeople.Name, 5, "Name", out var error
1))
            return Results.BadRequest(error1);
        people.Name = updatePeople.Name;
    }
})

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        }
        if (updatePeople.Email is not null)
        {
            if (!Valid.Email(updatePeople.Email, out var error2))
                return Results.BadRequest(error2);
            people.Email = updatePeople.Email;
        }
        return Results.Ok(people);
    })
    .WithSummary("Оновити частково конкретного користувача")
    .Produces<PeopleItems>(StatusCodes.Status200OK)
    .Produces(StatusCodes.Status404NotFound)
    .Produces(StatusCodes.Status400BadRequest)
    .WithName("PATCHPeople")
    .WithDescription("Оновити частково конкретного користувача")
    .WithOpenApi();
app.MapDelete("/peoples/{id:int}", (int id) =>
{
    var people = Elements.peopleItems.FirstOrDefault(A => A.Id == id);
    if (people is null)
        return Results.NotFound();
    Elements.peopleItems.Remove(people);
    return Results.NoContent();
})
    .WithSummary("Видалити конкретного користувача")
    .Produces(StatusCodes.Status204NoContent)
    .Produces(StatusCodes.Status404NotFound)
    .WithName("DELETEPeople")
    .WithDescription("Видалити конкретного користувача")
    .WithOpenApi();
}
}
}

SuscriptionsEndPoints.cs:
using CRUDAPI.Models;
using CRUDAPI.Data;
namespace CRUDAPI.EndPoints
{
    public static class SubscriptionsEndPoints
    {
        ///<summary>
        ///Реєструє всі маршрути до ресурсу Subscriptions
        ///</summary>
        ///<param name="app">Екземпляр<see cref="WebApplication"/>.</param>
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

public static void mapSubscriptions(this WebApplication app)
{
    app.MapGet("/subs/{id:int}", (int id) =>
    {
        var subItem = Elements.subscriptionItems.FirstOrDefault(A => A.
Id == id);
        return subItem is null ? Results.NotFound() : Results.Ok(subItem)
    ;
    })
    .WithSummary("Отримати підписку")
    .Produces<SubscriptionItems>(StatusCodes.Status200OK)
    .Produces(StatusCodes.Status404NotFound)
    .WithName("GETSub")
    .WithDescription("Отримати конкретну підписку за Id")
    .WithOpenApi();
    app.MapGet("/subs", (int? ownerId, string? service) =>
    {
        IEnumerable<SubscriptionItems> items = Elements.subscriptionIt
ems;
        if (ownerId is not null)
            items = items.Where(A => A.OwnerId == ownerId);
        if (service is not null)
            items = items.Where(A => A.Service == service);

        return items.Count() != 0 ? Results.Ok(items) : Results.NotFound(
);
    })
    .WithSummary("Отримати всі підписки або конкретні")
    .Produces<SubscriptionItems>(StatusCodes.Status200OK)
    .Produces(StatusCodes.Status404NotFound)
    .WithName("GETSubs")
    .WithDescription("Отримати всі підписки або конкретні за доп
омогою фільтрів")
    .WithOpenApi();
    app.MapPost("/subs", (SubscriptionItems subscription) =>
    {
        if (!Valid.StringField(subscription.Service, 3, "Service", out var er
ror))
            return Results.BadRequest(error);
        if (!Valid.CheckOwner(subscription.OwnerId))
            return Results.BadRequest("Немає вказаного 'OwnerId'");
        subscription.Id = Elements.subscriptionItems.Max(A => A.Id) +
1;
        Elements.subscriptionItems.Add(subscription);
        return Results.Created($""/subs/{subscription.Id}", subscription);
    })
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        })
        .WithSummary("Відправити підписку")
        .Produces<SubscriptionItems>(StatusCodes.Status201Created)
        .Produces(StatusCodes.Status400BadRequest)
        .WithName("POSTSub")
        .WithDescription("Відправити конкретну підписку")
        .WithOpenApi();
    app.MapPut("/subs/{id:int}", (int id, SubscriptionItems updateSub)
=>
{
    var subscription = Elements.subscriptionItems.Find(A => A.Id ==
id);
    if (subscription is null)
        return Results.NotFound();
    if (!Valid.CheckOwner(subscription.OwnerId))
        return Results.BadRequest("Немає вказаного 'ownerId'");
    if (!Valid.StringField(updateSub.Service, 3, "Service", out var err
or))
        return Results.BadRequest(error);
    subscription.OwnerId = updateSub.OwnerId;
    subscription.Service = updateSub.Service;
    return Results.Ok(subscription);
})
        .WithSummary("Оновити повністю підписку")
        .Produces<SubscriptionItems>(StatusCodes.Status200OK)
        .Produces(StatusCodes.Status404NotFound)
        .Produces(StatusCodes.Status400BadRequest)
        .WithName("PUTSub")
        .WithDescription("Оновити повністю конкретну підписку за Id
")
        .WithOpenApi();

    app.MapPatch("/subs/{id:int}", (int id, SubscriptionItems updateSub)
=>
{
    var subscription = Elements.subscriptionItems.Find(A => A.Id ==
id);
    if (subscription is null)
        return Results.NotFound();
    if (!Valid.StringField(updateSub.Service, 3, "Service", out var err
or))
        return Results.BadRequest(error);
    if (updateSub.OwnerId != 0)
    {
        if (!Valid.CheckOwner(subscription.OwnerId))

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        return Results.BadRequest("Немає вказаного 'ownerId'");
        subscription.OwnerId = updateSub.OwnerId;
    }
    if(updateSub.Service is not null)
        subscription.Service = updateSub.Service;
    return Results.Ok(subscription);
})
.WithSummary("Оновити частково підписку")
.Produces<SubscriptionItems>(StatusCodes.Status200OK)
.Produces(StatusCodes.Status404NotFound)
.Produces(StatusCodes.Status400BadRequest)
.WithName("PATCHSub")
.WithDescription("Оновити частково конкретну підписку за Id")
)
.WithOpenApi();
app.MapDelete("/subs/{id:int}", (int id) =>
{
    var sub = Elements.subscriptions.FirstOrDefault(A => A.Id ==
= id);
    if (sub is null)
        return Results.NotFound();
    Elements.subscriptions.Remove(sub);
    return Results.NoContent();
})
.WithSummary("Видалити підписку")
.Produces(StatusCodes.Status204NoContent)
.Produces(StatusCodes.Status404NotFound)
.WithName("DELETESub")
.WithDescription("Видалити конкретну підписку за Id")
.WithOpenApi();
}
}
}

MessageEndPoints.cs:
using CRUDAPI.Data;
using CRUDAPI.Models;
namespace CRUDAPI.EndPoints
{
    public static class MessageEndPoints
    {
        ///<summary>
        ///Реєструє всі маршрути до ресурсу Subscriptions
        ///</summary>
        ///<param name="app">Екземпляр<see cref="WebApplication"/>.</pa
ram>
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

public static void mapMessages(this WebApplication app)
{
    app.MapGet("/messages/{id:int}", (int id) =>
    {
        var mItem = Elements.messageItems.FirstOrDefault(A => A.Id == id);
        return mItem is null ? Results.NotFound() : Results.Ok(mItem);
    })
        .WithSummary("Отримати повідомлення")
        .Produces<MessageItems>(StatusCodes.Status200OK)
        .Produces(StatusCodes.Status404NotFound)
        .WithName("GETmessage")
        .WithDescription("Отримати конкретне повідомлення за Id")
        .WithOpenApi();
    app.MapGet("/messages", (int? ownerId, int? subId, string? title) =>
    {
        IEnumerable<MessageItems> items = Elements.messageItems;
        if (ownerId is not null)
            items = items.Where(A => A.OwnerId == ownerId);
        if (subId is not null)
            items = items.Where(A => A.SubId == subId);
        if (title is not null)
            items = items.Where(A => A.Title == title);
        return items.Count() != 0 ? Results.Ok(items) : Results.NotFound();
    })
        .WithSummary("Отримати всі повідомлення або конкретні")
        .Produces<MessageItems>(StatusCodes.Status200OK)
        .Produces(StatusCodes.Status404NotFound)
        .WithName("GETmessages")
        .WithDescription("Отримати всі повідомлення або використати фільтри для конкретизації")
        .WithOpenApi();
    app.MapPost("/messages", (MessageItems message) =>
    {
        if (!Valid.StringField(message.Title, 10, "Title", out var error))
            return Results.BadRequest(error);
        if (!Valid.Check_Subscribe_Owner(message.SubId, message.OwnerId, out string error1))
            return Results.BadRequest(error1);
        message.Id = Elements.messageItems.Max(A => A.Id) + 1;
        Elements.messageItems.Add(message);
        return Results.Created($""/messages/{message.Id}", message);
    })
        .WithSummary("Зафіксувати повідомлення в колекції")
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    .Produces<MessageItems>(StatusCodes.Status201Created)
    .Produces(StatusCodes.Status404NotFound)
    .WithName("POSTmessage")
    .WithDescription("Відправити повідомлення в колекцію")
    .WithOpenApi();
    app.MapPut("/messages/{id:int}", (int id, MessageItems updateM) =
    >
    {
        var m = Elements.messageItems.Find(A => A.Id == id);
        if (m is null)
            return Results.NotFound();
        if (!Valid.Check_Subscribe_Owner(updateM.SubId, updateM.Ow
        nerId, out string error1))
            return Results.BadRequest(error1);
        if (!Valid.StringField(updateM.Title, 10, "Title", out var error))
            return Results.BadRequest(error);
        m.Title = updateM.Title;
        m.OwnerId = updateM.OwnerId;
        m.SubId = updateM.SubId;
        return Results.Ok(m);
    })
    .WithSummary("Оновити повністю повідомлення")
    .Produces<MessageItems>(StatusCodes.Status200OK)
    .Produces(StatusCodes.Status404NotFound)
    .Produces(StatusCodes.Status400BadRequest)
    .WithName("PUTmessage")
    .WithDescription("Оновити повністю повідомлення в колекції з
        a Id")
        .WithOpenApi();
    app.MapPatch("/messages/{id:int}", (int id, MessageItems updateM) =
    =>
    {
        var m = Elements.messageItems.Find(A => A.Id == id);
        if (m is null)
            return Results.NotFound();
        int OId = updateM.OwnerId != 0? updateM.OwnerId : m.OwnerId
        ;
        int SId = updateM.SubId != 0? updateM.SubId : m.SubId;
        if (!Valid.Check_Subscribe_Owner(SId, OId, out string error1))
            return Results.BadRequest(error1);
        if (updateM.OwnerId != 0)
            m.OwnerId = updateM.OwnerId;
        if(updateM.SubId != 0)
            m.SubId = updateM.SubId;
        if (updateM.Title is not null)

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    {
        if (!Valid.StringField(updateM.Title, 10, "Title", out var error))
            return Results.BadRequest(error);
        m.Title = updateM.Title;
    }
    return Results.Ok(m);
})
.WithSummary("Оновити частково повідомлення")
.Produces<MessageItems>(StatusCodes.Status200OK)
.Produces(StatusCodes.Status404NotFound)
.Produces(StatusCodes.Status400BadRequest)
.WithName("PATCHmessage")
.WithDescription("Оновити частково повідомлення в колекції з
а Id")
.WithOpenApi();
app.MapDelete("/messages/{id:int}", (int id) =>
{
    var m = Elements.messageItems.FirstOrDefault(A => A.Id == id);
    if (m is null)
        return Results.NotFound();
    Elements.messageItems.Remove(m);
    return Results.NoContent();
})
.WithSummary("Видалити повідомлення")
.Produces(StatusCodes.Status204NoContent)
.Produces(StatusCodes.Status404NotFound)
.WithName("DELETEmessage")
.WithDescription("Видалити повідомлення з колекції за Id")
.WithOpenApi();
}
}
}

```

Підключенно в program.cs всі endpoints:

```

using CRUDAPI.EndPoints;
var builder = WebApplication.CreateBuilder(args);
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen(options =>
{
    var xmlFile = $"{System.Reflection.Assembly.GetExecutingAssembly().
GetName().Name}.xml";
    var xmlPath = Path.Combine(AppContext.BaseDirectory, xmlFile);
    options.IncludeXmlComments(xmlPath);
});
var app = builder.Build();
app.mapPeople();

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

app.MapSubscriptions();
app.MapMessages();
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}
app.Run();

```

Протестовано роботу програму за допомогою swagger:  
Запит1

The screenshot shows a Swagger UI interface. The 'Request URL' field contains `https://localhost:7211/peoples/1`. The 'Server response' section shows a successful 200 status code. The 'Response body' is a JSON object with fields `id`, `name`, and `email`, all set to their respective values for the user with ID 1. The 'Response headers' section includes `content-type: application/json; charset=utf-8`, `date: Wed, 08 Oct 2025 09:14:29 GMT`, and `server: Kestrel`. There are 'Copy' and 'Download' buttons for the response body.

Рисунок 1 – Отримання користувача за Id

Запит 2

The screenshot shows a Swagger UI interface. The 'Request URL' field contains `https://localhost:7211/peoples/10`. The 'Server response' section shows an error 404 status code with the message 'Error: response status is 404'. The 'Response headers' section includes `content-length: 0`, `date: Wed, 08 Oct 2025 09:17:56 GMT`, and `server: Kestrel`.

Рисунок 2 – Отримання користувача за відсутнім Id

Запит 3

The screenshot shows a Swagger UI interface. The 'Request URL' field contains `https://localhost:7211/peoples/1`. The 'Server response' section shows a successful 200 status code. The 'Response body' is a JSON object with fields `id`, `name`, and `email`, all updated to new values. The 'Response headers' section includes `content-type: application/json; charset=utf-8`, `date: Wed, 08 Oct 2025 09:19:21 GMT`, and `server: Kestrel`. There are 'Copy' and 'Download' buttons for the response body.

Рисунок 3 – Коректне оновлення користувача

Запит 4

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						16

Request URL  
**https://localhost:7211/peoples/1**

Server response

Code	Details
400	Error: response status is 400 Response body "Текстове поле Name пусте"

Response headers  
content-type: application/json; charset=utf-8  
date: Wed, 08 Oct 2025 09:20:59 GMT  
server: Kestrel

Рисунок 4 – Помилка під час оновлення

### Запит5

Request URL  
**https://localhost:7211/peoples/1**

Server response

Code	Details
400	Error: response status is 400 Response body "Невірна поштова адреса"

Response headers  
content-type: application/json; charset=utf-8  
date: Wed, 08 Oct 2025 09:22:11 GMT  
server: Kestrel

Рисунок 5 – Помилка валідації – невірна електронна адреса

### Запит 6

Request URL  
**https://localhost:7211/peoples/3**

Server response

Code	Details
200	Response body <pre>{   "id": 3,   "name": "Jane",   "email": "NewEmail@gmail.com" }</pre>

Response headers  
content-type: application/json; charset=utf-8  
date: Wed, 08 Oct 2025 09:24:25 GMT  
server: Kestrel

Рисунок 6 – Часткове оновлення користувача

### Запит 7

Request URL  
**https://localhost:7211/peoples/1**

Server response

Code	Details
204	Response headers date: Wed, 08 Oct 2025 09:25:49 GMT server: Kestrel

Рисунок 7 – Видалення користувача

### Запит 8

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						17

Request URL  
**https://localhost:7211/people**

Server response

Code	Details
200	Response body <pre>[   {     "id": 2,     "name": "John",     "email": "John@email.com"   },   {     "id": 3,     "name": "Jane",     "email": "NewEmail@gmail.com"   },   {     "id": 4,     "name": "Steve",     "email": "Steve@email.com"   },   {     "id": 5,     "name": "Alex",     "email": "Alex@email.com"   }]</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:26:36 GMT server: Kestrel</pre>

Рисунок 8 – Отримано список всіх користувачів

### Запит 9

Request URL  
**https://localhost:7211/people**

Server response

Code	Details
200	Response body <pre>[   {     "id": 2,     "name": "John",     "email": "John@email.com"   },   {     "id": 3,     "name": "Jane",     "email": "NewEmail@gmail.com"   },   {     "id": 4,     "name": "Steve",     "email": "Steve@email.com"   },   {     "id": 5,     "name": "Alex",     "email": "Alex@email.com"   }]</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:26:36 GMT server: Kestrel</pre>

Рисунок 9 – Отримано всіх користувачів

### Запит 10

Request URL  
**https://localhost:7211/people?name=John**

Server response

Code	Details
200	Response body <pre>[   {     "id": 2,     "name": "John",     "email": "John@email.com"   }]</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:27:53 GMT server: Kestrel</pre>

Рисунок 10 – Отримано конкретного користувача

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						18

## Запит 11

Request URL	
<code>https://localhost:7211/people?name=John&amp;email=invalidEmail1%40gmail.com</code>	
Server response	
Code	Details
404	Error: response status is 404
Response headers	
<code>content-length: 0 date: Wed, 08 Oct 2025 09:30:25 GMT server: Kestrel</code>	

### Рисунок 11 – Користувача не знайдено

Запит 12

Code	Details
201	<p>Response body</p> <pre>{     "id": 6,     "name": "Tomas",     "email": "Tom@email.com" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:33:26 GMT location: /people/6 server: Kestrel</pre> <div style="display: flex; justify-content: space-between;"><span> Copy</span><span>Download</span></div>

## Рисунок 12 – Створення нового користувача

Запит 13

Code	Details
400	<p>Error: response status is 400</p> <p>Response body</p> <div style="background-color: #f0f0f0; padding: 10px;"><p>"Невірна поштова адреса"</p></div> <p>Response headers</p> <div style="background-color: #f0f0f0; padding: 10px;"><p>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:38:10 GMT server: Kestrel</p></div> <p><a href="#">Copy</a> <a href="#">Download</a></p>

Рисунок 13 – Спроба додавання користувача з невалідною адресою

Запит 14

Code	Details
200	<p>Response body</p> <pre>{   "id": 10,   "ownerId": 4,   "service": "Netflix" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:40:19 GMT server: Kestrel</pre> <div style="text-align: right;"><a href="#">Copy</a> <a href="#">Download</a></div>

Рисунок 14 – Отримання конкретної підписки

Запит 15

Змін.	Арк.	№ докум.	Підпис

ЛР.ОК24.П231.03.04

Aprk.

19

Code	Details
404	Error: response status is 404 <b>Response headers</b> content-length: 0 date: Wed, 08 Oct 2025 09:41:48 GMT server: Kestrel

Рисунок 15 – спроба отримати неіснуючу підписку

## Запит 16

Request URL  
`https://localhost:7211/subs/1`

Server response

Code	Details
200	<p>Response body</p> <pre>{ "id": 1, "ownerId": 1, "service": "NewService" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:47:47 GMT server: Kestrel</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p>

## Рисунок 16 - Оновлення підписки

Запит 17

Request URL	
<code>https://localhost:7211/subs/1</code>	
Server response	
Code	Details
400	Error: response status is 400 Response body <code>"Немас указанного ownerId"</code>
<a href="#">Copy</a> <a href="#">Download</a>	
Response headers	
<code>content-type: application/json; charset=utf-8</code> <code>date: Wed, 08 Oct 2025 09:49:11 GMT</code> <code>server: Kestrel</code>	

Рисунок 17 – Оновлення підписки з Id користувача, якого не існує

Запит 18

Request URL  
<https://localhost:7211/subs/10>

Server response

Code	Details
200	<p>Response body</p> <pre>{     "id": 10,     "ownerId": 4,     "service": "NewService" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:51:49 GMT server: Kestrel</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p>

Рисунок 18 – Часткове оновлення користувача

Запит 19

					<b>ЛР.ОК24.ПІ231.03.04</b>	Арк.
						20
Змін.	Арк.	№ докум.	Підпис	Дата		

Request URL  
**https://localhost:7211/subs/10**

Server response

Code	Details
400	Error: response status is 400 Response body "Текстове поле Service пусте" <div style="text-align: right;"><a href="#">Edit</a> <a href="#">Download</a></div>

Response headers

content-type: application/json; charset=utf-8
date: Wed, 08 Oct 2025 09:52:55 GMT
server: Kestrel

Рисунок 19 – Спроба оновлення поля підписки на пусте

### Запит 20

Request URL  
**https://localhost:7211/subs/15**

Server response

Code	Details
204	Response headers date: Wed, 08 Oct 2025 09:54:15 GMT server: Kestrel

Рисунок 20 – Видалення підписки

### Запит 21

Request URL  
**https://localhost:7211/subs**

Server response

Code	Details
200	Response body <pre>{     "service": "YouTube" }, {     "id": 10,     "ownerId": 4,     "service": "NewService" }, {     "id": 11,     "ownerId": 4,     "service": "Megogo" }, {     "id": 12,     "ownerId": 4,     "service": "AppleFilms" }, {     "id": 13,     "ownerId": 5,     "service": "AppleSerice" }, {     "id": 14,     "ownerId": 5,     "service": "Xbox" }</pre> <div style="text-align: right;"><a href="#">Edit</a> <a href="#">Download</a></div>

Response headers

content-type: application/json; charset=utf-8
date: Wed, 08 Oct 2025 09:56:08 GMT
server: Kestrel

Рисунок 21 – Отримання списку всіх підписок

### Запит 22

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						21

Request URL  
**https://localhost:7211/subs?ownerId=1&service=Xbox**

Server response

Code	Details
200	Response body <pre>[   {     "id": 2,     "ownerId": 1,     "service": "Xbox"   }]</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:57:32 GMT server: Kestrel</pre>

Рисунок 22 – Отримання підписки за вказаними полями

### Запит 23

Request URL  
**https://localhost:7211/subs**

Server response

Code	Details
201	Response body <pre>{   "id": 15,   "ownerId": 1,   "service": "Xbox" }</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 09:58:50 GMT location: /subs/15 server: Kestrel</pre>

Рисунок 23 – Створення нової підписки

### Запит 24

Request URL  
**https://localhost:7211/subs**

Server response

Code	Details
400	Error: response status is 400 Response body <pre>"Немає вказаного 'ownerId'"</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 10:00:53 GMT server: Kestrel</pre>

Рисунок 24 – Спроба додавання нової підписки з неіснуючим користувачем

### Запит 25

Request URL  
**https://localhost:7211/subs**

Server response

Code	Details
400	Error: response status is 400 Response body <pre>"Занадто коротке поле Service"</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Wed, 08 Oct 2025 10:04:19 GMT server: Kestrel</pre>

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ПІ231.03.04**

Арк.

22

## Рисунок 25 – Спроба створення підписки з невалідним сервісом

### Запит 26

Request URL  
`https://localhost:7211/messages/10`

Server response

Code Details

200 Response body

```
{ "id": 10, "title": "До оплати підписки тиждень", "ownerId": 5, "subId": 15 }
```

Response headers

```
content-type: application/json; charset=utf-8
date: Wed, 08 Oct 2025 10:06:45 GMT
server: Kestrel
```

Рисунок 26 – Отримання конкретного повідомлення

### Запит 27

Request URL  
`https://localhost:7211/messages/11`

Server response

Code Details

404 Error: response status is 404

Response headers

```
content-length: 0
date: Wed, 08 Oct 2025 10:08:27 GMT
server: Kestrel
```

Рисунок 27 – Спроба отримання неіснуючої підписки

### Запит 28

Request URL  
`https://localhost:7211/messages/1`

Server response

Code Details

200 Response body

```
{ "id": 1, "title": "Підписка просрочена", "ownerId": 1, "subId": 1 }
```

Response headers

```
content-type: application/json; charset=utf-8
date: Wed, 08 Oct 2025 10:10:53 GMT
server: Kestrel
```

Рисунок 28 – Оновлення вмісту повідомлення

### Запит 29

Request URL  
`https://localhost:7211/messages/1`

Server response

Code Details

400 Error: response status is 400

Response body

```
"Неправильний власник підписки"
```

Response headers

```
content-type: application/json; charset=utf-8
date: Thu, 09 Oct 2025 17:27:58 GMT
server: Kestrel
```

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ПІ231.03.04**

Арк.

23

Рисунок 29 – Спроба відправити підписнику нагадування не про те повідомлення

### Запит 30

Request URL  
`https://localhost:7211/messages/5`

Server response

Code	Details
200	<p>Response body</p> <pre>{     "id": 5,     "title": "Підпису скасовано",     "ownerId": 3,     "subId": 7 }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 09 Oct 2025 17:34:31 GMT server: Kestrel</pre>

Responses

Рисунок 30 – Оновлення підписки в користувача

### Запит 31

Request URL  
`https://localhost:7211/messages/5`

Server response

Code	Details
400	<p>Error: response status is 400</p> <p>Response body</p> <pre>"Неправильний власник підписки"</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 09 Oct 2025 17:35:13 GMT server: Kestrel</pre>

Рисунок 31 – Спроба оновлення невірної підписки користувача

### Запит 32

Request URL  
`https://localhost:7211/messages/10`

Server response

Code	Details
204	<p>Response headers</p> <pre>date: Thu, 09 Oct 2025 17:39:59 GMT server: Kestrel</pre>

Responses

Рисунок 32 – Видалення повідомлення про підписку

### Запит 33

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						24

Request URL  
**https://localhost:7211/messages**

Server response

Code	Details
200	Response body <pre>[{"ownerId": 3, "subId": 7}, {"id": 6, "title": "Підпису просрочена", "ownerId": 4, "subId": 9}, {"id": 7, "title": "Підпису продовжено", "ownerId": 4, "subId": 9}, {"id": 8, "title": "Підписка просрочена", "ownerId": 4, "subId": 11}, {"id": 9, "title": "Підпису скасовано", "ownerId": 4, "subId": 11}],</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Thu, 09 Oct 2025 17:40:49 GMT server: Kestrel</pre>

Рисунок 33 -Отримання списку усіх повідомлень

### Запит 34

Request URL  
**https://localhost:7211/messages?ownerId=1**

Server response

Code	Details
200	Response body <pre>[{"id": 2, "title": "До оплати два дні", "ownerId": 4, "subId": 1}, {"id": 3, "title": "До оплати один день", "ownerId": 4, "subId": 1}],</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Thu, 09 Oct 2025 17:43:09 GMT server: Kestrel</pre>

Рисунок 34 – Отримання всіх повідомлень до користувача

### Запит 35

Request URL  
**https://localhost:7211/messages?subId=5**

Server response

Code	Details
200	Response body <pre>[{"id": 4, "title": "Підписка просрочена", "ownerId": 2, "subId": 5}],</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Thu, 09 Oct 2025 17:44:10 GMT server: Kestrel</pre>

Рисунок 35 – Отримання всіх повідомлень про конкретну підписку

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						25

## Запит 36

Request URL  
`https://localhost:7211/messages`

Server response

Code Details

400 Undocumented Error: response status is 400

Response body  
"Текстове поле Title пусте"

Response headers  
content-type: application/json; charset=utf-8  
date: Thu, 09 Oct 2025 17:45:17 GMT  
server: Kestrel

Рисунок 36 – Спроба відправити пусте повідомлення

## Запит 37

Request URL  
`https://localhost:7211/messages`

Server response

Code Details

201 Response body

```
{  
    "id": 10,  
    "title": "Відновлено підписку",  
    "ownerId": 5,  
    "subId": 15  
}
```

Response headers  
content-type: application/json; charset=utf-8  
date: Thu, 09 Oct 2025 17:46:23 GMT  
location: /messages/10  
server: Kestrel

Рисунок 37 – Додано повідомлення для користувача та його підписки

## Запит 38

Request URL  
`https://localhost:7211/messages`

Server response

Code Details

400 Undocumented Error: response status is 400

Response body  
"Неправильний власник підписки"

Response headers  
content-type: application/json; charset=utf-8  
date: Thu, 09 Oct 2025 17:47:09 GMT  
server: Kestrel

Рисунок 38 – Спроба відправити повідомлення користувачу не про його підписку

## Завдання 2

Розроблено проект за допомогою C# ASP.Net Core WebAPI.

Проект реалізує CRUD операції для власного API на тему: програма для керування підписками, надано три види ресурсів:

							Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		ЛР.ОК24.ПІ231.03.04	26

PeopleItems – модель, яка представляє користувача з властивостями: Id – унікальний ідентифікатор, Name – ім'я користувача та Email – електронна адреса користувача,

SubscriptionItems – модель, яка представляє об'єкт підписки з властивостями: Id – унікальний ідентифікатор, OwnerId – Ідентифікатор власника підписки та Service – сервіс на якому зроблено підписку, Status – перелік, для відображення станів підписки,

MessageItems – модель, яка представляє об'єкт повідомлення про підписку з властивостями: Id – унікальний ідентифікатор, Title – повідомлення, OwnerId – Ідентифікатор власника підписки та Service – сервіс на якому зроблено підписку, SubId – ідентифікатор підписки.

Реалізовано валідацію в класові PeopleItems за допомогою Data.Anotation, для класів SubscriptionItems, MessageItems за допомогою FluentValidator та винесено логіку в окремий каталог Validator.

Лістинг:

People.cs:

```
using System.ComponentModel.DataAnnotations;
namespace LW_4_2.Models
{
    /// <summary>
    /// Модель що представляє користувачів програми для управління під
    /// писками
    /// </summary>
    public class PeopleItems
    {
        /// <summary>
        /// Унікальний ідентифікатор
        /// </summary>
        public int Id { get; set; }
        /// <summary>
```

Змін.	Арк.	№ докум.	Підпис	Дата

```

/// Ім'я або логін користувача
/// </summary>
[Required(ErrorMessage = "Поле 'Name' Обов'язкове для заповнення")]
"")]
[RegularExpression(@"^[A-Z][a-z]{5,10}[0-
9]{0,5}$", ErrorMessage = "'Name' не відповідає вимогам:" +
"Перші велика англійська літера, далі від 5 до 10 малих англійсь-
ких символів та цифри за бажанням")]
public string? Name { get; set; }

/// <summary>
/// Поштова адреса користувача
/// </summary>
[Required(ErrorMessage = "Поле 'Email' є обов'язковим")]
[EmailAddress(ErrorMessage = "Не коректне значення 'Email'")]
public string? Email { get; set; }

}
}


```

### Subscription.cs:

```

namespace LW_4_2.Models
{
    /// <summary>
    /// Перелік статусів підписки
    /// </summary>
    public enum SubStatus
    {
        /// <summary>
        /// Статус очікування - Expectation - 0
        /// </summary>
        Expectation,
        /// <summary>

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

/// Статус активна - Active - 1
/// </summary>
Active,
/// <summary>
/// Статус просрочена - Overdue - 2
/// </summary>
Overdue,
/// <summary>
/// Статус Архівована - Archived - 3
/// </summary>
Archived
}

/// <summary>
/// Модель, яка описує підписку, яку може мати користувач
/// </summary>
public class SubscriptionItems
{
    /// <summary>
    /// Унікальний індентифікатор підписки
    /// </summary>
    public int? Id { get; set; }

    /// <summary>
    /// Ідентифікатор власника підписки
    /// </summary>
    public int? OwnerId { get; set; }

    /// <summary>
    /// Сервіс для кого виконана підписка
    /// </summary>
    public string? Service { get; set; }

    /// <summary>

```

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ПІ231.03.04**

Арк.

29

```

/// Статус в якому перебуває підписка користувача
/// </summary>
public SubStatus? Status { get; set; }

}
}

```

Message.cs:

```

namespace LW_4_2.Models
{
    /// <summary>
    /// Модель реалізує сповіщення, кі отримує користувач про підписку
    /// </summary>
    public class MessageItems
    {
        /// <summary>
        /// Унікальний ідентифікатор повідомлення
        /// </summary>
        public int Id { get; set; }

        /// <summary>
        /// Заголовок повідомлення
        /// </summary>
        public string? Title { get; set; }

        /// <summary>
        /// Кому надсилається повідомлення
        /// </summary>
        public int? OwnerId { get; set; }

        /// <summary>
        /// Індентифікатор підписки
        /// </summary>
        public int? SubId { get; set; }

    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```
}
```

PeopleData.cs:

```
using LW_4_2.Models;  
namespace LW_4_2.Data  
{  
    public static class PeopleData  
    {  
        public static List<PeopleItems> peopleItems = new List<PeopleItems>()  
        {  
            new PeopleItems() {Id = 1, Name = "Kate", Email = "Kate@Em  
ail.com"},  
            new PeopleItems() {Id = 2, Name = "John", Email = "John@Em  
ail.com"},  
            new PeopleItems() {Id = 3, Name = "Jane", Email = "Jane@Em  
ail.com"},  
            new PeopleItems() {Id = 4, Name = "Steve", Email = "Steve@E  
mail.com"},  
            new PeopleItems() {Id = 5, Name = "Alex", Email = "Alex@E  
mail.com"}  
        };  
    }  
}
```

SubData.cs:

```
using LW_4_2.Models;  
namespace LW_4_2.Data  
{  
    public static class SubData  
    {
```

Змін.	Арк.	№ докум.	Підпис	Дата

```

public static List<SubscriptionItems> subsriptionItems = new List<SubscriptionItems>()
{
    new SubscriptionItems(){Id = 1, OwnerId = 1, Service = "Netflix", Status = SubStatus.Active},
    new SubscriptionItems(){Id = 2, OwnerId = 1, Service = "Xbox", Status = SubStatus.Active},
    new SubscriptionItems(){Id = 3, OwnerId = 1, Service = "Amazon", Status = SubStatus.Active},
    new SubscriptionItems(){Id = 4, OwnerId = 2, Service = "Steam", Status = SubStatus.Active },
    new SubscriptionItems(){Id = 5, OwnerId = 2, Service = "Google", Status = SubStatus.Overdue},
    new SubscriptionItems(){Id = 6, OwnerId = 2, Service = "Amazon", Status = SubStatus.Active},
    new SubscriptionItems(){Id = 7, OwnerId = 3, Service = "YouTube", Status = SubStatus.Archived},
    new SubscriptionItems(){Id = 8, OwnerId = 3, Service = "Netflix", Status = SubStatus.Active},
    new SubscriptionItems(){Id = 9, OwnerId = 4, Service = "YouTube", Status = SubStatus.Active},
    new SubscriptionItems(){Id = 10, OwnerId = 4, Service = "Netflix", Status = SubStatus.Active},
    new SubscriptionItems(){Id = 11, OwnerId = 4, Service = "Mego", Status = SubStatus.Archived},
    new SubscriptionItems(){Id = 12, OwnerId = 4, Service = "AppleFilms", Status = SubStatus.Active},
    new SubscriptionItems(){Id = 13, OwnerId = 5, Service = "AppleService", Status = SubStatus.Active},

```

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ІІ231.03.04**

Арк.

32

```

        new SubscriptionItems(){Id = 14, OwnerId = 5, Service = "Xbo
x", Status = SubStatus.Active},
        new SubscriptionItems(){Id = 15, OwnerId = 5, Service = "Appl
eFilms", Status = SubStatus.Active},
    };
}
}

MessageData.cs:

using LW_4_2.Models;
namespace LW_4_2.Data
{
    public static class MessageData
    {
        public static List<MessageItems> messageItems = new List<MessageIt
ems>()
        {
            new MessageItems(){Id = 1,Title = "До оплати три дні", OwnerId =
1, SubId = 1 },
            new MessageItems(){Id = 2,Title = "До оплати два дні", OwnerId =
1, SubId = 1 },
            new MessageItems(){Id = 3,Title = "До оплати один день", OwnerI
d = 1, SubId = 1 },
            new MessageItems(){Id = 4,Title = "Підписка просрочена", Own
erId = 2, SubId = 5 },
            new MessageItems(){Id = 5,Title = "Підписку скасовано", OwnerI
d = 3, SubId = 7 },
            new MessageItems(){Id = 6,Title = "Підписку просрочена", Owner
Id = 4, SubId = 9 },
            new MessageItems(){Id = 7,Title = "Підписку продовжено", OwnerI
d = 4, SubId = 9 },
        }
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        new MessageItems(){Id = 8,Title = "Підписка прострочена", Owner
        rId = 4, SubId = 11 },
        new MessageItems(){Id = 9,Title = "Підписку скасовано", OwnerId
        = 4, SubId = 11 },
        new MessageItems(){Id = 10,Title = "До оплачі підписки тиждень",
        OwnerId = 5, SubId = 15 },
    };
}
}

```

MessageValidator.cs:

```

using FluentValidation;
using LW_4_2.Models;
using LW_4_2.Data;
namespace LW_4_2.Validator
{
    public class MessageValidator: AbstractValidator<MessageItems>
    {
        public MessageValidator()
        {
            RuleFor(x => x.Title)
                .NotEmpty().WithMessage("Поле 'Title' не може бути порожнім
")
                .Length(5, 100).WithMessage("Поле 'Title' має бути від трьох до
ста символів");
            RuleFor(x => x.SubId)
                .NotNull().WithMessage("Поле 'SubId' не може бути порожнім"
)
                .Must(SubId => SubData.subscriptions.Any(s => s.Id == SubI
d))
        }
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    .WithMessage("Не знайдено 'SubId' із таким індексом");
    RuleFor(x => x.OwnerId)
        .NotEmpty().WithMessage("Поле 'OwnerId' не можу бу порожні
        м")
        .Must(ownerId => PeopleData.peopleItems.Any(u => u.Id == own
        erId))
        .WithMessage("Немає людина із таким значенням 'Id'")
        .Must((mes, ownerId) =>
    {
        var sub = SubData.subscriptionItems.FirstOrDefault(s => s.Id =
        = mes.SubId);

        if (sub is null)
            return false;

        return sub.OwnerId == ownerId;
    }).WithMessage("Відсутня підписка за вказаним 'SubId' або для
        неї не існує користувача");
}
}
}

```

SubscriptionValidtor.cs:

```

using LW_4_2.Models;
using LW_4_2.Data;
using FluentValidation;
namespace LW_4_2.Validator
{
    public class SubscriptionValidator: AbstractValidator<SubscriptionItems
    >
    {
        public SubscriptionValidator()
        {

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        RuleFor(x => x.OwnerId)
            .NotEmpty().WithMessage("Поле 'OwnerId' не може бути порожнім")
            .Must(ownerID => SubData.subscriptionItems.Any(s => s.OwnerId == ownerID))
            .WithMessage("Не знайдено користувача за вказаним 'OwnerId'");

    
```

```

        RuleFor(x => x.Service)
            .NotEmpty().WithMessage("Поле 'Service' не може бути порожнім")
            .Length(3, 50).WithMessage("Поле 'Service' має бути в межах від 3 до 50 символів");
    
```

```

        RuleFor(x => x.Status)
            .NotEmpty().WithMessage("Поле 'Status' не можу бути порожнім")
            .IsInEnum().WithMessage("Поле 'Status' має бути вибране в заданих межах");
    
```

```

    }
}
}
```

PeopleController.cs:

```

using LW_4_2.Data;
using LW_4_2.Models;
using Microsoft.AspNetCore.Mvc;
using System.Text.Json;
namespace LW_4_2.Controllers
{
    [ApiController]

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

[Route("[controller]")]
public class PeopleController : ControllerBase
{
    [HttpGet]
    public ActionResult<IEnumerable<PeopleItems>> GetAll(
        [FromQuery] string? Name, [FromQuery] string? Email)
    {
        IEnumerable<PeopleItems> peoples = PeopleData.peopleItems;
        if (Name is not null)
            peoples = peoples.Where(x => x.Name == Name);
        if (Email is not null)
            peoples = peoples.Where(x => x.Email == Email);
        if (!peoples.Any())
            return NotFound("Не знайдено елементів");
        return Ok(peoples);
    }

    [HttpGet("{id}")]
    public ActionResult<PeopleItems> GetById(int id)
    {
        var people = PeopleData.peopleItems.FirstOrDefault(x => x.Id ==
id);
        if (people is null)
            return NotFound($"Не знайдено людину із вказаним 'Id': {id}");
        return Ok(people);
    }

    [HttpPost]
    public ActionResult<PeopleItems> Create(PeopleItems item)
    {
        item.Id = PeopleData.peopleItems.Max(x => x.Id) + 1;
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        PeopleData.peopleItems.Add(item);
        return CreatedAtAction(nameof(GetById), new { Id = item.Id }, item);
    }

    [HttpPut("{id}")]
    public IActionResult Update(int id, PeopleItems upPeople)
    {
        var people = PeopleData.peopleItems.FirstOrDefault(p => p.Id == id);
        if (people is null)
            return NotFound($"Не знайдено людини із Id {id}");
        people.Name = upPeople.Name;
        people.Email = upPeople.Email;
        return NoContent();
    }

    [HttpPatch("{id}")]
    public IActionResult UpdatePart(int id, [FromBody] JsonElement upPeople)
    {
        ModelState.ClearValidationState(nameof(upPeople));
        var people = PeopleData.peopleItems.FirstOrDefault(x => x.Id == id);
        if (people is null)
            return NotFound($"Не знайдено людини за вказаним Id {id}");
        var temp = new PeopleItems
        {
            Id = id,
            Email = people.Email,
            Name = people.Name
        }
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    }

        if(upPeople.TryGetProperty("name",out var name))
            temp.Name = name.GetString();

        if(upPeople.TryGetProperty("email",out var email))
            temp.Email = email.GetString();

        if (!TryValidateModel(temp))
            return BadRequest(ModelState);

        people.Name = temp.Name;
        people.Email = temp.Email;

        return NoContent();
    }

    [HttpDelete("{id}")]
    public IActionResult DeletePart(int id)
    {
        var people = PeopleData.peopleItems.FirstOrDefault(p => p.Id ==
id);

        if (people is null)
            return NotFound($"Не знайдено людину із вказаним Id {id}");

        ;

        PeopleData.peopleItems.Remove(people);

        return NoContent();
    }
}

```

SubController.cs:

```

using LW_4_2.Data;
using LW_4_2.Models;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.ApplicationParts;
using System.Text.Json;

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

namespace LW_4_2.Controllers
{
    [ApiController]
    [Route("[controller]")]
    public class SubController : ControllerBase
    {
        [HttpGet]
        public ActionResult<IEnumerable<SubscriptionItems>> GetAll(
            [FromQuery] int? ownerId, [FromQuery] string? service, [FromQuer
y] SubStatus? subStatus)
        {
            IEnumerable<SubscriptionItems> subItems = SubData.subscriptionIt
ems;
            if (ownerId is not null)
                subItems = subItems.Where(x => x.OwnerId == ownerId);
            if (service is not null)
                subItems = subItems.Where(x => x.Service == service);
            if (subStatus is not null)
                subItems = subItems.Where(x => x.Status == subStatus);
            if (subItems.Count() == 0)
                return NotFound("Не знайдено елементів за запитом");
            return Ok(subItems);
        }
        [HttpGet("{id}")]
        public ActionResult<SubscriptionItems> GetById(int id)
        {
            var sub = SubData.subscriptionItems.FirstOrDefault(x => x.Id == id)
            ;
            if (sub is null)
                return NotFound($"Не знайдено підписки з Id {id}");
        }
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        return Ok(sub);
    }

    [HttpPost]
    public ActionResult<SubscriptionItems> Create(SubscriptionItems item)
    {
        item.Id = SubData.subscriptionItems.Max(x => x.Id) + 1;
        SubData.subscriptionItems.Add(item);
        return CreatedAtAction(nameof(GetById), new { Id = item.Id }, item);
    }

    [HttpPut("{id}")]
    public IActionResult Update(int id, SubscriptionItems item)
    {
        var sub = SubData.subscriptionItems.FirstOrDefault(x => x.Id == id);
        if (sub is null)
            return NotFound($"Не знайдено підписки з Id {id}");
        sub.OwnerId = item.OwnerId;
        sub.Service = item.Service;
        sub.Status = item.Status;
        return NoContent();
    }

    [HttpPatch("{id}")]
    public IActionResult UpdatePart(int id, JsonElement item)
    {
        var sub = SubData.subscriptionItems.FirstOrDefault(x => x.Id == id);
        if (sub is null)
            return NotFound($"Не знайдено підписки з Id {id}");
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

var temp = new SubscriptionItems
{
    Id = sub.Id,
    OwnerId = sub.OwnerId,
    Service = sub.Service,
    Status = sub.Status
};

if(item.TryGetProperty("ownerId",out var upId))
    temp.OwnerId = upId.GetInt32();
if (item.TryGetProperty("status",out var upStatus))
    temp.Status = (SubStatus)upStatus.GetInt32();
if (item.TryGetProperty("service",out var upServ))
    temp.Service = upServ.GetString();

TryValidateModel(temp);
if (!ModelState.IsValid)
    return BadRequest(ModelState);

sub.OwnerId = temp.OwnerId;
sub.Service = temp.Service;
sub.Status = temp.Status;
return NoContent();
}

[HttpDelete("{id}")]
public IActionResult Delete(int id)
{
    var sub = SubData.subscriptions.FirstOrDefault(x => x.Id == id);

    if (sub is null)
        return NotFound($"Не знайдено підписки з Id {id}");

    SubData.subscriptions.Remove(sub);

    return NoContent();
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```
        }  
    }  
}
```

MessageController.cs:

```
using LW_4_2.Data;  
using LW_4_2.Models;  
using Microsoft.AspNetCore.Mvc;  
using Microsoft.AspNetCore.Mvc.ModelBinding;  
using System.Text.Json;  
namespace LW_4_2.Controllers  
{  
    [ApiController]  
    [Route("[controller]")]  
    public class MessageController: ControllerBase  
    {  
        [HttpGet]  
        public ActionResult<IEnumerable<MessageItems>> GetAll(  
            [FromQuery] int? ownerId,  
            [FromQuery] int? subId,  
            [FromQuery] string? title)  
        {  
            IEnumerable<MessageItems> messages = MessageData.messageIt  
ms;  
            if(ownerId is not null)  
                messages = messages.Where(x => x.OwnerId == ownerId);  
            if (subId is not null)  
                messages = messages.Where(x => x.SubId == subId);  
            if (title is not null)  
                messages = messages.Where(x => x.Title == title);  
            if (!messages.Any())
```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        return NotFound("Не знайдено елементів за вказаним запитом");
    }

    return Ok(messages);
}

[HttpGet("{id}")]
public ActionResult<MessageItems> GetById(int id)
{
    var message = MessageData.messageItems.FirstOrDefault(x => x.Id
== id);

    if (message is null)
        return NotFound($"Повідомлення із вказаним Id {id} не знайде
но");

    return Ok(message);
}

[HttpPost]
public ActionResult<MessageItems> Create(MessageItems message)
{
    message.Id = MessageData.messageItems.Max(x => x.Id) + 1;
    MessageData.messageItems.Add(message);
    return CreatedAtAction(nameof(GetById), new { Id = message.Id },
message);
}

[HttpPut("{id}")]
public IActionResult Update(int id, MessageItems message)
{
    var mes = MessageData.messageItems.FirstOrDefault(x => x.Id == i
d);

    if (mes is null)
        return NotFound($"Повідомлення із вказаним Id {id} не знайде
но");
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

mes.OwnerId = message.OwnerId;
mes.SubId = message.SubId;
mes.Title = message.Title;
return NoContent();
}

[HttpPatch("{id}")]
public IActionResult UpdatePart(int id, JsonElement message)
{
    var mes = MessageData.messageItems.FirstOrDefault(x => x.Id == i
d);
    if (mes is null)
        return NotFound($"Повідомлення із вказаним Id {id} не знайде
но");
    var temp = new MessageItems()
    {
        Id = id,
        OwnerId = mes.OwnerId,
        SubId = mes.SubId,
        Title = mes.Title,
    };
    if (message.TryGetProperty("ownerId", out var upO))
        temp.OwnerId = upO.GetInt32();
    if (message.TryGetProperty("subId", out var upS))
        temp.SubId = upS.GetInt32();
    if (message.TryGetProperty("title", out var upT))
        temp.Title = upT.GetString();
    TryValidateModel(temp);
    if (!ModelState.IsValid)
        return BadRequest(ModelState);
    mes.OwnerId = temp.OwnerId;
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        mes.SubId = temp.SubId;
        mes.Title = temp.Title;
        return NoContent();
    }

    [HttpDelete("{id}")]
    public IActionResult Delete(int id)
    {
        var mes = MessageData.messageItems.FirstOrDefault(x => x.Id == i
d);
        if (mes is null)
            return NotFound($"Повідомлення із вказаним Id {id} не знайде
но");
        MessageData.messageItems.Remove(mes);
        return NoContent();
    }
}

```

Program.cs:

```

using FluentValidation;
using FluentValidation.AspNetCore;
using LW_4_2.Models;
using LW_4_2.Validator;
var builder = WebApplication.CreateBuilder(args);
// Add services to the container.
builder.Services.AddControllers().AddFluentValidation();
// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetc
ore/swashbuckle
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen(options =>
{

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        var xmlFile = $"'{System.Reflection.Assembly.GetExecutingAssembly().
GetName().Name}'.xml";
        var xmlPath = Path.Combine(AppContext.BaseDirectory, xmlFile);
        options.IncludeXmlComments(xmlPath);
    });
builder.Services.AddValidatorsFromAssemblyContaining<SubscriptionValidator>();
builder.Services.AddValidatorsFromAssemblyContaining<MessageValidator>();
var app = builder.Build();
// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}
app.UseAuthorization();
app.MapControllers();
app.Run();

```

Протестовано роботу програми за допомогою swagger:

Запит 1

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ПІ231.03.04**

Арк.

47

Request URL  
**http://localhost:5140/People**

Server response

Code	Details
200	Response body <pre>[     {         "id": 1,         "name": "Kate",         "email": "Kate@email.com"     },     {         "id": 2,         "name": "John",         "email": "John@email.com"     },     {         "id": 3,         "name": "Jane",         "email": "Jane@email.com"     },     {         "id": 4,         "name": "Steve",         "email": "Steve@email.com"     },     {         "id": 5,         "name": "Alex",         "email": "Alex@email.com"     } ]</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 17:06:07 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 39 – Отримання всіх користувачів

### Запит 2

Request URL  
**http://localhost:5140/People?Name=Kate**

Server response

Code	Details
200	Response body <pre>[     {         "id": 1,         "name": "Kate",         "email": "Kate@email.com"     } ]</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 17:07:41 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 40 – Отримання користувачів за одним параметром

### Запит 3

Request URL  
**http://localhost:5140/People?Name=Kate&Email=Kate%40Email.com**

Server response

Code	Details
200	Response body <pre>[     {         "id": 1,         "name": "Kate",         "email": "Kate@email.com"     } ]</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 17:09:05 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 41 -Отримання користувачів за двома параметрами

### Запит 4

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						48

Request URL  
**http://localhost:5140/People?Name=Kate&Email=NoEmail%40Email.com**

Server response

Code	Details
404 Undocumented	Error: Not Found Response body Не знайдено елементів

Response headers

```
content-type: text/plain; charset=utf-8
date: Tue,14 Oct 2025 17:10:24 GMT
server: Kestrel
transfer-encoding: chunked
```

Рисунок 42 - Спроба отримати користувачів за хибним параметром запиту

### Запит 5

Request URL  
**http://localhost:5140/People/1**

Server response

Code	Details
200	Response body <pre>{   "id": 1,   "name": "Kate",   "email": "Kate@Email.com" }</pre>

Response headers

```
content-type: application/json; charset=utf-8
date: Tue,14 Oct 2025 17:11:44 GMT
server: Kestrel
transfer-encoding: chunked
```

Рисунок 43 – Отримання користувача за Id

### Запит 6

Request URL  
**http://localhost:5140/People/10**

Server response

Code	Details
404 Undocumented	Error: Not Found Response body Не знайдено людину із вказаним 'Id': 10

Response headers

```
content-type: text/plain; charset=utf-8
date: Tue,14 Oct 2025 17:12:33 GMT
server: Kestrel
transfer-encoding: chunked
```

Рисунок 44 – Спроба отримати неіснуючого користувача

### Запит 7

Request URL  
**http://localhost:5140/People**

Server response

Code	Details
201 Undocumented	Response body <pre>{   "id": 6,   "name": "Timcat1",   "email": "Time@Email.com" }</pre>

Response headers

```
content-type: application/json; charset=utf-8
date: Tue,14 Oct 2025 17:14:53 GMT
location: http://localhost:5140/People/6
server: Kestrel
transfer-encoding: chunked
```

Рисунок 45 – Додавання нового користувача

Змін.	Арк.	№ докум.	Підпис	Дата	Арк.
					49

**ЛР.ОК24.ПІ231.03.04**

## Запит 8

Request URL  
`http://localhost:5140/People`

Server response

Code Details

400 Undocumented Error: Bad Request

Response body

```
{ "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1", "title": "One or more validation errors occurred.", "status": 400, "errors": { "Name": [ "Name" не відповідає вимогам:Перша велика англійська літера, далі від 5 до 10 малих англійських символів та цифри за бажанням" ] }, "traceId": "00-8e01a42057ea453b4f73a55f752a3a90-366f7e4d94013a79-00" }
```

Response headers

```
content-type: application/problem+json; charset=utf-8
date: Tue, 14 Oct 2025 17:16:16 GMT
server: Kestrel
transfer-encoding: chunked
```

[Copy](#) [Download](#)

Рисунок 46 – Спроба додавання користувача із некоректним логіном

## Запит 9

Request URL  
`http://localhost:5140/People`

Server response

Code Details

400 Undocumented Error: Bad Request

Response body

```
{ "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1", "title": "One or more validation errors occurred.", "status": 400, "errors": { "Name": [ "Name" не відповідає вимогам:Перша велика англійська літера, далі від 5 до 10 малих англійських символів та цифри за бажанням" ], "Email": [ "Не коректне значення 'Email'" ] }, "traceId": "00-00deb9f2685c03214858a164829a32ba-18aa201d293fb963-00" }
```

Response headers

```
content-type: application/problem+json; charset=utf-8
date: Tue, 14 Oct 2025 17:17:34 GMT
server: Kestrel
transfer-encoding: chunked
```

[Copy](#) [Download](#)

Рисунок 47 – Спроба додавання користувача із некоректними полями

## Запит 10

Request URL  
`http://localhost:5140/People`

Server response

Code Details

400 Undocumented Error: Bad Request

Response body

```
{ "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1", "title": "One or more validation errors occurred.", "status": 400, "errors": { "Name": [ "Поле 'Name' Обов'язкове для заповнення" ], "Email": [ "Не коректне значення 'Email'" ] }, "traceId": "00-1f96ae5836edb34d53f8a99128e7e83-103fdc6fd76a7ca-00" }
```

Response headers

```
content-type: application/problem+json; charset=utf-8
date: Tue, 14 Oct 2025 17:19:11 GMT
server: Kestrel
transfer-encoding: chunked
```

[Copy](#) [Download](#)

Рисунок 48 – Спроба відправлення пустого поля

						ЛР.ОК24.ПІ231.03.04	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата			50

## Запит 11

Request URL	<code>http://localhost:5140/People/1</code>
Server response	
<b>Code</b>	Details
204	<i>Undocumented</i>
Response headers	
<code>date: Tue, 14 Oct 2025 17:21:02 GMT</code>	
<code>server: Kestrel</code>	

Рисунок 49 – Оновлення інформації про першого користувача

Запит 12

```
Request URL  
http://localhost:5140/People/1  
  
Server response  
  


| Code                       | Details            |
|----------------------------|--------------------|
| 400<br><i>Undocumented</i> | Error: Bad Request |

  
Response body  


```
{  
    "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1",  
    "title": "One or more validation errors occurred.",  
    "status": 400,  
    "errors": {  
        "Name": [  
            "Name' не відповідає вимогам:Перша велика англійська літера, далі від 5 до 10 малих англійських символів та цифри за бажанням"  
        ]  
    },  
    "traceId": "00-1f340787ec8b68abe2b685972300a1d9-bc60e0581ec02b45-00"  
}
```

  
Response headers  


```
content-type: application/problem+json; charset=utf-8  
date: Tue, 14 Oct 2025 17:22:00 GMT  
server: Kestrel  
transfer-encoding: chunked
```

  
Responses
```

Рисунок 50 – Спроа оновлення з використання некоректного логіну

Запит 13

Request URL  
<http://localhost:5140/People/1>

Server response

Code	Details
400 <i>Undocumented</i>	Error: Bad Request

Response body

```
{ "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1", "title": "One or more validation errors occurred.", "status": 400, "errors": { "Name": [ "None 'Name' Обов'язкове для заповнення" ], "Email": [ "None 'Email' є обов'язковим" ] }, "traceId": "00-fa94178695ec59d6a06169da7ab8bf68-27437703643d9d4f-00" }
```

 [Download](#)

Response headers

```
content-type: application/problem+json; charset=utf-8
date: Tue, 14 Oct 2025 17:23:08 GMT
server: Kestrel
transfer-encoding: chunked
```

Рисунок 51 – Спроба оновленн на пусті поля

Запит 14

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.	51

Request URL  
**http://localhost:5140/People/1**

Server response

Code	Details
204 Undocumented	Response headers date: Tue, 14 Oct 2025 18:45:48 GMT server: Kestrel

Рисунок 52 – Часткове оновлення користувача

### Запит 15

Request URL  
**http://localhost:5140/People/1**

Server response

Code	Details
400 Undocumented	Error: Bad Request Response body <pre>{   "Name": [     "Поле 'Name' обов'язкове для заповнення"   ] }</pre>
	Response headers content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 18:47:42 GMT server: Kestrel transfer-encoding: chunked

Рисунок 53 – Спроба оновлення на пусте поле

### Запит 16

Request URL  
**http://localhost:5140/People/1**

Server response

Code	Details
204 Undocumented	Response headers date: Tue, 14 Oct 2025 18:48:36 GMT server: Kestrel

Рисунок 54 – Видалення користувача

### Запит 17

Request URL  
**http://localhost:5140/People/1**

Server response

Code	Details
404 Undocumented	Error: Not Found Response body Не знайдено людину із вказанним Id 1
	Response headers content-type: text/plain; charset=utf-8 date: Tue, 14 Oct 2025 18:50:41 GMT server: Kestrel transfer-encoding: chunked

Рисунок 55 – Спроба видалення користувача

### Запит 18

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						52

Request URL  
**http://localhost:5140/Sub**

Server response

Code	Details
200	<p>Response body</p> <pre>{   "ownerId": 2,   "service": "Steam",   "status": 1 }, {   "id": 5,   "ownerId": 2,   "service": "Google",   "status": 2 }, {   "id": 6,   "ownerId": 2,   "service": "Amazon",   "status": 1 }, {   "id": 7,   "ownerId": 3,   "service": "YouTube",   "status": 3 }, {   "id": 8,   "ownerId": 3,   "service": "Netflix",   "status": 1 }</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 19:08:06 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 56 – Отримано список всіх підписок

### Запит 19

Request URL  
**http://localhost:5140/Sub?ownerId=1**

Server response

Code	Details
200	<p>Response body</p> <pre>[   {     "id": 1,     "ownerId": 1,     "service": "Netflix",     "status": 1   },   {     "id": 2,     "ownerId": 1,     "service": "Xbox",     "status": 1   },   {     "id": 3,     "ownerId": 1,     "service": "Amazon",     "status": 1   } ]</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 19:09:10 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 57 – Отримання підписок з викорисанням параметра

### Запит 20

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						53

Request URL  
**http://localhost:5140/Sub?ownerId=1&subStatus=1**

Server response

Code	Details
200	Response body <pre>[ { "id": 1, "ownerId": 1, "service": "Netflix", "status": 1 }, { "id": 2, "ownerId": 1, "service": "Xbox", "status": 1 }, { "id": 3, "ownerId": 1, "service": "Amazon", "status": 1 } ]</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 19:10:21 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 58 – Отримання підписок за деялькома параметрами

### Запит 21

Request URL  
**http://localhost:5140/Sub?ownerId=1&service=Random&subStatus=1**

Server response

Code	Details
404 Undocumented	Error: Not Found Response body <pre>Не знайдено елементів за запитом</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: text/plain; charset=utf-8 date: Tue, 14 Oct 2025 19:11:52 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 59 – Спроба отримати підписки за відсутнім параметром

### Запит 22

Request URL  
**http://localhost:5140/Sub/1**

Server response

Code	Details
200	Response body <pre>{ "id": 1, "ownerId": 1, "service": "Netflix", "status": 1 }</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 19:12:55 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 60 – Отримання підписки за Id

### Запит 23

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						54

Request URL  
`http://localhost:5140/Sub/111`

Server response

Code	Details
404 <i>Undocumented</i>	Error: Not Found Не знайдено підписки з Id 111

Response body

Response headers

```
content-type: text/plain; charset=utf-8
date: Tue, 14 Oct 2025 19:13:50 GMT
server: Kestrel
transfer-encoding: chunked
```

 [Download](#)

Рисунок 61 – Спроба отримання неіснуючої підписки за Id

Запит 24

Request URL  
<http://localhost:5140/Sub>

Server response

Code	Details
201 Undocumented	<p>Response body</p> <pre>{     "id": 15,     "ownerId": 1,     "service": "AmazonMinPlus",     "status": 1 }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 19:16:28 GMT location: http://localhost:5140/Sub/15 server: Kestrel transfer-encoding: chunked</pre>

Responses

Code Details

201 Response body

```
{  
    "id": 15,  
    "ownerId": 1,  
    "service": "AmazonMinPlus",  
    "status": 1  
}
```

Download

### Рисунок 62 – Додавання нової підписки

### Запит 25

```
Request URL  
http://localhost:5140/Sub  
  
Server response  


| Code                       | Details            |
|----------------------------|--------------------|
| 400<br><i>Undocumented</i> | Error: Bad Request |

  
Response body  


```
{  
    "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1",  
    "title": "One or more validation errors occurred.",  
    "status": 400,  
    "errors": {  
        "OwnerId": [  
            "Не знайдено користувача за вказаним 'OwnerId'"  
        ]  
    },  
    "traceId": "00-c9d583bf8c7fe07e3586e05929e7c59a-12c618641147e82d-00"  
}
```

  
Response headers  


```
content-type: application/problem+json; charset=utf-8  
date: Tue,14 Oct 2025 19:17:32 GMT  
server: Kestrel  
transfer-encoding: chunked
```


```

Рисунок 64 – Способ додавання підписки із неіснуючим користувачем

Запит 26

						Арк.
Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	55

Request URL  
**http://localhost:5140/Sub**

Server response

Code	Details
400 Undocumented	Error: Bad Request

Response body

```
{
  "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1",
  "title": "One or more validation errors occurred.",
  "status": 400,
  "errors": {
    "Status": [
      "Поле 'Status' має бути виbrane в заданих межах"
    ],
    "OwnerId": [
      "Поле 'OwnerId' не може бути порожнім",
      "Не знайдено користувача за вказаним 'OwnerId'"
    ]
  ],
  "traceId": "00-9dc2d41315b359fb0912bde01dc312e0-8c1b8509ccb49f7e-00"
}
```

Response headers

```
content-type: application/problem+json; charset=utf-8
date: Tue, 14 Oct 2025 19:18:47 GMT
server: Kestrel
transfer-encoding: chunked
```

[Copy](#) [Download](#)

Рисунок 65 – Ще одна спроба додати невірну інформацію про підписку  
 Запит 27

Request URL  
**http://localhost:5140/Sub/1**

Server response

Code	Details
204 Undocumented	Response headers

```
date: Tue, 14 Oct 2025 19:20:35 GMT
server: Kestrel
```

Responses

Рисунок 66 – Оновлення інформації про підписку  
 Запит 28

Request URL  
**http://localhost:5140/Sub/1**

Server response

Code	Details
400 Undocumented	Error: Bad Request

Response body

```
{
  "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1",
  "title": "One or more validation errors occurred.",
  "status": 400,
  "errors": {
    "OwnerId": [
      "Не знайдено користувача за вказаним 'OwnerId'"
    ]
  },
  "traceId": "00-8e0035c7b64bb32589ebc088a6cc5ed9-6a4eb9cd6e958366-00"
}
```

Response headers

```
content-type: application/problem+json; charset=utf-8
date: Tue, 14 Oct 2025 19:21:47 GMT
server: Kestrel
transfer-encoding: chunked
```

[Copy](#) [Download](#)

Рисунок 67 – Спроба некоректного оновлення підписки  
 Запит 29

Request URL  
**http://localhost:5140/Sub/1**

Server response

Code	Details
204 Undocumented	Response headers

```
date: Tue, 14 Oct 2025 19:23:41 GMT
server: Kestrel
```

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ПІ231.03.04**

Арк.

56

Рисунок 68 – Часткове оновлення власника підписки

### Запит 30

Request URL  
`http://localhost:5140/Sub/1`

Server response

Code	Details
400 Undocumented	Error: Bad Request Response body <pre>{     "OwnerId": [         "Не знайдено користувача за вказаним 'OwnerId'"     ] }</pre>

Response headers  
`content-type: application/json; charset=utf-8  
date: Tue,14 Oct 2025 19:24:41 GMT  
server: Kestrel  
transfer-encoding: chunked`

[Copy](#) [Download](#)

Рисунок 69 – Спроба часткового оновлення підписки невірним користувачем

### Запит 31

Request URL  
`http://localhost:5140/Sub/11`

Server response

Code	Details
204 Undocumented	Response headers <code>date: Tue,14 Oct 2025 19:25:39 GMT server: Kestrel</code>

Рисунок 70 – Видалення підписки за Id

### Запит 32

Request URL  
`http://localhost:5140/Sub/111`

Server response

Code	Details
404 Undocumented	Error: Not Found Response body <code>Не знайдено підписки з Id 111</code>

Response headers  
`content-type: text/plain; charset=utf-8  
date: Tue,14 Oct 2025 19:26:33 GMT  
server: Kestrel  
transfer-encoding: chunked`

[Copy](#) [Download](#)

Рисунок 71 – Спроба видалення неіснуючої підписки

### Запит 33

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						57

http://localhost:5140/Message

Server response

Code	Details
200	<p>Response body</p> <pre>[{"ownerId": 4, "subId": 9}, {"id": 7, "title": "Підпису продовжено", "ownerId": 4, "subId": 9}, {"id": 8, "title": "Підписка прострочена", "ownerId": 4, "subId": 11}, {"id": 9, "title": "Підписку скасовано", "ownerId": 4, "subId": 11}, {"id": 10, "title": "До оплати підписки тиждень", "ownerId": 5, "subId": 15}]</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 19:27:50 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 72 – Отримання списку всіх повідомлень

### Запит 34

Request URL

http://localhost:5140/Message?ownerId=1&subId=1

Server response

Code	Details
200	<p>Response body</p> <pre>[{"id": 1, "title": "До оплати три дні", "ownerId": 1, "subId": 1}, {"id": 2, "title": "До оплати два дні", "ownerId": 1, "subId": 1}, {"id": 3, "title": "До оплати один день", "ownerId": 1, "subId": 1}]</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Tue, 14 Oct 2025 19:29:54 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 73 – Отримання повідомлень за вказаними параметрами

### Запит 35

Request URL

http://localhost:5140/Message?ownerId=1&subId=1&title=1

Server response

Code	Details
404 Undocumented	<p>Error: Not Found</p> <p>Не знайдено елементів за вказаним запитом</p> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: text/plain; charset=utf-8 date: Tue, 14 Oct 2025 19:30:43 GMT server: Kestrel transfer-encoding: chunked</pre>

Змін.	Арк.	№ докум.	Підпис	Дата

ЛР.ОК24.ПІ231.03.04

Арк.

58

Рисунок 74 – Спроба отримати неіснуюче повідомлення

### Запит 36

Request URL  
`http://localhost:5140/Message/1`

Server response

Code Details

200 Response body

```
{ "id": 1, "title": "Во оплати три дні", "ownerId": 1, "subId": 1 }
```

Response headers

```
content-type: application/json; charset=utf-8
date: Tue,14 Oct 2025 19:32:09 GMT
server: Kestrel
transfer-encoding: chunked
```

[Copy](#) [Download](#)

Рисунок 75 – Отримання повідомлення за Id

### Запит 37

Request URL  
`http://localhost:5140/Message/111`

Server response

Code Details

404 Undocumented Error: Not Found

Response body

```
Повідомлення із вказаним Id 111 не знайдено
```

Response headers

```
content-type: text/plain; charset=utf-8
date: Tue,14 Oct 2025 19:33:00 GMT
server: Kestrel
transfer-encoding: chunked
```

[Copy](#) [Download](#)

Рисунок 76 – Спроба отримання неіснуючого повідомлення за Id

### Запит 38

Request URL  
`http://localhost:5140/Message`

Server response

Code Details

201 Undocumented Response body

```
{ "id": 10, "title": "Підписка просрочена", "ownerId": 1, "subId": 3 }
```

Response headers

```
content-type: application/json; charset=utf-8
date: Tue,14 Oct 2025 19:34:40 GMT
location: http://localhost:5140/Message/10
server: Kestrel
transfer-encoding: chunked
```

[Copy](#) [Download](#)

Рисунок 77 – Додавання нової підписки

### Запит 39

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ПІ231.03.04**

Арк.

59

Request URL  
**http://localhost:5140/Message**

Server response

Code	Details
400 Undocumented	Error: Bad Request  Response body  <pre>{   "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1",   "title": "One or more validation errors occurred.",   "status": 400,   "errors": [     "Title": [       "Поле 'Title' не може бути порожнім",       "Поле 'Title' має бути від трьох до ста символів"     ],     "OwnerId": [       "Відсутня підписка за вказаним 'SubId' або для неї не існує користувача"     ]   ],   "traceId": "00-62fe2d8dd958ab132e24aa9f06b4e7f0-17af6db3c1e9b08b-00" }</pre> <div style="text-align: right; margin-top: -20px;"> <a href="#">Copy</a> <a href="#">Download</a> </div>
	Response headers  <pre>content-type: application/problem+json; charset=utf-8 date: Tue, 14 Oct 2025 19:35:51 GMT server: Kestrel transfer-encoding: chunked</pre>

**Рисунок 78 – Спроба додавання повідомлення про підписку не тому користувачеві та з порожніс текстом**

### Запит 40

Request URL  
**http://localhost:5140/Message/1**

Server response

Code	Details
204 Undocumented	Response headers  <pre>date: Tue, 14 Oct 2025 19:37:42 GMT server: Kestrel</pre>

**Рисунок 79 – Коректне оновлення повідомлення**

### Запит 41

400  
Undocumented Error: Bad Request

Response body

<pre>{   "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1",   "title": "One or more validation errors occurred.",   "status": 400,   "errors": [     "Title": [       "Поле 'Title' не може бути порожнім"     ],     "OwnerId": [       "Відсутня підписка за вказаним 'SubId' або для неї не існує користувача"     ]   ],   "traceId": "00-a186d7ff4bbcd3c3d32e132d7ddf193-ac9107b4cd152008-00" }</pre>	<a href="#">Copy</a> <a href="#">Download</a>
Response headers  <pre>content-type: application/problem+json; charset=utf-8 date: Tue, 14 Oct 2025 19:38:38 GMT server: Kestrel transfer-encoding: chunked</pre>	

**Рисунок 80 – Спроба оновити повідомлення, присвоївши підписку не ому користувачеві та з порожнім підписом**

### Запит 42

Request URL  
**http://localhost:5140/Message/1**

Server response

Code	Details
204 Undocumented	Response headers  <pre>date: Tue, 14 Oct 2025 19:44:11 GMT server: Kestrel</pre>

Змін.	Арк.	№ докум.	Підпис	Дата	Арк.
					60

**ЛР.ОК24.ПІ231.03.04**

### Рисунок 81 – Коректне часткове оновлення

Запит 43

```
Request URL  
http://localhost:5140/Message/1  
  
Server response  


| Code                       | Details            |
|----------------------------|--------------------|
| 400<br><i>Undocumented</i> | Error: Bad Request |

  
Response body  


```
{  
    "title": [  
        "Поле 'Title' не може бути порожнім",  
        "Поле 'Title' має бути від трьох до ста символів"  
    ]  
}
```

  
Response headers  


```
content-type: application/json; charset=utf-8  
date: Tue, 14 Oct 2025 19:45:28 GMT  
server: Kestrel  
transfer-encoding: chunked
```



Download


```

Рисунок 82 – Спроба часткового оновлення повідомлення пустим текстом

Запит 44

Request URL	
<code>http://localhost:5140/Message/1</code>	
Server response	
Code	Details
204 <small>Undocumented</small>	<p>Response headers</p> <pre>date: Tue, 14 Oct 2025 19:47:06 GMT server: Kestrel</pre>

### Рисунок 83 – Видалення повідомлення

### Запит 45

The screenshot shows a web browser interface. At the top, it says "Request URL" followed by "http://localhost:5140/Message/1". Below that, it says "Server response". A table follows with two columns: "Code" and "Details". The "Code" column contains "404" and "Undocumented". The "Details" column contains "Error: Not Found". Under the "Details" column, there is a section titled "Response body" containing the text "Повідомлення із вказанним Id 1 не знайдено". At the bottom right of the main content area, there are "Copy" and "Download" buttons. Below the main content area, there is a section titled "Response headers" with the following content:

Content-Type	text/plain; charset=utf-8
Date	Tue, 14 Oct 2025 19:47:30 GMT
Server	Kestrel
Transfer-Encoding	chunked

Рисунок 84 – Спроба видалення повідolenня ще раз

## Завдання 3

Створено діаграму компонентів для програми по відображеню підписок користувача. Діаграма складається Controller Layer – відповідає за прийняття та відправлення http – запитів, валідацію вхідної інформації, Service Layer – відповідає за бізнес логіку, Repository Layer – відповідає за логіку

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк. 61

звернення до клієнта БД, DB Layer – відповідає за створення клієнта БД із потрібним вмістом.

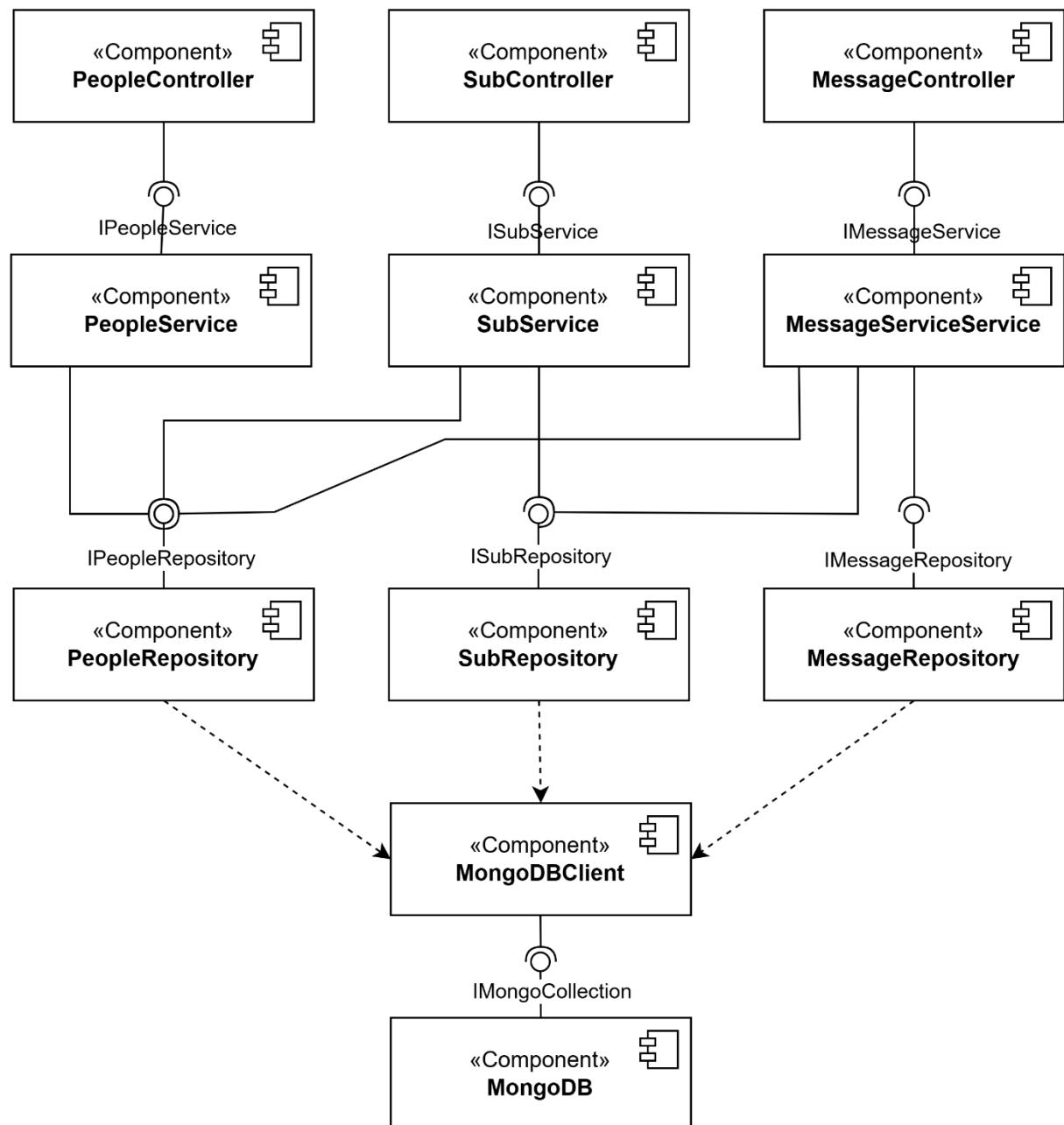


Рисунок 85 – Component Diagram

Controller Layer – надає клієнтові методи для звернення до ресурсів за вказаними ендпоїнтами. Він складається із трьох компонентів(контролер для користувачів, підписок, повідомлень), кожний із яких надає можливість виконання CRUD – операцій, приклади методів Create – створення сущності, Get – отримання, Update – Оновлення, Delete – Видалення.

Змін.	Арк.	№ докум.	Підпис	Дата

Service Layer – виконує бізнес логіку для запитів/об'єктів від контролерів. Також складається із трьох компонентів (сервіс для користувачів, підписок, повідомлень), реалізує відповідний для нього інтерфейс із набором методів, які має виконати сервіс. Наприклад має реалізувати такі методи: CreateAsync – створення, GetAsync – отримання, UpdateAsync – Оновлення, DeleteAsync – Видалення.

Repository Layer – виконує логіку звернення до клієнта БД для отримання елементів БД чи оновлення. Також складається із трьох компонентів (репозиторій для користувачів, підписок, повідомлень), реалізує відповідний для нього інтерфейс із набором методів, які має виконати репозиторій. Наприклад має реалізувати такі методи: CreateAsync – створення, GetAsync – отримання, UpdateAsync – Оновлення, DeleteAsync – Видалення.

DB Layer – реалізує логіку клієнта для отримання колекції/таблиці елементів із БД. Наприклад реалізує метод GetCollection, за яким можна отримати колекцію потрібних елементів.

## Завдання 4

Вдосконалено проект із другого завдання, використовуючи компонентну діаграму із третього завдання. В проектові створено підключення до MongoDB, оновлено моделі, створено шар репозиторіїв, які реалізують відповідні до них інтерфейси, та використовують MongoDBClient, створено шар сервісів, які реалізують відповідні їм інтерфейси, та які використовують репозиторії, також оновлено шар контролерів, які використовують сервіси

Виконано CRUD операції із використанням MongoDB:

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
63						

**Elements.Peoples**

STORAGE SIZE: 86KB LOGICAL DATA SIZE: 421B TOTAL DOCUMENTS: 6 INDEXES TOTAL SIZE: 36KB

**Find** Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass **INSERT DOCUMENT**

**Filter** Type a query: { field: 'value' } **Reset** **Apply** **Options ▾**

QUERY RESULTS: 1-6 OF 6

```
_id: ObjectId('68f5e00af21b02f6aece46ec')
name : "Jakekold"
email : "Jake@email.com"

_id: ObjectId('68f5e023f21b02f6aece46ed')
name : "Solosolo"
email : "solosolosolo52@gmail.com"
```

System Status: All Good  
©2025 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

Рисунок 86 – Початковий вміст БД

http://localhost:5129/People

Server response

Code	Details
200	Response body

```
[
  {
    "id": "68f5e00af21b02f6aece46ec",
    "name": "Jakekold",
    "email": "Jake@email.com"
  },
  {
    "id": "68f5e023f21b02f6aece46ed",
    "name": "Solosolo",
    "email": "solosolosolo52@gmail.com"
  },
  {
    "id": "68f5e054f21b02f6aece46ee",
    "name": "Susanteam",
    "email": "Susan@email.com"
  },
  {
    "id": "68f5e06bf21b02f6aece46ef",
    "name": "Mikefreez",
    "email": "Mike@email.com"
  },
  {
    "id": "68ff7822a866d7d74955a11c",
    "name": "Chorpovniik",
    "email": "Chort@email.com"
  },
  {
    "id": "690113f4b26a5f31097a0fad",
    "name": null,
    "email": null
  }
]
```

Download

Response headers	
content-type: application/json; charset=utf-8	date: Thu, 30 Oct 2025 14:53:06 GMT
server: Kestrel	transfer-encoding: chunked

Рисунок 87 – Отримання всіх користувачів

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						64

Curl

```
curl -X 'GET' \
'http://localhost:5129/People?name=Susanteam' \
-H 'accept: text/plain'
```

Request URL

<http://localhost:5129/People?name=Susanteam>

Server response

Code	Details	Links
200	<p>Response body</p> <pre>[   {     "id": "68f5e054f21b02f6aece46ee",     "name": "Susanteam",     "email": "Susan@email.com"   } ]</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 14:54:57 GMT server: Kestrel transfer-encoding: chunked</pre>	<a href="#">Copy</a> <a href="#">Download</a>
Responses		Links
Code	Description	
200	OK	No links

Рисунок 88 – Отримано користувача з використанням фільтру

Request URL

<http://localhost:5129/People/68f5e054f21b02f6aece46ee>

Server response

Code	Details	Links
200	<p>Response body</p> <pre>{   "id": "68f5e054f21b02f6aece46ee",   "name": "Susanteam",   "email": "Susan@email.com" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 14:58:08 GMT server: Kestrel transfer-encoding: chunked</pre>	<a href="#">Copy</a> <a href="#">Download</a>

Рисунок 89 – Отримано користувача за Id

Request URL

<http://localhost:5129/People>

Server response

Code	Details	Links
201 Undocumented	<p>Response body</p> <pre>{   "id": "69037d4bfc61cdb6bbb93f34",   "name": "Newuser",   "email": "NewUser@email.com" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 14:59:24 GMT location: http://localhost:5129/People/69037d4bfc61cdb6bbb93f34 server: Kestrel transfer-encoding: chunked</pre>	<a href="#">Copy</a> <a href="#">Download</a>

Рисунок 90 – Створено нового користувача

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						65

The screenshot shows the MongoDB Compass interface. On the left, the sidebar has 'Data Explorer' selected. In the main area, the 'Elements' section is expanded, showing 'Peoples'. Below it, two documents are listed:

```

_id: ObjectId('6901134b26ab3109/a0fad')
name : "Bashuk"
email : "bashuk777@gmail.com"

_id: ObjectId('69037d4bfc61cdb6bbb93f34')
name : "Newuser"
email : "NewUser@Email.com"

```

At the bottom, the status bar shows 'System Status: All Good'.

Рисунок 91 – Вміст бази даних після додавання нового користувача

The screenshot shows the Network tab of a browser developer tools window. A successful POST request is listed with the following details:

- Request URL:** `http://localhost:5129/People/69037d4bfc61cdb6bbb93f34`
- Server response:**
  - Code:** 204
  - Response headers:**
    - `date: Thu, 30 Oct 2025 15:02:01 GMT`
    - `server: Kestrel`

Рисунок 92 – Повне оновлення користувача

The screenshot shows the MongoDB Compass interface, identical to Figure 91 but with a different document ID. The document now has a different name and email address:

```

_id: ObjectId('69037d4bfc61cdb6bbb93f34')
name : "Updateuser"
email : "Some@email.com"

```

Рисунок 93 – Вміст після повного оновлення

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						66

Request URL  
<http://localhost:5129/People/69037d4bfcc61cdb6bbb93f34>

Server response

Code	Details
204 Undocumented	Response headers date: Thu, 30 Oct 2025 15:03:55 GMT server: Kestrel

Рисунок 94 – Часткове оновлення користувача

The screenshot shows the MongoDB Compass interface. On the left, the sidebar has 'Data Explorer' selected. In the main area, under 'Elements', 'Peoples', there is a list of documents. One document is expanded, showing its fields: '\_id: ObjectId('69037d4bfcc61cdb6bbb93f34'), name: 'Patchuser'', and 'email: 'Some@email.com''. A query builder at the top right shows the filter: 'Type a query: { field: 'value' }'. Below the document list, status information includes 'System Status: All Good' and copyright details: '©2025 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales'.

Рисунок 95 – Вміст бази даних після часткового оновлення

Curl  

```
curl -X 'DELETE' \
  'http://localhost:5129/People/69037d4bfcc61cdb6bbb93f34' \
  -H 'accept: */*'
```

Request URL  
<http://localhost:5129/People/69037d4bfcc61cdb6bbb93f34>

Server response

Code	Details
204 Undocumented	Response headers date: Thu, 30 Oct 2025 15:07:01 GMT server: Kestrel

Рисунок 96 – Видалення користувача

Змін.	Арк.	№ докум.	Підпис	Дата	Арк.
					67

ЛР.ОК24.ПІ231.03.04

**Elements.Peoples**

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 421B TOTAL DOCUMENTS: 6 INDEXES TOTAL SIZE: 36KB

**Find** Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass **INSERT DOCUMENT**

Filter Type a query: { field: 'value' } **Reset** **Apply** Options ▾

`_id: ObjectId('68ff7822a866d7d74955a1c')`  
`name : "Chorpovni"`  
`email : "Chort@Email.com"`

`_id: ObjectId('690113f4b26a5f31097a0fad')`  
`name : "Bashuk"`  
`email : "bashuk777@gmail.com"`

System Status: All Good  
 ©2025 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

Рисунок 97 – Вміст бази даних після видалення

**Elements.Subscriptions**

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 1.46KB TOTAL DOCUMENTS: 16 INDEXES TOTAL SIZE: 36KB

**Find** Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass **INSERT DOCUMENT**

Filter Type a query: { field: 'value' } **Reset** **Apply** Options ▾

`service : "Unortiv"`  
`status : 1`

`_id: ObjectId('69011d9e8a51e5b05552f128')`  
`ownerId : "68f5e00af21b02f6aece46ec"`  
`service : "ChortoveVT"`  
`status : 1`

System Status: All Good  
 ©2025 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

Рисунок 98 – Вміст колекції підписок

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						68

http://localhost:5129/Sub

Server response

Code	Details
200	<p>Response body</p> <pre>[   {     "id": "68f5e0eaf21b02f6aece46f1",     "ownerId": "68f5dff5f21b02f6aece46eb",     "service": "Megogo",     "status": 3   },   {     "id": "68f5e1b1006ab12b84b5db86",     "ownerId": "68f5dff5f21b02f6aece46eb",     "service": "XBox",     "status": 1   },   {     "id": "68f5e1c5006ab12b84b5db87",     "ownerId": "68f5e00af21b02f6aece46ec",     "service": "Steam",     "status": 1   },   {     "id": "68f5e1d9006ab12b84b5db88",     "ownerId": "68f5e00af21b02f6aece46ec",     "service": "YouTube Music",     "status": 1   },   {     "id": "68f5e1e5006ab12b84b5db89",     "ownerId": "68f5e00af21b02f6aece46ec",     "service": null,     "status": 1   } ]</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 15:09:10 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 99 – Отримано всі підписки користувачів

Request URL

http://localhost:5129/Sub?service=XBox&status=1

Server response

Code	Details
200	<p>Response body</p> <pre>[   {     "id": "68f5e1b1006ab12b84b5db86",     "ownerId": "68f5dff5f21b02f6aece46eb",     "service": "XBox",     "status": 1   },   {     "id": "68f5e245006ab12b84b5db8c",     "ownerId": "68f5e054f21b02f6aece46ee",     "service": "XBox",     "status": 1   } ]</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 15:10:06 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 100 – Отримано підписки за вказаними фільтрами

Request URL

http://localhost:5129/Sub/68f5e1b1006ab12b84b5db86

Server response

Code	Details
200	<p>Response body</p> <pre>{   "id": "68f5e1b1006ab12b84b5db86",   "ownerId": "68f5dff5f21b02f6aece46eb",   "service": "XBox",   "status": 1 }</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 15:10:48 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 101 – Отримано підписку за конкретним Id

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						69

Request URL  
**http://localhost:5129/Sub**

Server response

Code	Details
201 Undocumented	<b>Response body</b> <pre>{   "id": "6903802bfc61cdb6bbb93f35",   "ownerId": "68f5e054f21b02f6aece46ee",   "service": "Amazon",   "status": 1 }</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div>
<b>Response headers</b>	
<pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 15:11:39 GMT location: http://localhost:5129/Sub/6903802bfc61cdb6bbb93f35 server: Kestrel transfer-encoding: chunked</pre>	

Рисунок 102 – Створено нову підписку для користувача

The screenshot shows the MongoDB Compass interface. On the left, the sidebar has 'Cluster' selected under 'Data Explorer'. Under 'Elements', 'Subscriptions' is also selected. In the main pane, the title is 'Elements.Subscriptions'. A query is being run: 'service : "ChortoveVI" status : 1'. The results show one document:

```
_id: ObjectId('6903802bfc61cdb6bbb93f35')
ownerId: "68f5e054f21b02f6aece46ee"
service: "Amazon"
status: 1
```

Рисунок 103 – Вміст бази даних після створення підписки

Request URL  
**http://localhost:5129/Sub/6903802bfc61cdb6bbb93f35**

Server response

Code	Details
204 Undocumented	<b>Response headers</b> <pre>date: Thu, 30 Oct 2025 15:19:30 GMT server: Kestrel</pre>

Рисунок 104 – Повне оновлення підписки

Змін.	Арк.	№ докум.	Підпис	Дата	Арк.
					70

**ЛР.ОК24.ПІ231.03.04**

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with a tree view of the cluster structure. The 'Data Explorer' section is selected. In the main area, the 'Elements' tab is active, showing the 'Subscriptions' collection. A search bar at the top right says 'Generate queries from natural language in Compass'. Below it, a 'Find' button is highlighted. A query builder allows for filtering by fields like 'service' and 'status'. Two document snippets are shown:

```

service : "ChortoveVI"
status : 1

_id: ObjectId('6903802bfc61cdb6bbb93f35')
ownerId: "68f5e054f21b02f6aece46ee"
service: "UpdatedService"
status : 1

```

At the bottom, system status is shown as 'All Good'.

Рисунок 105 – Вміст колекції після повного оновлення

This screenshot shows a browser developer tools Network tab. It displays a request to the URL `http://localhost:5129/Sub/6903802bfc61cdb6bbb93f35`. The response code is 204 (Undocumented). The response headers show the date as `Thu, 30 Oct 2025 15:21:33 GMT` and the server as `Kestrel`.

Рисунок 106 – Часткове оновлення підписки

This screenshot is identical to Figure 105, showing the MongoDB Compass interface with the 'Subscriptions' collection. However, the document content in the main panel has been partially updated. The first document now includes the field 'service' with the value 'Patch'.

```

service : "ChortoveVI"
status : 1

_id: ObjectId('6903802bfc61cdb6bbb93f35')
ownerId: "68f5e054f21b02f6aece46ee"
service: "Patch"
status : 1

```

Рисунок 107 – Вміст колекції після часткового оновлення

This screenshot shows a browser developer tools Network tab. It displays a request to the URL `http://localhost:5129/Sub/6903802bfc61cdb6bbb93f35`. The response code is 204 (Undocumented). The response headers show the date as `Thu, 30 Oct 2025 15:23:58 GMT` and the server as `Kestrel`.

Змін.	Арк.	№ докум.	Підпис	Дата

ЛР.ОК24.ПІ231.03.04

Арк.

71

Рисунок 108 – Видалення підписки

The screenshot shows the MongoDB Compass interface with the 'Elements/Subscriptions/find' query page. The left sidebar has 'Data Explorer' selected under 'Cluster'. In the main area, the 'Find' tab is active, and a query is being run:

```
service : "Chortiv"
status : 1
```

The results pane shows one document:

```
_id: ObjectId('69011d9e8a51e5b05552f128')
ownerId: "68f5e08af21b02f6aece46ec"
service : "ChortoveVT"
status : 1
```

Рисунок 109 – Вміст колекції після видалення підписки

The screenshot shows the MongoDB Compass interface with the 'Elements/Messages/find' query page. The left sidebar has 'Data Explorer' selected under 'Cluster'. In the main area, the 'Find' tab is active, and a query is being run:

```
ownerId : "68f5e06dt21b02f6aece46ef"
subId : "68f5e2ca006ab12b84b5db99"
```

The results pane shows one document:

```
_id: ObjectId('68f5e5aa006ab12b84b5db9d')
title: "Підписка просрочена"
ownerId: "68f5e06b21b02f6aece46ef"
subId: "68f5e2ca006ab12b84b5db99"
```

Рисунок 110 – Вміст колекції повідомлень

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						72

Request URL  
<http://localhost:5129/Message>

Server response

Code	Details
200	Response body <pre>{   "ownerId": "68f5e054f21b02f6aece46ee",   "subId": "68f5e253096ab12b84b5db8d" }, {   "id": "68f5e504006ab12b84b5db99",   "title": "Лінійка вступила в дію",   "ownerId": "68f5e054f21b02f6aece46ee",   "subId": "68f5e253096ab12b84b5db8d" }, {   "id": "68f5e591006ab12b84b5db9b",   "title": "Лінійка вступила в дію",   "ownerId": "68f5e06bf21b02f6aece46ef",   "subId": "68f5e2ca006ab12b84b5db9b" }, {   "id": "68f5e5a0006ab12b84b5db9c",   "title": "До оплати підписки день",   "ownerId": "68f5e06bf21b02f6aece46ef",   "subId": "68f5e2ca006ab12b84b5db9c" }, {   "id": "68f5e5aa006ab12b84b5db9d",   "title": "Лінійка просрочена",   "ownerId": "68f5e06bf21b02f6aece46ef",   "subId": "68f5e2ca006ab12b84b5db9d" } ]</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 15:29:28 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 111 – Отримання списку всіх повідомлень

Request URL  
<http://localhost:5129/Message/68f5e5aa006ab12b84b5db9d>

Server response

Code	Details
200	Response body <pre>{   "id": "68f5e5aa006ab12b84b5db9d",   "title": "Лінійка просрочена",   "ownerId": "68f5e06bf21b02f6aece46ef",   "subId": "68f5e2ca006ab12b84b5db9d" }</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 15:30:19 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 112 – Отримання конкретного повідомлення

Request URL  
<http://localhost:5129/Message>

Server response

Code	Details
201 Undocumented	Response body <pre>{   "id": "69038551b40392ba552ffff8a",   "title": "Лінійка додана",   "ownerId": "68f5e00af21b02f6aece46ec",   "subId": "68f5e1c5006ab12b84b5db87" }</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 15:33:37 GMT location: http://localhost:5129/Message/69038551b40392ba552ffff8a server: Kestrel transfer-encoding: chunked</pre>

Рисунок 113 – Створення нового повідомлення

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						73

Рисунок 114 – Вміст колекції після додавання

Рисунок 115 – Повне оновлення повідомлення

Рисунок 116 – Вміст колекції після повного оновлення

Рисунок 117 – Часткове оновлення повідомлення

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						74

**Elements.Messages**

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 1.7KB TOTAL DOCUMENTS: 11 INDEXES TOTAL SIZE: 36KB

**Find** Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass

Filter Type a query: { field: 'value' } **Apply** Options

`_id: ObjectId('69038551b40392ba552fff8a')  
title: 'Вміст повідомлення після часткового оновлення'  
ownerId: '68f5e00fa0fb21b02f6aece46ec'  
subId: '68f5e1c5006ab12b84b5db87'`

System Status: All Good

Рисунок 118 – Вміст колекції після часткового оновлення повідомлення

Request URL  
`http://localhost:5129/Message/69038551b40392ba552fff8a`

Server response

Code	Details
204 Undocumented	Response headers <code>date: Thu, 30 Oct 2025 15:50:55 GMT server: Kestrel</code>

Рисунок 119 – Видалення повідомлення

**Elements.Messages**

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 1.51KB TOTAL DOCUMENTS: 10 INDEXES TOTAL SIZE: 36KB

**Find** Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass

Filter Type a query: { field: 'value' } **Apply** Options

`_id: ObjectId('68f5e5aa006ab12b84b5db9d')  
title: 'Підписка просрочена'  
ownerId: '68f5e00fb21b02f6aece46ef'  
subId: '68f5e2ca006ab12b84b5db90'`

System Status: All Good

Рисунок 120 – Вміст колекції після оновлення

## Завдання 5

В проект із попереднього завдання було додано нові моделі, автомаминг, налаштовано використання репозиторій, сервісів, автомаперів за допомогою DI.

Змін.	Арк.	№ докум.	Підпис	Дата	Арк.
					75

**ЛР.ОК24.ПІ231.03.04**

Приклади використання мапінгу в програмі:

PeopleController.cs:

```
using AutoMapper;
using LW4_task_3.Interfaces;
using LW4_task_3.Models.Entities;
using LW4_task_3.Models.Request;
using LW4_task_3.Models.Response;
using LW4_task_3.Validators;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using System.Text.Json;
namespace LW4_task_3.Controllers
{
    [Route("[controller]")]
    [ApiController]
    public class PeopleController : ControllerBase
    {
        private readonly IPeopleService _peopleService;
        // Створення змінної для подальшого вкористання за допомогою DI
        private readonly IMapper _mapper;
        public PeopleController(IPeopleService peopleService, IMapper mapper)
        {
            _peopleService = peopleService;
            _mapper = mapper;
        }
        [HttpGet]
        public async Task<ActionResult<IEnumerable<PeopleResponse>>> Get(
            [FromQuery] string? name, [FromQuery] string? email)
        {

```

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ПІ231.03.04**

Арк.

76

```

    {
        try
        {
            var peoplesItems = await _peopleService.GetPeoplesItemsAsync(n
ame, email);
            // Перетворення DTO в Entity
            var peoples = _mapper.Map<IEnumerable<PeopleResponse>>(pe
oplesItems);
            return Ok(peoples);
        }
        catch (KeyNotFoundException kex)
        {
            return NotFound(kex.Message);
        }
    }

    [HttpGet("{id}")]
    public async Task<ActionResult<PeopleResponse>> GetById(string id
)
{
    try
    {
        ValidElement.ValidId(id);
        var peopleItem = await _peopleService.GetByIdAsync(id);
        // Перетворення DTO в Entity
        var people = _mapper.Map<PeopleResponse>(peopleItem);
        return Ok(people);
    }
    catch (KeyNotFoundException kex)
    {
        return NotFound(kex.Message);
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        }

        catch(ArgumentException aex)
        {
            return BadRequest(aex.Message);
        }
    }

    [HttpPost]
    public async Task<IActionResult> Create(PeopleRequest item)
    {
        // Перетворення DTO в Entity
        var element = _mapper.Map<PeopleItem>(item);
        await _peopleService.CreateAsync(element);

        // Перетворення Entity в DTO
        var resp = _mapper.Map<PeopleResponse>(element);
        return CreatedAtAction(nameof(GetById), new { Id = resp.Id }, resp);
    }

    [HttpPut("{id}")]
    public async Task<IActionResult> Update(string id, PeopleRequest item)
    {
        try
        {
            ValidElement.ValidId(id);

            // Перетворення DTO в Entity
            var element = _mapper.Map<PeopleItem>(item);
            await _peopleService.UpdateAsync(id, element);
            return NoContent();
        }

        catch (KeyNotFoundException kex)
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    {
        return NotFound(kex.Message);
    }
    catch (ArgumentException aex)
    {
        return BadRequest(aex.Message);
    }
}

[HttpPatch("{id}")]
public async Task<IActionResult> UpdatePart(string id, JsonElement e
lement)
{
    try
    {
        ValidElement.ValidId(id);
        var upPeople = await _peopleService.GetByIdAsync(id);

        // Перетворення Entity в DTO
        var temp = _mapper.Map<PeopleRequest>(upPeople);
        if (element.TryGetProperty("name", out var name))
            temp.Name = name.GetString();
        if (element.TryGetProperty("email", out var email))
            temp.Email = email.GetString();
        if (!TryValidateModel(temp))
            return BadRequest(ModelState);

        //Маппінг DTO в Entity
        _mapper.Map(temp, upPeople);
        await _peopleService.UpdateAsync(id, upPeople);
        return NoContent();
    }
    catch (KeyNotFoundException kex)

```

Змін.	Арк.	№ докум.	Підпис	Дата

```
    {
        return NotFound(kex.Message);
    }
    catch(ArgumentException aex)
    {
        return BadRequest(aex.Message);
    }
}
```

Створено профілі для маппінгу кожного типу моделей:

namespace LW4 task 3.Mapping

{

```
public class MessageProfile: Profile
```

{

```
public MessageProfile()
```

{

```
CreateMap<MessageRequest, MessageItem>().ReverseMap();
```

```
CreateMap<MessageItem, MessageResponse>();
```

}

}

namespace LW4\_task\_3.Mapping

{

public class PeopleProfile: Profile

{

```
public PeopleProfile()
```

{

```
CreateMap<PeopleRequest, PeopleItem>().ReverseMap();
```

```
CreateMap<PeopleItem, PeopleResponse>();
```

Змін.	Арк.	№ докум.	Підпис	Дата

ЛР ОК24 П1231 03 04

Aprk.

80

```

        }
    }

}

namespace LW4_task_3.Mapping
{
    public class SubProfile: Profile
    {
        public SubProfile()
        {
            CreateMap<SubscriptionRequest, SubscriptionItem>()
                .ForMember(d => d.Status, o => o.MapFrom(s => Enum.Parse<SubscriptionStatus>(s.Status,true)));
            CreateMap<SubscriptionItem, SubscriptionRequest>()
                .ForMember(d => d.Status, o => o.MapFrom(s => s.Status.ToString()));
            CreateMap<SubscriptionItem, SubscriptionResponse>()
                .ForMember(d => d.Status, o => o.MapFrom(s => s.Status.ToString()));
        }
    }
}

```

Додано до конструкторів контролерів, сервісів прив'язку до інтерфейсів, а не до об'єктів.

Виконано запити для перевірки роботи DI та Mapping

MessageController:

```

private readonly IMessageService _messageService;
private readonly IMapper _mapper;
public MessageController(IMessageService messageService, IMapper mapper)
{

```

Змін.	Арк.	№ докум.	Підпис	Дата

```
        _messageService = messageService;
        _mapper = mapper;
    }
```

SubController:

```
private readonly ISubService _subService;
private readonly IMapper _mapper;
public SubController(ISubService subService, IMapper mapper)
{
    _subService = subService;
    _mapper = mapper;
}
```

PeopleController:

```
private readonly IPeopleService _peopleService;
private readonly IMapper _mapper;
public PeopleController(IPeopleService peopleService, IMapper mappe
r)
{
    _peopleService = peopleService;
    _mapper = mapper;
}
```

MessageService:

```
private readonly IMessageRepository _messageRepository;
private readonly ISubRepository _subRepository;
private readonly IPeopleRepository _peopleRepository;
public MessageService(IMessageRepository messageRepository, ISub
Repository subRepository, IPeopleRepository peopleRepository)
{
    _messageRepository = messageRepository;
    _subRepository = subRepository;
    _peopleRepository = peopleRepository;
```

Змін.	Арк.	№ докум.	Підпис	Дата

```
}
```

SubService:

```
private readonly ISubRepository _subRepository;  
private readonly IPeopleRepository _peopleRepository;  
public SubService(ISubRepository subRepository, IPeopleRepository peopleRepository)  
{  
    _subRepository = subRepository;  
    _peopleRepository = peopleRepository;  
}
```

PeopleService:

```
private readonly IPeopleRepository _peopleRepository;  
public PeopleService(IPeopleRepository peopleRepository)  
{  
    _peopleRepository = peopleRepository;  
}
```

В Program.cs додано прив'язку між інтерфейсом та класом для реалізації DI:

```
using FluentValidation;  
using FluentValidation.AspNetCore;  
using LW4_task_3.Models;  
using LW4_task_3.Validators;  
using LW4_task_3.Services;  
using LW4_task_3.Interfaces;  
using LW4_task_3.InterfacesRepository;  
using LW4_task_3.Repositories;  
using LW4_task_3.Mapping;  
var builder = WebApplication.CreateBuilder(args);  
// Add services to the container.  
builder.Services.AddControllers().AddFluentValidation();
```

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ПІ231.03.04**

Арк.

83

```

builder.Services.AddScoped<IPeopleRepository,PeopleRepository>();
builder.Services.AddScoped<ISubRepository,SubRepository>();
builder.Services.AddScoped<IMessageRepository,MessageRepository>();
builder.Services.AddScoped<IPeopleService, PeopleService>();
builder.Services.AddScoped<ISubService, SubService>();
builder.Services.AddScoped<IMessageService, MessageService>();
// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen(options =>
{
    var xmlFile = $"'{System.Reflection.Assembly.GetExecutingAssembly().GetName().Name}'.xml";
    var xmlPath = Path.Combine(AppContext.BaseDirectory, xmlFile);
    options.IncludeXmlComments(xmlPath);
});
builder.Services.AddValidatorsFromAssemblyContaining<PeopleValidator>();
builder.Services.AddValidatorsFromAssemblyContaining<SubValidator>();
builder.Services.AddValidatorsFromAssemblyContaining<MessageValidator>();
builder.Services.AddAutoMapper(typeof(PeopleProfile));
builder.Services.AddAutoMapper(typeof(SubProfile));
builder.Services.AddAutoMapper(typeof(MessageProfile));
var app = builder.Build();
// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    }

    app.UseAuthorization();

    app.MapControllers();

    app.Run();

```

В програмі DI працює за допомогою заміни створення об'єктів у конструкторах класів на прив'язку до інтерфейсів, які підставляються автоматично за допомогою реєстрації в Program.cs. AutoMapper також використовується за допомогою DI, клас має об'єкт типу IMapper за допомогою якого робить маппінг, зареєстрований в класові – профілі.

Здійснено перевірку роботи CRUD – операцій після оновлення проекту.

```

http://localhost:5129/People

Server response

Code Details

200 Response body
[{"email": "Email@a.com"}, {"id": "68f5e023f21b02f6aece46ed", "name": "Solosolo", "email": "solosolosolo5@gmail.com"}, {"id": "68f5e054f21b02f6aece46ee", "name": "Susanteam", "email": "Susan@Email.com"}, {"id": "68f5eb6bf21b02f6aece46ef", "name": "Mikefreez", "email": "Mike@Email.com"}, {"id": "68ff7822a866d7d74955a11c", "name": "Chorpovnii", "email": "Chort@email.com"}, {"id": "690113f4b26a5f31097a0fad", "name": "Bashuk", "email": "bashuk77@gmail.com"}]

Response headers
content-type: application/json; charset=utf-8
date: Thu, 30 Oct 2025 16:56:08 GMT
server: Kestrel
transfer-encoding: chunked

```

Рисунок 121 – Отримано список всіх користувачів

```

Curl
curl -X 'GET' \
'http://localhost:5129/People?name=Somename' \
-H 'accept: text/plain'

Request URL
http://localhost:5129/People?name=Somename

Server response

Code Details

200 Response body
[{"id": "68f5e00af21b02f6aece46ec", "name": "Somename", "email": "Email@a.com"}]

Response headers
content-type: application/json; charset=utf-8
date: Thu, 30 Oct 2025 16:59:01 GMT
server: Kestrel
transfer-encoding: chunked

```

Змін.	Арк.	№ докум.	Підпис	Дата

Рисунок 122 – Отримано користувачів за вказаним фільтром

Request URL  
http://localhost:5129/People/68f5e00af21b02f6aece46ec

Server response

Code	Details
200	Response body <pre>{   "id": "68f5e00af21b02f6aece46ec",   "name": "Somename",   "email": "Email@o.com" }</pre> <div style="display: flex; justify-content: space-end;"> <span></span> <span></span> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 16:59:53 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 123 – Отримано користувача за Id

Request URL  
http://localhost:5129/People

Server response

Code	Details
201 <i>Undocumented</i>	Response body <pre>{   "id": "69039a91e402e8cfbe2a51e1",   "name": "Newuser",   "email": "Newuser@email.com" }</pre> <div style="display: flex; justify-content: space-end;"> <span></span> <span></span> </div> Response headers <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 17:04:17 GMT location: http://localhost:5129/People/69039a91e402e8cfbe2a51e1 server: Kestrel transfer-encoding: chunked</pre>

Рисунок 124 – Створено нового користувача

Request URL  
http://localhost:5129/People/68f5e00af21b02f6aece46ec

Server response

Code	Details
204 <i>Undocumented</i>	Response headers <pre>date: Thu, 30 Oct 2025 17:06:16 GMT server: Kestrel</pre>

Рисунок 125 – Оновлено повністю користувача

Request URL  
http://localhost:5129/People/68f5e00af21b02f6aece46ec

Server response

Code	Details
204 <i>Undocumented</i>	Response headers <pre>date: Thu, 30 Oct 2025 17:07:18 GMT server: Kestrel</pre>

Рисунок 126 – Здійснено часткове оновлення

Request URL  
http://localhost:5129/People/68f5e00af21b02f6aece46ec

Server response

Code	Details
204 <i>Undocumented</i>	Response headers <pre>date: Thu, 30 Oct 2025 17:09:42 GMT server: Kestrel</pre>

Рисунок 127 – Видалено користувача

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						86

Request URL  
http://localhost:5129/Sub

Server response

Code	Details
200	<p>Response body</p> <pre>{   "service": "XBox",   "status": "Overdue" }, {   "id": "68f5e301006ab12b84b5db91",   "ownerId": "68f5e06bf21b02f6aece46ef",   "service": "Steam",   "status": "Active" }, {   "id": "68f5e338006ab12b84b5db92",   "ownerId": "68f5e06bf21b02f6aece46ef",   "service": "Apple",   "status": "Active" }, {   "id": "69011d6e8a51e5b05552f127",   "ownerId": "68f5e00af21b02f6aece46ec",   "service": "ChortTV",   "status": "Expectation" }, {   "id": "69011d9e8a51e5b05552f128",   "ownerId": "68f5e00af21b02f6aece46ec",   "service": "ChortloveV",   "status": "Expectation" } ]</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 17:11:03 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 128 – Отримано список всіх підписок

Request URL  
http://localhost:5129/Sub?status=1

Server response

Code	Details
200	<p>Response body</p> <pre>{   "service": "YouTube Music",   "status": "Expectation" }, {   "id": "68f5e222006ab12b84b5db8b",   "ownerId": "68f5e023f21b02f6aece46ed",   "service": "AplicFilms",   "status": "Expectation" }, {   "id": "68f5e245006ab12b84b5db8c",   "ownerId": "68f5e054f21b02f6aece46ee",   "service": "XBox",   "status": "Expectation" }, {   "id": "69011d6e8a51e5b05552f127",   "ownerId": "68f5e00af21b02f6aece46ec",   "service": "ChortTV",   "status": "Expectation" }, {   "id": "69011d9e8a51e5b05552f128",   "ownerId": "68f5e00af21b02f6aece46ec",   "service": "ChortloveV",   "status": "Expectation" } ]</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 17:11:46 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 129 – Отримано список підписок, що очікують на активацію

Request URL  
http://localhost:5129/Sub/68f5e1b1006ab12b84b5db86

Server response

Code	Details
200	<p>Response body</p> <pre>{   "id": "68f5e1b1006ab12b84b5db86",   "ownerId": "68f5df5f21b02f6aece46eb",   "service": "XBox",   "status": "Expectation" }</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 17:12:52 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 130 – Отримано підписку за Id

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						87

Request URL  
**http://localhost:5129/Sub**

Server response

Code	Details
201 Undocumented	<b>Response body</b> <pre>{     "id": "69039d0ae402e8cfbe2a51e2",     "ownerId": "69039a91e402e8cfbe2a51e1",     "service": "XBox",     "status": "Active" }</pre> <div style="text-align: right;"> <a href="#">Copy</a> <a href="#">Download</a> </div>
<b>Response headers</b>	
content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 17:14:50 GMT location: http://localhost:5129/Sub/69039d0ae402e8cfbe2a51e2 server: Kestrel transfer-encoding: chunked	

Рисунок 130 – Створено нову підписку для користувача

Request URL  
**http://localhost:5129/Sub/69039d48e402e8cfbe2a51e3**

Server response

Code	Details
204 Undocumented	<b>Response headers</b> <pre>date: Thu, 30 Oct 2025 17:20:44 GMT  server: Kestrel</pre>

Рисунок 131 – Здійснено повне оновлення підписки

Request URL  
**http://localhost:5129/Sub/69039d48e402e8cfbe2a51e3**

Server response

Code	Details
204 Undocumented	<b>Response headers</b> <pre>date: Thu, 30 Oct 2025 17:21:46 GMT  server: Kestrel</pre>

Рисунок 132 – Здійснено часткове оновлення підписки

Request URL  
**http://localhost:5129/Sub/69039d48e402e8cfbe2a51e3**

Server response

Code	Details
204 Undocumented	<b>Response headers</b> <pre>date: Thu, 30 Oct 2025 17:22:35 GMT  server: Kestrel</pre>

Рисунок 133 – Видалено підписку

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						88

Request URL  
**http://localhost:5129/Message**

Server response

Code	Details
200	<p>Response body</p> <pre>{     "ownerId": "68f5e054f21b02f6aee46ee",     "subId": "68f5e253006ab12b84b5db8d" }, {     "id": "68f5e504006ab12b84b5db99",     "title": "Підписка вступила в дію",     "ownerId": "68f5e054f21b02f6aee46ee",     "subId": "68f5e253006ab12b84b5db8d" }, {     "id": "68f5e591006ab12b84b5db9b",     "title": "Підписка вступила в дію",     "ownerId": "68f5e06bf21b02f6aee46ef",     "subId": "68f5e2ca006ab12b84b5db9b" }, {     "id": "68f5e5a0006ab12b84b5db9c",     "title": "До оплати пірниски день",     "ownerId": "68f5e06bf21b02f6aee46ef",     "subId": "68f5e2ca006ab12b84b5db9b" }, {     "id": "68f5e5aa006ab12b84b5db9d",     "title": "Підписка просрочена",     "ownerId": "68f5e06bf21b02f6aee46ef",     "subId": "68f5e2ca006ab12b84b5db9b" } ]</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 17:23:31 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 134 – Отримано список всіх повідомлень

Request URL  
**http://localhost:5129/Message/68f5e504006ab12b84b5db99**

Server response

Code	Details
200	<p>Response body</p> <pre>{     "id": "68f5e504006ab12b84b5db99",     "title": "Підписка вступила в дію",     "ownerId": "68f5e054f21b02f6aee46ee",     "subId": "68f5e253006ab12b84b5db8d" }</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 17:24:27 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 135 – Отримано повідомлення за Id

Request URL  
**http://localhost:5129/Message**

Server response

Code	Details
201 Undocumented	<p>Response body</p> <pre>{     "id": "69039f92e402e8cfbe2a51e4",     "title": "Підписка активована",     "ownerId": "68f5e023f21b02f6aee46ed",     "subId": "68f5e210006ab12b84b5db8a" }</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Thu, 30 Oct 2025 17:25:38 GMT location: http://localhost:5129/Message/69039f92e402e8cfbe2a51e4 server: Kestrel transfer-encoding: chunked</pre>

Рисунок 136 – Створено нове повідомлення

Request URL  
**http://localhost:5129/Message/69039f92e402e8cfbe2a51e4**

Server response

Code	Details
204 Undocumented	<p>Response headers</p> <pre>date: Thu, 30 Oct 2025 17:26:38 GMT server: Kestrel</pre>

Змін.	Арк.	№ докум.	Підпис	Дата

**ЛР.ОК24.ПІ231.03.04**

Арк.

89

Рисунок 137 – Повністю оновлено повідомлення



Рисунок 138 – Частково оновлено повідомлення



Рисунок 139 – Видалено повідомлення

## Завдання 6

Встановлено необхідні пакети та налаштовано appsetting.json:

```
{  
    "Logging": {  
        "LogLevel": {  
            "Default": "Information",  
            "Microsoft.AspNetCore": "Warning"  
        }  
    },  
    "AllowedHosts": "*",  
    "JwtSettings": {  
        "SecretKey": "My_Super_Secret_Key_2w22w3_4w52q2728=93+_some",  
        "Issuer": "http://localhost:5129",  
        "Audience": "http://localhost:5129",  
        "AccessTokenExpirationMinutes": 60,  
        "RefreshTokenExpirationDays": 30  
    }  
}
```

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						90

Створено модель JwtSetting та сутність для роботи із БД, і DTO моделі для логіну та реєстрації:

```
namespace LW4_task_3.Models

{
    public class JwtSettings
    {
        public string SecretKey { get; set; }
        public string Issuer { get; set; }
        public string Audience { get; set; }
        public int AccessTokenExpirationMinutes { get; set; }
        public int RefreshTokenExpirationDays { get; set; }
    }

}

using LW4_task_3.Enums;
using Microsoft.AspNetCore.Identity;
using MongoDB.Bson;
using MongoDB.Bson.Serialization.Attributes;
namespace LW4_task_3.Models.Entities
{
    public class UserItem
    {
        [BsonId]
        [BsonRepresentation(BsonType.ObjectId)]
        public string? Id { get; set; }
        [BsonElement("username")]
        public string? UserName { get; set; }
        [BsonElement("email")]
        public string? Email { get; set; }
        [BsonElement("passwordHash")]
        public string? Password { get; set; }
    }
}
```

Змін.	Арк.	№ докум.	Підпис	Дата

```

[BsonElement("refreshToken")]
public string? RefreshToken { get; set; }

[BsonElement("refreshTokenExpireTime")]
public DateTime? RefreshTokenExpireTime { get; set; }

[BsonElement("role")]
public UserRole Role { get; set; } = UserRole.User;

}

}

namespace LW4_task_3.Models.Request
{
    public class LoginModel
    {
        public string Email { get; set; }
        public string Password { get; set; }
    }
}

namespace LW4_task_3.Models.Request
{
    public class UserRegistration
    {
        public string? UserName { get; set; }
        public string? Email { get; set; }
        public string? Password { get; set; }
    }
}

```

Створено сервіс для хешування та сервіс для генерування accesstoken, refresh token:

```

using LW4_task_3.Interface.Interfaces;
using System.Security.Cryptography;
using System.Text;

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

namespace LW4_task_3.Services
{
    public class PasswordHasher : IPasswordHasher
    {
        public string Hash(string password)
        {
            using var sha256 = SHA256.Create();
            var bytes = Encoding.UTF8.GetBytes(password);
            var hash = sha256.ComputeHash(bytes);
            return Convert.ToString(hash);
        }

        public bool Verify(string password, string hash)
        {
            var hashPas = Hash(password);
            return hashPas == hash;
        }
    }

    using LW4_task_3.Models;
    using Microsoft.Extensions.Options;
    using Microsoft.IdentityModel.Tokens;
    using System.Text;
    using System.Security.Claims;
    using System.IdentityModel.Tokens.Jwt;
    using LW4_task_3.Models.Entities;
    using System.Security.Cryptography;
    using LW4_task_3.Enums;
    namespace LW4_task_3.Services
    {
        public class JwtTokenGenerator

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    {

        private readonly JwtSettings _jwtSettings;
        public JwtTokenGenerator(IOptions<JwtSettings> options)
        {
            _jwtSettings = options.Value;
        }

        public string GenerateToken(UserItem user)
        {
            var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(_jwtSettings.SecretKey));
            var creds = new SigningCredentials(key, SecurityAlgorithms.HmacSha256);

            var claims = new List<Claim>
            {
                new Claim(JwtRegisteredClaimNames.Sub, user.Id),
                new Claim(JwtRegisteredClaimNames.Email, user.Email),
                new Claim(JwtRegisteredClaimNames.Name, user.UserName)
            };

            foreach(UserRole role in Enum.GetValues(typeof(UserRole)))
            {
                if(role == UserRole.None)
                    continue;

                if(user.Role.HasFlag(role))
                    claims.Add(new Claim(ClaimTypes.Role, role.ToString()));
            }

            claims.Add(new Claim("rolesValue", ((int)user.Role).ToString()));

            var token = new JwtSecurityToken(
                issuer: _jwtSettings.Issuer,
                audience: _jwtSettings.Audience,
                claims: claims,

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        expires: DateTime.UtcNow.AddMinutes(_jwtSettings.AccessToke
nExpirationMinutes),
        signingCredentials: creds
    );
    return new JwtSecurityTokenHandler().WriteToken(token);
}

public string GeneretaRefreshToken()
{
    var randomNumbers = new byte[32];
    using var random = RandomNumberGenerator.Create();
    random.GetBytes(randomNumbers);
    return Convert.ToBase64String(randomNumbers);
}

public ClaimsPrincipal? GetClaimsPrincipalFromExpiredToken(string t
oken)
{
    var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(_j
wtSettings.SecretKey));
    var tokenValidationParametr = new TokenValidationParameters
    {
        ValidateAudience = true,
        ValidateIssuer = true,
        ValidateIssuerSigningKey = true,
        ValidIssuer = _jwtSettings.Issuer,
        ValidAudience = _jwtSettings.Audience,
        IssuerSigningKey = key,
        ValidateLifetime = false
    };
}

var tokenHandler = new JwtSecurityTokenHandler();

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        try
        {
            var principal = tokenHandler.ValidateToken(token, tokenValidationParametr, out var securityToken);

            if (securityToken is not JwtSecurityToken jwtSecurityToken ||
                !jwtSecurityToken.Header.Alg.Equals(SecurityAlgorithms.HmacSha256, StringComparison.InvariantCultureIgnoreCase))
                return null;

            return principal;
        }
        catch
        {
            return null;
        }
    }
}

```

Створено сервіс та репозиторій для зберігання та роботи із сущностями в MongoDB:

```

using LW4_task_3.Clients;
using LW4_task_3.Interface.InterfacesRepository;
using LW4_task_3.Models.Entities;
using MongoDB.Driver;
namespace LW4_task_3.Repositories
{
    public class UserRepository : IUserRepository
    {
        IMongoCollection<UserItem> _users;

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

public UserRepository()
{
    _users = MongoDBClient.Instance.GetCollection<UserItem>("Users");
}

public async Task CreateAsync(UserItem element) =>
    await _users.InsertOneAsync(element);
public async Task DeleteAsync(string id) =>
    await _users.DeleteOneAsync(x => x.Id == id);
public async Task<List<UserItem>> GetAllAsync() =>
    await _users.Find(x => true).ToListAsync();
public async Task<UserItem> GetByEmailAsync(string email) =>
    await _users.Find(x => x.Email == email).FirstOrDefaultAsync();
public async Task<UserItem> GetByIdAsync(string id) =>
    await _users.Find(x => x.Id == id).FirstOrDefaultAsync();
public async Task<bool> IsExist(string id) =>
    await _users.Find(x => x.Id == id).AnyAsync();
public async Task UpdateAsync(string id, UserItem element) =>
    await _users.ReplaceOneAsync(x => x.Id == id, element);
}
}

```

Створено TestController для роботи перевірки роботи автентифікації:

```

using LW4_task_3.Enums;
using LW4_task_3.Interface.Interfaces;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.SignalR;
using System.Security.Claims;

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

namespace LW4_task_3.Controllers
{
    [Route("[controller]")]
    [ApiController]
    public class TestController : ControllerBase
    {
        IUserService _userService;
        public TestController(IUserService userService)
        {
            _userService = userService;
        }
        [Authorize]
        [HttpGet("private")]
        public async Task<IActionResult> GetIdEmail()
        {
            var email = User?.Claims.FirstOrDefault(c => c.Type == ClaimType
s.Email)?.Value;
            var user = await _userService.GetByEmailAsync(email);
            return Ok(new
            {
                UserId = user.Id,
                Email = email,
            });
        }
    }
}

```

Створено контролер для автентифікації, логіну та оновлення токену:

```

using AutoMapper;
using LW4_task_3.Interface.Interfaces;
using LW4_task_3.Models.Entities;

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

using LW4_task_3.Models.Request;
using LW4_task_3.Services;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Configuration.UserSecrets;
using System.Security.Claims;
namespace LW4_task_3.Controllers
{
    [Route("[controller]")]
    [ApiController]
    public class AuthController : ControllerBase
    {
        private readonly IUserService _userService;
        private readonly IPasswordHasher _passwordHasher;
        private readonly JwtTokenGenerator _jwtGenerator;
        private readonly IMapper _mapper;
        public AuthController(IUserService userService,
            IPasswordHasher passwordHasher,
            JwtTokenGenerator jwtTokenGenerator, IMapper mapper)
        {
            _userService = userService;
            _passwordHasher = passwordHasher;
            _jwtGenerator = jwtTokenGenerator;
            _mapper = mapper;
        }
        [HttpPost("register")]
        public async Task<IActionResult> Reg(UserRegistration user)
        {
            var userItem = _mapper.Map<UserItem>(user);
            try

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    {
        await _userService.CreateAsync(userItem);
        return Ok("Користувача зареєстровано");
    }
    catch (ArgumentException aex)
    {
        return BadRequest(aex.Message);
    }
}

[HttpPost("login")]
public async Task<IActionResult> LogIn(LoginModel loginModel)
{
    string message = "Невірна пошта чи пароль";
    UserItem user;
    try
    {
        user = await _userService.GetByEmailAsync(loginModel.Email);
    }
    catch (KeyNotFoundException)
    {
        return Unauthorized(message);
    }
    if(!_passwordHasher.Verify(loginModel.Password,user.Password))
        return BadRequest(message);
    var token = _jwtGenerator.GenerateToken(user);
    user.RefreshToken= _jwtGenerator.GenerateRefreshToken();
    user.RefreshTokenExpireTime = DateTime.UtcNow.AddDays(7);
    await _userService.UpdateAsync(user.Id, user);
    return Ok(new
    {

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        Token = token,
        TokenExpireTime = DateTime.UtcNow.AddMinutes(60),
        ResreshToken = user.RefreshToken,
        ResreshTokenExpireTime = user.RefreshTokenExpireTime
    });
}

[HttpPost("refresh")]
public async Task<IActionResult> Refresh([FromHeader(Name = "Authorization")] string authHead, string refreshToken)
{
    if (string.IsNullOrEmpty(authHead) || !authHead.StartsWith("Bearer"))
        return BadRequest("Немає або неправильний заголовок");

    var token = authHead.Substring("Bearer ".Length);
    var principal = _jwtGenerator.GetClaimsPrincipalFromExpiredToken(
        token);

    if (principal == null)
        return Unauthorized("Невірний token");

    var id = principal?.Claims.FirstOrDefault(c => c.Type == ClaimTypes
        .NameIdentifier)?.Value;

    UserItem user;
    try
    {
        user = await _userService.GetByIdAsync(id);
    }
    catch (KeyNotFoundException kex)
    {
        return NotFound(kex.Message);
    }

    if (user.RefreshToken != refreshToken)

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    || user.RefreshTokenExpireTime <= DateTime.UtcNow
        return Unauthorized("Невірний RefreshToken");

    var newToken = _jwtGenerator.GenerateToken(user);
    var newrefreshToken = _jwtGenerator.GeneretaRefreshToken();
    user.RefreshToken = newrefreshToken;
    user.RefreshTokenExpireTime = DateTime.UtcNow.AddDays(7);
    await _userService.UpdateAsync(id, user);
    return Ok(new
    {
        Token = token,
        TokenExpireTime = DateTime.UtcNow.AddMinutes(60),
        ResreshToken = user.RefreshToken,
        ResreshTokenExpireTime = user.RefreshTokenExpireTime
    });
}
}
}

```

Створено контролер для роботи із User:

```

using AutoMapper;
using LW4_task_3.Enums;
using LW4_task_3.Interface.Interfaces;
using LW4_task_3.Models.Entities;
using LW4_task_3.Models.Request;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.IdentityModel.Protocols.OpenIdConnect;
using System.Data;
using ZstdSharp.Unsafe;
namespace LW4_task_3.Controllers.A_A

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

{
    [Route("[controller]")]
    [ApiController]
    [Authorize(Roles = nameof(UserRole.Admin))]
    public class UserController : ControllerBase
    {
        private readonly IUserService _userService;
        private readonly IMapper _mapper;
        private readonly IPasswordHasher _passwordHasher;
        public UserController(IUserService userService, IMapper mapper, IPasswordHasher passwordHasher)
        {
            _userService = userService;
            _mapper = mapper;
            _passwordHasher = passwordHasher;
        }
        [Authorize(Roles = $" {nameof(UserRole.Admin)} , {nameof(UserRole.Manager)}")]
        [HttpGet]
        public async Task<IActionResult> GetAllUser()
        {
            try
            {
                var users = await _userService.GetAllAsync();
                return Ok(users);
            }
            catch (KeyNotFoundException kex)
            {
                return NotFound(kex.Message);
            }
        }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

    }

    [Authorize(Roles = $"{{nameof(UserRole.Admin)}},{{nameof(UserRole.
Manager)}}")]
    [HttpGet("{id}")]
    public async Task<IActionResult> GetOneUser(string id)
    {
        try
        {
            var user = await _userService.GetByIdAsync(id);
            return Ok(user);
        }
        catch(NotFoundException kex)
        {
            return NotFound(kex.Message);
        }
    }

    [HttpPut("{id}")]
    public async Task<IActionResult> UpdateUser(string id, UserRegistrat
ion userRegistration)
    {
        try
        {
            var updateUser = await _userService.GetByIdAsync(id);
            var emailUser = await _userService.GetByEmailAsync(userRegist
ration.Email);

            if (emailUser is not null && emailUser.Id != updateUser.Id)
                return BadRequest("Ця пошта вже використовується");

            userRegistration.Password = _passwordHasher.Hash(userRegistrat
ion.Password);

            _mapper.Map(userRegistration, updateUser);
        }
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        await _userService.UpdateAsync(id, updateUser);
        return NoContent();
    }

    catch (KeyNotFoundException kex)
    {
        return NotFound(kex.Message);
    }
}

[HttpDelete("{id}")]
public async Task<IActionResult> DeleteUser(string id)
{
    try
    {
        await _userService.DeleteAsync(id);
        return NoContent();
    }

    catch (KeyNotFoundException kex)
    {
        return NotFound(kex.Message);
    }
}

[HttpPost("setRole")]
public async Task<IActionResult> SetRole(string id, UserRole role)
{
    try
    {
        var user = await _userService.GetByIdAsync(id);
        user.Role = role;
        await _userService.UpdateAsync(id, user);
    }
}

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        return Ok(new { UserName = user.UserName, Email = user.Email
, Role = (int)user.Role });
    }

    catch (KeyNotFoundException kex)
    {
        return NotFound(kex.Message);
    }
}

[HttpPost("addRole")]
public async Task<IActionResult> AddRole(string id, UserRole role)
{
    try
    {
        var user = await _userService.GetByIdAsync(id);
        user.Role |= role;
        await _userService.UpdateAsync(id, user);
        return Ok(new { UserName = user.UserName, Email = user.Email
, Role = (int)user.Role });
    }

    catch (KeyNotFoundException kex)
    {
        return NotFound(kex.Message);
    }
}

[HttpPost("removeRole")]
public async Task<IActionResult> RemoveRole(string id, UserRole rol
e)
{
    try
    {

```

Змін.	Арк.	№ докум.	Підпис	Дата

```

        var user = await _userService.GetByIdAsync(id);
        user.Role &= ~role;
        await _userService.UpdateAsync(id, user);
        return Ok(new { UserName = user.UserName, Email = user.Email
            , Role = (int)user.Role });
    }
    catch (KeyNotFoundException kex)
    {
        return NotFound(kex.Message);
    }
}
}
}
}

```

Протестовано роботу програми з різними ролями:

Request URL  
http://localhost:5129/Auth/register

Server response

Code	Details
200	Response body Користувача зареєстровано <a href="#">Copy</a> <a href="#">Download</a>

Response headers

content-type: text/plain; charset=utf-8
date: Sun,09 Nov 2025 12:50:49 GMT
server: Kestrel
transfer-encoding: chunked

Рисунок 140 – Реєстрація нового користувача

Request URL  
http://localhost:5129/Auth/login

Server response

Code	Details
400 Undocumented	Error: Bad Request
	Response body Невірна пошта чи пароль <a href="#">Copy</a> <a href="#">Download</a>

Response headers

content-type: text/plain; charset=utf-8
date: Sun,09 Nov 2025 12:47:49 GMT
server: Kestrel
transfer-encoding: chunked

Рисунок 141 – Невірно введений пароль

Змін.	Арк.	№ докум.	Підпис	Дата

Request URL  
**http://localhost:5129/Auth/login**

Server response

Code	Details
200	Response body <pre>{   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJ2OTEwNGZkODM3NjM1ZWfjODkwZThhYmMilCJ1bWFpbCI6IkFkbWluQGdtYmIsLmNbSIsImh0dHA6Ly9zY2h1bWFzLm1pY3Jvc29ndC5jb20vd3MvMjAwOCBwNi9pZGVudGl0e59jbGpbWvcms9ZS16IkfkbnUuixicm9sZXNlYlx1ZS16IjQilCJ1eHai0jE3NjI2OTU500gsIm1zcyI6Imh0dHA6Ly9sb2NhGhvc3Q6NEYOSIsImF1ZC16Imh0dHA6Ly9sb2NhbGwv3Q6NEYOSj9.eyJkZS5pdlUeZztb-505uCutIK2_11zKs17Qe04InI",   "tokenExpiresTime": "2025-11-09T13:46:29.2233787Z",   "refreshToken": "Cir0n5dczLNth9935xaTeUElsorhmly5wq/01GShj4=",   "refreshTokenExpiresTime": "2025-11-16T12:46:28.7366383Z" }</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 12:46:28 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 142 – Вхід із роллю Admin

Request URL  
**http://localhost:5129/Test/private**

Server response

Code	Details
200	Response body <pre>{   "userId": "69104fd837635eac890e8abc",   "email": "Admin@gmail.com" }</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 12:49:01 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 143 – Запит на TestController

Request URL  
**http://localhost:5129/User**

Server response

Code	Details
200	Response body <pre>[   {     "id": "69104fd837635eac890e8abc",     "userName": "Admin",     "email": "Admin@gmail.com",     "password": "GWE/3J2k+3Kxof62aRdujTyQ/5TVQZ4FI2PuqJ3+4d0=",     "refreshToken": "Cir0n5dczLNth9935xaTeUElsorhmly5wq/01GShj4=",     "refreshTokenExpiresTime": "2025-11-16T12:46:28.736Z",     "role": 4   },   {     "id": "69108e2adb14a7b661bc38b9",     "userName": "Manager",     "email": "Manager@gmail.com",     "password": "G+A1j1Cg84iauVtdMTuhLk/xBGR0cCirR3n0tScwlyM=",     "refreshToken": null,     "refreshTokenExpiresTime": null,     "role": 1   },   {     "id": "69108e7edb14a7b661bc38ba",     "userName": "User",     "email": "User@gmail.com",     "password": "JTjxdU/NhYCrPJCvqj+cZFsPpVVMFFgu/mcZ0rstUgI=",     "refreshToken": null,     "refreshTokenExpiresTime": null,     "role": 1   } ]</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 12:52:29 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 144 – Переглянути список всіх User-ів

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						108

Request URL  
**http://localhost:5129/User/69108e7edb14a7b661bc38ba**

Server response

Code	Details
200	<p>Response body</p> <pre>{   "id": "69108e7edb14a7b661bc38ba",   "userName": "User",   "email": "User@gmail.com",   "password": "JTJxU/NhYcRcPJcvqj+czFsPpVMMFFgu/mcZ0rstUgI=",   "refreshToken": null,   "refreshTokenExpireTime": null,   "role": 1 }</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 12:54:06 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 145 – Отримано інформацію про конкретного User-а

Request URL  
**http://localhost:5129/User/setRole?id=69108e2adb14a7b661bc38b9&role=1**

Server response

Code	Details
200	<p>Response body</p> <pre>{   "userName": "Manager",   "email": "Manager@gmail.com",   "role": 1 }</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 12:55:51 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 146 – Вкористано /setRole для зміни ролі менеджера

Request URL  
**http://localhost:5129/User/addRole?id=69108e2adb14a7b661bc38b9&role=2**

Server response

Code	Details
200	<p>Response body</p> <pre>{   "userName": "Manager",   "email": "Manager@gmail.com",   "role": 3 }</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 12:56:50 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 147 – Додано роль менеджера через /addRole

Request URL  
**http://localhost:5129/User/removeRole?id=69108e2adb14a7b661bc38b9&role=1**

Server response

Code	Details
200	<p>Response body</p> <pre>{   "userName": "Manager",   "email": "Manager@gmail.com",   "role": 2 }</pre> <p><a href="#">Copy</a> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 12:57:57 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 148 – Видалено роль User через /removeRole

Змін.	Арк.	№ докум.	Підпис	Дата

Request URL  
**http://localhost:5129/User/69108e2adb14a7b661bc38b9**

Server response

<b>Code</b>	<b>Details</b>
-------------	----------------

204 *Undocumented* Response headers

```
date: Sun, 09 Nov 2025 13:00:18 GMT
server: Kestrel
```

Рисунок 149 – Оновлено ім’я для Manager

Request URL  
**http://localhost:5129/User/69108e2adb14a7b661bc38b9**

Server response

<b>Code</b>	<b>Details</b>
-------------	----------------

200 Response body

```
{
  "id": "69108e2adb14a7b661bc38b9",
  "userName": "NewManager",
  "email": "Manager@gmail.com",
  "password": "G+AiJ1Cq84iauVtdmTuhLk/xBGR0cC1rR3n0t5cwMyM=",
  "refreshToken": null,
  "refreshTokenExpireTime": null,
  "role": 2
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Sun, 09 Nov 2025 13:01:14 GMT
server: Kestrel
transfer-encoding: chunked
```

Рисунок 150 – Результат оновлення

Request URL  
**http://localhost:5129/User/69108e2adb14a7b661bc38b9**

Server response

<b>Code</b>	<b>Details</b>
-------------	----------------

204 *Undocumented* Response headers

```
date: Sun, 09 Nov 2025 13:02:04 GMT
server: Kestrel
```

Рисунок 151 – Видалено оновленого користувача

Request URL  
**http://localhost:5129/User/69108e2adb14a7b661bc38b9**

Server response

<b>Code</b>	<b>Details</b>
-------------	----------------

404 *Undocumented* Error: Not Found

Response body

```
Не знайдено user з id: 69108e2adb14a7b661bc38b9
```

Response headers

```
content-type: text/plain; charset=utf-8
date: Sun, 09 Nov 2025 13:02:42 GMT
server: Kestrel
transfer-encoding: chunked
```

Рисунок 152 – Спроба отримання видаленогокористувача

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						110

Request URL  
<http://localhost:5129/Auth/refresh?refreshToken=C1r0nSdCzLNth99J5xaTeUEls0rhmlY5wqgK2F01GShJ4%3D>

Server response

Code	Details
200	<p>Response body</p> <pre>{   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiZ0OTEwOTE2MWRlMTRNZI2NjF1YzM4YmIiLCJlbWFpbCI6Ik1hbFnZXJAZ21hawuY29tIiwbmFtZSI6Ik1hbFnZXIiLCJodHRwOi8vc2NoZWlhcy5taWByb3NvZnQuV29tL3dzLzlwDgv0YvaRlbnRpdrIkvY2xhaW1zL3JvbGUoIiVc2VyyIiwm9sZXNvYw1kZSI6IjEiLCJleHAIoJE3NjI20TczNzEsIm1zcyl6Imh0dHA6Ly9sb2NhbgHvc3Q6NTeyOSIsImF1ZC16Imh0dHA6Ly9sb2NhbGhvC3Q6NTeyOS39_AL4kzQ5nPaiUEztb-50SuCutIK2_11zkns17Qe041uI",   "tokenExpiresTime": "2025-11-09T14:07:46.4491256Z",   "refreshToken": "7hEX+Ed7NWQaAutvpdxN4yeJ2G0khE2cdO1jy2Hw=",   "refreshTokenExpiresTime": "2025-11-16T13:07:46.3153626Z" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 13:07:45 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 154 – Оновлення токену

Request URL  
<http://localhost:5129/Auth/login>

Server response

Code	Details
200	<p>Response body</p> <pre>{   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiZ0OTEwOTE2MWRlMTRNZI2NjF1YzM4YmIiLCJlbWFpbCI6Ik1hbFnZXJAZ21hawuY29tIiwbmFtZSI6Ik1hbFnZXIiLCJodHRwOi8vc2NoZWlhcy5taWByb3NvZnQuV29tL3dzLzlwDgv0YvaRlbnRpdrIkvY2xhaW1zL3JvbGUoIiVc2VyyIiwm9sZXNvYw1kZSI6IjEiLCJleHAIoJE3NjI20TczNzEsIm1zcyl6Imh0dHA6Ly9sb2NhbgHvc3Q6NTeyOSIsImF1ZC16Imh0dHA6Ly9sb2NhbGhvC3Q6NTeyOS39_QNAFmIiv2Q8jagwCYMFm_AN7FKzH53oZSkatCIBwFE",   "tokenExpiresTime": "2025-11-09T14:09:32.0935208Z",   "refreshToken": "b/DZ37y3nc+bersnlhUYWAREJk0MDFvvzCubtIV0MM=",   "refreshTokenExpiresTime": "2025-11-16T13:09:31.9573391Z"</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 13:09:31 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 155 – Вхід з роллю Manager

Request URL  
<http://localhost:5129/Test/private>

Server response

Code	Details
200	<p>Response body</p> <pre>{   "userId": "69109161db14a7b661bc38bb",   "email": "Manager@gmail.com" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 13:11:15 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 156 – Виконання запиту за допомогою TestController

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						111

http://localhost:5129/User

Server response

Code	Details
200	Response body <pre>[   {     "id": "69104fd837635eac890e8abc",     "userName": "Admin",     "email": "Admin@gmail.com",     "password": "cVE/3J2k+3KkoF62aRdUjTyQ/5TVQZ4fI2PuqJ3+4d0=",     "refreshToken": "tHeX7+E+d7NWqAutvpdxNMye32G0kEg2zC0iJy2Hm=",     "refreshTokenExpireTime": "2025-11-16T13:07:46.315Z",     "role": 4   },   {     "id": "69108e7edb14a7b661bc38ba",     "userName": "User",     "email": "User@gmail.com",     "password": "t1Jxd/NHyCrPJCvqj+czFsPpVWMFFgu/mcZ0rstUgI=",     "refreshToken": "v7hzSn5JpGamIBcvqilYi/1IpQ0K28/aKnA0022sLY=",     "refreshTokenExpireTime": "2025-11-16T13:09:14.155Z",     "role": 1   },   {     "id": "69109161db14a7b661bc38bb",     "userName": "Manager",     "email": "Manager@gmail.com",     "password": "G+ai1Cq4iauVtMnTuhLk/xBG80cC1rR3n0tScwlyH=",     "refreshToken": "U1tELHuJ2q+h/W#06510SMvdI+rxD8e7xQRN55aIbc=",     "refreshTokenExpireTime": "2025-11-16T13:50:59.543Z",     "role": 2   } ]</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 13:51:31 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 157 – Отримано список всіх User-ів

Request URL

http://localhost:5129/User/69109187db14a7b661bc38bd

Server response

Code	Details
200	Response body <pre>{   "id": "69109187db14a7b661bc38bd",   "userName": "User3",   "email": "User3@gmail.com",   "password": "gVDvEd/qn4ZHnnSPb50R1ssSo4UNkMRksjBY2fSw9UPI=",   "refreshToken": null,   "refreshTokenExpireTime": null,   "role": 1 }</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Sun, 09 Nov 2025 13:52:25 GMT server: Kestrel transfer-encoding: chunked</pre>

Рисунок 158 – Отримано конкретного User-а

Request URL

http://localhost:5129/User/69109187db14a7b661bc38bd

Server response

Code	Details
403 <i>Undocumented</i>	Error: Forbidden
	Response headers <pre>content-length: 0 date: Sun, 09 Nov 2025 13:53:17 GMT server: Kestrel</pre>

Рисунок 159 – Спроба видалити User-а із роллю Manager

Змін.	Арк.	№ докум.	Підпис	Дата

```
Request URL
http://localhost:5129/Auth/login

Server response
Code Details
200 Response body
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2OTEwOTE4N2RiMTRNZiIjF1YzMAYmQjLCJlbnRpLCI6I1VzZXI1QGdtYmlsLmNbSisIm5hbWUiO1JVc2VNSisImh0dHA6Ly9zY2h1MFZlm1pY3Jvc29mdC5jb20vd3MvIjAwOC8wNi9pZGVudGl0eS9jbGpbXVcm9sZS16I1VzXl1lCjy62x1c1ZhbhV1jjoimSisImV4cC16Mtc2mjcwMDM1MCwiaXnZ1joiHR0cDovL2xvY2FaG9zD0e1MT15i1iviYXVki1joiHR0cDovL2xvY2FsaG9zD0e1MT15i1iviYXVki1joiHR0cDovL2xvY2F",
  "tokenExpiresTime": "2025-11-09T14:59:10.6282516Z",
  "refreshToken": "i3Suge12Pi70QP8ghbXEVJ4041uWoccg/DPF8XhuOFU=",
  "refreshTokenExpiresTime": "2025-11-16T13:59:10.4649829Z"
}

Response headers
content-type: application/json; charset=utf-8
date: Sun, 09 Nov 2025 13:59:10 GMT
server: Kestrel
transfer-encoding: chunked
```

Рисунок 160 – Здійснено вхід із роллю User

Request URL  
`http://localhost:5129/Test/private`

Server response

Code	Details
200	<p>Response body</p> <pre>{   "userId": "69109197db14a7b661bc38bd",   "email": "User5@gmail.com" }</pre> <p> </p>

Response headers

```
content-type: application/json; charset=utf-8  
date: Sun, 09 Nov 2025 14:00:54 GMT  
server: Kestrel  
transfer-encoding: chunked
```

Рисунок 161 – Виконано запит на TestController

Request URL  
`http://localhost:5129/User`

Server response

Code	Details
403 <i>Undocumented</i>	Error: Forbidden

Response headers

```
content-length: 0
date: Sun, 09 Nov 2025 14:01:43 GMT
server: Kestrel
```

Рисунок 162 – Спроба отримати список User-ів з невідповідною роллю

<http://localhost:5129/People>

Server response

Code Details

200 Response body

```
[ { "id": "68f5e054f21b02f6aece46ee", "name": "Susanteam", "email": "Susan@email.com" }, { "id": "68f5e06bf21b02f6aece46ef", "name": "Mikefreez", "email": "Mike@email.com" }, { "id": "68ff7822a866d7d74955a11c", "name": "Chorpovnii", "email": "Chort@email.com" }, { "id": "690113f4b26a5f31097a0fad", "name": "Bashuk", "email": "bashuk777@gmail.com" }, { "id": "69039a91e402e8cfbe2a51e1", "name": "Newuser", "email": "Newuser@email.com" } ]
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Sun, 09 Nov 2025 14:02:57 GMT
server: Kestrel
transfer-encoding: chunked
```


Рисунок 163 – Здійснено запит для отримання списку користувачів підписок

Request URL  
http://localhost:5129/People/68f5e054f21b02f6aece46ee

Server response

Code Details

200 Response body

```
{ "id": "68f5e054f21b02f6aece46ee", "name": "Susanteam", "email": "SusanEmail.com" }
```

Response headers

```
content-type: application/json; charset=utf-8
date: Sun, 09 Nov 2025 14:04:02 GMT
server: Kestrel
transfer-encoding: chunked
```

Download

Рисунок 164 – Отримано конкретного користувача підписки

Request URL  
http://localhost:5129/People/68f5e054f21b02f6aece46ee

Server response

Code Details

403 Error: Forbidden

Undocumented Response headers

```
content-length: 0
date: Sun, 09 Nov 2025 14:04:46 GMT
server: Kestrel
```

Рисунок 165 – Способ видалення користувача підписки

Висновок: на даній лабораторній роботі я реалізував CRUD – операції у C# ASP.Net Web – API, Minimal API та навчився працювати із ними, також здобув практичні навики по створенню RESTful API для взаємодії із клієнтом. Також засвоїв як реалізувати та працювати з тришаровою архітектурою, як використовувати DI. Також я засвоїв роботу із автентифікацією та авторизацією

Змін.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.03.04	Арк.
						114