

## Лабораторна робота №4

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

### Завдання 1

Ознайомитися з кроками створення **ASP.NET Core Minimal API** | [Check out the](#).

Створити **ASP.NET Core Minimal API**:

- Створити **3–4 групи маршрутів** відповідно до Додатку А (наприклад: /todos, /users, /categories).
- Використати **in-memory “сховище”** (простий список List в коді) для зберігання даних.
- Реалізувати **CRUD** для кожної групи маршрутів:
  - Create** (POST) — створення нового об’єкта.
  - Read** (GET) — отримання списку об’єктів або одного об’єкта за ID.
  - Update** (PUT/PATCH) — оновлення існуючого об’єкта.
  - Delete** (DELETE) — видалення об’єкта за ID.
- Виконати **валідацію даних** під час створення нових об’єктів (наприклад, перевірка порожніх полів, мінімальної довжини рядка, формату email тощо).
- Повертає стандартні HTTP-відповіді:
  - 200 OK — успішний запит,
  - 201 Created — створення нового ресурсу,
  - 400 Bad Request — помилка валідації,
  - 404 Not Found — об’єкт не знайдено.
- Для кожної групи створити **окремий набір ендпойнтів** у коді.
- Винести ендпойнти в окремі файли** (наприклад: Endpoints/ToDoEndpoints.cs, Endpoints/UserEndpoints.cs) і підключити їх у Program.cs.
- Організувати структуру проєкту** за папками:  
/Models  
/Endpoints  
/Data (за потреби)  
Program.cs
- Додати Swagger (OpenAPI)** для зручного тестування API через браузер:

					ЛР.ОК24.ПІ231.13.04		
Змн.	Арк.	№ докум.	Підпис	Дата	<b>Створення RESTful API</b>		
Розроб.		Євчук М.В.					
Перевір.		Жереб Д.В.			Літ.		
						Арк.	Акрушів
						1	35
Н. Контр.					ХПК		
Затверд.							

## 10.Перевірити роботу API через **Swagger** або **Postman**.

### Завдання 2

Ознайомитися з кроками створення **ASP.NET Core Web API** | [Check out the.](#)

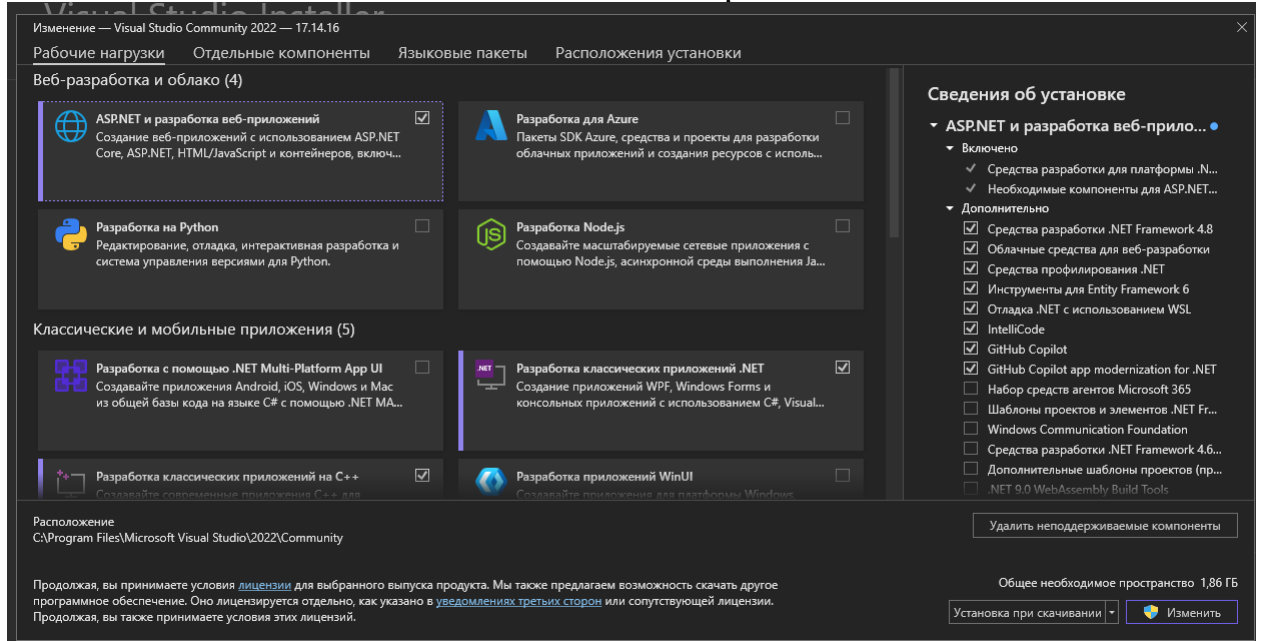
Створити **ASP.NET Core Web API**:

1. Створити **3–4 контролера** відповідно до Додатку А(наприклад: `TodoController`, `UserController`, `CategoryController`).
2. Створити відповідні моделі даних. Використати `Enum`, як поле класу. [Про Enum тут](#)
3. Використати **in-memory “сховище”** (простий список `List` в коді) для зберігання даних.
4. Реалізувати **CRUD** для кожної моделі в контролерах:
  - **Create** (POST) — створення нового об’єкта.
  - **Read** (GET) — отримання списку об’єктів або одного об’єкта за ID.
  - **Update** (PUT/PATCH) — оновлення існуючого об’єкта.
  - **Delete** (DELETE) — видалення об’єкта за ID.
5. Виконати **валідацію даних** під час створення нових об’єктів (наприклад, перевірка порожніх полів, мінімальної довжини рядка, формату email тощо). Для однієї моделі (самої меншої за к-тю полів) виконати валідацію через `DataAnnotations`. Для інших моделей виконати валідацію через `FluentValidation`. Для одного з полів використати [Regular Expressions](#)
6. Повертає стандартні HTTP-відповіді:
  - 200 OK — успішний запит,
  - 201 Created — створення нового ресурсу,
  - 400 Bad Request — помилка валідації,
  - 404 Not Found — об’єкт не знайдено.
7. **Організувати структуру проєкту** за папками:
  - /Models
  - /Controllers
  - /Data (за потреби)
  - /Validators
  - Program.cs
8. Перевірити роботу API через **Swagger** або **Postman**.
9. Підготувати **звіт**, який містить:
  - короткий опис проєкту;
  - код `Program.cs`, моделей та ендпойнтів;
  - скріншоти виконання запитів (GET, POST, помилки валідації);

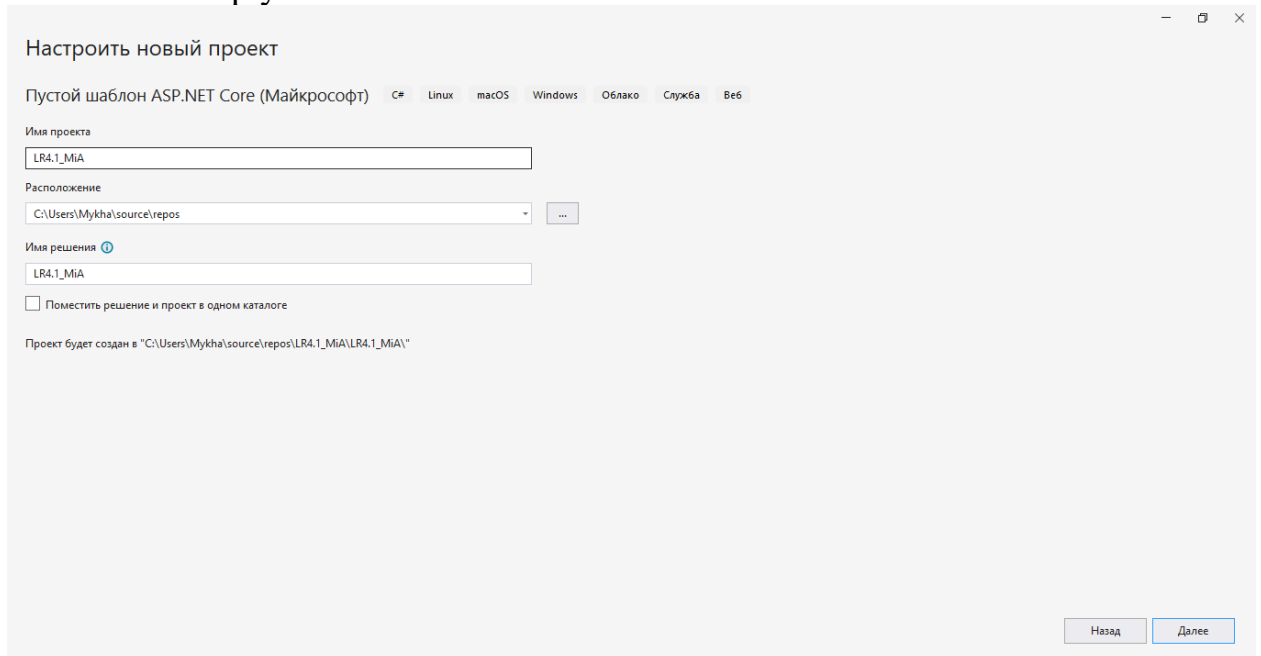
## Хід роботи

### 40.Онлайн-сервіс для обміну кулінарними рецептами з рейтингами

#### 1. Під час встановлення Visual Studio обираємо ASP.NET



#### 2. Перейдемо до створення проєкту. Для цього знаходимо проєкт ASP.NET Core empty.

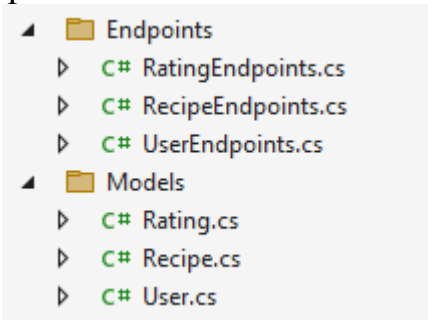


Перейдемо до написання програми.

Створимо папки Models, Endpoints. В папці Models створюємо 3 класа, а саме User.cs, Recipe.cs, Rating.cs. В папці Endpoints створюємо також 3 класа, а саме UserEndpoints.cs, RecipeEndpoints.cs, RatingEndpoints.cs.

					ЛР.ОК24.ПІ231.13.04	Арк.
						3
Змн.	Арк.	№ докум.	Підпис	Дата		

Ось як виглядає наша ієрархія.



Тепер перейдемо до написання коду.

### Models

#### User.cs

```
namespace LW4.Task1_MiA.Models
{
    public class User
    {
        public int Id { get; set; }
        public string Username { get; set; }
        public string Email { get; set; }
    }
}
```

#### Recipe.cs

```
namespace LW4.Task1_MiA.Models
{
    public class Recipe
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Instructions { get; set; }
        public List<Rating> Ratings { get; set; }
    }
}
```

#### Rating.cs

```
namespace LW4.Task1_MiA.Models
{
    public class Rating
    {
        public int Id { get; set; }
        public int RecipeId { get; set; }
        public int UserId { get; set; }
        public int Score { get; set; }
        public string Review { get; set; }
    }
}
```

### Endpoints

#### UserEndpoints.cs

```
using LW4.Task1_MiA.Models;
namespace LW4.Task1_MiA.Endpoints
{
    public static class UserEndpoints
    {
        public static void MapUserEndpoints(this WebApplication app)
        {
            var users = new List<User>
            {
                new User { Id = 1, Username = "john_doe", Email = "john@example.com" },
                new User { Id = 2, Username = "jane_smith", Email = "jane@example.com" },
            },
        }
    }
}
```

					ЛР.ОК24.ПІ231.13.04	Арк.
						4
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        new User { Id = 3, Username = "alice_wonder", Email =
"alice@example.com" }
    };
    app.MapGet("/users", () => Results.Ok(users));
    app.MapGet("/users/{id:int}", (int id) =>
    {
        var user = users.FirstOrDefault(u => u.Id == id);
        return user is not null ? Results.Ok(user) : Results.NotFound();
    });

    app.MapPost("/users", (User newUser) =>
    {
        newUser.Id = users.Count + 1;
        users.Add(newUser);
        return Results.Created($"/users/{newUser.Id}", newUser);
    });

    app.MapPut("/users/{id:int}", (int id, User updatedUser) =>
    {
        var user = users.FirstOrDefault(u => u.Id == id);
        if (user is null) return Results.NotFound();
        user.Username = updatedUser.Username;
        user.Email = updatedUser.Email;
        return Results.Ok(user);
    });

    app.MapDelete("/users/{id:int}", (int id) =>
    {
        var user = users.FirstOrDefault(u => u.Id == id);
        if (user is null) return Results.NotFound();
        users.Remove(user);
        return Results.NoContent();
    });
}
}
}

```

## RecipeEndpoints.cs

```

using LW4.Task1_MiA.Models;
namespace LW4.Task1_MiA.Endpoints
{
    public static class RecipeEndpoints
    {
        public static void MapRecipeEndpoints(this WebApplication app)
        {
            var recipes = new List<Recipe>
            {
                new Recipe { Id = 1, Name = "Pasta", Instructions = "Boil water, add
pasta" },
                new Recipe { Id = 2, Name = "Salad", Instructions = "Mix vegetables"
},
                new Recipe { Id = 3, Name = "Sandwich", Instructions = "Put
ingredients between bread slices" },
                new Recipe { Id = 4, Name = "Omelette", Instructions = "Beat eggs,
cook in a pan" },
                new Recipe { Id = 5, Name = "Soup", Instructions = "Boil broth, add
ingredients" },
                new Recipe { Id = 6, Name = "Steak", Instructions = "Season meat,
cook to desired doneness" },
                new Recipe { Id = 7, Name = "Pancakes", Instructions = "Mix batter,
cook on griddle" },
                new Recipe { Id = 8, Name = "Tacos", Instructions = "Fill tortillas
with ingredients" },
                new Recipe { Id = 9, Name = "Pizza", Instructions = "Prepare dough,
add toppings, bake" },
                new Recipe { Id = 10, Name = "Curry", Instructions = "Cook spices,
add meat/vegetables, simmer" },
            };
        }
    }
}

```

					ЛР.ОК24.ПІ231.13.04	Арк.
						5
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        new Recipe { Id = 11, Name = "Grilled Cheese", Instructions =
" Butter bread, add cheese, grill" },
        new Recipe { Id = 12, Name = "Fruit Smoothie", Instructions = "Blend
fruits with yogurt or milk" },
        new Recipe { Id = 13, Name = "Roast Chicken", Instructions = "Season
chicken, roast in oven" },
    };
    app.MapGet("/recipes", () => Results.Ok(recipes));
    app.MapGet("/recipes/{id:int}", (int id) =>
    {
        var recipe = recipes.FirstOrDefault(r => r.Id == id);
        return recipe is not null ? Results.Ok(recipe) : Results.NotFound();
    });
    app.MapPost("/recipes", (Recipe newRecipe) =>
    {
        // Перевірка: чи заповнене поле Name
        if (string.IsNullOrEmpty(newRecipe.Name))
            return Results.BadRequest("Recipe name cannot be empty.");
        // Додавання нового рецепту
        newRecipe.Id = recipes.Count + 1;
        recipes.Add(newRecipe);
        return Results.Created($" /recipes/{newRecipe.Id}", newRecipe);
    });
    app.MapPut("/recipes/{id:int}", (int id, Recipe updatedRecipe) =>
    {
        var recipe = recipes.FirstOrDefault(r => r.Id == id);
        if (recipe is null) return Results.NotFound();
        recipe.Name = updatedRecipe.Name;
        recipe.Instructions = updatedRecipe.Instructions;
        return Results.Ok(recipe);
    });
    app.MapDelete("/recipes/{id:int}", (int id) =>
    {
        var recipe = recipes.FirstOrDefault(r => r.Id == id);
        if (recipe is null) return Results.NotFound();
        recipes.Remove(recipe);
        return Results.NoContent();
    });
}
}
}
}

```

## RatingEndpoints.cs

```

using LW4.Task1_MiA.Models;
namespace LW4.Task1_MiA.Endpoints {
    public static class RatingEndpoints
    {
        public static void MapRatingEndpoints(this WebApplication app)
        {
            var ratings = new List<Rating>
            {
                new Rating { Id = 1, RecipeId = 1, UserId = 1, Score = 5, Review =
"Delicious!" },
                new Rating { Id = 2, RecipeId = 2, UserId = 2, Score = 4, Review =
"Tasty, but could use more dressing." },
                new Rating { Id = 3, RecipeId = 3, UserId = 3, Score = 3, Review =
"Average." },
                new Rating { Id = 4, RecipeId = 1, UserId = 2, Score = 4, Review =
"Pretty good!" },
                new Rating { Id = 5, RecipeId = 2, UserId = 3, Score = 5, Review =
"Loved it!" },
                new Rating { Id = 6, RecipeId = 3, UserId = 1, Score = 2, Review =
"Not my favorite." },
                new Rating { Id = 7, RecipeId = 1, UserId = 3, Score = 5, Review =
"Absolutely fantastic!" },
            };
        }
    }
}

```

					ЛР.ОК24.ПІ231.13.04	Арк.
						6
Змн.	Арк.	№ докум.	Підпис	Дата		

```

app.MapGet("/ratings", () => Results.Ok(ratings));
app.MapGet("/ratings/{id:int}", (int id) =>
{
    var rating = ratings.FirstOrDefault(r => r.Id == id);
    return rating is not null ? Results.Ok(rating) : Results.NotFound();
});
app.MapPost("/ratings", (Rating newRating) =>
{
    newRating.Id = ratings.Count + 1;
    ratings.Add(newRating);
    return Results.Created($"/ratings/{newRating.Id}", newRating);
});
app.MapPut("/ratings/{id:int}", (int id, Rating updatedRating) =>
{
    var rating = ratings.FirstOrDefault(r => r.Id == id);
    if (rating is null) return Results.NotFound();
    rating.Score = updatedRating.Score;
    rating.Review = updatedRating.Review;
    return Results.Ok(rating);
});
app.MapDelete("/ratings/{id:int}", (int id) =>
{
    var rating = ratings.FirstOrDefault(r => r.Id == id);
    if (rating is null) return Results.NotFound();
    ratings.Remove(rating);
    return Results.NoContent();
});
}
}
}

```

### Program.cs

```

using LW4.Task1_MiA.Endpoints;

var builder = WebApplication.CreateBuilder(args);

// Swagger
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();
var app = builder.Build();
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI(opt =>
    {
        opt.SwaggerEndpoint("/swagger/v1/swagger.json", "Recipes API v1");
        opt.RoutePrefix = "swagger";
    });
}
app.MapRecipeEndpoints();
app.MapUserEndpoints();
app.MapRatingEndpoints();

app.Run();

```

					ЛР.ОК24.ПІ231.13.04	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		7

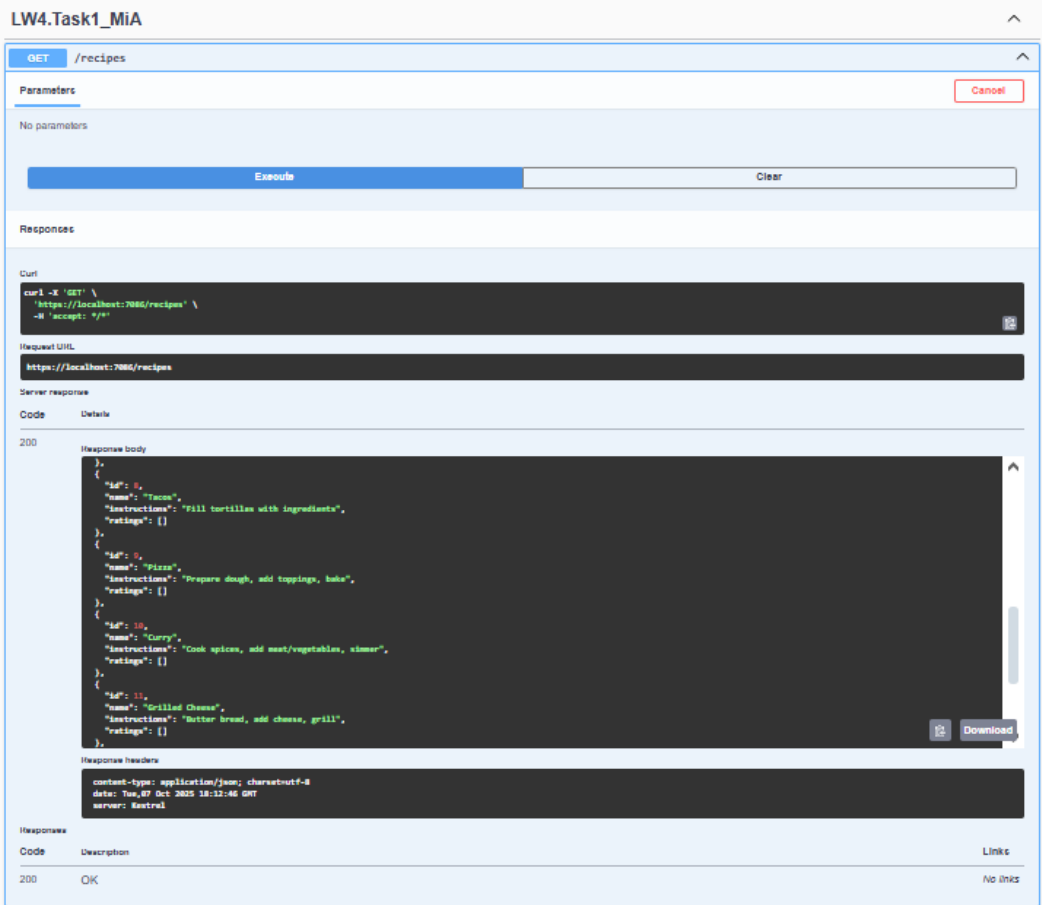
Результат роботи:

```
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: https://localhost:7086
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5128
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\Mykha\source\repos\LW4.Task1_MiA\LW4.Task1_MiA
```

Програма показує те, що нам доступна вся інформація по посиланню.

Запити

Запит на отримання рецептів





Запит на публікацію рецепта

POST /recipes

Рекордів: 0

Request body: application/json

API Notes | Schema

```
{
  "id": 1,
  "name": "Borscht",
  "instructions": "To prepare classic borscht, first boil the beets, then add chopped potatoes and cabbage, and finally add a dressing of sautéed beef, carrots, onions, and tomato paste. It is important to preserve the bright color of the beets by adding citric acid or lemon juice during cooking. Borscht is served with sour cream and herbs, letting it steep before serving.",
  "tags": [
    "ukr", "beet",
    "vegetarian": false
  ],
  "ingredients": [
    "beet",
    "potato",
    "cabbage",
    "beef",
    "carrot",
    "onion",
    "tomato paste",
    "citric acid",
    "sour cream",
    "herbs"
  ],
  "nutrition": {
    "calories": 120,
    "protein": 10,
    "carbs": 20,
    "fat": 5
  }
}
```

Скасувати

Очистити

Рекордів: 0

curl -X POST -H 'Content-Type: application/json' -d '{
 "id": 1,
 "name": "Borscht",
 "instructions": "To prepare classic borscht, first boil the beets, then add chopped potatoes and cabbage, and finally add a dressing of sautéed beef, carrots, onions, and tomato paste. It is important to preserve the bright color of the beets by adding citric acid or lemon juice during cooking. Borscht is served with sour cream and herbs, letting it steep before serving.",
 "tags": [
 "ukr", "beet",
 "vegetarian": false
 ],
 "ingredients": [
 "beet",
 "potato",
 "cabbage",
 "beef",
 "carrot",
 "onion",
 "tomato paste",
 "citric acid",
 "sour cream",
 "herbs"
 ],
 "nutrition": {
 "calories": 120,
 "protein": 10,
 "carbs": 20,
 "fat": 5
 }
}'

Request URL: http://localhost:3000/recipes/1

Request response: Code: 200, Status: OK

Request body: {
 "id": 1,
 "name": "Borscht",
 "instructions": "To prepare classic borscht, first boil the beets, then add chopped potatoes and cabbage, and finally add a dressing of sautéed beef, carrots, onions, and tomato paste. It is important to preserve the bright color of the beets by adding citric acid or lemon juice during cooking. Borscht is served with sour cream and herbs, letting it steep before serving.",
 "tags": [
 "ukr", "beet",
 "vegetarian": false
 ],
 "ingredients": [
 "beet",
 "potato",
 "cabbage",
 "beef",
 "carrot",
 "onion",
 "tomato paste",
 "citric acid",
 "sour cream",
 "herbs"
 ],
 "nutrition": {
 "calories": 120,
 "protein": 10,
 "carbs": 20,
 "fat": 5
 }
}

Request headers: Content-Type: application/json; charset=utf-8, Host: localhost:3000, User-Agent: curl/7.81.0, Accept: \*/\*

Response: Code: 200, Status: OK, Link: /#/this

Запит на отримання конкретного рецепта

GET /recipes/:id

Рекордів: 0

id: 1, type: integer(32), (path): 1

Скасувати

Очистити

Рекордів: 0

curl -X GET -H 'Accept: application/json' -d ''

Request URL: http://localhost:3000/recipes/1

Request response: Code: 200, Status: OK

Request body: {
 "id": 1,
 "name": "Borscht",
 "instructions": "To prepare classic borscht, first boil the beets, then add chopped potatoes and cabbage, and finally add a dressing of sautéed beef, carrots, onions, and tomato paste. It is important to preserve the bright color of the beets by adding citric acid or lemon juice during cooking. Borscht is served with sour cream and herbs, letting it steep before serving.",
 "tags": [
 "ukr", "beet",
 "vegetarian": false
 ],
 "ingredients": [
 "beet",
 "potato",
 "cabbage",
 "beef",
 "carrot",
 "onion",
 "tomato paste",
 "citric acid",
 "sour cream",
 "herbs"
 ],
 "nutrition": {
 "calories": 120,
 "protein": 10,
 "carbs": 20,
 "fat": 5
 }
}

Request headers: Accept: application/json; charset=utf-8, Host: localhost:3000, User-Agent: curl/7.81.0, Accept: \*/\*

Response: Code: 200, Status: OK, Link: /#/this

					ЛР.ОК24.ПІ231.13.04	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		9

## Запит на оновлення рецепта

The screenshot displays the AWS IAM console interface. At the top, the 'Groups' tab is selected, showing a list of IAM groups. The group 'arn:aws:iam::123456789012:group/MyGroup' is highlighted. Below the list, the 'Permissions' tab is visible, showing a list of permissions. The 'Groups' tab shows a table with columns for Name, ARN, and Status. The 'Permissions' tab shows a table with columns for Name, ARN, and Status. The 'Groups' tab shows a table with columns for Name, ARN, and Status. The 'Permissions' tab shows a table with columns for Name, ARN, and Status.

## Запит на видалення рецепта

**DELETE** /recipies/{id}

---

### Parameters

Name	Description
Id	integer (int64) (path)

[Cancel](#) [Clear](#)

---

### Responses

Code	Description
201	Created successfully.

```
curl -X DELETE "http://localhost:7000/recipies/1"
```

**Response:**

```
{
  "message": "Recipe deleted successfully."
}
```

Запит на отримання користувачів

GET /users

Cancel

No parameters

Cancel

Clear

Request

Call

```
curl -d '{"id": 1, "username": "john_doe", "email": "john.doe@example.com"}' https://localhost:3000/users
```

Response (JSON)

```
https://localhost:3000/users
```

Response headers

Code

Details

200

Response body

```
{  "id": 1,  "username": "john_doe",  "email": "john.doe@example.com"}  {  "id": 2,  "username": "jane_smith",  "email": "jane.smith@example.com"}  {  "id": 3,  "username": "mike_roberts",  "email": "mike.roberts@example.com"}  }
```

Download

Response headers

```
Content-Type: application/json; charset=utf-8
Date: Tue, 27 Oct 2025 10:14:11 GMT
Server: React
```

Response

Code

Details

Links

200

OK

No link

Запит на додавання користувача

POST /users

Cancel

Reset

No parameters

Request body

application/json

Full Value | Schema

```
{  "id": 1,  "username": "mike",  "email": "mike.doe@gmail.com"}  }
```

Cancel

Clear

Request

Call

```
curl -d '{"id": 1, "username": "john_doe", "email": "john.doe@example.com"}' https://localhost:3000/users
```

Response (JSON)

```
https://localhost:3000/users
```

Response headers

Code

Details

201

Response body

```
{  "id": 1,  "username": "mike",  "email": "mike.doe@gmail.com"}  }
```

Download

Response headers

```
Content-Type: application/json; charset=utf-8
Date: Tue, 27 Oct 2025 10:14:11 GMT
Server: React
```

Response

Code

Details

Links

200

OK

No link

Запит на отримання конкретного користувача

GET

/users/{id}

Cancel

Parameters

Name	Description
<b>id</b> *required	
integer(\$int32)	3
(path)	

ExecuteClear

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7060/users/3' \
  -H 'accept: */*'
```

Request URL

https://localhost:7060/users/3

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{   "id": 3,   "username": "alice_wonder",   "email": "alice@example.com" }</pre></div><div>Download</div></div> <div><div>Response headers</div><div><pre>content-type: application/json; charset=utf-8 date: Tue, 07 Oct 2025 18:16:42 GMT server: Kestrel</pre></div></div>

Response

Code	Description	Links
200	OK	No links

Запит на оновлення користувача

PUT

/users/{id}

Cancel

Reset

Parameters

Name	Description
<b>id</b> *required	
integer(\$int32)	2
(path)	

Request body

\*required

application/json

Full Value | Syntax

```
{
  "id": 12,
  "username": "bob",
  "email": "bob@example.com"
}
```

ExecuteClear

Responses

Curl

```
curl -X 'PUT' \
  'https://localhost:7060/users/2' \
  -H 'content-type: application/json' \
  -d '{
    "id": 12,
    "username": "bob",
    "email": "bob@example.com"
  }'
```

Request URL

https://localhost:7060/users/2

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{   "id": 12,   "username": "bob",   "email": "bob@example.com" }</pre></div><div>Download</div></div> <div><div>Response headers</div><div><pre>content-type: application/json; charset=utf-8 date: Tue, 07 Oct 2025 18:17:09 GMT server: Kestrel</pre></div></div>

Response

Code	Description	Links
200	OK	No links

Змн.	Арк.	№ докум.	Підпис	Дата

## Запит на видалення користувача

SELECT

/users/{id}

Cancel

Parameters

Name

Description

id \* required

Integer(1 to 10)

1

Clear

Clear

Responses

200

```

{
  "id": "12345678",
  "username": "john.doe@example.com",
  "email": "john.doe@example.com"
}

```

201

```

{
  "id": "12345678",
  "username": "john.doe@example.com",
  "email": "john.doe@example.com"
}

```

204

```

{
  "id": "12345678",
  "username": "john.doe@example.com",
  "email": "john.doe@example.com"
}

```

## Запит на отримання оцінок

[illegible]

Змн.	Арк.	№ докум.	Підпис	Дата

ЛР.ОК24.ПІ231.13.04

Арк.

13

Запит на додавання оцінки

POST /ratings

Cancel

Результат

No results

Request body

application/json

Raw Value

JSON

```
{
  "id": 1,
  "userId": 12,
  "score": 5,
  "comment": "Great!"
}
```

Скопіювати

Clear

Відповісти

Call

```
curl -X POST \
  https://localhost:7000/ratings \
  -H 'Content-Type: application/json' \
  -d '{
    "id": 1,
    "userId": 12,
    "score": 5,
    "comment": "Great!"
  }'
```

Response (201)

https://localhost:7000/ratings

Response headers

Code

Details

201

Created

Response body

```
{
  "id": 1,
  "userId": 12,
  "score": 5,
  "comment": "Great!"
}
```

Download

Response headers

```
Content-Type: application/json; charset=utf-8
Date: Tue, 27 Oct 2020 10:27:48 GMT
Server: /api/v1
Server: /api/v1
```

Response

Code

Details

200

OK

Links

No links

Запит на отримання конкретної оцінки

GET /ratings/{id}

Cancel

Результат

Header

Details

Id

1

Скопіювати

Clear

Відповісти

Call

```
curl -X GET \
  https://localhost:7000/ratings/1 \
  -H 'Content-Type: application/json'
```

Response (200)

https://localhost:7000/ratings/1

Response headers

Code

Details

200

OK

Response body

```
{
  "id": 1,
  "userId": 12,
  "score": 5,
  "comment": "Great!"
}
```

Download

Response headers

```
Content-Type: application/json; charset=utf-8
Date: Tue, 27 Oct 2020 10:27:48 GMT
Server: /api/v1
Server: /api/v1
```

Response

Code

Details

200

OK

Links

No links

					ЛР.ОК24.ПІ231.13.04	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		14

Запит на оновлення оцінки

PUT /ratings/{id}

Cancel

Save

Назва

Опис/Повн.

Id \*

Integer(32x102)

5

Request body \*

application/json

Full Value | Schema

```
{
  "id": 42,
  "reviewId": 88,
  "score": 12,
  "score": 4,
  "review": "test text"
}
```

Скасувати

Очистити

Вихідні дані

Code

Full Value

Request URL:

https://localhost:7084/ratings/5

Request headers:

Code

Full Value

200

Request body:

```
{
  "id": 42,
  "reviewId": 88,
  "score": 12,
  "score": 4,
  "review": "test text"
}
```

Download

Request headers:

Content-Type: application/json; charset=utf-8

Accept: \*/\*

Accept-Range: bytes 0-262

Server: Apache/2.4.18 (Ubuntu)

Response:

Code

Full Value

Links

200

OK

No data

Запит на видалення оцінки

DELETE /ratings/{id}

Cancel

Назва

Опис/Повн.

Id \*

Integer(32x102)

5

Скасувати

Очистити

Вихідні дані

Code

Full Value

Request URL:

https://localhost:7084/ratings/5

Request headers:

Code

Full Value

204

Content removed.

Accept: \*/\*

Accept-Range: bytes 0-262

Server: Apache/2.4.18 (Ubuntu)

Response:

Code

Full Value

Links

200

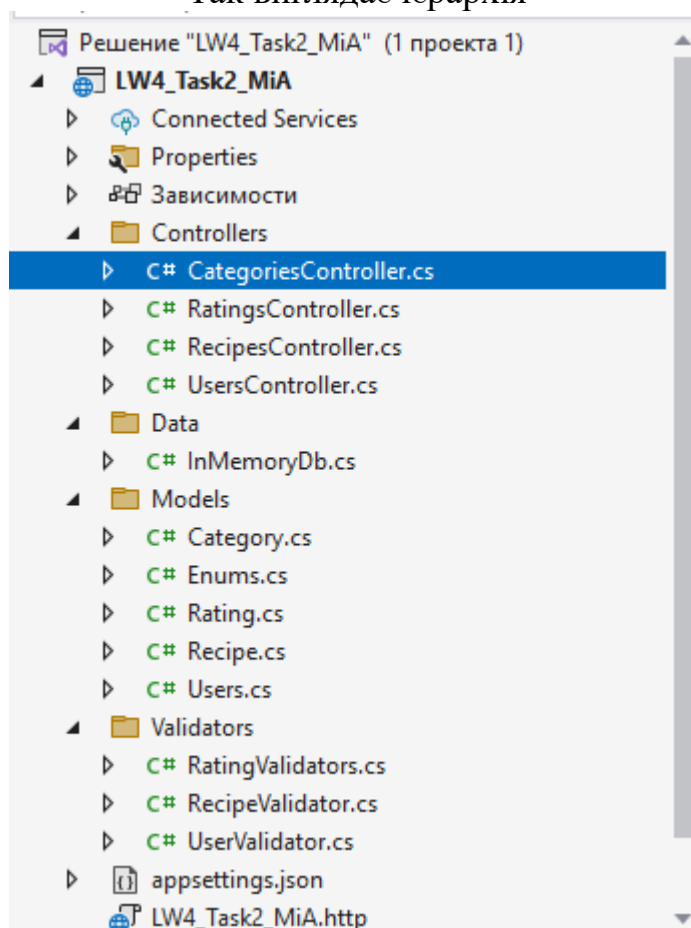
OK

No data

Завдання №2

Створимо проєкт з назвою LW4\_Task2\_MiA.

Так виглядає ієрархія



Тепер перейдемо до написання коду

## Controllers

### CategoriesControllers.cs:

```
using LW4_Task2_MiA.Data;
using LW4_Task2_MiA.Models;
using Microsoft.AspNetCore.Mvc;

namespace LW4_Task2_MiA.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class CategoriesController : ControllerBase
    {
        private readonly InMemoryDb _db;
        public CategoriesController(InMemoryDb db) => _db = db;

        [HttpGet]
        public ActionResult<IEnumerable<Category>> GetAll() => Ok(_db.Categories);

        [HttpGet("{id:int}")]
        public ActionResult<Category> GetById(int id)
        {
            var item = _db.Categories.FirstOrDefault(x => x.Id == id);
            return item is null ? NotFound() : Ok(item);
        }

        [HttpPost]
        public ActionResult<Category> Create([FromBody] Category model)
```

					ЛР.ОК24.ПІ231.13.04	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		16



```

    {
        if (!ModelState.IsValid) return BadRequest(ModelState);
        model.Id = _db.NextCategoryId();
        _db.Categories.Add(model);
        return CreatedAtAction(nameof(GetById), new { id = model.Id }, model);
    }

    [HttpPut("{id:int}")]
    public ActionResult<Category> Update(int id, [FromBody] Category model)
    {
        if (!ModelState.IsValid) return BadRequest(ModelState);
        var existing = _db.Categories.FirstOrDefault(x => x.Id == id);
        if (existing is null) return NotFound();
        existing.Name = model.Name;
        existing.Type = model.Type;
        return Ok(existing);
    }

    [HttpDelete("{id:int}")]
    public IActionResult Delete(int id)
    {
        var existing = _db.Categories.FirstOrDefault(x => x.Id == id);
        if (existing is null) return NotFound();
        _db.Categories.Remove(existing);
        return Ok();
    }
}

}

RatingsControllers.cs:
using LW4_Task2_MiA.Data;
using LW4_Task2_MiA.Models;
using Microsoft.AspNetCore.Mvc;

namespace LW4_Task2_MiA.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class RatingsController : ControllerBase
    {
        private readonly InMemoryDb _db;
        public RatingsController(InMemoryDb db) => _db = db;

        [HttpGet]
        public ActionResult<IEnumerable<Rating>> GetAll() => Ok(_db.Ratings);

        [HttpGet("{id:int}")]
        public ActionResult<Rating> GetById(int id)
        {
            var r = _db.Ratings.FirstOrDefault(x => x.Id == id);
            return r is null ? NotFound() : Ok(r);
        }

        [HttpPost("/api/recipes/{recipeId:int}/ratings")]
        public ActionResult<Rating> CreateForRecipe(int recipeId, [FromBody] Rating
model)
        {
            if (!ModelState.IsValid) return BadRequest(ModelState);

            var recipe = _db.Recipes.FirstOrDefault(r => r.Id == recipeId);
            if (recipe is null) return NotFound("Recipe не знайдено.");

            // Перевіряємо існування користувача
            var user = _db.Users.FirstOrDefault(u => u.Id == model.UserId);
            if (user is null) return BadRequest("UserId не існує.");
        }
    }
}

```

```

        // Форсуємо правильні зв'язки
        var rating = new Rating
        {
            Id = _db.NextRatingId(),
            RecipeId = recipeId,           // ігноруємо model.RecipeId
            UserId = user.Id,
            Value = model.Value,
            Comment = model.Comment
        };

        _db.Ratings.Add(rating);
        return CreatedAtAction(nameof(GetById), new { id = rating.Id }, rating);
    }

    [HttpPut("{id:int}")]
    public ActionResult<Rating> Update(int id, [FromBody] Rating model)
    {
        if (!ModelState.IsValid) return BadRequest(ModelState);
        if (model.Id != 0 && model.Id != id)
            return BadRequest("Змінювати Id рейтингу заборонено.");

        var r = _db.Ratings.FirstOrDefault(x => x.Id == id);
        if (r is null) return NotFound();

        if (_db.Recipes.All(rc => rc.Id != model.RecipeId))
            return BadRequest("RecipeId не існує.");
        if (_db.Users.All(u => u.Id != model.UserId))
            return BadRequest("UserId не існує.");

        r.RecipeId = model.RecipeId;
        r.UserId = model.UserId;
        r.Value = model.Value;
        r.Comment = model.Comment;
        return Ok(r);
    }

    [HttpDelete("{id:int}")]
    public IActionResult Delete(int id)
    {
        var r = _db.Ratings.FirstOrDefault(x => x.Id == id);
        if (r is null) return NotFound();
        _db.Ratings.Remove(r);
        return Ok();
    }
}

```

### RecipesControllers.cs:

```

using LW4_Task2_MiA.Data;
using LW4_Task2_MiA.Models;
using Microsoft.AspNetCore.Mvc;

namespace LW4_Task2_MiA.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class RecipesController : ControllerBase
    {
        private readonly InMemoryDb _db;
        public RecipesController(InMemoryDb db) => _db = db;

        [HttpGet]
        public ActionResult<IEnumerable<Recipe>> GetAll() => Ok(_db.Recipes);

        [HttpGet("{id:int}")]
    }
}

```

					ЛР.ОК24.ПІ231.13.04	Арк.
						18
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public ActionResult<Recipe> GetById(int id)
{
    var r = _db.Recipes.FirstOrDefault(x => x.Id == id);
    return r is null ? NotFound() : Ok(r);
}

[HttpPost]
public ActionResult<Recipe> Create([FromBody] Recipe model)
{
    if (!ModelState.IsValid) return BadRequest(ModelState);
    if (_db.Categories.All(c => c.Id != model.CategoryId)) return
BadRequest("CategoryId не існує.");
    if (_db.Users.All(u => u.Id != model.AuthorUserId)) return
BadRequest("AuthorUserId не існує.");

    model.Id = _db.NextRecipeId();
    _db.Recipes.Add(model);
    return CreatedAtAction(nameof(GetById), new { id = model.Id }, model);
}

[HttpPut("{id:int}")]
public ActionResult<Recipe> Update(int id, [FromBody] Recipe model)
{
    if (!ModelState.IsValid) return BadRequest(ModelState);
    var r = _db.Recipes.FirstOrDefault(x => x.Id == id);
    if (r is null) return NotFound();

    if (_db.Categories.All(c => c.Id != model.CategoryId)) return
BadRequest("CategoryId не існує.");
    if (_db.Users.All(u => u.Id != model.AuthorUserId)) return
BadRequest("AuthorUserId не існує.");

    r.Title = model.Title;
    r.Slug = model.Slug;
    r.Description = model.Description;
    r.Difficulty = model.Difficulty;
    r.CategoryId = model.CategoryId;
    r.AuthorUserId = model.AuthorUserId;
    return Ok(r);
}

[HttpDelete("{id:int}")]
public IActionResult Delete(int id)
{
    var r = _db.Recipes.FirstOrDefault(x => x.Id == id);
    if (r is null) return NotFound();
    _db.Recipes.Remove(r);
    _db.Ratings.RemoveAll(rt => rt.RecipeId == id); // каскадне очищення
рейтингів
    return Ok();
}
}

```

### UsersControllers.cs:

```

using LW4_Task2_MiA.Data;
using LW4_Task2_MiA.Models;
using Microsoft.AspNetCore.Mvc;

namespace LW4_Task2_MiA.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class UsersController : ControllerBase
    {
        private readonly InMemoryDb _db;
    }
}

```

					ЛР.ОК24.ПІ231.13.04	Арк.
						19
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public UsersController(InMemoryDb db) => _db = db;

[HttpGet]
public ActionResult<IEnumerable<User>> GetAll() => Ok(_db.Users);

[HttpGet("{id:int}")]
public ActionResult<User> GetById(int id)
{
    var u = _db.Users.FirstOrDefault(x => x.Id == id);
    return u is null ? NotFound() : Ok(u);
}

[HttpPost]
public ActionResult<User> Create([FromBody] User model)
{
    if (!ModelState.IsValid) return BadRequest(ModelState);
    model.Id = _db.NextUserId();
    _db.Users.Add(model);
    return CreatedAtAction(nameof(GetById), new { id = model.Id }, model);
}

[HttpPut("{id:int}")]
public ActionResult<User> Update(int id, [FromBody] User model)
{
    if (!ModelState.IsValid) return BadRequest(ModelState);
    var u = _db.Users.FirstOrDefault(x => x.Id == id);
    if (u is null) return NotFound();
    u.DisplayName = model.DisplayName;
    u.Email = model.Email;
    u.Role = model.Role;
    return Ok(u);
}

[HttpDelete("{id:int}")]
public IActionResult Delete(int id)
{
    var u = _db.Users.FirstOrDefault(x => x.Id == id);
    if (u is null) return NotFound();
    _db.Users.Remove(u);
    return Ok();
}
}
}

```

## Data

### InMemoryDb.cs:

```

using LW4_Task2_MiA.Models;

namespace LW4_Task2_MiA.Data
{
    public class InMemoryDb
    {
        public List<Category> Categories { get; } = new();
        public List<User> Users { get; } = new();
        public List<Recipe> Recipes { get; } = new();
        public List<Rating> Ratings { get; } = new();

        private int _catId = 1, _userId = 1, _recipeId = 1, _ratingId = 1;

        public InMemoryDb()
        {
            // Початкові дані
            Categories.AddRange(new[]
            {
                new Category { Id = _catId++, Name="Десерти",
                    Type=CategoryType.Dessert },
            }

```

Змн.	Арк.	№ докум.	Підпис	Дата

ЛР.ОК24.ПІ231.13.04

Арк.

20

```

        new Category{ Id = _catId++, Name="Сніданки",
Type=CategoryType.Breakfast },
        new Category{ Id = _catId++, Name="Обіди", Type=CategoryType.Lunch
},
        new Category{ Id = _catId++, Name="Вечері", Type=CategoryType.Dinner
},
        new Category{ Id = _catId++, Name="Інше", Type=CategoryType.Unknown
},
    });
    Users.Add(new User { Id = _userId++, DisplayName = "Alice", Email =
"alice@example.com", Role = UserRole.Regular });
    Users.Add(new User { Id = _userId++, DisplayName = "Bob", Email =
"bob@gmail.com", Role = UserRole.Admin });
    Users.Add(new User { Id = _userId++, DisplayName = "Charlie", Email =
"charlie@yahoo.com", Role = UserRole.Regular });
    Users.Add(new User { Id = _userId++, DisplayName = "Diana", Email =
"diana@duckduckgo.com", Role = UserRole.Regular });
    Recipes.Add(new Recipe
    {
        Id = _recipeId++,
        Title = "Яблучний пиріг",
        Slug = "yabluchnyi-pyrig",
        Description = "Класичний рецепт.",
        Difficulty = RecipeDifficulty.Medium,
        CategoryId = 1,
        AuthorUserId = 1
    });
    Recipes.Add(new Recipe
    {
        Id = _recipeId++,
        Title = "Омлет",
        Slug = "omlet",
        Description = "Швидкий сніданок.",
        Difficulty = RecipeDifficulty.Easy,
        CategoryId = 2,
        AuthorUserId = 1
    });
    Recipes.Add(new Recipe
    {
        Id = _recipeId++,
        Title = "Млинці",
        Slug = "mlynci",
        Description = "Тонкі млинці з начинкою.",
        Difficulty = RecipeDifficulty.Easy,
        CategoryId = 2,
        AuthorUserId = 1
    });
    Recipes.Add(new Recipe
    {
        Id = _recipeId++,
        Title = "Тірамісу",
        Slug = "tiramisu",
        Description = "Італійський десерт.",
        Difficulty = RecipeDifficulty.Hard,
        CategoryId = 1,
        AuthorUserId = 1
    });
    Ratings.Add(new Rating { Id = _ratingId++, RecipeId = 1, UserId = 1,
Value = 5, Comment = "Смачно!" });
    Ratings.Add(new Rating { Id = _ratingId++, RecipeId = 1, UserId = 2,
Value = 4, Comment = "Добре, але можна краще." });
    Ratings.Add(new Rating { Id = _ratingId++, RecipeId = 2, UserId = 3,
Value = 5, Comment = "Просто і смачно." });
    Ratings.Add(new Rating { Id = _ratingId++, RecipeId = 3, UserId = 4,
Value = 3, Comment = "Млинці вийшли трохи товсті." });

```

					ЛР.ОК24.ПІ231.13.04	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		21

```

        Ratings.Add(new Rating { RecipeId = _ratingId++, UserId = 2, Value = 5,
Comment = "Обожнюю тіпаміцу!" });
        Ratings.Add(new Rating { Id = _ratingId++, RecipeId = 4, UserId = 3,
Value = 4, Comment = "Дуже смачно, але складно готувати." });
        Ratings.Add(new Rating { Id = _ratingId++, RecipeId = 2, UserId = 4,
Value = 4, Comment = "Хороший рецепт для швидкого сніданку." });
    }

    public int NextCategoryId() => _catId++;
    public int NextUserId() => _userId++;
    public int NextRecipeId() => _recipeId++;
    public int NextRatingId() => _ratingId++;
}
}

```

## Models

### Category.cs:

```

using System.ComponentModel.DataAnnotations;

namespace LW4_Task2_MiA.Models
{
    public class Category
    {
        public int Id { get; set; }

        [Required, StringLength(40, MinimumLength = 2)]
        public string Name { get; set; } = string.Empty;

        [Required]
        public CategoryType Type { get; set; } = CategoryType.Unknown;
    }
}

```

### Enums.cs:

```

namespace LW4_Task2_MiA.Models
{
    public enum RecipeDifficulty { Easy = 1, Medium = 2, Hard = 3 }
    public enum UserRole { Regular = 1, Moderator = 2, Admin = 3 }
    public enum CategoryType { Unknown = 0, Breakfast = 1, Lunch = 2, Dinner =
3, Dessert = 4 }
}

```

### Rating.cs:

```

namespace LW4_Task2_MiA.Models
{
    public class Rating
    {
        public int Id { get; set; }
        public int RecipeId { get; set; }
        public int UserId { get; set; }
        public int Value { get; set; }
        public string? Comment { get; set; }
    }
}

```

### Recipe.cs:

```

namespace LW4_Task2_MiA.Models
{
    public class Recipe
    {
        public int Id { get; set; }
        public string Title { get; set; } = string.Empty;
        public string Slug { get; set; } = string.Empty;
        public string Description { get; set; } = string.Empty;
        public RecipeDifficulty Difficulty { get; set; } = RecipeDifficulty.Easy;
        public int CategoryId { get; set; }
        public int AuthorUserId { get; set; }
    }
}

```

					ЛР.ОК24.ПІ231.13.04	Арк.
						22
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }
}

Users.cs:
namespace LW4_Task2_MiA.Models
{
    public class User
    {
        public int Id { get; set; }
        public string DisplayName { get; set; } = string.Empty;
        public string Email { get; set; } = string.Empty;
        public UserRole Role { get; set; } = UserRole.Regular;
    }
}

```

## Validators

### RatingValidator.cs:

```

using LW4_Task2_MiA.Models;
using FluentValidation;
namespace LW4_Task2_MiA.Validators
{
    public class RatingValidator : AbstractValidator<Rating>
    {
        public RatingValidator()
        {
            RuleFor(x => x.RecipeId).GreaterThan(0);
            RuleFor(x => x.UserId).GreaterThan(0);
            RuleFor(x => x.Value).InclusiveBetween(1, 5);
            RuleFor(x => x.Comment).MaximumLength(500);
        }
    }
}

```

### RecipeValidator.cs:

```

using LW4_Task2_MiA.Models;
using FluentValidation;
using System.Text.RegularExpressions;
namespace LW4_Task2_MiA.Validators
{
    public class RecipeValidator : AbstractValidator<Recipe>
    {
        public RecipeValidator()
        {
            RuleFor(x => x.Title).NotEmpty().MinimumLength(3).MaximumLength(80);

            RuleFor(x => x.Slug).NotEmpty().Must(s => Regex.IsMatch(s, "[a-z0-9-]+$")).WithMessage("Slug має містити лише малі латиницю, цифри та тире.");

            RuleFor(x =>
x.Description).NotEmpty().MinimumLength(10).MaximumLength(1000);

            RuleFor(x => x.Difficulty).IsInEnum();

            RuleFor(x => x.CategoryId).GreaterThan(0);
            RuleFor(x => x.AuthorUserId).GreaterThan(0);
        }
    }
}

```

### UserValidator.cs:

```

using LW4_Task2_MiA.Models;
using FluentValidation;
using System.Text.RegularExpressions;
namespace LW4_Task2_MiA.Validators
{
    public class UserValidator : AbstractValidator<User>
    {

```

					ЛР.ОК24.ПІ231.13.04	Арк.
						23
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        private const string EmailRegex =
            @"^[A-Za-z0-9._%+\-]+\@[A-Za-z0-9.\-]+\.[A-Za-z]{2,}$";

        public UserValidator()
        {
            RuleFor(x =>
                x.DisplayName).NotEmpty().MinimumLength(2).MaximumLength(50);

            RuleFor(x => x.Email).NotEmpty().Must(v => Regex.IsMatch(v,
                EmailRegex)).WithMessage("Невірний формат email.");

            RuleFor(x => x.Role).IsInEnum();
        }
    }
}

Program.cs:
using FluentValidation;
using FluentValidation.AspNetCore;
using LW4_Task2_MiA.Data;

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddSingleton<InMemoryDb>();
builder.Services.AddControllers();
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

// FluentValidation
builder.Services.AddFluentValidationAutoValidation();
builder.Services.AddValidatorsFromAssemblyContaining<LW4_Task2_MiA.Validators.Recipe
Validator>();

var app = builder.Build();

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

app.UseHttpsRedirection();
app.MapControllers();

app.Run();

```

## Результат програми

```

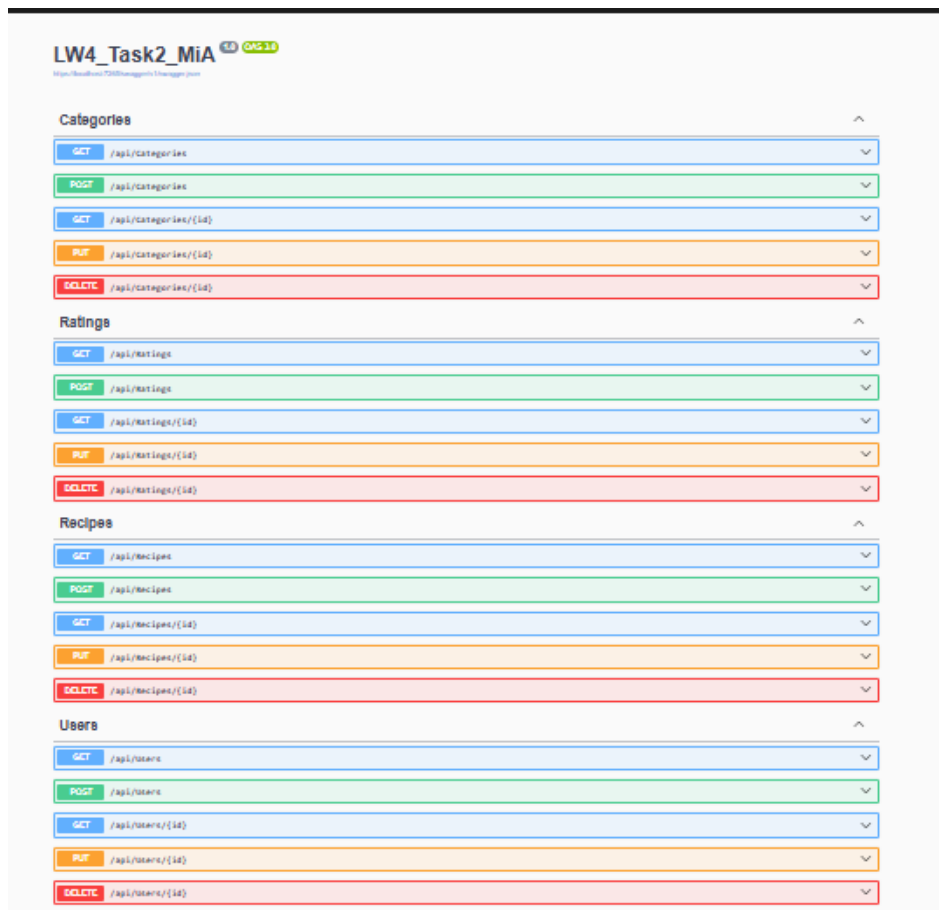
C:\Users\Mykha\source\repos\ x + v
[info]: Microsoft.Hosting.Lifetime[14]
Now listening on: https://localhost:7265
[info]: Microsoft.Hosting.Lifetime[14]
Now listening on: http://localhost:5141
[info]: Microsoft.Hosting.Lifetime[0]
Application started. Press Ctrl+C to shut down.
[info]: Microsoft.Hosting.Lifetime[0]
Hosting environment: Development
[info]: Microsoft.Hosting.Lifetime[0]
Content root path: C:\Users\Mykha\source\repos\LW4_Task2_MiA\LW4_Task2_MiA
|

```

Програма працює. Переходимо до Swagger.

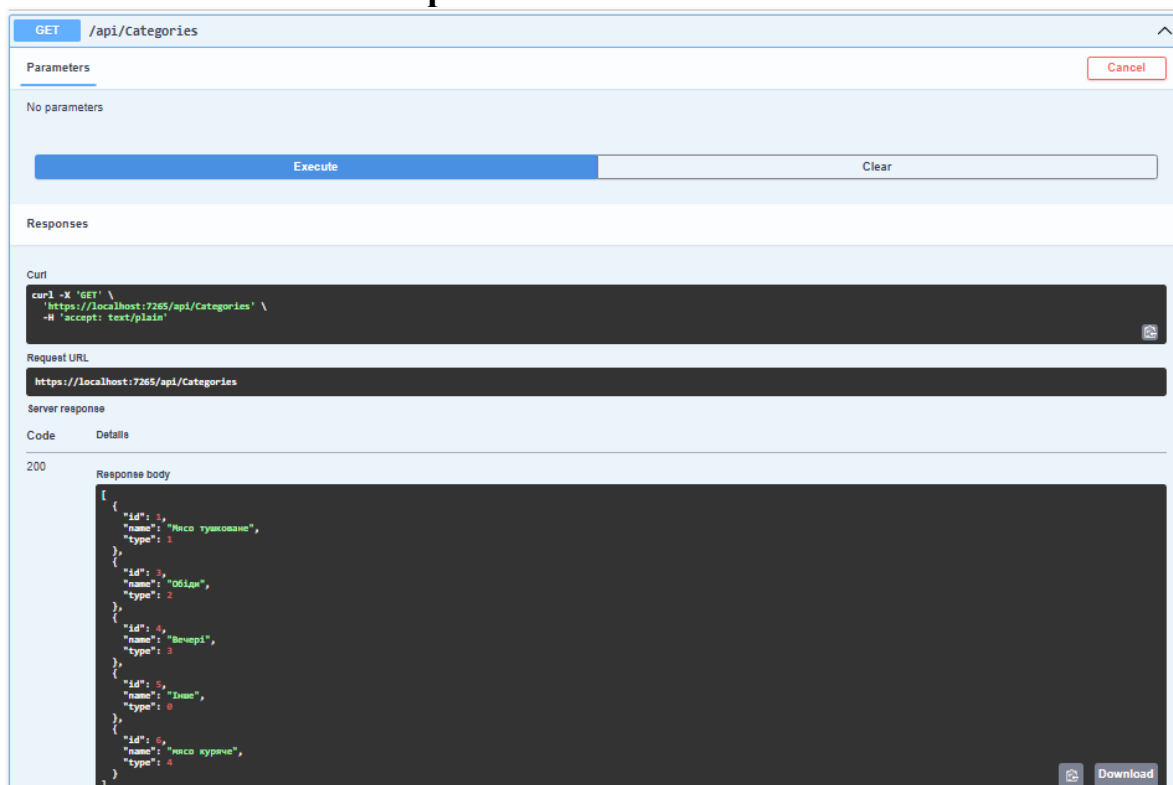
					ЛР.ОК24.ПІ231.13.04	Арк.
						24
Змн.	Арк.	№ докум.	Підпис	Дата		





В програмі є 4 контролера: Categories, Ratings, Recipes, Users.  
Для кожного з них є CRUD (C - Create, R - Read, U - Update, D – Delete).  
**Почнемо з Categories**

**Виконаємо запит на категорії.**



Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 13 Oct 2025 18:23:41 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	OK	No links

Media type  
text/plain

Controls Accept header.

Example Value | Schema

```
{
  "id": 0,
  "name": "string",
  "type": 0
}
```

Виконаємо запит на додавання категорії.

POST /api/Categories

Parameters

No parameters

Request body

application/json

```
{
  "id": 18,
  "name": "м'ясо куряче",
  "type": 4
}
```

Execute

Clear

Responses

Responses

Curl

```
curl -X 'POST' \
'https://localhost:7265/api/Categories' \
-H 'accept: text/plain' \
-H 'Content-Type: application/json' \
-d '{
  "id": 18,
  "name": "м'ясо куряче",
  "type": 4
}'
```

Request URL

https://localhost:7265/api/Categories

Server response

Code	Details
201	<div>Response body</div> <div><pre>{   "id": 0,   "name": "м'ясо куряче",   "type": 4 }</pre></div> <div>Response headers</div> <div><pre>content-type: application/json; charset=utf-8 date: Mon, 13 Oct 2025 18:19:11 GMT location: https://localhost:7265/api/Categories/6 server: Kestrel</pre></div>

Responses

Code	Description	Links
200	OK	No links

Media type  
text/plain

Controls Accept header.

Example Value | Schema

```
{
  "id": 0,
  "name": "string",
  "type": 0
}
```

Виконаємо запит на отримання конкретної категорії.

GET

/api/Categories/{id}

Parameters

Cancel

Name	Description
id * required	
integer(\$int32)	3
(path)	

Execute

Clear

Responses

Curl

curl -X 'GET' \

'https://localhost:7265/api/Categories/3' \

-H 'accept: text/plain'

Request URL

https://localhost:7265/api/Categories/3

Server response

Code	Details
200	<div><div>Response body</div><div>{</div><div>"id": 3,</div><div>"name": "Овощи",</div><div>"type": 2</div><div>}</div><div>Download</div></div> <div><div>Response headers</div><div>content-type: application/json; charset=utf-8</div><div>date: Mon, 13 Oct 2025 18:18:29 GMT</div><div>server: Kestrel</div></div>

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

{

"id": 0,

"name": "string",

"type": 0

}

Виконаємо запит на оновлення конкретної категорії.

PUT

/api/Categories/{id}

Parameters

Cancel

Reset

Name	Description
id * required	
integer(\$int32)	1
(path)	

Request body

application/json

{

"id": 1,

"name": "М'ясо тушковане",

"type": 1

}

Execute

Clear

Responses

					ЛР.ОК24.ПІ231.13.04	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		27

Responses

Curl

```
curl -X 'PUT' \
  'https://localhost:7265/api/Categories/1' \
  -H 'accept: text/plain' \
  -H 'Content-Type: application/json' \
  -d '{
    "id": 1,
    "name": "Рісco тyкoмaнa",
    "type": 1
  }'
```

Request URL

https://localhost:7265/api/Categories/1

Server response

CodeDetails

200

Response body

```
{
  "id": 1,
  "name": "Рісco тyкoмaнa",
  "type": 1
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 13 Oct 2025 18:15:00 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{
  "id": 0,
  "name": "string",
  "type": 0
}
```

Виконаємо запит на видалення конкретної категорії.

DELETE /api/Categories/{id}

Parameters

Name	Description
id <span>required</span>	integer(\$int32) (path)

Execute

Clear

Responses

Curl

```
curl -X 'DELETE' \
  'https://localhost:7265/api/Categories/2' \
  -H 'accept: */*'
```

Request URL

https://localhost:7265/api/Categories/2

Server response

CodeDetails

200

Response headers

```
content-length: 0
date: Mon, 13 Oct 2025 18:13:20 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	OK	No links

Змн.	Арк.	№ докум.	Підпис	Дата

ЛР.ОК24.ПІ231.13.04

Арк.

28

# Ratings

Виконаємо запит на отримання рейтингів.

GET /api/Ratings

Parameters

Cancel

No parameters

Execute

Clear

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7265/api/Ratings' \
  -H 'accept: text/plain'
```

Request URL

https://localhost:7265/api/Ratings

Server response

Code

Details

200

Response body

```
{
  "id": 1,
  "recipeId": 1,
  "userId": 1,
  "value": 5,
  "comment": "Смачно!"
},
{
  "id": 2,
  "recipeId": 1,
  "userId": 2,
  "value": 4,
  "comment": "Добре, але можна краще."
},
{
  "id": 3,
  "recipeId": 2,
  "userId": 3,
  "value": 5,
  "comment": "Просто і смачно."
},
{
  "id": 4,
  "recipeId": 3,
  "userId": 4,
  "value": 3,
  "comment": "Немає смаку, дуже гіркий"
}
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 13 Oct 2025 17:31:18 GMT
server: Kestrel
```

Responses

Code

Description

Links

200

OK

No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{
  "id": 0,
  "recipeId": 0,
  "userId": 0,
  "value": 0,
  "comment": "string"
}
```

Змн.	Арк.	№ докум.	Підпис	Дата

ЛР.ОК24.ПІ231.13.04

Арк.29

Виконаємо запит на отримання конкретного рейтингу.

GET

/api/Ratings/{id}

Parameters

Cancel

Name	Description
id * required	
integer(\$int32)	2
(path)	

ExecuteClear

Responses

Curl

curl -X 'GET' \n'https://localhost:7265/api/Ratings/2' \n-H 'accept: text/plain'

Request URL

https://localhost:7265/api/Ratings/2

Server response

Code	Details
200	<div><div>Response body</div><div>{\n  "id": 2,\n  "recipeId": 1,\n  "userId": 2,\n  "value": 4,\n  "comment": "Добре, але можна краще."\n}</div><div>Download</div></div> <div><div>Response headers</div><div>content-type: application/json; charset=utf-8\ndate: Mon, 13 Oct 2025 17:32:51 GMT\nserver: Kestrel</div></div>

Responses

Responses

Links

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

{\n "id": 0,\n "recipeId": 0,\n "userId": 0,\n "value": 0,\n "comment": "string"\n}

Виконаємо запит на оновлення конкретного рейтингу.

PUT

/api/Ratings/{id}

Parameters

Cancel

Reset

Name	Description
id * required	
integer(\$int32)	3
(path)	

Request body

application/json

{\n "id": 3,\n "recipeId": 3,\n "userId": 4,\n "value": 4,\n "comment": "it's amazing!"\n}

ExecuteClear

Responses

Curl

```
curl -X 'PUT' \
  'https://localhost:7265/api/Ratings/3' \
  -H 'accept: text/plain' \
  -H 'Content-Type: application/json' \
  -d '{
    "id": 3,
    "recipeId": 3,
    "userId": 4,
    "value": 4,
    "comment": "it's amazing!"
  }'
```

Request URL

https://localhost:7265/api/Ratings/3

Server response

Code

Details

200

Response body

```
{
  "id": 3,
  "recipeId": 3,
  "userId": 4,
  "value": 4,
  "comment": "it's amazing!"
}
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 13 Oct 2025 17:48:04 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{
  "id": 0,
  "recipeId": 0,
  "userId": 0,
  "value": 0,
  "comment": "string"
}
```

Виконаємо запит на видалення конкретного рейтингу.

DELETE /api/Ratings/{id}

Cancel

Parameters

Name	Description
id * required	
integer(\$int32)	
(path)	

1

Execute

Clear

Responses

Curl

```
curl -X 'DELETE' \
  'https://localhost:7265/api/Ratings/1' \
  -H 'accept: */*'
```

Request URL

https://localhost:7265/api/Ratings/1

Server response

Code

Details

200

Response headers

```
content-length: 0
date: Mon, 13 Oct 2025 18:02:31 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	OK	No links

Виконаємо запит на додавання рейтингу до конкретного рецепту.

POST

/api/recipes/{recipeId}/ratings

Parameters

Cancel

Reset

Name	Description
recipeId <small>* required</small>	
<small>integer(\$int32)</small>	4
<small>(path)</small>	

Request body

application/json

```
{
  "id": 13,
  "recipeId": 3,
  "userId": 2,
  "value": 2,
  "comment": "shitt!!!"
}
```

Execute

Clear

Responses

Curl

```
curl -X 'POST' \
  'https://localhost:7265/api/recipes/4/ratings' \
  -H 'accept: text/plain' \
  -H 'Content-Type: application/json' \
  -d '{
    "id": 13,
    "recipeId": 3,
    "userId": 2,
    "value": 2,
    "comment": "shitt!!!"
  }'
```

Request URL

https://localhost:7265/api/recipes/4/ratings

Server response

Code	Details
201	<div><div>Response body</div><div><pre>{   "id": 9,   "recipeId": 4,   "userId": 2,   "value": 2,   "comment": "shitt!!!" }</pre></div><div><div>Download</div></div></div> <div><div>Response headers</div><div><pre>content-type: application/json; charset=utf-8 date: Mon, 13 Oct 2025 18:03:17 GMT location: https://localhost:7265/api/Ratings/9 server: Kestrel</pre></div></div>

Responses

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header

Example Value | Schema

```
{
  "id": 0,
  "recipeId": 0,
  "userId": 0,
  "value": 0,
  "comment": "string"
}
```



# Recipes

Виконаємо запит на отримання усіх рецептів.

GET

/api/Recipes

Cancel

Parameters

No parameters

ExecuteClear

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7265/api/Recipes' \
  -H 'accept: text/plain'
```

Request URL

https://localhost:7265/api/Recipes

Server response

CodeDetails

200

Response body

```
{
  "id": 1,
  "title": "Різдвяний мпир",
  "slug": "yabluchnyi-pyrig",
  "description": "Класичний рецепт.",
  "difficulty": 2,
  "categoryId": 1,
  "authorUserId": 1
},
{
  "id": 2,
  "title": "Омлет",
  "slug": "omlet",
  "description": "Швидкий снідачок.",
  "difficulty": 1,
  "categoryId": 2,
  "authorUserId": 1
},
{
  "id": 3,
  "title": "Млинці",
  "slug": "mlynci",
  "description": "Тонкі млинці з начинкою.",
  "difficulty": 2,
  "categoryId": 3,
  "authorUserId": 1
}
```

Download

Виконаємо запит на отримання конкретного рецепту.

GET

/api/Recipes/{id}

Cancel

Parameters

Name	Description
id * required	
integer(\$int32)	3
(path)	

ExecuteClear

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7265/api/Recipes/3' \
  -H 'accept: text/plain'
```

Request URL

https://localhost:7265/api/Recipes/3

Server response

CodeDetails

200

Response body

```
{
  "id": 3,
  "title": "Млинці",
  "slug": "mlynci",
  "description": "Тонкі млинці з начинкою.",
  "difficulty": 2,
  "categoryId": 3,
  "authorUserId": 1
}
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 13 Oct 2025 18:05:02 GMT
server: Kestrel
```

Responses

Code	Description
------	-------------

Links

Змн.	Арк.	№ докум.	Підпис	Дата

ЛР.ОК24.ПІ231.13.04

Виконаємо запит на оновлення конкретного рецепту.

PUT

/api/Recipes/{id}

Parameters

Cancel

Reset

Name	Description
id * required	
integer(\$int32)	
(path)	

Request body

application/json

```
{
  "id": 2,
  "title": "Млинці",
  "slug": "швидко та смачно",
  "description": "Молоко хліб сало яйця",
  "difficulty": 1,
  "categoryId": 2,
  "authorUserId": 1
}
```

Execute

Clear

Responses

```
{
  "id": 2,
  "title": "Млинці",
  "slug": "швидко та смачно",
  "description": "Молоко хліб сало яйця",
  "difficulty": 1,
  "categoryId": 2,
  "authorUserId": 1
}
```

Request URL

https://localhost:7265/api/Recipes/2

Server response

Code	Details
400	Error: response status is 400

Response body

```
{
  "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1",
  "title": "One or more validation errors occurred.",
  "status": 400,
  "errors": {
    "slug": {
      "Slug має містити лише малі латиниці, цифри та тире."
    }
  },
  "traceId": "00-c1ea3285bc899c44aa5817eacc4fb1f-0d39ffd9ff07077f-00"
}
```

Response headers

```
content-type: application/problem+json; charset=utf-8
date: Mon, 13 Oct 2025 18:06:05 GMT
server: Kestrel
```

Response

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header.

Example Value

Schema

```
{
  "id": 0,
  "title": "string",
  "slug": "string",
  "description": "string",
  "difficulty": 1,
  "categoryId": 0,
  "authorUserId": 0
}
```

					ЛР.ОК24.ПІ231.13.04	Арк.
						34
Змн.	Арк.	№ докум.	Підпис	Дата		

Виконаємо запит на видалення конкретного рецепту.

DELETE

/api/Recipes/{id}

Parameters

Cancel

Name	Description
id <small>* required</small>	
<small>integer(\$int32)</small>	4
<small>(path)</small>	

ExecuteClear

Responses

Curl

```
curl -X 'DELETE' \
'https://localhost:7265/api/Recipes/4' \
-H 'accept: */*'
```

Request URL

https://localhost:7265/api/Recipes/4

Server response

Code	Details
200	<div>Response headers<div><div>content-length: 0</div><div>date: Mon, 13 Oct 2025 18:06:59 GMT</div><div>server: Kestrel</div></div></div>

Responses

Code	Description	Links
200	OK	No links

Виконаємо запит на публікацію рецепта.

POST

/api/Recipes

Parameters

CancelReset

No parameters

Request body

application/json

```
{
  "id": 16,
  "title": "М'ясо",
  "slug": "14",
  "description": "смачне потужне м'ясо",
  "difficulty": 1,
  "categoryId": 1,
  "authorUserId": 2
}
```

ExecuteClear

Responses

Request URL

https://localhost:7265/api/Recipes

Server response

Code

Details

201

Undocumented

Response body

```
{
  "id": 5,
  "title": "М'ясо",
  "slug": "14",
  "description": "смажене поручем м'ясо",
  "difficulty": 1,
  "categoryId": 1,
  "authorUserId": 2
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 13 Oct 2025 18:08:48 GMT
location: https://localhost:7265/api/Recipes/5
server: Kestrel
```

Responses

Code

Description

Links

200

OK

No links

Media type

text/plain

Controls Accept header:

Example Value | Schema

```
{
  "id": 0,
  "title": "string",
  "slug": "string",
  "description": "string",
  "difficulty": 1,
  "categoryId": 0,
  "authorUserId": 0
}
```

Users

Виконаємо запит на отримання усіх юзерів.

GET /api/Users

Parameters

No parameters

Execute Clear

Responses

Curl

```
curl -X 'GET' \
'https://localhost:7265/api/Users' \
-H 'accept: text/plain'
```

Request URL

https://localhost:7265/api/Users

Server response

Code

Details

200

Response body

```
{
  "id": 1,
  "displayName": "Alice",
  "email": "alice@example.com",
  "role": 1
},
{
  "id": 2,
  "displayName": "Bob",
  "email": "bob@gmail.com",
  "role": 3
},
{
  "id": 3,
  "displayName": "Charlie",
  "email": "charlie@yahoo.com",
  "role": 1
},
{
  "id": 4,
  "displayName": "Diana",
  "email": "diana@duckduckgo.com",
  "role": 1
}
}
```

Response headers

Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 13 Oct 2025 18:30:04 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{
  "id": 0,
  "displayName": "string",
  "email": "string",
  "role": 1
}
```

Виконаємо запит на публікацію юзера.

POST /api/Users

CancelReset

No parameters

Request body

application/json

```
{
  "id": 52,
  "displayName": "Misha",
  "email": "mishaxgod22@gmail.com",
  "role": 2
}
```

Execute

Clear

Responses

Curl

```
curl -X 'POST' \
  https://localhost:7265/api/Users \
  -H 'accept: text/plain' \
  -H 'Content-Type: application/json' \
  -d '{
    "id": 52,
    "displayName": "Misha",
    "email": "mishaxgod22@gmail.com",
    "role": 2
  }'
```

Request URL

https://localhost:7265/api/Users

Server response

Code	Details
201	<div>Response body<div><pre>{   "id": 5,   "displayName": "Misha",   "email": "mishaxgod22@gmail.com",   "role": 2 }</pre></div><div>Download</div></div> <div>Response headers<div><pre>content-type: application/json; charset=utf-8 date: Mon, 13 Oct 2025 18:31:04 GMT location: https://localhost:7265/api/Users/5 server: Kestrel</pre></div></div>

Responses

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{
  "id": 0,
  "displayName": "string",
  "email": "string",
  "role": 1
}
```

Змн.	Арк.	№ докум.	Підпис	Дата

Виконаємо запит на отримання конкретного юзера.

GET

/api/Users/{id}

Parameters

Cancel

Name	Description
id * required	
integer(\$int32)	4
(path)	

ExecuteClear

Responses

Curl

curl -X 'GET' \n'https://localhost:7265/api/Users/4' \n-H 'accept: text/plain'

Request URL

https://localhost:7265/api/Users/4

Server response

Code	Details
200	<div><div>Response body</div><div>{\n  "id": 4,\n  "displayName": "Diana",\n  "email": "diana@duckduckgo.com",\n  "role": 1\n}</div><div>Download</div></div> <div><div>Response headers</div><div>content-type: application/json; charset=utf-8\ndate: Mon, 13 Oct 2025 18:31:58 GMT\nserver: Kestrel</div></div>

Responses

CodeDescriptionLinks

200OKNo links

Media type

text/plain

Controls Accept header.

Example Value | Schema

{\n "id": 0,\n "displayName": "string",\n "email": "string",\n "role": 1\n}

Виконаємо запит на оновлення юзера.

PUT

/api/Users/{id}

Parameters

Cancel

Reset

Name	Description
id * required	
integer(\$int32)	2
(path)	

Request body

application/json

{\n "id": 2,\n "displayName": "Ilusha",\n "email": "ilushamoney@gmail.com",\n "role": 1\n}

Execute

Clear

Responses

Curl

curl -X 'PUT' \n'https://localhost:7265/api/Users/2' \n-H 'accept: text/plain' \n-H 'Content-Type: application/json' \n-d '{\n "id": 2,\n "displayName": "Ilusha",\n "email": "ilushamoney@gmail.com",\n "role": 1\n}'

Request URL

https://localhost:7265/api/Users/2

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{   "id": 2,   "displayName": "Iluska",   "email": "iluhamonesy@gmail.com",   "role": 1 }</pre></div><div> Download</div></div> <div><div>Response headers</div><div><pre>content-type: application/json; charset=utf-8 date: Mon, 13 Oct 2025 18:32:46 GMT server: Kestrel</pre></div></div>

Responses

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{
  "id": 0,
  "displayName": "string",
  "email": "string",
  "role": 1
}
```

Виконаємо запит на видалення юзера.

DELETE /api/Users/{id}

Cancel

Parameters

Name	Description
id * required	
integer(\$int32)	
(path)	

1

Execute

Clear

Responses

Curl

```
curl -X 'DELETE' \
'https://localhost:7265/api/Users/1' \
-H 'accept: */*'
```

Request URL

https://localhost:7265/api/Users/1

Server response

Code	Details
200	<div><div>Response headers</div><div><pre>content-length: 0 date: Mon, 13 Oct 2025 18:33:31 GMT server: Kestrel</pre></div></div>

Responses

Code	Description	Links
200	OK	No links

**Висновок:** Під час виконання лабораторної роботи, я ознайомився з процесом створення клієнт-серверної архітектури. Навчився створювати RESTful API для взаємодії між клієнтом і сервером.