

**Міністерство освіти і науки України**  
**Національний технічний університет України «Київський політехнічний**  
**інститут імені Ігоря Сікорського»**  
**Факультет інформатики та обчислювальної техніки**  
  
**Кафедра інформатики та програмної інженерії**

**Звіт**

з лабораторної роботи № 3 з дисципліни  
«Сучасні технології розробки WEB-застосувань на платформі Microsoft.NET»  
  
«Проектування REST веб-API»

**Виконав**

IC-13, Харчук А.В.  
(шифр, прізвище, ім'я, по батькові)

**Перевірів**

Бардін В.  
(прізвище, ім'я, по батькові)

Київ 2023

## Завдання

### Теоретична частина:

1. Ознайомитися з основами створення REST веб-API та методологією С4 для відображення архітектури системи.
2. Ознайомитися з основами створення ER-діаграм для представлення структури бази даних.

### Практична частина:

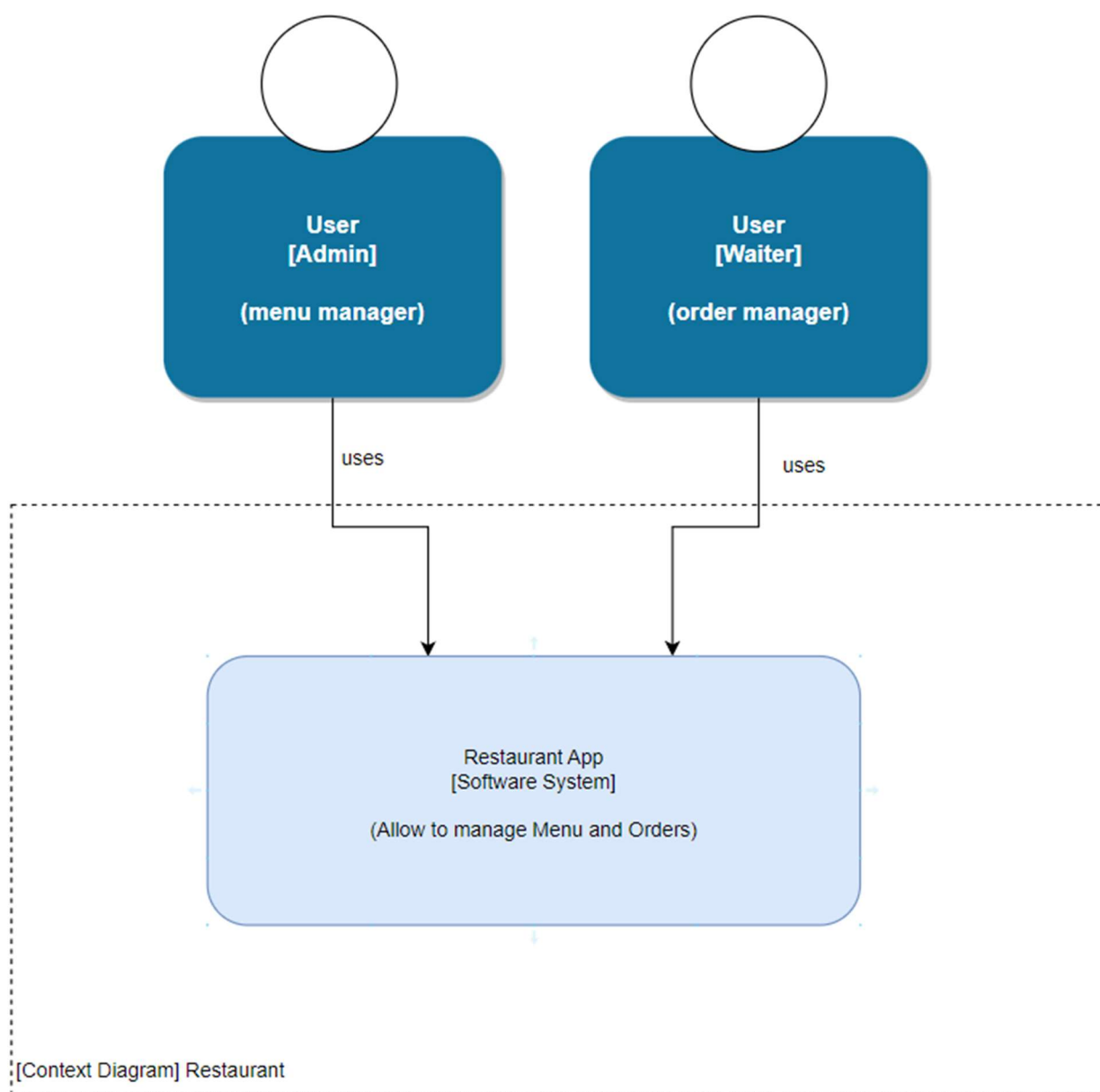
1. З дотриманням вимог REST-у спроектувати веб-API для обраної(згідно варіанту) доменної області, використовуючи методологію С4 для створення діаграми архітектури системи.
2. Створити ER-діаграму для DAL (Data Access Layer), яка відображатиме структуру бази даних веб-API.
3. Оформити спроектоване рішення у вигляді звіту до лабораторної роботи.

## Варіант 1

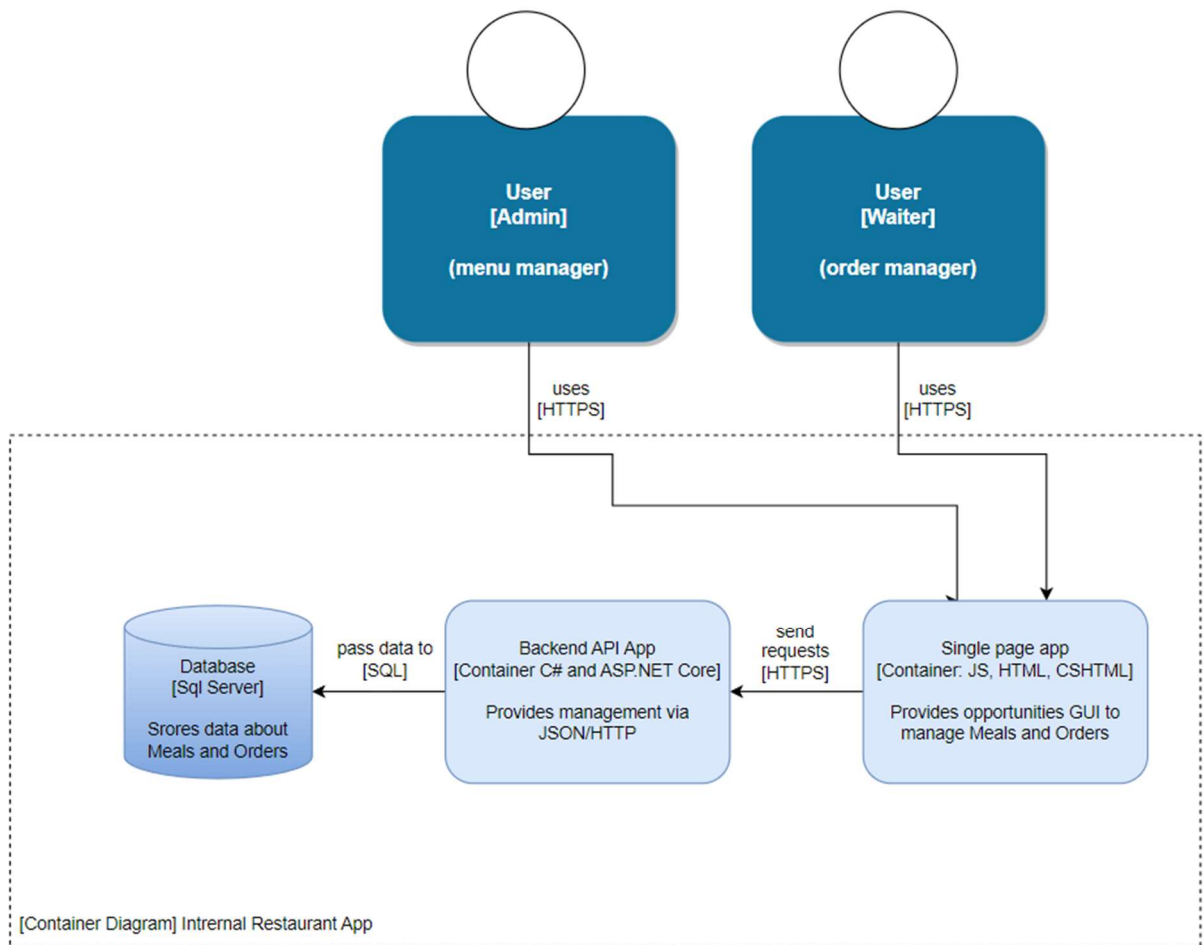
<b>1</b>	Ресторан. Формування замовлень.	<p>1. Страви складаються із інгредієнтів. Інгредієнти можуть складати різні страви. Страви складають прайсліст, в якому вказано ціну для різних порцій страви.</p> <p>2. Замовлення може містити в собі набір декількох порцій різних страв.</p> <p><b>Функціональні вимоги:</b></p> <p>1. Складання страв та меню; 2. Формування замовлень.</p>
----------	------------------------------------	--

## Діаграми архітектури системи за методологією C4

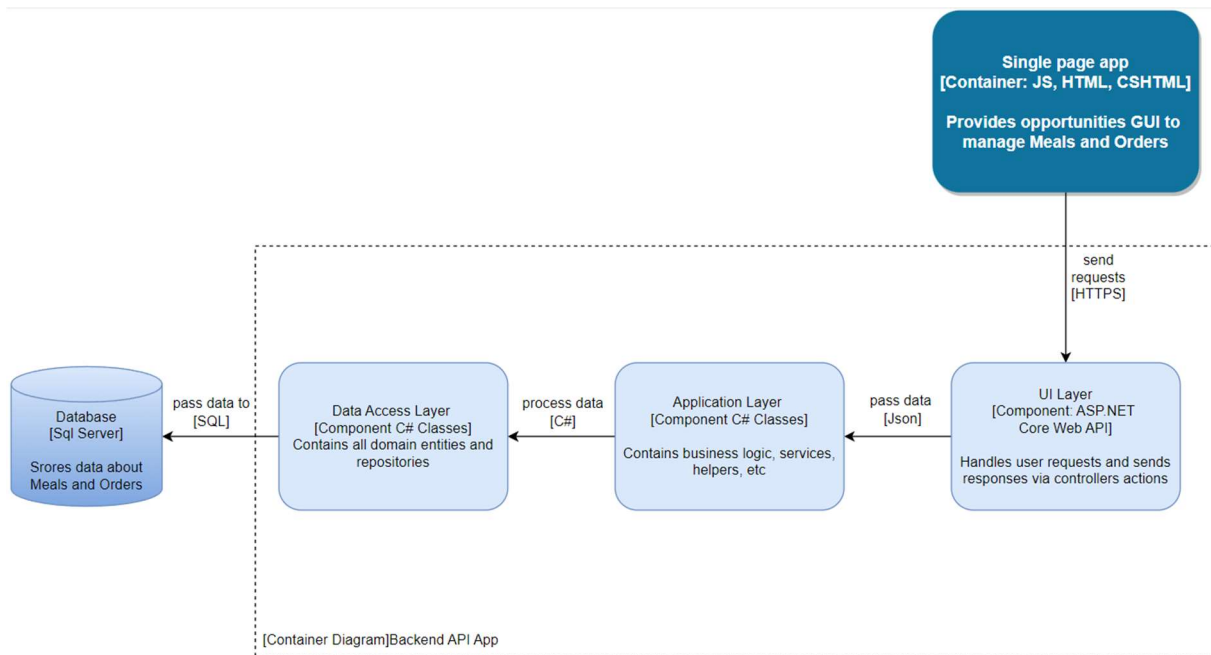
### Level 1



## Level 2

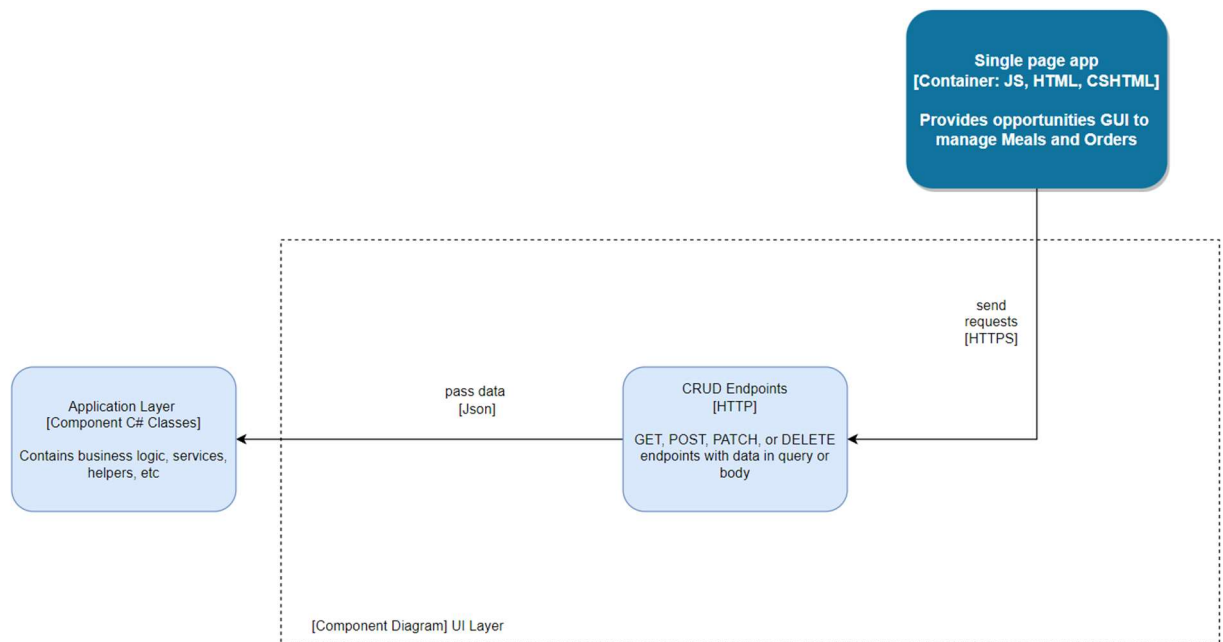


## Level 3

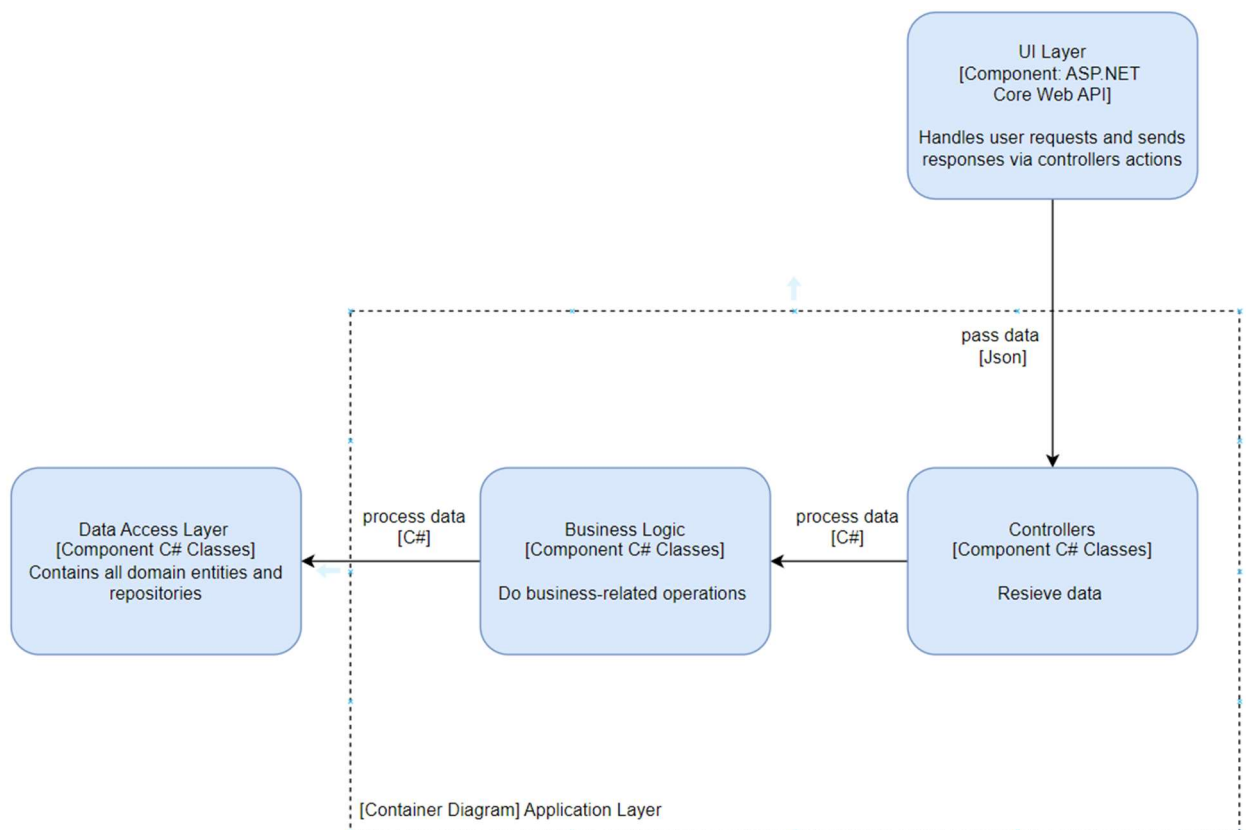


## Level 4

### UI Layer



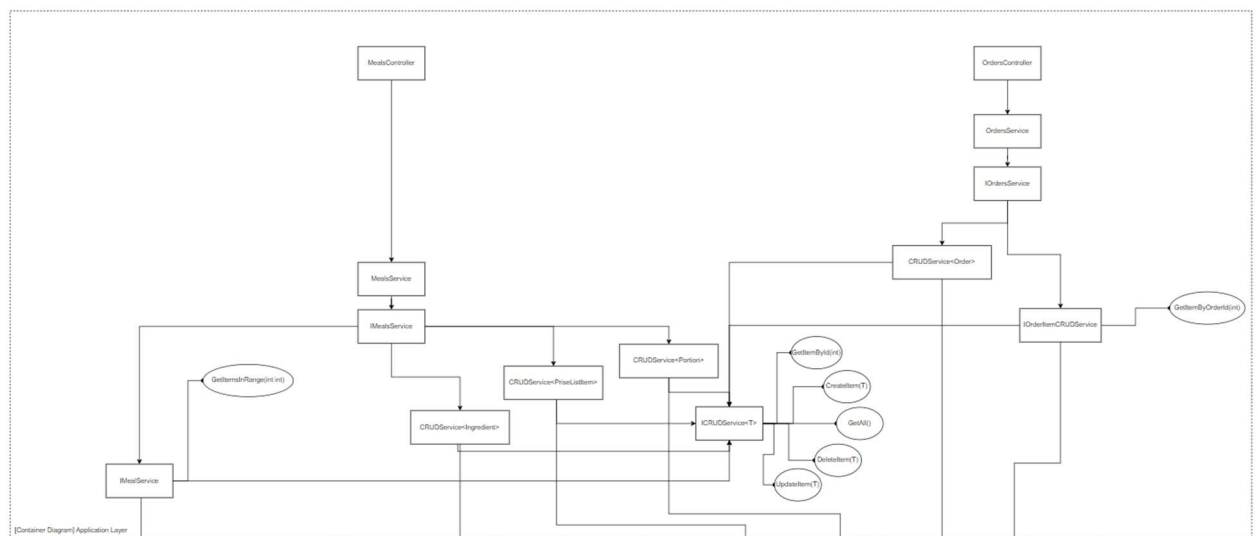
### Application Layer

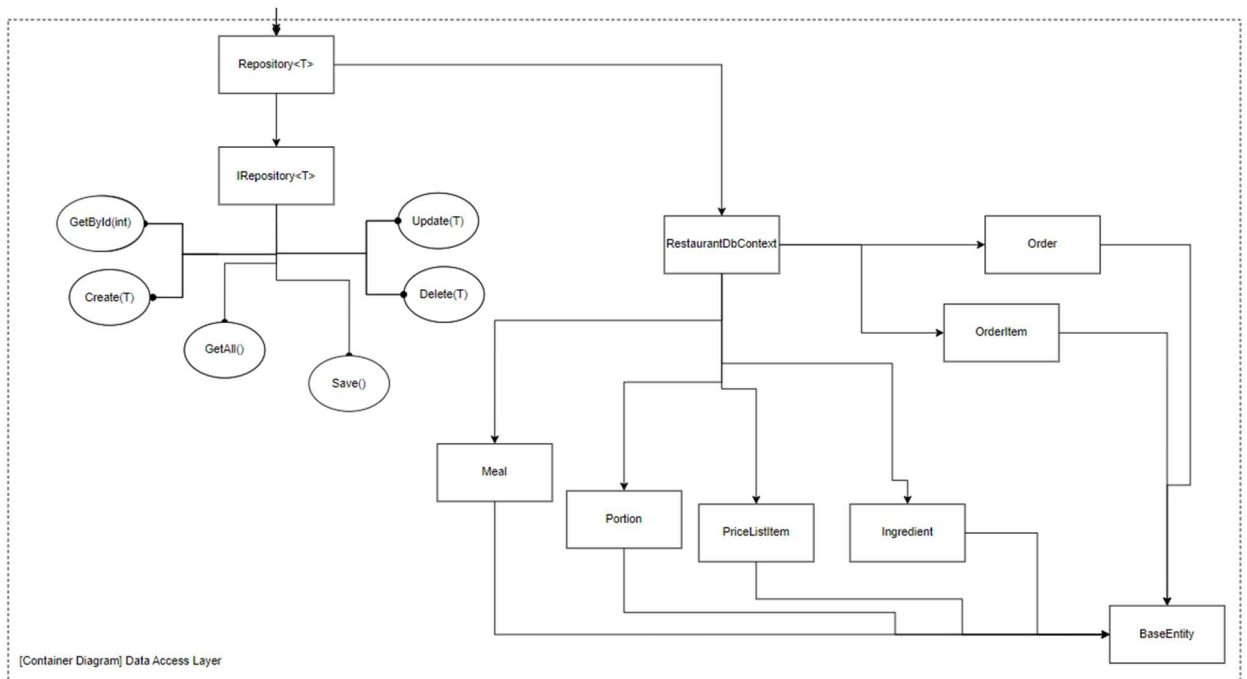


```
graph TD
    subgraph DAL [Data Access Layer]
        direction TB
        Parser["Parser  
[Component C# Classes]  
Parses data to match db classes"]
        DSaver["Data saver  
[Component C# Classes]  
Generate SQL code to access DB"]
        Parser -- "process data [C#]" --> DSaver
    end
    Parser -- "process data [C#]" --> AppLayer["Application Layer  
[Component C# Classes]  
Contains business logic, services, helpers, etc"]
    DSaver -- "pass data to [SQL]" --> DB[(Database  
[Sql Server]  
Stores data about Meals and Orders)]
```

The diagram illustrates the Data Access Layer (DAL) structure. It features three main components: the Application Layer, the Parser, and the Data saver. The Application Layer, located at the top, contains business logic, services, and helpers. It interacts with the Parser via a 'process data [C#]' relationship. The Parser, a component C# class, parses data to match database classes and also interacts with the Data saver via a 'process data [C#]' relationship. The Data saver, another component C# class, generates SQL code to access the database and passes data to the Database via a 'pass data to [SQL]' relationship. The Database, represented by a cylinder, stores data about meals and orders. A dashed box encloses the Parser and Data saver components, indicating they are part of the Data Access Layer.

- \*Всі методи асинхронні.
- \*Всі Get методи мають параметр `string include=""` в якому користувач вводить поля, які повинні бути включеними у вивід.





## Endpoints:

Base address:

<https://<your host>/api/>

Ендпоінти для страв

Get all Ingredients:

Endpoint: GET /api/Meal/Ingredients

Method: GetAllIngredients

Get Ingredient by ID:

Endpoint: GET /api/Meal/Ingredients/{id}

Method: GetIngredientById

Create Ingredient:

Endpoint: POST /api/Meal/Ingredients

Method: CreateIngredient

Update Ingredient:

Endpoint: PUT /api/Meal/Ingredients

Method: UpdateIngredient

Delete Ingredient:

Endpoint: DELETE /api/Meal/Ingredients/{id}

Method: DeleteIngredient

Get all Meals:

Endpoint: GET /api/Meal/Meals

Query parameters:

includePath: (Optional) A query parameter to specify the include path for related entities.

Method: GetAllMeals

Get Meal by ID:

Endpoint: GET /api/Meal/Meals/{id}

Query parameters:

includePath: (Optional) A query parameter to specify the include path for related entities.

Method: GetMealById

Get Meals by Page:

Endpoint: GET /api/Meal/Meals/Page/{num}

Query parameters:

includePath: (Optional) A query parameter to specify the include path for related entities.



Method: GetMealByPage

Create Meal:

Endpoint: POST /api/Meal/Meals

Method: CreateMeal

Update Meal:

Endpoint: PUT /api/Meal/Meals/{id}

Method: UpdateMeal

Delete Meal:

Endpoint: DELETE /api/Meal/Meals/{id}

Method: DeleteMeal

Get all Portions:

Endpoint: GET /api/Meal/Portions

Method: GetAllPortions

Get Portion by ID:

Endpoint: GET /api/Meal/Portions/{id}

Method: GetPortionById

Create Portion:

Endpoint: POST /api/Meal/Portions

Method: CreatePortion

Update Portion:

Endpoint: PUT /api/Meal/Portions

Method: UpdatePortion

Delete Portion:

Endpoint: DELETE /api/Meal/Portions/{id}

Method: DeletePortion

Get all PriceListItems:

Endpoint: GET /api/Meal/PriceListItems

Query parameters:

includePath: (Optional) A query parameter to specify the include path for related entities.

Method: GetAllPriceListItems

Get PriceListItem by ID:

Endpoint: GET /api/Meal/PriceListItems/{id}

Query parameters:

includePath: (Optional) A query parameter to specify the include path for related entities.

Method: GetPriceListItemById

Create PriceListItem:

Endpoint: POST /api/Meal/PriceListItems

Method: CreatePriceListItem

Update PriceListItem:

Endpoint: PUT /api/Meal/PriceListItems/{id}

Method: UpdatePriceListItem

Delete PriceListItem:

Endpoint: DELETE /api/Meal/PriceListItems/{id}

Method: DeletePriceListItem

Ендпоінти для замовлень

Get All Orders:

Endpoint: GET /api/Order/Orders

Method: GetAllOrders

Get Order by ID:

Endpoint: GET /api/Order/Orders/{id}

Method: GetOrderById

Create Order:

Endpoint: POST /api/Order/Orders

Method: CreateOrder

Update Order:

Endpoint: PUT /api/Order/Orders/{id}

Method: UpdateOrder

Delete Order:

Endpoint: DELETE /api/Order/Orders/{id}

Method: DeleteOrder

Get All Order Items:

Endpoint: GET /api/Order/OrderItems

Method: GetAllOrderItems

Query Parameters:

orderId: (Optional) Filter by order ID.

includePath: (Optional) A query parameter to specify the include path for related entities.

Get Order Item by ID:

Endpoint: GET /api/Order/OrderItems/{id}

Method: GetOrderItemById

Query Parameters:

includePath: (Optional) A query parameter to specify the include path for related entities.

Create Order Item:

Endpoint: POST /api/Order/OrderItems

Method: CreateOrderItem

Update Order Item:

Endpoint: PUT /api/Order/OrderItems/{id}

Method: UpdateOrderItem

Delete Order Item:

Endpoint: DELETE /api/Order/OrderItems/{id}

Method: DeleteOrderItem

ER-діаграма БД

