Національний технічний університет України «КПІ»
Факультет інформатики та обчислювальної техніки
Кафедра Інформаційних систем та технологій

Лабораторна робота №1

з дисципліни « Сучасні технології розробки WEB-застосувань на платформі Microsoft.NET»

на тему: « Проектування REST веб-API»

Виконав:
студент гр. ІС-13
Хробатенко Андрій
Викладач:
Бардін В.

2023 рік

Завдання:

Теоретична частина:

1. Ознайомитися з основами створення REST веб-API та методологією C4 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 C4 для створення діаграми архітектури системи.

2. Створити ER-діаграму для DAL (Data Access Layer), яка відображатиме структуру бази даних веб-API.

3. Оформити спроєктоване рішення у вигляді звіту до лабораторної роботи.

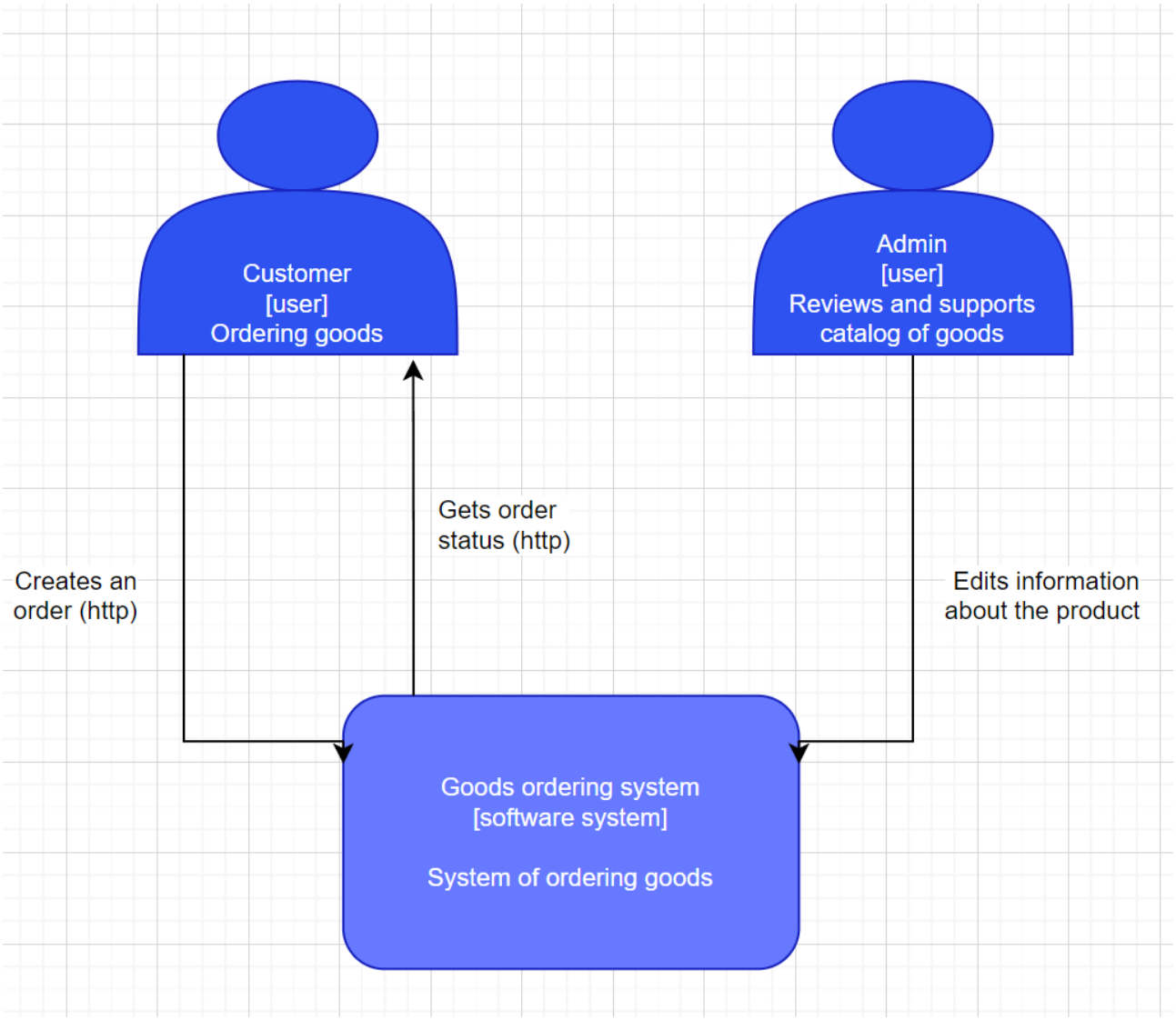
5	Склад. Облік товарів	1. На складі зберігаються товари, які можуть бути відпущені замовнику зі складу. 2. Товари можливо заказати до продажу навіть при відсутності його на складі. При відсутності товару потрібно записати його у чергу на придбання та завезення на склад. Після доставки відсутніх товарів на склад відвантаження за замовленням може бути виконано. **Функціональні вимоги:** 1. Облік товарів на складі; 2. Реалізація товарів зі складу.

Посилання на GitHub:

Виконання:

Для цієї роботи використаємо модель C4, яка описує архітектуру програмних систем, використовуючи кілька кутів зору, які пояснюють декомпозицію системи на контейнери та компоненти, відносини між ними та, де доречно, їх відношення до користувача.

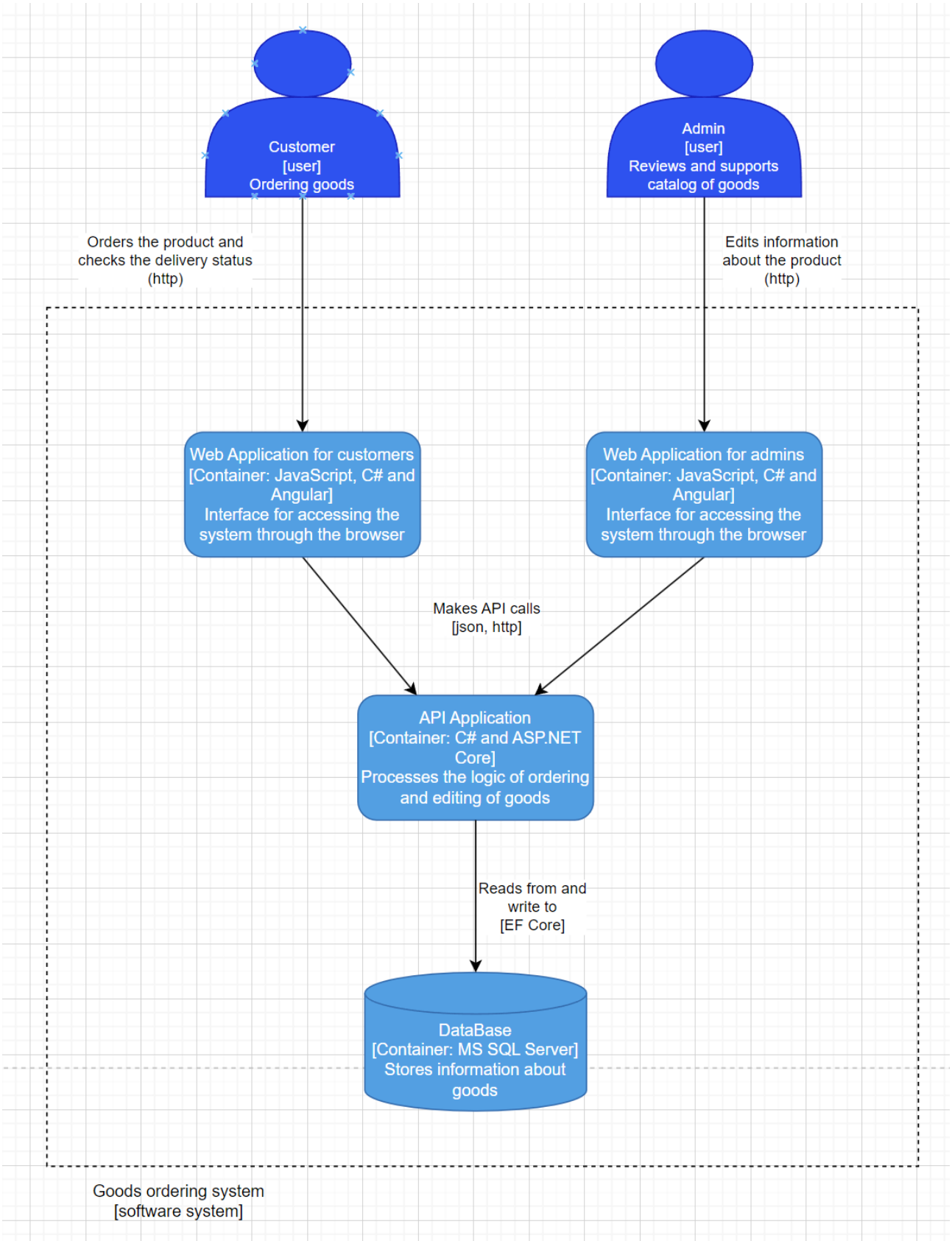
Contex diagram:



Маємо двох користувачів: замовник, адмін. Замовник може замовляти товар, дивитись статус його відправки та стати на чергу отримання цього товару, якщо на данний момент його не буде на складі. Адмін може редагувати товар,

змінюватии його кількість, працювати з чергою замовлень і з самими замовленнями.

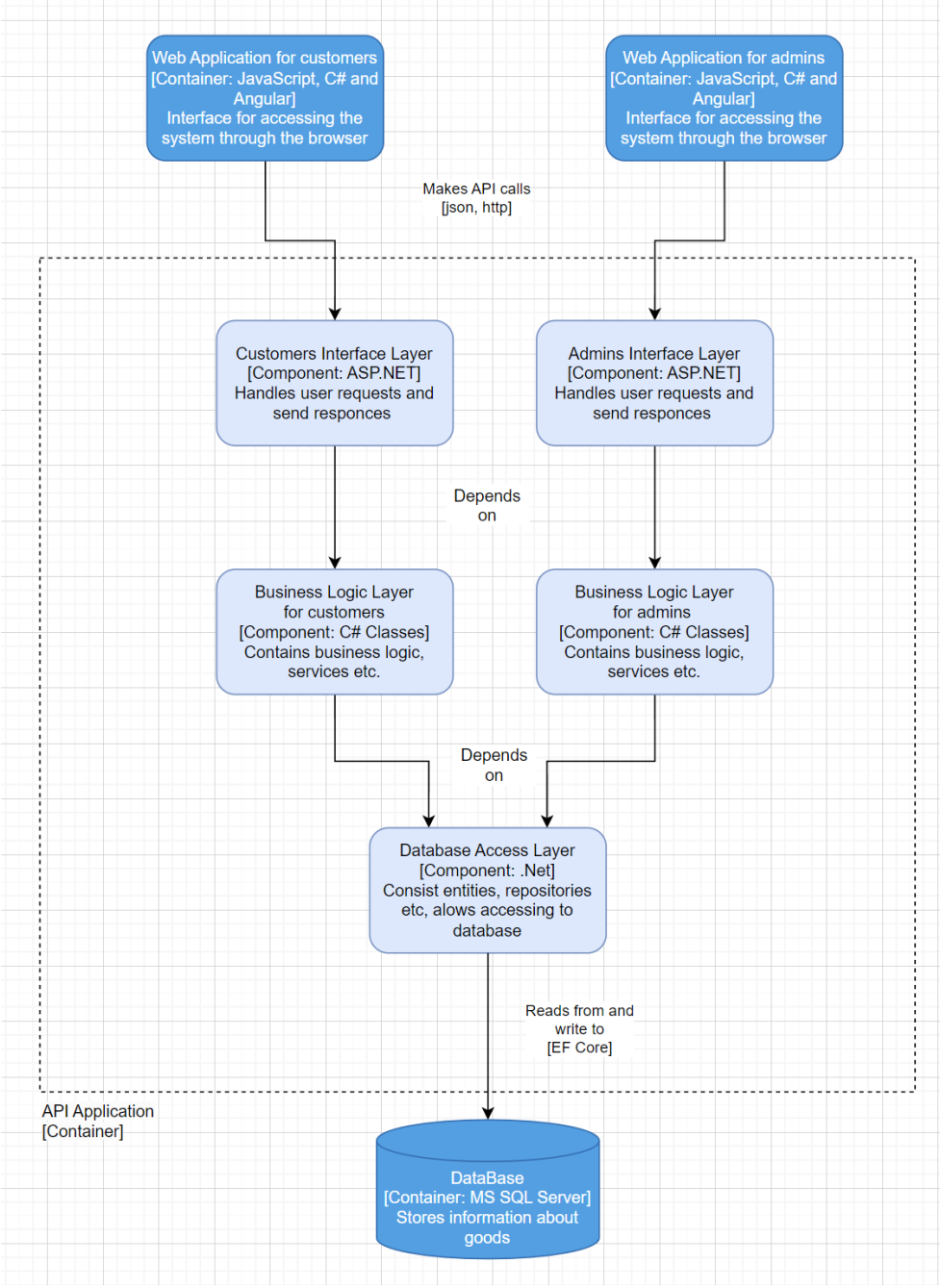
Container diagram:



Ця система складається з трьох компонентів:

1) Веб-застосунок, за допомогою якого користувачі і адміни зможуть отримати функціонал, який вони потребують.
2) Серверна програма, яка реалізується за допомогою REST API і приймає запити від браузера.
3) База данних, яка зберігає всю інформаці і сутності, потрібні для функціонування програми.

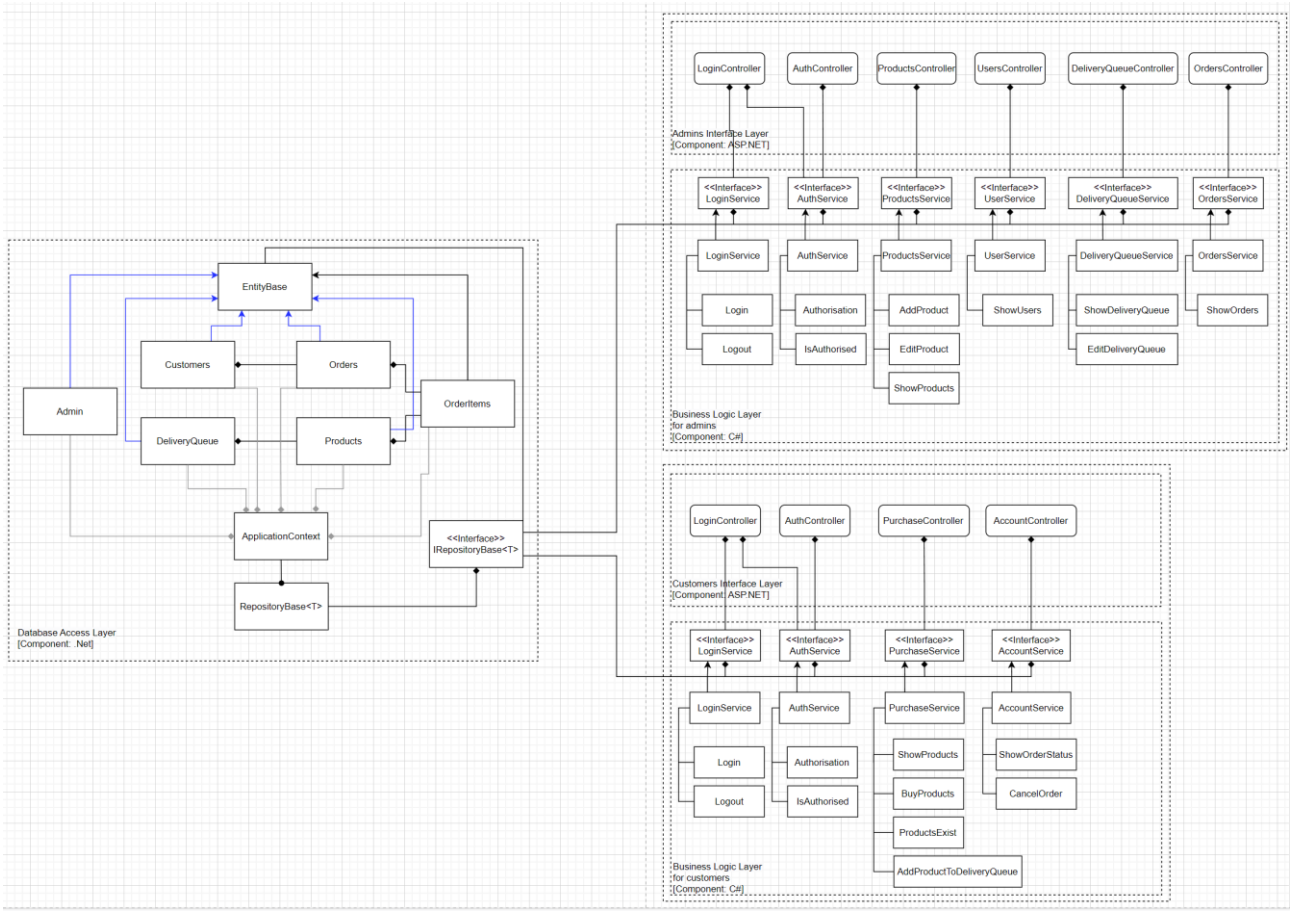
Component diagram:



В цьому випадку застосовується N-Layer/Tier архітектура, яка поділяє програму на 3 шари:

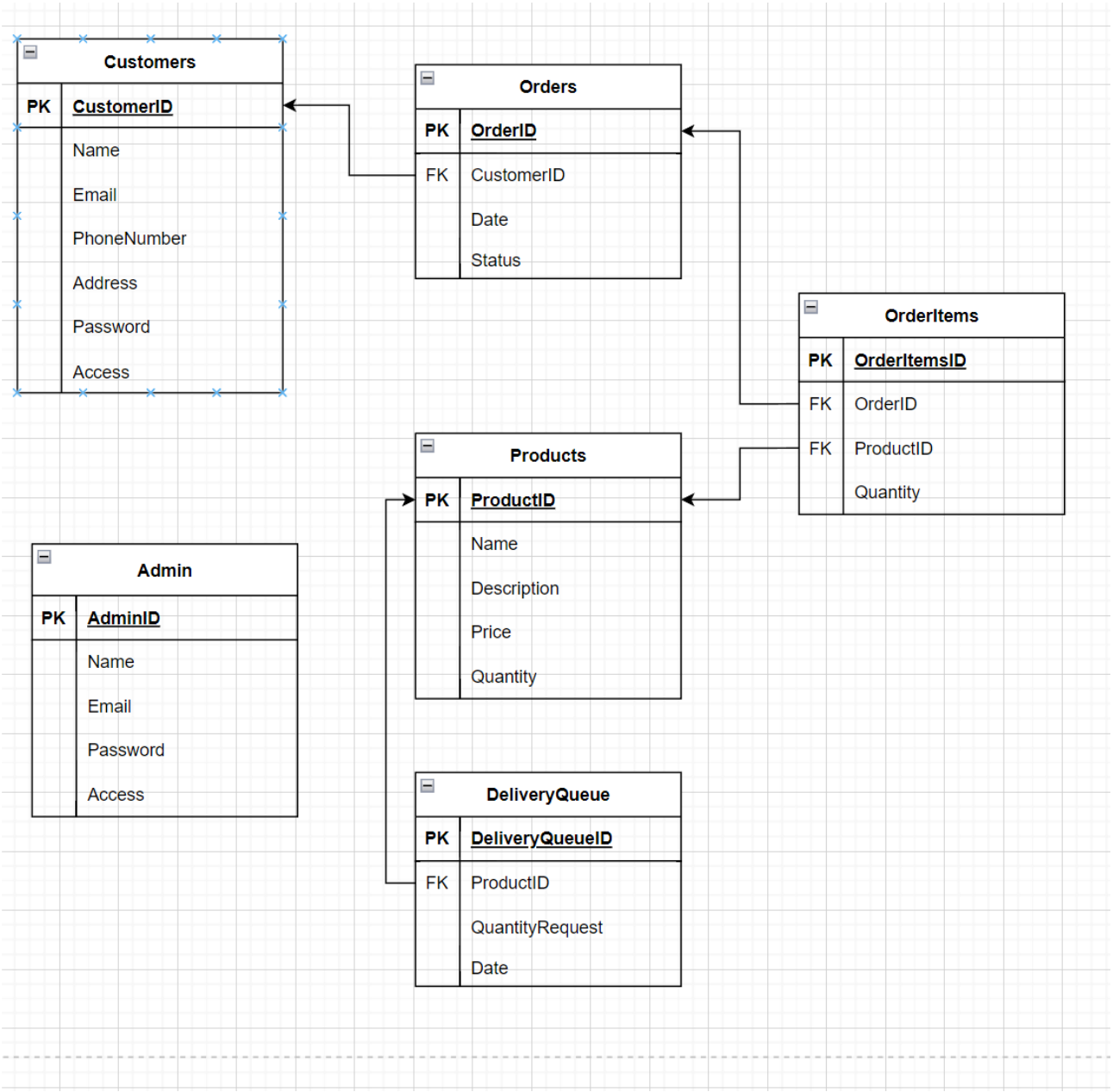
1) Customers/Admins Interface Layer: на цьому рівні знаходяться контролери та графічний інтерфейс з яким будуть працювати користувачі
2) Business Logic Layer for customers/admins: тут міститься вся бізнес логіка програми. Містить сервіси, їх інтерфейси, логіка обробки даних.
3) Data Access Layer: за допомогою якого ми маємо можливість працювати з базою даних.

Code diagram:



На цій діаграмі розписано про контролери, інтерфейси, класи та методи які вони виконують. Також тут присутня реалізація репозиторію RepositoryBase<T> та контекст бази даних ApplicationContext який містить усі сутності.

ER-diagram для DAL:



Таблиці:

Customers

CustomerID (INTEGER, PRIMARY KEY): Унікальний ідентифікатор клієнта.

Name (VARCHAR): Ім'я клієнта.

Email (VARCHAR): Електронна пошта клієнта.

PhoneNumber(VARCHAR): Номер телефону клієнта.

Address (VARCHAR): Адреса клієнта.

Password (VARCHAR): Пароль клієнта.

Access(BOOLEAN): Доступ до даних.

Admin

AdminID (INTEGER, PRIMARY KEY): Унікальний ідентифікатор адміністратора.

Name (VARCHAR): Ім'я адміністратора.

Email (VARCHAR): Електронна пошта адміністратора.

Password (VARCHAR): Пароль адміна.

Access(BOOLEAN): Доступ до даних.

Products

ProductID (INTEGER, PRIMARY KEY): Унікальний ідентифікатор товару.

Name (VARCHAR): Назва товару.

Description (TEXT): Опис товару.

Price (DECIMAL): Ціна товару.

Quantity (INTEGER): Кількість доступних одиниць товару на складі.

Orders

OrderID (INTEGER, PRIMARY KEY): Унікальний ідентифікатор замовлення.

CustomerID (INTEGER, FOREIGN KEY): Посилання на унікальний ідентифікатор клієнта з таблиці Customer.

Date (DATE/TIMESTAMP): Дата замовлення.

Status (VARCHAR): Статус замовлення.

Order_Items

OrderItemsID (INTEGER, PRIMARY KEY): Унікальний ідентифікатор деталі замовлення.

OrderID (INTEGER, FOREIGN KEY): Посилання на унікальний ідентифікатор замовлення з таблиці Order.

ProductID (INTEGER, FOREIGN KEY): Посилання на унікальний ідентифікатор товару з таблиці Product.

Quantity (INTEGER): Кількість товару в замовленні.

DeliveryQueue

DeliveryQueueID (INTEGER, PRIMARY KEY): Унікальний ідентифікатор черги на придбання товару.

ProductID (INTEGER, FOREIGN KEY): Посилання на унікальний ідентифікатор товару з таблиці Product.

QuantityRequested (INTEGER): Кількість товару, яку потрібно придбати.

Date (DATE/TIMESTAMP): Дата, коли товар був запрошений на закупівлю.

Кінцеві точки REST API(EndPoints):

1) Створення замовлення:

Метод: Post

URL: /api/orders

2) Додавання товару в чергу:

Метод: Post

URL: /api/deliveryqueues

3) Додавання товару в чергу:

Метод: Post

URL: /api/deliveryqueues

4) Отримання списку клієнтів:

Метод: Get

URL: /api/orders

5) Отримання списку продуків:

Метод: Get

URL: /api/products

6) Отримання інформації про конкретного клієнта:

Метод: Get

URL: /api/customers/{CustomerID}

7) Отримання адреси конкретного клієнта:

Метод: Get

URL: /api/customers/{CustomerID}/address

8) Отримання інформації про черги:

Метод: Get

URL: /api/deliveryqueues

9)Оновлення інформації про продукт:

Метод: PUT

URL: /api/products/{ProductID}

10)Оновлення інформації про продукт:

Метод: PUT

URL: /api/products/{ProductID}

10)Видалення продукту:

Метод: DELETE

URL: /api/products/{ProductID}

10)Видалення продукту:

Метод: DELETE

URL: /api/products/{ProductID}

11)Видалення черги:

Метод: DELETE

URL: /api/ deliveryqueues/{ DeliveryQueueID}

Висновок:

Я ознайомився з основами створення REST веб-API та методологією

C4 для відображення архітектури системи, ознайомився з основами створення ER-діаграм для представлення структури бази даних.