# 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

# Практикум №4

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

на тему: «Реалізація веб-АРІ»

Викладач: Бардін В. Виконав: Хільчук А.В. студент 3 курсу групи ІП-14 ФІОТ

# Практична робота №4

Тема: Реалізаця веб-АРІ

#### Завдання:

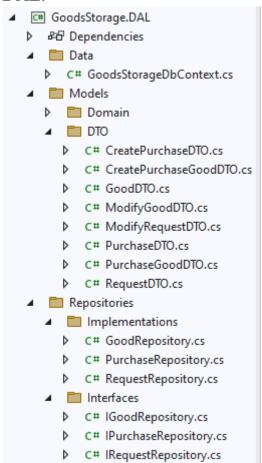
- 1. За виконання першого пункту можна отримати до 10 балів, розподілені наступним чином:
  - а. За правильний поділ застосунку на шари до 3 балів.
  - b. Реалізація DAL 3 бали.
  - с. Реалізація BLL 3 бали.
  - d. Реалізація Presentation Layer (REST web-API) 1 бал.
- 2. За виконання другого пункту ЛР студент може отримати до 5 балів, а саме:
  - а. Модульні тести 2 бали,
  - b. Інтеграційні тести 3 бали.
- 3. Експорт розробленого API до Postman оцінюється в 1 (один) бал.
- 4. Відповіді на теоретичні запитання до 4 балів.

		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		
_	Склад. Облік	придбання та завезення на склад. Після доставки		
5	товарів	відсутніх товарів на склад відвантаження за		
		замовленням може бути виконано.		
		Функціональні вимоги:		
		1. Облік товарів на складі;		
		2. Реалізація товарів зі складу.		

#### Виконання:

1. Реалізація проектук

# Результуючий проект має наступний вигляд: DAL:



# BAL:

4	C	# 6	iood	sSto	rage.BAL
	$\triangleright$	84	5 De	per	idencies
	4	2	M	ode	ls
		4		DT	0
			Þ	C#	LoginUserDTO.cs
			Þ	C#	RegisterUserDTO.cs
			Þ	C#	UpdateUserSummaryDTO.cs
			Þ	C#	UserSummaryDTO.cs
		Þ	C#	Ba	seResponse.cs
	4	2	■ Se	rvic	es
		4		lm	plementations
			$\triangleright$	C#	AuthService.cs
			Þ	C#	GoodsService.cs
			Þ	C#	PurchaseService.cs
			Þ	C#	RequestService.cs
			Þ	C#	UserSummaryService.cs
		4		Int	erfaces
			Þ	C#	IAuthService.cs
			Þ	C#	IGoodsService.cs
			$\triangleright$	C#	IPurchaseService.cs
			$\triangleright$	C#	IRequestService.cs
			$\triangleright$	C#	IUserSummaryService.cs

### API:

Connected Services ▶ & Dependencies Properties Authorization D C# CanAccessPurchaseHandler.cs D C# CanAccessPurchaseRequiremer C# CanAccessRequestHandler.cs C# CanAccessRequestRequirement ▶ C# CanAccessUserSummaryHandle C# CanAccessUserSummaryRequir C# CanCreateRequestHandler.cs D C# CanCreateRequestRequirement Controllers C# AuthController.cs C# GoodsController.cs D C# PurchasesController.cs C# RequestsController.cs C# UserSummaryController.cs Migrations b C# 20231211173324\_Intial.cs C# 20231212105846\_Add-Identity.c C# 20231212232325\_Small-Change: C# GoodsStorageDbContextModel appsettings.json ■ GoodsStorageApi.http C# Program.cs

# 2. Реалізація тестування:

# Юніт-тестування:

- - ▶ ₽₽ Dependencies
  - C# AuthServiceTests.cs
    - (i) coverage.json
    - C# GlobalUsings.cs
  - C# GoodsServiceTests.cs
  - ▶ C# MockConfiguration.cs
  - C# MockUserManager.cs
  - C# PurchaseServiceTests.cs
  - D C# RequestServiceTests.cs
  - D C# UserSummaryServiceTests.cs

- GoodsStorage.DAL.UnitTests
  - ▶ ₽₽ Dependencies
  - C# DbContextMocker.cs
    - C# GlobalUsings.cs
  - C# GoodRepositoryTests.cs
  - C# PurchaseRepositoryTests.cs
  - C# RequestRepositoryTests.cs

# Інтеграційне тестування:

- - ▶ ₽₽ Dependencies
  - C# AuthControllerTests.cs
    - C# GlobalUsings.cs
  - ▶ C# GoodsControllerTests.cs
  - C# JwtTokenMock.cs
  - C# PurchaseControllerTests.cs
  - C# RequestControllerTests.cs
  - ▶ C# WebAppFactory.cs
    - 3. Експорт у Postman:

✓ Goods Storage Api						
→     api						
→ Auth						
→ ☐ register						
> POST /api/Auth/register						
→ login						
> POST /api/Auth/login						
✓ ☐ Goods						
√ [id]						
> GET /api/Goods/:id						
> PUT /api/Goods/:id						
> DEL /api/Goods/:id						
> GET /api/Goods						
> POST /api/Goods						
→ Purchases						
√ [] {id}						
> GET /api/Purchases/:id						
> GET /api/Purchases						
> POST /api/Purchases						
→ Requests						
√ [] {id}						
> GET /api/Requests/:id						
> DEL /api/Requests/:id						
> PUT /api/Requests/:id						
> GET /api/Requests						
> POST /api/Requests						
→ ☐ UserSummary						
✓ ☐ {id}						
> GET /api/UserSummary/:id						
> PUT /api/UserSummary/:id						

#### Висновок

Отож, у ході виконання лабораторної роботи було спроектовано реалізовано API на основі проектувань попередньої лабороторної роботи. У рамках виконання даної діаграми було реалізовано 5 контроллерів, 5 сервісів, а також 3 репозиторіїв. Додано авторизацію- як контролерів так і ресурсів- та јwt-автентифікацію за допомогою Identity Core. Створено низку інтеграційних та юніт-тестів. Документацію результату додано в Postman. Набуто практичних навичок реалізації та тестування API за допомогою ASP.Net core.