Міністерство освіти й науки України

Івано-Франківський національний технічний

університет нафти і газу

Кафедри ІПЗ

Лабораторна робота №4

# Тема: «Написання ІНТЕГРАЦІЙНИХ тестів»

Варінт №10

Виконав студент

Групи ІП-21-2

Кавецький О.О

Перевірив

Храбатин Р. І.

м. Івано-Франківськ

2025р.

**Мета:** Навчитися створювати інтеграційні тести, розібратися в існуючих інструментах для тестування API та навчитися їх обирати.

**Короткі теоретичні відомості:**

Інтеграційне тестування (Integration Testing) — це вид тестування, який перевіряє взаємодію між модулями або компонентами системи. Його мета — виявити дефекти, що виникають під час обміну даними між компонентами.

**Тестування API:**

API (Application Programming Interface) — це набір правил, що дозволяє програмним компонентам взаємодіяти між собою. Тестування API дозволяє перевірити коректність обробки запитів та відповідей без необхідності взаємодії з користувацьким інтерфейсом.

**Основні методи HTTP-запитів для тестування API:**

* **GET** — отримання даних із сервера.
* **POST** — створення нового ресурсу.
* **PUT** — оновлення існуючого ресурсу.
* **DELETE** — видалення ресурсу.

**Інструменти для тестування API:**

* **Postman** — популярний інструмент для ручного та автоматизованого тестування API.
* **Jest** — тестовий фреймворк для JavaScript, який використовується для написання автоматизованих тестів.
* **Axios** — бібліотека для виконання HTTP-запитів, яка спрощує взаємодію з API.

**Переваги інтеграційного тестування API:**

* Виявлення проблем на ранніх стадіях розробки.
* Мінімізація кількості помилок у взаємодії між сервісами.
* Автоматизація перевірок дозволяє швидше знаходити помилки.

**Виконання:**

**1. Вибір інструменту для тестування API**

Зі списку доступних інструментів було обрано **Axios**, оскільки він є простим у використанні та добре підтримується спільнотою.

**2. Налаштування середовища**

1. Створення нового каталогу.
2. Встановлення Axios:

npm install axios

1. Створення нового тестового файлу **api.test.js**.

const axios = require('axios');

const BASE\_URL = 'https://jsonplaceholder.typicode.com';

describe('JSONPlaceholder API Tests', () => {

    test('GET /posts should return a list of posts', async () => {

        const response = await axios.get(`${BASE\_URL}/posts`);

        expect(response.status).toBe(200);

        expect(Array.isArray(response.data)).toBeTruthy();

    });

    test('GET /posts/:id should return a single post', async () => {

        const response = await axios.get(`${BASE\_URL}/posts/1`);

        expect(response.status).toBe(200);

        expect(response.data.id).toBe(1);

    });

    test('POST /posts should create a new post', async () => {

        const newPost = {

            title: 'Test Post',

            body: 'This is a test post.',

            userId: 1

        };

        const response = await axios.post(`${BASE\_URL}/posts`, newPost);

        expect(response.status).toBe(201);

        expect(response.data.title).toBe(newPost.title);

    });

    test('PUT /posts/:id should update an existing post', async () => {

        const updatedPost = {

            id: 1,

            title: 'Updated Title',

            body: 'Updated body content.',

            userId: 1

        };

        const response = await axios.put(`${BASE\_URL}/posts/1`, updatedPost);

        expect(response.status).toBe(200);

        expect(response.data.title).toBe(updatedPost.title);

    });

    test('DELETE /posts/:id should delete a post', async () => {

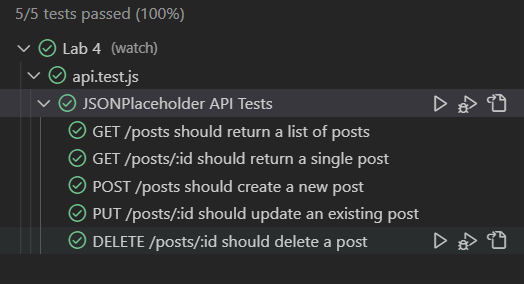
        const response = await axios.delete(`${BASE\_URL}/posts/1`);

        expect(response.status).toBe(200);

    });

});

Ось результат виконання тестів:



**Висновок:** В ході виконання лабораторної роботи було виконано наступне:

* Ознайомлено з основними принципами тестування API.
* Встановлено інструмент для тестування (Axios).
* Написано тестові набори для методів GET, POST, PUT, DELETE.
* Автоматизовано перевірку роботи API.
* Завантажено код у репозиторій.

Автоматизоване тестування дозволяє швидко перевіряти роботу API, мінімізуючи потребу у ручному тестуванні через Postman. Використання Jest у поєднанні з Axios значно спрощує написання тестів і робить їх зручними для підтримки в майбутньому.

Посилання на репозиторій - <https://github.com/olehkavetskyi/FAT-Labs>