Міністерство освіти й науки України

Івано-Франківський національний технічний

університет нафти і газу

Кафедри ІПЗ

Лабораторна робота №5

# Тема: «НАПИСАННЯ ТЕСТІВ ДЛЯ ГРАФІЧНОГО ІНТЕРФЕЙСУ»

Виконав студент

Групи ІП-21-2

Кавецький О.О

Перевірив

Храбатин Р. І.

м. Івано-Франківськ

2025р.

**Мета:** навчитися створювати UI тести, розібратися в існуючих інструментах для створення UI тестів та навчитися їх обирати.

**Короткі теоретичні відомості:**

1. **Автоматизоване тестування**

Автоматизоване тестування — це процес використання програмного забезпечення для автоматизації виконання тестів, що дозволяє перевіряти функціональність, продуктивність і безпеку програмних продуктів без участі людини. Основною метою є зменшення людських помилок та підвищення швидкості перевірок, що дозволяє проводити тестування повторно з мінімальними витратами часу та ресурсів.

2. **Інструменти для автоматизованого тестування**

Існує багато інструментів для автоматизованого тестування, серед яких:

* **Selenium** — популярний фреймворк для тестування веб-застосунків через браузер. Дозволяє автоматизувати взаємодію з веб-сторінками, що містять елементи форм, кнопок, списків тощо.
* **Puppeteer** — бібліотека для Node.js, що надає API для керування браузером Chrome або Chromium. Вона використовує DevTools протокол для автоматизації тестів.
* **Cypress** — сучасний інструмент для автоматизованого тестування веб-застосунків, що підтримує швидкий запуск тестів і інтерактивний режим.

### 3. **Типи тестування**

* **UI тести (інтерфейсу користувача)** — перевіряють правильність роботи елементів інтерфейсу, таких як кнопки, поля вводу, меню, а також взаємодію користувача з ними.
* **Функціональне тестування** — перевіряє, чи працюють функції веб-застосунку відповідно до вимог.
* **Тестування продуктивності** — вимірює, як веб-застосунок працює під навантаженням.
* **Тестування сумісності** — перевіряє, чи працює застосунок коректно на різних браузерах та пристроях.

**Виконання:**

Для виконання лабораторної роботи був використаний популярний сайт для тестування: <https://magento.softwaretestingboard.com>, тож напишу декілька коротких тестів:

const { launch } = require('puppeteer');

describe('UI Tests for E-Commerce Website', () => {

    let browser;

    let page;

    beforeAll(async () => {

        browser = await launch({ headless: false });

        page = await browser.newPage();

        await page.goto('https://magento.softwaretestingboard.com/men/tops-men/jackets-men.html');

    });

    afterAll(async () => {

        await browser.close();

    });

    test('Page title should be correct', async () => {

        const title = await page.title();

        expect(title).toBe('Jackets - Tops - Men');

    });

    test('Product grid should be visible', async () => {

        const productGrid = await page.$('.products-grid');

        expect(productGrid).not.toBeNull();

    });

    test('There should be products listed', async () => {

        const products = await page.$$('.products-grid .product-item');

        expect(products.length).toBeGreaterThan(0);

    });

    test('Sorting functionality should work', async () => {

        await page.select('.sorter-options', 'price');

        await page.waitForSelector('.price');

        const prices = await page.$$eval('.price', elements =>

          elements.map(el => parseFloat(el.textContent.replace('$', '')))

        );

      });

      test('Filtering by size should work', async () => {

        await page.click('.filter-options-title');

        await page.waitForSelector('div[aria-label="M"]', { visible: true });

        await page.click('div[aria-label="M"]');

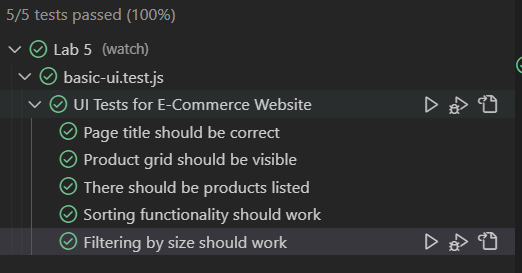
        await page.waitForSelector('.products-grid .product-item');

        const filteredProducts = await page.$$('.products-grid .product-item');

    });

});

**Результат виконання:**



**Висновок:** В результаті виконання лабораторної роботи було розглянуто можливості автоматизованого тестування веб-застосунку за допомогою Puppeteer. Успішно реалізовані тести підтвердили коректну роботу функцій веб-застосунку, що свідчить про їхню стабільність та відповідність вимогам.

**Додаткові рекомендації:**

* Розширити набір тестів для перевірки мобільної версії сайту.
* Додати тести на перевірку взаємодії з кошиком.
* Використовувати CI/CD для автоматичного запуску тестів після внесення змін у код.

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