Міністерство освіти й науки України

Івано-Франківський національний технічний

університет нафти і газу

Кафедри ІПЗ

Лабораторна робота №8

# Тема: «ОГЛЯД СУЧАСНОГО ПОПУЛЯРНОГО АВТОМАТИЗОВАНОГО ІНСТРУМЕНТАРІЮ»

Виконав студент

Групи ІП-21-2

Кавецький О.О

Перевірив

Храбатин Р. І.

м. Івано-Франківськ

2025р.

**Мета:** зрозуміти та навчитися будувати прості базові автоматизовані рішення.

**Короткі теоретичні відомості:**

### 1. **Jest**

**Тип тестування**: Юніт-тести, інтеграційні тести, тестування компонентів у JavaScript/TypeScript. **Опис**: Jest — це популярний фреймворк для тестування JavaScript, розроблений Facebook. Він має багато вбудованих можливостей, таких як мокінг, тайм-аут, спостереження за змінами, автоматичне створення звітів про тести, асинхронні тести та інше. **Переваги**:

* Легка інтеграція з React.
* Вбудовані функції для мокінгу і асинхронних тестів.
* Широко підтримується спільнотою.

### 2. **Mocha**

**Тип тестування**: Юніт-тести, інтеграційні тести, тести API. **Опис**: Mocha — це фреймворк для тестування JavaScript, який дозволяє використовувати різні бібліотеки для асерцій та мокінгу (наприклад, Chai для асерцій). Mocha гнучкий і дозволяє налаштовувати багато аспектів тестування. **Переваги**:

* Підтримує багато різних плагінів і бібліотек.
* Дуже гнучкий.
* Добре працює для тестування Node.js програм.

### 3. **Cypress**

**Тип тестування**: UI/E2E тестування. **Опис**: Cypress — це інструмент для тестування інтерфейсу користувача та E2E тестування. Він працює безпосередньо в браузері, що дозволяє візуально відслідковувати виконання тестів у реальному часі. **Переваги**:

* Відображення тестів у реальному часі.
* Вбудовані функції для асинхронного тестування.
* Дуже швидке і надійне для тестування браузера.

### 4. **Playwright**

**Тип тестування**: UI/E2E тестування. **Опис**: Playwright — це фреймворк для автоматизації браузера від Microsoft. Він підтримує всі основні браузери (Chromium, Firefox, WebKit) і дозволяє писати тести для веб-додатків. **Переваги**:

* Підтримка багатьох браузерів (Chromium, Firefox, WebKit).
* Підтримка паралельного виконання тестів.
* Вбудовані функції для тестування мобільних додатків.

### 5. **Puppeteer**

**Тип тестування**: Автоматизація браузера (UI тести, скрейпінг). **Опис**: Puppeteer — це бібліотека для Node.js, яка надає API для управління браузером Chromium. Puppeteer найкраще підходить для автоматизації браузерних задач та тестування, а також для скрейпінгу. **Переваги**:

* Легке налаштування для автоматизації браузера.
* Добре підходить для створення скриншотів і запису відео.
* Підтримує глибоке управління браузером.

### 6. **Selenium**

**Тип тестування**: UI/E2E тестування. **Опис**: Selenium — це один з найстаріших і найпопулярніших інструментів для автоматизованого тестування веб-додатків. Він дозволяє писати тести для браузерів на різних мовах програмування, таких як Java, Python, C# і JavaScript. **Переваги**:

* Підтримка великої кількості мов програмування.
* Може працювати з усіма основними браузерами.
* Широко використовуваний і добре задокументований.

### 7. **Appium**

**Тип тестування**: Тестування мобільних додатків. **Опис**: Appium — це інструмент для автоматизації тестів для мобільних додатків на платформі Android і iOS. Він підтримує багатоплатформенність і дозволяє писати тести на різних мовах програмування. **Переваги**:

* Підтримує мобільні додатки для Android і iOS.
* Підтримка множинних мов програмування.
* Зручний для тестування гібридних та нативних мобільних додатків.

### 8. **Jasmine**

**Тип тестування**: Юніт-тести, тести API. **Опис**: Jasmine — це фреймворк для тестування JavaScript, який має вбудовану підтримку асерцій і мока. Він використовує BDD (Behavior Driven Development) стиль тестів. **Переваги**:

* Простий і зручний для написання тестів.
* Підтримка асинхронних тестів.
* Підходить для тестування на фронтенді та сервері.

### 9. **Karma**

**Тип тестування**: Юніт-тести, інтеграційні тести. **Опис**: Karma — це тестовий сервер для запуску тестів у реальних браузерах. Він часто використовується з іншими фреймворками, такими як Jasmine, Mocha або QUnit. **Переваги**:

* Підтримка запуску тестів у реальних браузерах.
* Легко інтегрується з CI/CD системами.
* Дозволяє виконувати тести у різних браузерах паралельно.

### 10. **TestCafe**

**Тип тестування**: UI/E2E тестування. **Опис**: TestCafe — це фреймворк для автоматизованого тестування веб-додатків, який підтримує тестування в різних браузерах без необхідності мати додаткові драйвери. **Переваги**:

* Простота налаштування і використання.
* Не потребує браузерних драйверів, як Selenium.
* Може бути використаний для тестування мобільних браузерів.

**Виконання:**

**Playwright** — значно зручніший для початку роботи з UI-тестуванням.

**Чому:** • Має все вбудовано: автоматичне очікування елементів, перевірки (assertions), підтримка кількох браузерів (Chrome, Firefox, WebKit). • Має вбудовані можливості для виконання тестів у кількох браузерах, що дає змогу проводити кросбраузерне тестування. • Має розширений API, який дозволяє автоматизувати складні сценарії, наприклад, взаємодію з iframe або пристроями з різними екранами. • Зручна інтеграція з CI/CD.

**Puppeteer** — це гнучкий інструмент для автоматизації браузера, але не тестовий фреймворк сам по собі.  
Тобто з ним потрібно: • Самостійно підключати бібліотеки перевірок (наприклад, Jest або Mocha). • Більше кодувати вручну для простих речей (типу “дочекатися елемента” чи “натиснути”).

Мені більше сподобався **Playwright** як інструмент для тестування, оскільки: • Це тестовий фреймворк із коробки, розроблений спеціально для UI тестів і кросбраузерного тестування. • Простий для навчання і демонстрації в лабораторній роботі. • Має дуже хорошу документацію та спільноту, що спростить написання тестів для багатьох сценаріїв. • Підтримує велику кількість браузерів і мобільних пристроїв, що зручно для автоматизації кросбраузерного тестування.

**Puppeteer** можна використати як другий інструмент — для порівняння і демонстрації гнучкості, але він більше підходить для: • Автоматизації скрейпінгу (збирання даних з веб-сторінок). • Кастомних сценаріїв, де потрібен детальний контроль над браузером, наприклад, для створення скріншотів, запису відео чи автоматизації специфічних задач браузера.

Це дозволить більш чітко розуміти, чому для UI тестування **Playwright** є більш зручним вибором, але **Puppeteer** може бути корисним для специфічних завдань.

|  |  |  |  |
| --- | --- | --- | --- |
| Інструмент | Сильна сторона | Слабка сторона | Типові задачі |
| Puppeteer | |  | | --- | | Високий контроль над Chromium, простий у налаштуванні, стабільний API |  |  | | --- | |  | | |  | | --- | | Працює тільки з Chrome/Chromium (офіційно), менша кросбраузерна підтримка |  |  | | --- | |  | | |  | | --- | | UI тести, PDF/скріншоти, веб-скрейпінг, автоматизація браузера |  |  | | --- | |  | |
| Playwright | |  | | --- | | Підтримка кількох браузерів (Chrome, Firefox, WebKit), автоочікування, багатомовність |  |  | | --- | |  | | |  | | --- | | Вищий поріг входу порівняно з Puppeteer, більший розмір бібліотеки |  |  | | --- | |  | | UI тести, кросбраузерне тестування, інтеграція в CI/CD, складна веб-автоматизація |

Код до тестів знаходиться в репозиторії.

**Висновок:** В даній роботі ми зрозуміли та навчилися будувати прості базові автоматизовані рішення.

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