Ольга Стецюк, Ігор Крошний*
[1] НЛТУ України, Львів, Україна, olga.stetsyuk@nltu.lviv.ua
[2] НЛТУ України, Львів, Україна. ORCID: 0000-0003-0018-7998,
kroshny.igor@nltu.edu.ua

Фреймворк автоматизованого тестування вебзастосунків з використанням засобів Cypress

Анотація. *У роботі представлено підхід до розроблення фреймворку автоматизованого тестування вебзастосунків із використанням засобів Cypress та інтегрованого AI-модуля генерації тестів. Фреймворк реалізує архітектуру на основі Page Object Model, підтримує централізовані конфігурації, фікстури тестових даних і гнучку систему звітності (Mochawesome). AI-модуль дозволяє автоматично створювати тести за текстовим описом користувача, що значно скорочує час написання сценаріїв і підвищує ефективність тестування.*

Ключові слова – *Cypress, автоматизоване тестування, AI-генерація тестів, Page Object Model, Mochawesome.*

Вступ. Автоматизоване тестування відіграє ключову роль у забезпеченні стабільності та якості сучасних вебзастосунків. Воно дозволяє швидко перевіряти працездатність основного функціоналу після змін у коді, зменшуючи ризик людських помилок і прискорюючи випуск оновлень. Одним із найефективніших інструментів для End–to–end тестування є Cypress – фреймворк, що поєднує простоту написання тестів із потужними можливостями роботи безпосередньо у браузері, забезпечуючи реалістичне виконання сценаріїв та зручні засоби налагодження.

У межах цієї роботи створено фреймворк автоматизованого тестування вебзастосунків на основі Cypress, який реалізує архітектуру Page Object Model, підтримку тегованих сценаріїв, централізовані конфігурації та звітність через Mochawesome. Особливістю розробленого рішення є інтеграція AI-модуля генерації тестів, що дозволяє створювати тестові сценарії за текстовим описом користувача. Такий підхід значно скорочує час написання тестів, підвищує ефективність автоматизації та робить фреймворк універсальним інструментом для тестування різних типів вебзастосунків [1].

* науковий керівник

Внутрішня архітектура та ключові механізми фреймворку Cypress. Ключовим елементом роботи Cypress є його тісна інтеграція з браузером, що дозволяє виконувати тести безпосередньо у середовищі реального вебзастосунку. Завдяки цьому фреймворк має прямий доступ до DOM, мережевих запитів і консолі браузера, що забезпечує точність перевірок і швидкий зворотний зв'язок [2].

Архітектура Cypress складається з двох частин: Node.js–сервера, який керує тестовими процесами, та браузерного середовища, де виконується код застосунку. Між ними встановлюється двосторонній канал зв'язку, через який передаються команди, події та результати. Завдяки внутрішньому циклу обробки подій Cypress автоматично повторює дії, доки умови не будуть виконані, усуваючи потребу у ручних затримках чи додаткових очікуваннях [3].

Особливістю розробленого фреймворку є вбудований AI–модуль генерації тестів, який застосовує моделі штучного інтелекту для створення тестових сценаріїв за текстовим описом користувача. Цей механізм автоматично формує структуру тесту, підбирає необхідні кроки взаємодії з елементами та зберігає готовий файл у проєкті. Таким чином поєднання Cypress із AI–підходом значно спрощує процес створення тестів, робить його швидшим, інтелектуальнішим і доступним навіть для користувачів без глибоких технічних знань.

Програмна реалізація. Розроблений фреймворк побудовано на основі Cypress із використанням архітектури Page Object Model (POM). Такий підхід дозволяє розділити логіку тестів і елементи сторінок, що спрощує підтримку коду та його повторне використання [4]. Структура проєкту (рис. 1) включає каталоги з тестами, фікстурами тестових даних, сторінками об'єктів і службовими модулями.

У межах реалізації створено AI-модуль генерації тестів, який дає змогу користувачу формувати нові тестові сценарії за допомогою текстового опису (рис. 2). Після введення сценарію система надсилає запит до модуля штучного інтелекту, отримує згенерований код тесту й автоматично додає його до катало-

* науковий керівник

гу cypress/e2e/ai. Це значно прискорює процес створення тестів і знижує потребу в ручному програмуванні.

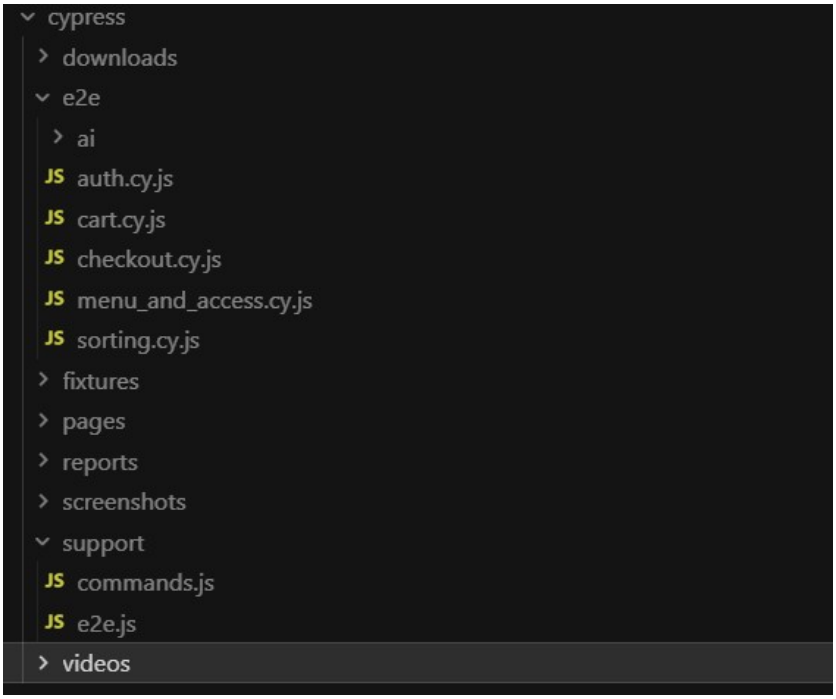

Рисунок 1. Структура розробленого фреймворку

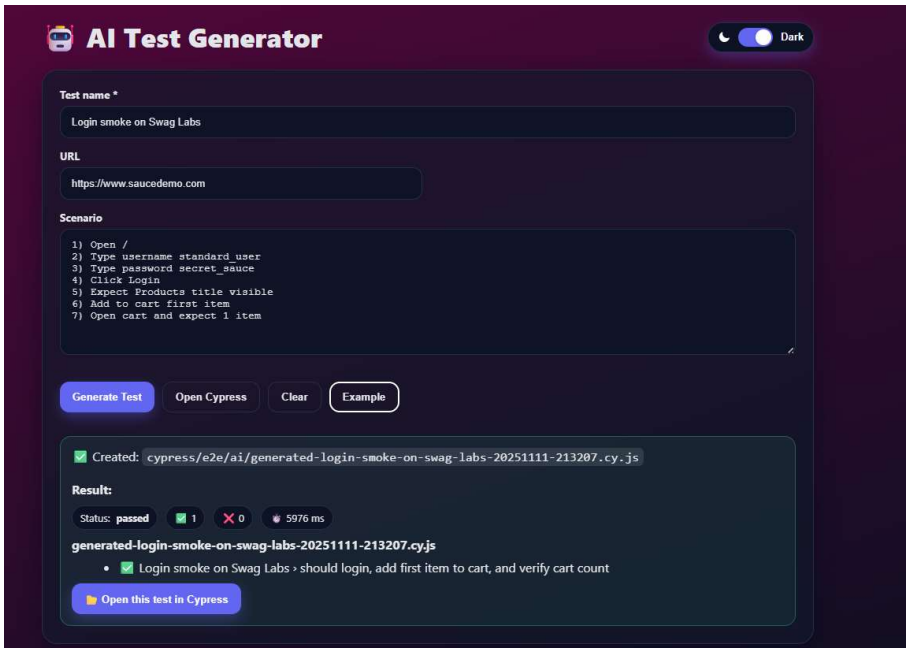

Рисунок 2. Реалізація AI-модуля для генерації тестів

Фреймворк було випробувано на демо-застосунку Swag Labs, де виконано серію функціональних і smoke-тестів (рис. 3). Результати показали стабільне проходження всіх сценаріїв, автоматичне створення звітів (Mochawesome) та відео-записів виконання. Тестові набори виконувались у середньому за 1–2 хвилини, що підтверджує ефективність і швидкодію розробленого рішення.

* науковий керівник

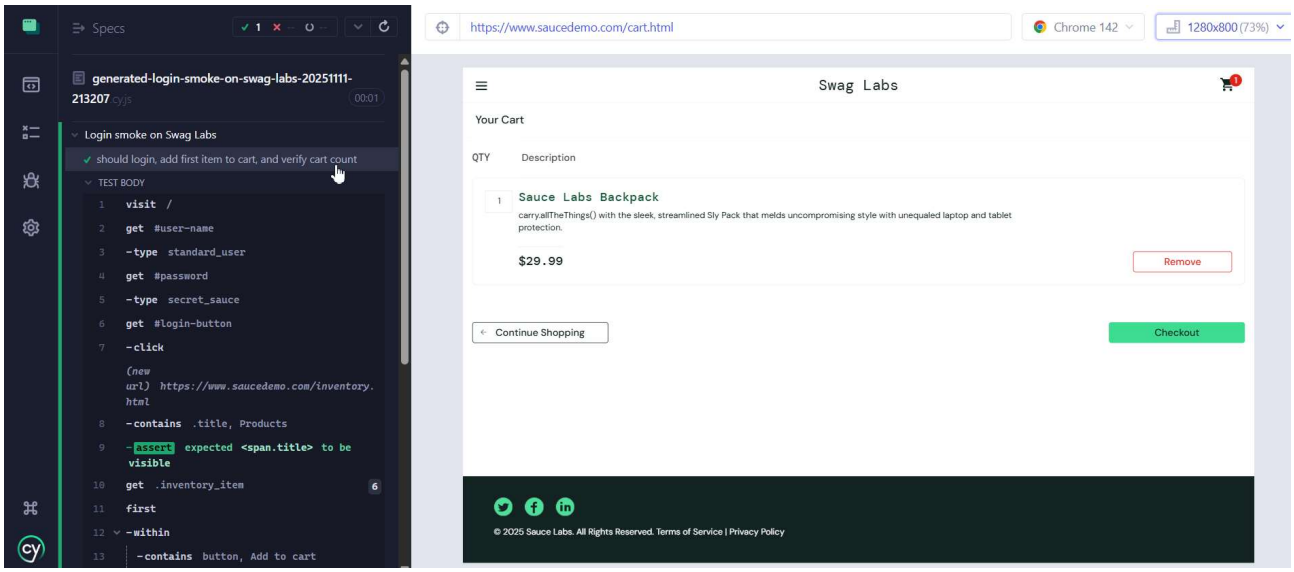

Рисунок 3. Приклад виконання автоматизованих тестів на демо-застосунку Swag Labs

Висновок. У результаті роботи розроблено фреймворк автоматизованого тестування вебзастосунків із використанням засобів Cypress та вбудованого AI–модуля генерації тестів. Реалізована архітектура на основі Page Object Model забезпечує зручність супроводу та масштабування тестів, а використання фікстур і звітності підвищує прозорість результатів перевірок.

AI–модуль довів свою ефективність у прискоренні процесу створення тестів і зниженні рутинного навантаження на тестувальника. Проведені випробування підтвердили стабільність, швидкість і практичну придатність фреймворку для застосування в автоматизованому тестуванні вебзастосунків.

Список використаних літературних джерел

[1] Cypress testing solution [Електронний ресурс]. URL: https://docs.cypress.io/app/get-started/why-cypress

[2] Introduction to Cypress Testing Framework [Електронний ресурс]. URL: https://www.geeksforgeeks.org/software-testing/introduction-to-cypress-testing-framework/

[3] Cypress For Test Automation: A Step-by-Step Guide [Електронний ресурс]. URL: https://testgrid.io/blog/cypress-testing/;

[4] Vasiljevic S. Mastering Cypress: End-to-End Testing for Web Applications [Текст]. – United States : Independently published, 2023. – 312 с.

* науковий керівник

Framework for Automated Web Application Testing Using Cypress Tools

Olha Stetsiuk, Igor Kroshny

This paper presents an approach to developing a framework for automated web application testing using Cypress and an integrated AI–based test generation module. The framework is built on the Page Object Model architecture and supports centralized configurations, test data fixtures, and a flexible reporting system (Mochawesome). The AI module enables automatic creation of test cases based on a user's text description, significantly reducing test development time and increasing the efficiency of the testing process.

* науковий керівник