DOI: https://doi.org/10.36910/6775-2524-0560-2025-59-20
УДК: 004.415
Краснокутська Інесса Володимирівна, к. ф.-м.н., доцент
https://orcid.org/0000-0002-7034-7291
Дутчак Ольга Олексіївна, студентка
https://orcid.org/0009-0006-2941-8648
Чернівецький національний університет імені Юрія Федьковича, м. Чернівці, Україна

АВТОМАТИЗОВАНЕ ТЕСТУВАННЯ САЙТУ ФАКУЛЬТЕТУ З ВИКОРИСТАННЯМ CYPRESS JS ТА ІНТЕГРУВАННЯ BDD ФРЕЙМВОРКУ CUCUMBER

Краснокутська І. В., Дутчак О. О. Автоматизоване тестування сайту факультету з використанням Cypress JS та інтегрування BDD фреймворку Cucumber. У статті розглянута методологія Behavior Driven Development (BDD) в контексті наскрізного тестування веб-додатків за допомогою фреймворку Cypress. Описано суть цієї методології, а також основні переваги її застосування при автоматизації тестування. Зокрема, підкреслюється як інтеграція Cypress з BDD дозволяє створювати тести, що відображають реальні користувацькі сценарії, використовуючи природну мову. Особливу увагу приділено використанню Cucumber, який забезпечує можливість написання тестових сценаріїв у форматі Gherkin. Це значно покращує комунікацію між розробниками, тестувальниками та бізнес-стейкхолдерами, оскільки всі учасники процесу можуть легко зрозуміти вимоги та сценарії тестування, викладені простою й доступною мовою. У статті також обґрунтовано, чому використання Cypress у поєднанні з BDD забезпечує високу ефективність тестування веб-додатків, дозволяючи знизити ризики помилок і підвищити якість кінцевого продукту.
Ключові слова: автоматизоване тестування, E2E тести, Cypress, BDD, Cucumber, Gherkin.

Krasnokutska I., Dutchak O. Automated Testing of a Faculty Website Using Cypress JS and Integration with the BDD Framework Cucumber. The article explores the methodology of Behavior Driven Development (BDD) in the context of end-to-end testing of web applications using the Cypress framework. It discusses the essence of this methodology and the key advantages of its application in test automation. In particular, the article highlights how the integration of Cypress with BDD allows the creation of tests that reflect real user scenarios using natural language. Special attention is given to the use of Cucumber, which enables writing test scenarios in the Gherkin format. This significantly improves communication between developers, testers, and business stakeholders, as it ensures a shared understanding of requirements and test scenarios written in a language that is clear to everyone. The article also justifies why the use of Cypress in combination with BDD ensures high testing efficiency for web applications, reducing the risk of errors and improving the quality of the final product.
Keywords: automated testing, E2E tests, Cypress, BDD, Cucumber, Gherkin.

Постановка наукової проблеми. У сучасному програмному забезпеченні важливим аспектом є забезпечення його стабільності, функціональності та відповідності очікуванням користувачів. Збільшення складності програмних продуктів, а також швидкість змін у вимогах та технологіях потребують від розробників більш ефективних підходів до тестування. Автоматизоване тестування є потужним інструментом для зменшення ризиків помилок, покращення якості продукту та прискорення процесу розробки. Воно дозволяє не тільки автоматизувати повторювані перевірки, але й значно знизити ймовірність людських помилок, що виникають під час тестування вручну.

Одним із найефективніших підходів до перевірки роботи веб-додатків є наскрізне (E2E) тестування, яке забезпечує перевірку всієї системи в умовах, максимально наближених до реальних сценаріїв використання. Це тестування дозволяє виявити не тільки локальні помилки в коді, але й проблеми взаємодії різних компонентів системи, що є важливим аспектом для веб-додатків, які часто мають складну архітектуру та безліч залежностей між мікросервісами, базами даних та іншими зовнішніми системами. Застосування E2E тестування дає змогу розробникам і тестувальникам виявляти і виправляти помилки на ранніх етапах розробки, що в кінцевому результаті підвищує стабільність і якість кінцевого продукту.

Проте попри поширення E2E тестування на практиці, існує відносно небагато наукових досліджень і публікацій, що висвітлюють особливості застосування сучасних інструментів для такого тестування, зокрема у контексті інтеграції засобів типу Cypress із підходами, орієнтованими на поведінку (BDD), такими як Cucumber. Це створює потребу в детальному аналізі ефективності подібних рішень та розробці практичних рекомендацій щодо їх застосування. Саме це і визначає актуальність дослідження, спрямованого на вивчення можливостей наскрізного тестування з використанням сучасних інструментів в умовах реального проєктування веб-застосунків.

Аналіз досліджень. У наукових статтях спостерігається зростаючий інтерес до автоматизованого E2E-тестування як ключового підходу до перевірки складних веб-додатків. Дослідники відзначають, що на відміну від модульного чи інтеграційного тестування, E2E дозволяє охопити всю систему цілісно, перевіряючи роботу компонентів у реальному середовищі [1, 2, 6].

Серед популярних інструментів особливу увагу привертає Cypress, який завдяки зручному API, високій продуктивності та можливостям візуалізації тестів отримав широке визнання в спільноті. Важливою перевагою Cypress є підтримка інтеграції з підходом Behavior Driven Development (BDD), що забезпечує написання тестів у форматі Gherkin — мовою, зрозумілою як технічним, так і нетехнічним фахівцям. Така інтеграція, зокрема за допомогою Cucumber, сприяє кращій комунікації в командах і підвищує прозорість процесу тестування [3, 4, 5]. Додатково підкреслюється доцільність використання E2E-тестування в рамках CI/CD-процесів, де воно дозволяє швидко виявляти регресії та знижувати ризики на ранніх етапах розробки [7].

Невирішені аспекти проблеми. Попри значну увагу до автоматизованого E2E-тестування, недостатньо дослідженим залишається питання формалізації та оптимізації структури тестів у межах Cypress з урахуванням принципів повторного використання коду, масштабованості та зручності супроводу. Особливо актуальним є впровадження підходу Page Object Model у поєднанні з BDD, що дозволяє підвищити ефективність написання й підтримки E2E-тестів.

Мета дослідження. Метою даного дослідження є аналіз та реалізація практичної моделі організації E2E-тестів у середовищі Cypress із використанням Page Object Model та підходу BDD з акцентом на структурованість, повторне використання компонентів та покращення комунікації між розробниками й аналітиками.

Основна частина. Швидкий прогрес технологій став ключовим фактором у розвитку сучасного світу, особливо в галузі веб-розробки. Інтернет, що колись слугував лише джерелом інформації, сьогодні є потужною платформою для комунікації, ведення бізнесу та освіти. Тому у цифрову епоху ефективність та функціональність веб-ресурсів вищих навчальних закладів стають дедалі важливішими.

З метою забезпечення високої якості сайту факультету математики та інформатики Чернівецького національного університету імені Юрія Федьковича (https://fmi.chnu.edu.ua/), важливо здійснювати регулярне тестування. Це забезпечить зручний доступ до важливої інформації для всіх користувачів сайту, включаючи навчальний та адміністративний персонал, а також сприятиме формуванню позитивного іміджу факультету в онлайн-просторі. Крім того, постійний контроль за роботою сайту дозволяє оперативно виявляти та усувати технічні проблеми, що можуть вплинути на користувацький досвід. Регулярне тестування також сприяє удосконаленню ресурсу, дозволяючи впроваджувати нові функції та підвищувати його зручність і ефективність.

Сайт факультету надає широкий набір функцій, зокрема доступ до розкладу занять, навчальних матеріалів, контактів викладацького складу та адміністрації. Також він містить інтегровані засоби комунікації, зокрема електронну пошту та онлайн-платформи. Однак під час користування можуть виникати певні труднощі, такі як повільне завантаження сторінок, неочікувані помилки або незручна навігація. Саме тому наскрізне (E2E) тестування відіграє ключову роль у забезпеченні стабільності та коректної роботи сайту.

Наскрізне або E2E тестування дозволяє перевіряти роботу всього додатка від початку до кінця, імітуючи реальні сценарії використання. Це дає змогу не лише виявляти помилки в окремих модулях, а й перевіряти коректну взаємодію між різними компонентами системи, що є особливо важливим для сучасних веб-додатків [2].

Основна перевага наскрізного тестування полягає в його здатності перевіряти функціональність додатка в умовах, максимально наближених до реального використання. На відміну від модульного тестування, яке зосереджується на окремих частинах коду, наскрізні тести охоплюють весь процес – від взаємодії користувача з інтерфейсом до обробки запитів на сервері. Це допомагає забезпечити стабільність роботи сайту та покращити користувацький досвід (Рис. 1).

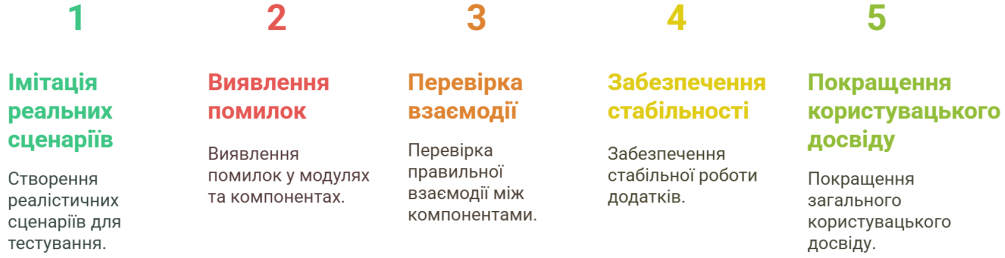

Рис. 1. Переваги наскрізного тестування

Одним із найефективніших інструментів для автоматизованого тестування є Cypress. Цей фреймворк є потужною платформою для наскрізного тестування веб-додатків з відкритим кодом, що використовує JavaScript і підтримує операційні системи Windows, Linux і macOS. Його основними перевагами є простота у використанні, швидке виконання тестів і можливість їх перевірки в режимі реального часу, що значно полегшує процес пошуку та виправлення помилок. А завдяки своїй орієнтації на потреби фронтенд-розробників Cypress став одним із найкращих інструментів для тестування. Його можна використовувати навіть у випадках, коли бекенд ще не повністю готовий, що значно спрощує процес розробки та перевірки функціональності.

З метою подальшого підвищення ефективності тестування та оптимізації комунікації між учасниками команди доцільним є впровадження у Cypress фреймворку Cucumber, що реалізує підхід поведінково-орієнтованої розробки (BDD). Cucumber дозволяє створювати тести природною мовою, що робить їх зрозумілими не лише розробникам, а й бізнес-стейкхолдерам. Суть цього підходу ілюструється нижче (Рис. 2).

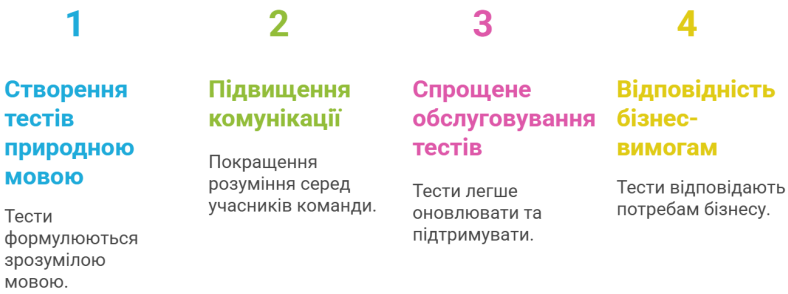

Рис. 2. Переваги BDD-фреймворку Cucumber

Структура тестового проєкту передбачала використання папки E2E з двома основними підпапками: *tests* для зберігання тестових файлів та *pages* для реалізації локаторів і взаємодії з елементами сторінок (Рис. 3). Ця організація базувалася на принципах шаблону Page Object Model, що спрощувало написання й підтримку тестів, а також сприяло їх масштабованості та повторному використанню [3]. Однак для подальшого вдосконалення процесу автоматизованого тестування було вирішено впровадити фреймворк Cucumber, що дозволило ще більше підвищити ефективність тестування та забезпечити кращу інтеграцію з іншими частинами проєкту.

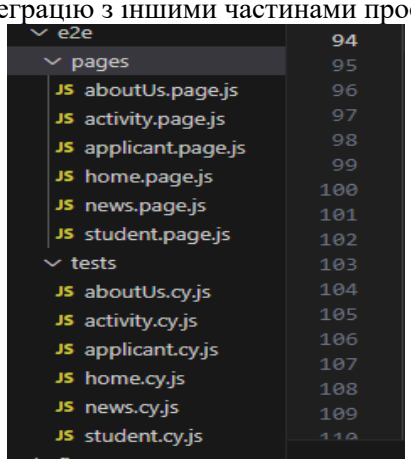

Рис. 3. Підхід Page Object Model

У процесі побудови автоматизованих тестів, зокрема з використанням фреймворку Cypress, застосовується структурний підхід, що ґрунтується на усталених принципах модульного та поведінково-орієнтованого тестування [4]. Конструкція *describe* використовується для визначення загального контексту тестування, групування тестових сценаріїв за функціональним призначенням та забезпечення логічної організації тестів. Такий підхід сприяє покращенню структурованості коду та підвищує його читабельність.

Блок *beforeEach* виконує ініціалізаційні дії, необхідні для підготовки тестового середовища перед виконанням кожного окремого тесту. Це дозволяє гарантувати сталість початкових умов і запобігти взаємному впливу тестів, що є необхідною умовою для забезпечення достовірності та відтворюваності результатів.

Конструкція *it* застосовується для опису конкретних тестових випадків, кожен з яких

перевіряє окремий аспект поведінки системи. Така деталізація дозволяє локалізовано тестувати функціональність, спрощує аналіз результатів і полегшує подальшу підтримку тестової бази. Загалом, така структура відповідає принципам логічної ієрархії, що є характерною рисою якісно побудованих E2E тестів.

Із впровадженням підходу Behavior Driven Development (BDD) та інтеграцією фреймворку Cucumber у Cypress, структура тестового проєкту зазнала розширення та формалізації. Вона доповнилась спеціалізованими папками, такими як *features* та *step_definitions*, що дозволило реалізувати тестування у формі природномовних сценаріїв, які легше інтерпретуються всіма учасниками розробки. У папці *features* зберігаються файли тестових сценаріїв, описаних за допомогою мови Gherkin. Мова Gherkin дозволяє сформулювати тести в природній мові, що робить їх зрозумілими не лише для розробників, але й для інших учасників проєкту, таких як менеджери або тестувальники. Файли, що містяться у цій папці, включають опис тестових сценаріїв, які моделюють очікувану поведінку системи в типових користувацьких ситуаціях, як наприклад, проілюстровано на рис. 4.

```
Scenario: VALIDATE THE SEARCH FUNCTIONALITY WITH NO RESULTS
  When I click on the search button one time
  Then the search field should be visible
  And I type "Невідомий текст" into the search field
  And the URL should contain "https://fmi.chnu.edu.ua/poshuk/?query="
  And I should see "Нічого не знайдено."
```

Рис. 4. Сценарій валідації поля пошуку

У папці *steps* розміщуються відповідні файли з реалізацією кроків, що відповідають сценаріям, описаним у форматі Gherkin. Такий підхід забезпечує чітке розмежування між формальним описом поведінки системи та її технічною реалізацією, що сприяє покращенню супроводжуваності та масштабованості тестів. Кожен крок описується у вигляді окремої функції, яка виконує конкретну дію або перевірку відповідно до логіки тестового сценарію (Рис. 5).

```
261  // VALIDATE THE SEARCH FUNCTIONALITY WITH NO RESULTS
262  When("I click on the search button one time", () => {
263    homePage.clickSearchButton();
264  });
265
266  Then("the search field should be visible", () => {
267    homePage.searchwrp.should("have.class", "show");
268    homePage.searchinput.should("be.visible");
269  });
270
271  Then("I type {string} into the search field", (query) => {
272    homePage.typeSearchTermForSearch(query);
273  });
274
275  Then("the URL should contain {string}", (expectedUrl) => {
276    cy.url().should("include", expectedUrl);
277  });
278
279  Then("I should see {string}", (text) => {
280    cy.contains(text).should("exist");
281  });
```

Рис. 5. Реалізація кроків сценарію

Умови прийняття тесту детально описані в таблиці нижче (Таб.1). У ній наведено всі критерії, яким має відповідати тест для успішного проходження.

Таблиця 1. Критерії прийняття

Крок	Очікуваний результат	Фактичний результат	Статус
Натискання на кнопку пошуку один раз	Поле пошуку стає видимим	Поле пошуку стало видимим	Успішно
Введення "Невідомий текст" у поле пошуку	Текст "Невідомий текст" введено у поле пошуку	Текст успішно введено	Успішно

Перевірка URL після введення запиту	URL містить https://fmi.chnu.edu.ua/poshuk/?query=	URL містить очікуваний параметр	Успішно
Перевірка відображення повідомлення про відсутність результатів	Відображається текст "Нічого не знайдено."	Текст успішно відображається	Успішно

У процесі дослідження було здійснено всебічне тестування веб-сайту факультету математики та інформатики Чернівецького національного університету імені Юрія Федьковича, що дозволило оцінити стабільність його функціонування, виявити потенційні проблеми та запропонувати шляхи їх усунення. Для цього використовувалася методологія наскрізного (E2E) тестування, що є одним із найбільш ефективних підходів до автоматизованої перевірки веб-додатків. Основним інструментом тестування став Cypress, який забезпечує швидке виконання тестів у реальних умовах браузерного середовища та дозволяє ефективно взаємодіяти з елементами сторінки.

В ході тестування було створено структуру тестового проєкту, що базується на принципах Page Object Model. Це дало змогу підвищити організованість тестового коду, зменшити його дублювання та спростити подальше супроводження. В рамках цієї архітектури у папці E2E було створено дві основні директорії: *tests* для зберігання тестових сценаріїв та *pages* для управління локаторами та методами взаємодії з елементами веб-сторінок. Завдяки такому підходу тести стали більш структурованими, що сприяло їх зручності та повторному використанню.

Крім традиційного E2E тестування, у проєкті також було реалізовано методологію Behavior Driven Development (BDD) [5] з використанням Cucumber [6]. Цей підхід дозволив писати тестові сценарії у форматі Gherkin, що зробило їх більш зрозумілими навіть для тих учасників команди, які не мають глибоких технічних знань. У структурі тестового середовища було створено окрему папку *features*, що містить файли сценаріїв у форматі *.feature*, де тести описуються у вигляді читабельних поведінкових сценаріїв, використовуючи ключові слова Given, When, Then. Для реалізації відповідних кроків тестів була створена папка *step_definitions*, де кожен сценарій отримав свою логічну реалізацію на рівні коду. Реалізацію всієї тестової інфраструктури можна переглянути у GitHub-репозиторії https://github.com/olhadutchak/fmitestcypress.

Тестування надало можливість оцінити функціональні можливості сайту та рівень зручності його використання для кінцевих користувачів. У процесі перевірки було проаналізовано ефективність навігації, доступність ключової інформації та логіку структури веб ресурсу. Особлива увага приділялася роботі пошукової системи: виконання пошукових запитів дозволило переконатися в її працездатності, а також у відповідності отриманих результатів заданим ключовим словам. Такий підхід забезпечив комплексне уявлення про якість реалізованих механізмів та їхню відповідність очікуванням користувачів. За підсумками тестування сформовано базу для подальшого аналізу й удосконалення функціоналу сайту з метою підвищення загального рівня зручності та ефективності користування.

Важливо зазначити, що автоматизоване тестування є безперервним процесом, оскільки усунення одних помилок може призвести до появи інших. Регулярне тестування дозволяє мінімізувати ризики виникнення критичних збоїв та забезпечує стабільність роботи веб-додатків у довгостроковій перспективі. Успішна інтеграція BDD у тестове середовище значно підвищує ефективність командної роботи, оскільки тестові сценарії можуть створюватися та аналізуватися не лише тестувальниками, а й аналітиками та розробниками.

Завдяки комплексному підходу, що поєднує традиційне E2E тестування та методологію BDD, було отримано цілісну картину роботи сайту, що сприяє його подальшій оптимізації та вдосконаленню.

Висновки. У результаті проведеного дослідження було реалізовано прикладну модель організації наскрізного (E2E) тестування з використанням фреймворку Cypress, архітектурного підходу Page Object Model та методології Behavior Driven Development (BDD) у поєднанні з Cucumber. Запропонований підхід дозволив створити структуровану, читабельну та ефективну систему тестування, що забезпечує зручність підтримки, повторне використання коду та зрозуміле

формулювання тестових сценаріїв. Тестування веб-сайту факультету математики та інформатики Чернівецького національного університету імені Юрія Федьковича показало його стабільну роботу, відповідність функціональних можливостей очікуванням користувачів та дотримання сучасних стандартів якості інтерфейсу.

Використання мови Gherkin для опису поведінки системи у формі, близькій до звичайної мови має перспективу значно спростити комунікацію між членами команди, зокрема у міждисциплінарних колективах, де технічні та нетехнічні фахівці спільно формулюють вимоги до функціональності. Інтеграція Gherkin-сценаріїв разом з програмною реалізацією через Cucumber створює передумови для досягнення більшої узгодженості між очікуваною та фактичною поведінкою системи, що у майбутньому може сприяти підвищенню прозорості розробки та кращому взаєморозумінню в команді.

У подальших дослідженнях доцільно зосередитися на таких напрямах, як розширення масштабованості тестових проєктів, інтеграція E2E тестів у CI/CD пайплайни [7] для забезпечення безперервної перевірки якості, а також застосування візуального тестування й автоматизованого аналізу продуктивності. Перспективним є також вивчення можливостей використання штучного інтелекту та машинного навчання для динамічної генерації тестів, адаптивного виявлення нових сценаріїв використання та підвищення ефективності виявлення дефектів. Крім того, подальше дослідження методів тестування доступності (accessibility testing) може посилити інклюзивність веб-продуктів і розширити їх аудиторію.

Таким чином, впроваджений підхід демонструє практичну ефективність і створює надійне підґрунтя для подальших розробок у сфері автоматизованого тестування сучасних веб-застосунків.

Список бібліографічного опису:

1. Jyolsna J., Anuar S. Modern Web Automation with Cypress.Io // Open International Journal of Informatics (OIJI). – 2022. – Vol. 10, No. 2. – Published: 15.12.2022. – DOI: https://doi.org/10.11113/oiji2022.10n2.229
2. Maurizio Leotta, Diego Clerissi, Filippo Ricca, and Paolo Tonella, (2016) "Approaches and Tools for Automated End-to End Web Testing", Advances in Computers 101 (2016), 193–237, Retrieved from: https://www.sciencedirect.com/science/article/abs/pii/S0065245815000686
3. I. V. Krasnokutska and O. S. Krasnokutskyi, (2024) "Implementing E2E tests with Cypress and Page Object Model: evolution of approaches," CEUR Workshop Proceedings, vol. 3662, 101–110, Retrieved from: https://ceur-ws.org/Vol-3662/paper24.pdf.
4. "Cypress Documentation." Cypress.io. Retrieved from: https://docs.cypress.io/
5. Farooq M. S., Omer U., Ramzan A., Rasheed M. A., Atal Z. Behavior Driven Development: A Systematic Literature Review // IEEE Access. – 2019. – Vol. 11. – DOI: 10.1109/ACCESS.2023.3302356
6. Psujek, M., Radzik, A., Kozieł, G.Comparative analysis of solutions used in Automated Testing of Internet Applications // Journal of Computer Sciences Institute. – 2021. – Vol. 18. – P. 7–14. – DOI: http://dx.doi.org/10.35784/jcsi.2373.
7. Kulkarni N. Automated testing as part of CI/CD pipeline – shift left implementation // North American Journal of Engineering Research. – 2020. – Vol. 1, No. 3. – URL: https://najer.org/najer/article/view/62.

References:

1. Jyolsna J., Anuar S. Modern Web Automation with Cypress.Io // Open International Journal of Informatics (OIJI). – 2022. – Vol. 10, No. 2. – Published: 15.12.2022. – DOI: https://doi.org/10.11113/oiji2022.10n2.229
2. Maurizio Leotta, Diego Clerissi, Filippo Ricca, and Paolo Tonella, (2016) "Approaches and Tools for Automated End-to End Web Testing", Advances in Computers 101 (2016), 193–237, Retrieved from: https://www.sciencedirect.com/science/article/abs/pii/S0065245815000686
3. I. V. Krasnokutska and O. S. Krasnokutskyi, (2024) "Implementing E2E tests with Cypress and Page Object Model: evolution of approaches," CEUR Workshop Proceedings, vol. 3662, 101–110, Retrieved from: https://ceur-ws.org/Vol-3662/paper24.pdf
4. "Cypress Documentation." Cypress.io. Retrieved from: https://docs.cypress.io/
5. Farooq M. S., Omer U., Ramzan A., Rasheed M. A., Atal Z. Behavior Driven Development: A Systematic Literature Review // IEEE Access. – 2019. – Vol. 11. – DOI: 10.1109/ACCESS.2023.3302356
6. Psujek, M., Radzik, A., Kozieł, G.Comparative analysis of solutions used in Automated Testing of Internet Applications // Journal of Computer Sciences Institute. – 2021. – Vol. 18. – P. 7–14. – DOI: http://dx.doi.org/10.35784/jcsi.2373.
7. Kulkarni N. Automated testing as part of CI/CD pipeline – shift left implementation // North American Journal of Engineering Research. – 2020. – Vol. 1, No. 3. – URL: https://najer.org/najer/article/view/62.