2. Artificial intelligence and games / G. N. Yannakakis, J. Togelius. Cham: Springer, 2018.

3. Game AI Pro: Collected wisdom of game AI professionals / ed. by S. Rabin. Boca Raton: CRC Press, 2013.

4. AI in modern game design. Analytics Insight. URL: https://www.analyticsinsight.net/ai-in-modern-game-design (дата звернення: 30.10.2025).

УДК 004.738.5

Бєглова А. В., здобувачка 2 курсу
ОС «Магістр» спеціальності
F3 Комп'ютерні науки,
Веселовська Н. Р., д-р техн. наук, професор,
професор кафедри інформаційних технологій

ЗАСТОСУВАННЯ ГТУЧНОГО ІНТЕЛЕКТУ ДЛЯ АВТОМАТИЧНОГО ГЕНЕРУВАННЯ UI-ТЕСТІВ ВЕБДОДАТКІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Стрімке зростання складності вебдодатків формує потребу у масштабованих та ефективних підходах до тестування їх користувацьких інтерфейсів. Традиційні методи створення UI-тестів потребують значних часових та людських ресурсів, що сповільнює розробку та ускладнює підтримку тестових сценаріїв. У таких умовах актуальним стає використання інтелектуальних систем, здатних автоматично генерувати тестові сценарії та адаптувати їх до змін в інтерфейсі.

Приблизно 30–50 % витрат на забезпечення якості у великих проєктах припадає саме на створення та підтримку UI-тестів. Класичні сценарії є крихкими, залежать від структури DOM та потребують постійного оновлення. Штучний інтелект дає змогу автоматизувати ці процеси, зменшити навантаження на команду тестування та підвищити стабільність UI-тестів. Сучасні моделі генерації коду та аналізу інтерфейсів створюють можливість адаптивного тестування без ручного опису сценаріїв.

Метою роботи є дослідити можливості застосування штучного інтелекту для автоматичного генерування UI-тестів вебдодатків та оцінити ефективність такого підходу, порівняно з традиційними методами створення тестових сценаріїв.

Сучасний розвиток інструментів автоматизації тестування призвів до появи спеціалізованих платформ, що використовують штучний інтелект саме для автоматичного генерування UI-тестів. Одним із найвідоміших рішень є Testim.io, який застосовує машинне навчання для аналізу інтерфейсу та автоматичного формування тестових сценаріїв на основі зафіксованих дій користувача. Завдяки алгоритмам ML система здатна самостійно обирати найбільш стабільні селектори та відновлювати тести після змін у DOM, що значно зменшує потребу у ручному супроводі. Інший яскравий приклад – платформа Mabl, яка комбінує аналіз DOM,

поведінкові моделі й комп'ютерний зір. Вона може генерувати UI-тести автоматично, прогнозувати потенційні зміни інтерфейсу, а також розширювати тестове покриття шляхом виявлення важливих користувацьких шляхів.

Помітну роль відіграє і Test.ai – система, що використовує нейронні мережі для розпізнавання елементів інтерфейсу на візуальному рівні. Такий підхід дає змогу платформі створювати тести, подібні до того, як би це робила людина-тестувальник: через «розуміння» зовнішнього вигляду сторінки, а не через прив'язку до селекторів, які часто змінюються. Так само Applitools розвиває напрям Autonomous Testing, що поєднує великі мовні моделі та візуальний аналіз інтерфейсу. Система може автоматично перетворювати текстові описи вимог на робочі UI-тести, формуючи як кроки сценарію, так і очікувані результати перевірок. Подібну логіку використовує і Katalon, де модулі штучного інтелекту допомагають генерувати початкові тестові сценарії на основі записаних дій користувача та оптимізують їх за допомогою автовідновлення селекторів [1].

Окрім повноцінних платформ, у сфері автоматизації набули поширення генератори, що інтегруються з наявними фреймворками та працюють на базі великих мовних моделей. Такі системи здатні автоматично створювати тести з текстових інструкцій або з аналізу HTML-структури сторінки, доповнювати їх відсутніми перевірками, та виявляти непокриті елементи інтерфейсу. Їх поява демонструє зміну парадигми: якщо раніше UI-тести вимагали повністю ручного програмування, то тепер значну частину роботи може виконувати ШІ, залишаючи інженеру роль коректора, а не автора коду.

Найбільш актуальне дослідження, що заслуговує уваги – GenIA-E2ETest. В ньому запропоновано використовувати генеративний ШІ для створення виконуваних end-to-end (E2E) UI-тестів на основі натурально-мовних описів. За результатами авторів: тестові скрипти, згенеровані системою, показали середню «completeness» і «correctness» приблизно 77 %, точність виконання («precision») – 82 %, а «recall» – 85 %. Причому потреба в ручних доробках була невеликою – в середньому лише ~10 % сценаріїв потребували підправлення після генерації. Це демонструє, що сучасні LLM + підходи до генерації можуть значно скоротити час на створення E2E-тестів, зменшити «вхідний поріг» для QA-інженерів і зробити автоматизацію більш доступною навіть для непрофільних користувачів [2].

Ще одним актуальним дослідженням є AutoQALLMs, яке комбінує LLM (наприклад, GPT-4, Claude, Grok) з аналізом HTML/DOM та регулярними виразами для автоматичного створення UI-тестів у Selenium середовищі. Система здатна «з нуля» (zero-shot) – без спеціального навчання чи шаблонів – приймати HTML сторінку, аналізувати її структуру, генерувати Selenium скрипт, а потім «очищати» / покращувати скрипт за допомогою regex-модуля, щоб підвищити стійкість до змін UI.

Згідно з оцінками на понад 30 різних веб-додатків, покриття UI-елементів досягало ~ 96 % (порівняно з ~ 98 % у ручних скриптів), а час генерації – значно менший, ніж написання вручну. Водночас експерти вказували, що хоча скрипти були робочими і їх швидко можна було запустити, вони іноді були надто «загальними» та пропускали логіку специфічних бізнес-валідацій, які зазвичай додає QA вручну. Це підкреслює важливу властивість сучасного AI-підходу: інструменти

створюють основу, але для глибокої перевірки потрібно доповнювати скрипти вручну, особливо коли йдеться про складну бізнес-логіку [3].

Огляд літератури (так званої «grey literature» + академічних праць) показує, що AI-інструменти для тестування (наприклад, mabl, Test.ai, Appvance, Testim та ін.) прийшли як спроба подолати проблеми «звичайної» автоматизації: висока вартість підтримки, громіздкість скриптів, залежність від стабільності UI та DOM, трудомісткість написання тестів. Їх переваги: спрощена генерація тестів (часто без коду), автоматичне оновлення скриптів у разі зміни UI (self-healing), інтеграція з CI/CD, можливість швидкого створення regression-suite навіть не командою QA [4].

Але існують і суттєві обмеження: для складної бізнес-логіки або нетривіальних сценаріїв (наприклад, MFA, динамічний контент, специфічні валідації) AI-згенеровані тести часто недостатні – їх треба доопрацьовувати вручну. Також – ризик «false positives / negatives», складність у підтримці, залежність від стабільності моделей, витрати на виклики AI/LLM.

Отже, впровадження штучного інтелекту в автоматичне генерування UI-тестів вебдодатків істотно підвищує ефективність і масштабованість процесу тестування. AI-орієнтовані інструменти здатні самостійно створювати та адаптувати тестові сценарії, зменшуючи залежність від ручної роботи і підвищуючи стійкість тестів до змін інтерфейсу. Результати сучасних досліджень підтверджують, що такі підходи забезпечують високу точність і скорочують витрати на підтримку автоматизації. Водночас залишаються завдання, які потребують експертного втручання, зокрема у сфері складної бізнес-логіки та нестандартних сценаріїв. Отже, найбільш ефективною стратегією є поєднання AI-генерації з професійним контролем якості з боку інженера.

Список використаних джерел

1. Ricca F., Marchetto A., Stocco A. AI-based test automation: A grey literature analysis. 2021.

2. GenIA-E2ETest: a generative AI-based approach for end-to-end test automation. 2025. URL: https://arxiv.org/abs/2510.01024 (дата звернення 20.11.2025).

3. AutoQALLMs: Automating Web Application Testing Using Large Language Models (LLMs) and Selenium. URL: https://www.mdpi.com/2073-431X/14/11/501 (дата звернення 20.11.2025).

4. Improving web element localization by using a large language model. URL: https://arxiv.org/abs/2310.02046 (дата звернення 20.11.2025).