Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ ТА АНАЛІЗ МЕТОДІВ ВЕБТЕСТУВАННЯ

(тема)

Виконав:
студент 2 курсу, групи ІНФм-23-1
Шаповалов В.В.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник ст. викл. Путятіна О.Є.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Шаповалову Владиславу Вадимовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та аналіз методів вебтестування

затверджена наказом по університету від 25 листопада 2024 року № 1246Ст

2. Термін подання студентом роботи до екзаменаційної комісії 30 грудня 2024 р.

3. Вихідні дані до роботи наукові та методичні джерела, теоретичні основи тестування ПЗ, інструменти для автоматизації тестування, методологічні підходи до тестування, програмні засоби для проведення дослідження, інформація про вимоги до якості ПЗ.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд основних методів вебтестування.
2. Дослідження та аналіз методів вебтестування.
3. Практичні аспекти та порівняння методів вебтестування.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми дослідження та аналізу вебтестування, постановка задачі, об'єкт та мета роботи, етапи проведення дослідження, висновок, опис апробації.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	25.11.2024	
2	Аналіз завдання, підбір літератури	25.11.24-26.11.24	
3	Аналіз літератури з досліджуваної проблеми	26.11.24-27.11.24	
4	Аналіз технічних засобів	28.11.24-30.11.24	
5	Розробка методу	01.12.24-04.12.24	
6	Програмна реалізація	05.12.24-09.12.24	
7	Оформлення пояснювальної записки	09.12.24-11.12.24	
8	Перевірка на плагіат	13.12.2024	
9	Рецензування	20.12.2024	
10	Підготовка презентації та доповіді	24.12.2024	
11	Занесення роботи в електронний архів	01.01.2025	
12	Попередній захист кваліфікаційної роботи	09.01.2025	

Дата видачі завдання 25 листопада 2024 р.

Студент _____
(підпис)

Керівник роботи _____ ст. викл. Путятіна О.Є.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 52 с., 3 табл., 3 рис., 40 джерел.

ДОСЛІДЖЕННЯ ТА АНАЛІЗ МЕТОДІВ ВЕБТЕСТУВАННЯ ТЕХНОЛОГІЙ HTML, CSS, PHP, MYSQL, JAVASCRIPT.

Об’єктом дослідження є вебсторінки та вебзастосунки, створені за допомогою технологій HTML, CSS, PHP, MySQL, JavaScript.

Метою дослідження є розробка методів та інструментів для тестування вебзастосунків, що використовують технології HTML, CSS, PHP, MySQL, JavaScript, для забезпечення їх функціональності, безпеки та сумісності.

Методи для дослідження застосовувалися методи теоретичного аналізу, експериментальні методи тестування, а також числове моделювання. Проведено аналіз існуючих підходів до тестування вебзастосунків, включаючи функціональне тестування, тестування продуктивності, безпеки та сумісності вебтехнологій.

У результаті дослідження проведене функціональне тестування вебзастосунків, тестування сумісності, тестування.

RESEARCH AND ANALYSIS OF WEB TESTING METHODS FOR HTML, CSS, PHP, MYSQL, JAVASCRIPT TECHNOLOGIES.

The object of the research is web pages and web applications created using HTML, CSS, PHP, MySQL, JavaScript technologies.

The aim of the research is the development of methods and tools for testing web applications using HTML, CSS, PHP, MySQL, JavaScript technologies to ensure their functionality, security, and compatibility.

The research utilized theoretical analysis methods, experimental testing methods, and numerical modeling. The study analyzed existing approaches to web application testing, including functional testing, performance testing, security testing, and compatibility testing of web technologies.

As a result of the research, functional testing of web applications, compatibility testing, and performance testing were conducted.

ЗМІСТ

ВСТУП

Вебтестування є важливою складовою забезпечення якості вебзастосунки та сайтів, оскільки сучасні вебрішення стали невід'ємною частиною бізнесу та повсякденного життя. Вебсайти та вебзастосунки використовуються у різних сферах, таких як електронна комерція, соціальні мережі, фінансові послуги, освіта тощо. Через їхній стрімкий розвиток зростають вимоги до якості та надійності цих систем, тому тестування відіграє ключову роль у забезпеченні безперебійної та коректної роботи вебпродуктів.

Актуальність теми дослідження визначається тим, що з появою нових технологій та інструментів для розробки вебзастосунків підвищуються вимоги до їхнього тестування. Висока конкурентність на ринку вимагає від розробників швидкого випуску оновлень, що не повинно впливати на якість продукту. Для досягнення цих цілей використовуються різноманітні методи вебтестування, які дозволяють автоматизувати перевірки, скоротити час на виявлення дефектів та забезпечити стабільність вебпродукту в умовах підвищеного навантаження.

Метою дипломної роботи є дослідження та аналіз сучасних методів вебтестування для визначення їхньої ефективності у різних аспектах тестування вебзастосунків, таких як функціональне тестування, тестування продуктивності, безпеки та юзабіліті.

Завдання роботи:

– провести огляд існуючих методів вебтестування;

– визначити переваги та недоліки ручного та автоматизованого тестування;

– дослідити інструменти для автоматизації вебтестування та їх застосування;

– провести порівняльний аналіз методів тестування на прикладі реального вебзастосунку;

– запропонувати рекомендації щодо вибору методів тестування залежно від специфіки проекту.

Об’єктом дослідження є процес тестування вебзастосунків.

Предметом дослідження є методи тестування, які використовуються для забезпечення якості вебпродуктів.

У дипломній роботі будуть розглянуті як ручні, так і автоматизовані підходи до вебтестування, проаналізовано їхню ефективність залежно від мети тестування, а також запропоновано оптимальні методи для використання в умовах сучасної розробки. Тема дослідження має практичне значення для фахівців з розробки та тестування програмного забезпечення, адже правильний вибір методів тестування впливає на загальну якість продукту, його безпеку та зручність для кінцевих користувачів.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ ВЕБТЕСТУВАННЯ

1.1 Визначення вебтестування

Тестування є одним з ключових методів контролю якості, що охоплює планування, розробку, проведення тестів та аналіз отриманих результатів. Тестування програмного забезпечення полягає у процесі аналізу програмних рішень та супровідної документації з метою виявлення дефектів і підвищення якості продукту [1].

Тестування ПЗ включає не лише безпосереднє проведення тестів, а й інші елементи забезпечення якості, такі як: аналіз та планування вимог, визначення критеріїв початку тестування, розробка стратегії тестування, створення тестових сценаріїв (тест-планів, тест-кейсів), виконання тестів, реєстрація дефектів, аналіз результатів та складання звітів.

Основними цілями тестування є:

– надання інформації замовнику про якість програмного забезпечення;

– покращення якості тестованого ПЗ;

– запобігання виникненню нових дефектів.

1.2 Класифікація видів тестування вебтестування

До основних видів тестування належать:

Ручне тестування (Manual Testing) – це процес, у якому тестувальник перевіряє ПЗ вручну, не використовуючи інструменти автоматизації. Тестувальник виступає як кінцевий користувач та шукає дефекти в програмі. Незважаючи на тривалість, ручне тестування є економічно вигідним у довгостроковій перспективі [1]. Воно включає різні типи: модульне тестування, інтеграційне, системне, операційне та приймальне тестування.

Для ефективного тестування застосовуються тест-плани, що описують весь обсяг робіт, який потрібно виконати тестувальнику. Тест-план містить опис об’єкта, мети, ресурсів, графіка тестувальних активностей, стратегії тестування та критерії початку й завершення тестування. Також тест-план включає оцінку можливих ризиків та їх вирішення [2].

Переваги ручного тестування:

– реальні відгуки від користувачів;

– можливість тестування користувацького інтерфейсу вручну;

– економічна вигода;

– можливість почати тестування на ранніх етапах розробки;

– дослідницьке тестування.

Недоліки ручного тестування:

– залежність від людського фактора;

– неможливість навантажувального тестування;

– складність повторної перевірки після змін.

Автоматизоване тестування (Automated Testing) – це використання інструментів для автоматизації деяких задач тестування [3]. Автотестування відбувається на трьох рівнях: тестування коду (модульне), функціональне та тестування графічного інтерфейсу користувача (GUI) [4]. Незважаючи на автоматизацію, розробка та виконання тест-кейсів потребують участі тестувальника. Тест-кейс визначає вхідні дані, умови виконання та очікуваний результат.

Переваги автоматизованого тестування:

– швидке виконання тестів;

– можливість повторного тестування після змін;

– перевірка навантаженості системи.

Недоліки автоматизованого тестування:

– високі витрати;

– висока вартість інструментів;

– можливість пропуску дрібних помилок.

Напівавтоматизоване тестування (Semiautomated Testing) – поєднує ручне тестування з автоматизацією окремих завдань. Це дозволяє ефективніше розподіляти ресурси та збільшити точність тестування.

Тестування можна класифікувати за багатьма критеріями, на основі яких створюються різні стратегії тестування, що відповідають конкретним цілям проекту [5].

Розглянемо схему класифікації тестування (рис. 1.1).

Перший тип тестування передбачає перевірку коду, і його можна розділити на два підвиди:

– статичне тестування, яке не потребує запуску коду;

– динамічне тестування, що виконується із запуском коду.

До статичного тестування належить перевірка таких елементів, як: документація (користувацькі та бізнес-вимоги, тест-кейси, користувацькі історії), графічні прототипи, програмний код, параметри програми, підготовка тестових даних. Динамічне тестування охоплює роботу з самим кодом, його частинами та перевірку поведінки програми.

Методи доступу до коду та програми поділяються на:

– метод білого ящика (повний доступ до коду);

– метод сірого ящика (доступ до частини коду);

– метод чорного ящика (без доступу до коду).

Залежно від ступеня автоматизації розрізняють ручне та автоматизоване тестування.

З точки зору рівня деталізації застосунку:

– модульне тестування перевіряє функціональність окремих частин застосунку;

– інтеграційне тестування фокусується на взаємодії між компонентами після їх окремого тестування;

– системне тестування охоплює перевірку як функціональних, так і нефункціональних вимог системи в цілому.

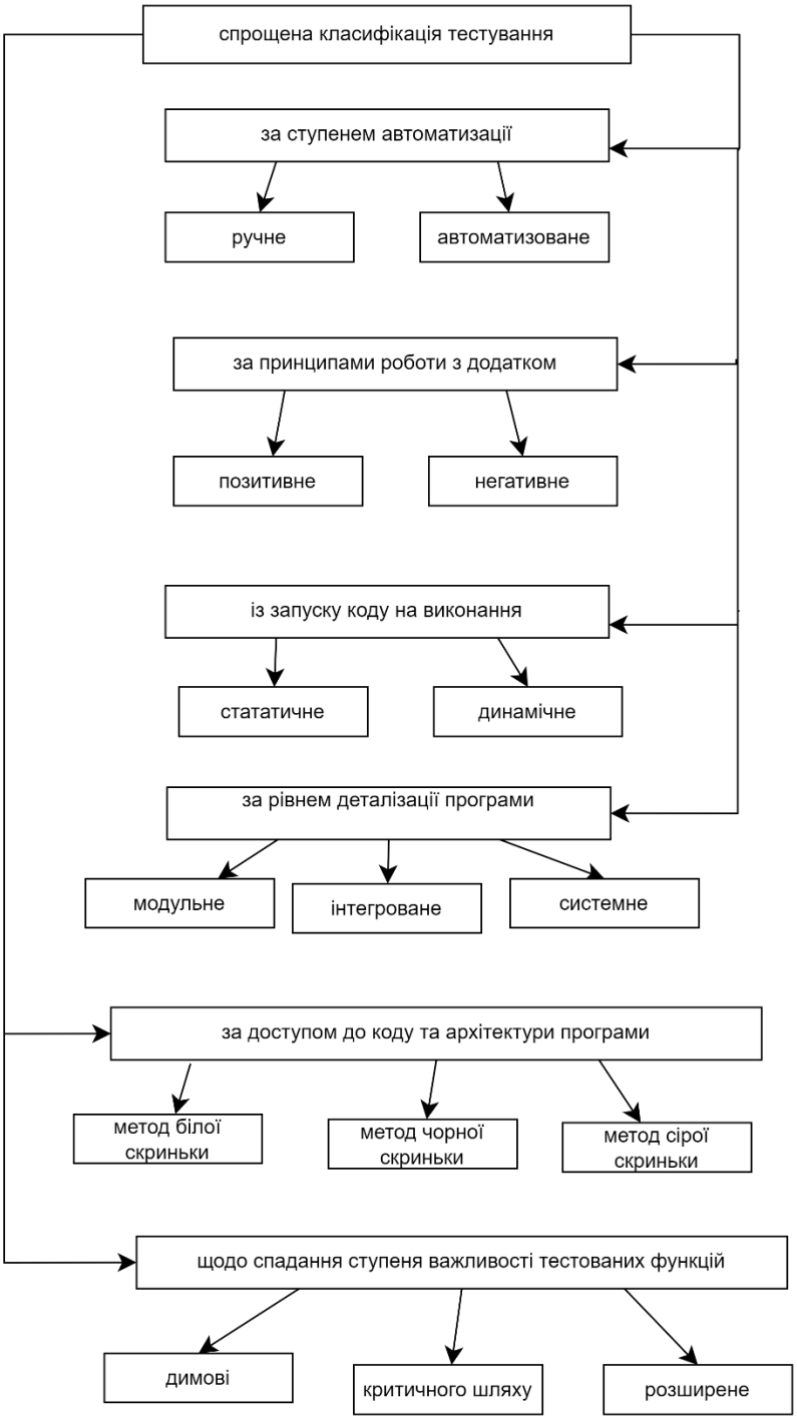

Рисунок 1.1 – Схематичне подання класифікації тестування

За критерієм важливості тестованих функцій виділяють:

– димове тестування, яке перевіряє ключові функції застосунку;

– тестування критичного шляху, що перевіряє найважливіші елементи та функції;

– розширене тестування, що охоплює іншу функціональність.

Залежно від підходу до тестування застосунку:

– позитивне тестування перевіряє коректну роботу програми при введенні валідних даних;

– негативне тестування перевіряє поведінку програми на некоректні дані та помилки [6].

За типом застосунку розрізняють:

– тестування вебзастосунків, яке фокусується на тестуванні сумісності та продуктивності;

– тестування мобільних застосунків, що вимагає уваги до сумісності, продуктивності та автоматизації тестування за допомогою емуляторів;

– тестування класичних застосунків, що залежить від специфіки архітектури та ключових показників якості.

Автоматизація тестування є найефективнішим способом покращити швидкість виконання тестів, охоплення та ефективність процесу. Вона важлива через такі причини: ручне тестування всіх сценаріїв займає багато часу та ресурсів; багатомовні сайти важко тестувати вручну; автоматизація не вимагає втручання людини, підвищує швидкість та зменшує ризик помилок [7].

1.3 Інструменти для автоматизації вебтестування

Інструменти для автоматизації тестування – це програмні рішення, які дозволяють тестувати настільні, веб та мобільні застосунки. Вони автоматизують процес тестування, пропонуючи можливості для перевірки графічних інтерфейсів, продуктивності, навантаження та API.

До найпопулярніших інструментів автоматизації належать: Selenium, Katalon Studio, UFT One. Ці інструменти обирають за кількома ключовими критеріями:

– підтримка тестування API та сервісів;

– можливості штучного інтелекту, машинного навчання та аналітики;

– популярність і завершеність функцій.

Selenium – це набір інструментів для автоматизації веббраузерів на різних платформах. Він підтримує автоматизацію різних браузерів, інтегрується з численними мовами програмування та тестовими фреймворками. Selenium (рис.1.2) – це масштабний open-source проєкт, який зазвичай використовується для автоматизованого тестування вебзастосунків.

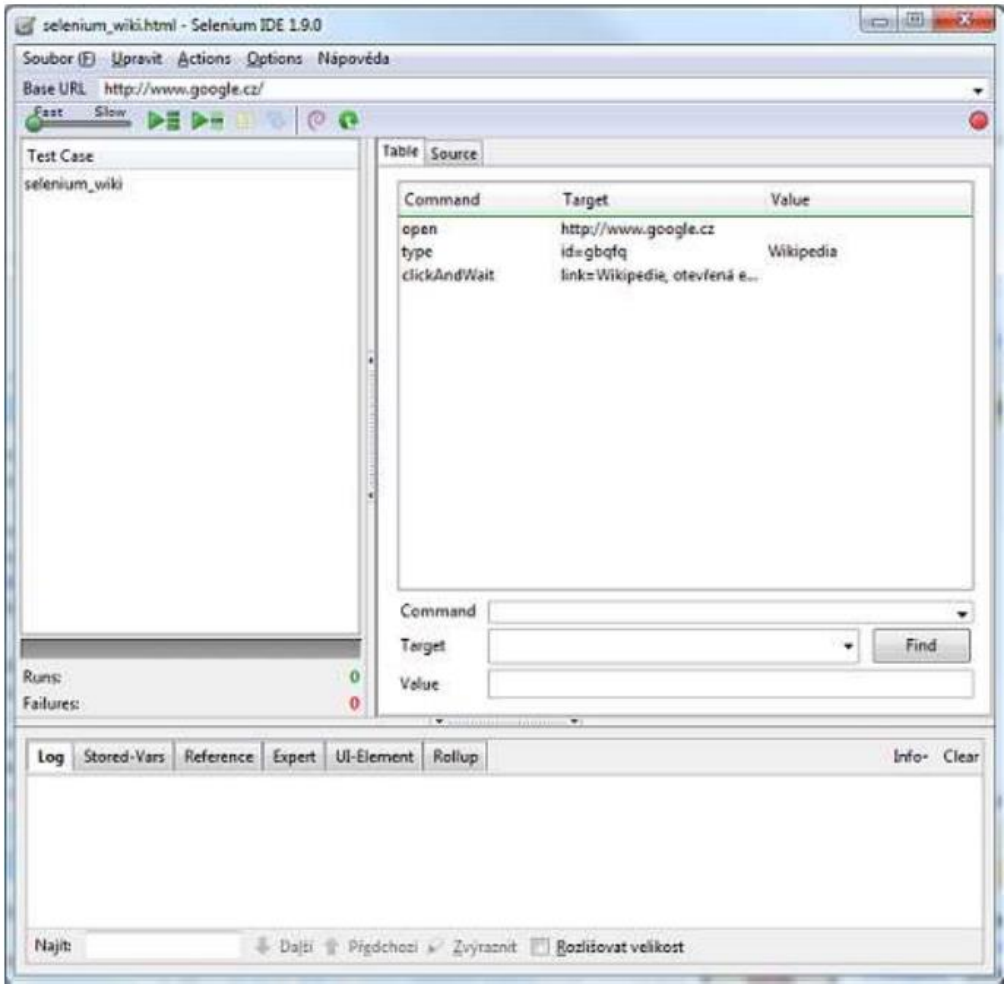

Рисунок 1.2 – Інтерфейс Selenium

Проєкт «Selenium Core», або ж «Selenium 1.0», зародився на основі JavaScript-бібліотеки «JavaScriptTestRunner», яку в 2004 році створив Джейсон Хаггінс у компанії ThoughtWorks для виконання тестів у браузері на Python-сайтах. Згодом до розвитку проєкту долучилися інші співробітники. Починаючи з 2007 року, Хаггінс разом із колегами продовжував працювати

над Selenium вже в компанії Google. Було випущено кілька версій, включно з довгоочікуваною Selenium 3.0, а також анонсовані майбутні версії Selenium 4.0 і Selenium 5.0.

Завдяки їхній роботі Selenium на сьогодні включає численні бібліотеки, написані різними мовами програмування, основною з яких є кросплатформенна Java.

До компонентів Selenium входять: Selenium WebDriver, Selenium Server + Selenium Grid, Selenium IDE, Selenium RC. Головна мета створення системи автоматизованого тестування вебзастосунків полягає в тому, щоб надати розробникам можливість виявляти проблеми під час взаємодії користувачів із вебзастосунками на клієнтському рівні.

Основні вимоги до систем автоматизованого тестування вебзастосунків:

– реєстрація користувачів;

– можливість створення запитів на автоматичне тестування за посиланням;

– налаштування відкладеного або повторюваного тестування;

– наочне відображення проблем із продуктивністю вебзастосунків;

– збереження локальних результатів тестування [8].

1.4 Математичні підходи у вебтестуванні

Математичні підходи у вебтестуванні включають різні методи і моделі, що використовуються для підвищення якості та ефективності процесу тестування. Основні підходи, які використовуються, зосереджуються на формалізації, оцінці ризиків, оптимізації процесів і кількісному аналізі результатів. Деякі з таких підходів:

Теорія графів. Використовується для моделювання шляхів користувачів через різні сторінки вебзастосунка або сценарії взаємодії з інтерфейсом. Графи можуть представляти структуру вебсайту або різні етапи користувацьких дій,

що допомагає створювати тести, охоплюючи всі можливі шляхи і сценарії використання.

Матричне представлення графа: Для графа

G=(V, E), де V – множина вершин, а E – множина ребер, матриця суміжності A може бути використана для представлення зв'язків між сторінками вебзастосунка (1.1):

$$A[i][j] = \begin{cases} 1, \text{якщо є ребро між } V_i \text{ і } V_j \\ 0, \text{якщо немає ребра} \end{cases}. \quad (1.1)$$

Це дозволяє визначати всі можливі шляхи між сторінками вебсайту.

Комбінаторне тестування. Використовується для перевірки різних комбінацій вхідних даних і умов. Комбінаторний аналіз дозволяє мінімізувати кількість тестових випадків, покриваючи всі можливі варіанти параметрів, що зменшує витрати на тестування та водночас забезпечує високу ефективність.

Теорія ймовірностей і статистика. Використовується для оцінки надійності і стійкості вебзастосунків. Ці методи можуть бути застосовані для аналізу ймовірності виникнення дефектів, визначення найвразливіших частин системи або прогнозування поведінки системи за різних умов.

Машинне навчання. У сучасному вебтестуванні часто використовуються алгоритми машинного навчання для автоматизації процесу тестування, аналізу результатів і виявлення дефектів. Машинне навчання допомагає виявляти аномалії в поведінці застосунків та покращувати точність тестування.

Функція помилки для визначення алгоритмів класифікації наведена у формулі (1.2):

$$E(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y_i\, log\left(h_\theta(x_i)\right) + (1-y_i) log\left(1-h_\theta(x_i)\right)\right], \quad (1.2)$$

де $h_\theta(x_i)$ – передбачення моделі;

y_i – реальне значення;

m – кількість прикладів.

Метод кластеризації і класифікації. Ці математичні методи допомагають групувати тестові сценарії за схожими характеристиками або на основі поведінки системи, що дозволяє оптимізувати процес тестування, зосереджуючи увагу на найбільш критичних функціональних елементах.

Логічні та алгебраїчні методи. Формальні методи, засновані на логіці, використовуються для перевірки правильності виконання алгоритмів, відповідності специфікаціям і вимогам до системи. Алгебраїчні підходи допомагають формалізувати та автоматизувати валідацію програмних компонентів [9].

Таким чином, математичні підходи у вебтестуванні забезпечують систематичний та науково обґрунтований підхід до тестування, що дозволяє не лише підвищити якість кінцевого продукту, а й оптимізувати процеси тестування, зменшуючи витрати часу та ресурсів. Застосування цих методів сприяє більш ефективному виявленню дефектів та підвищенню загальної надійності вебзастосунків.

2 ДОСЛІДЖЕННЯ ТА АНАЛІЗ МЕТОДІВ ВЕБТЕСТУВАННЯ

2.1 Математичні моделі в вебтестуванні

Коли об’єкт дослідження має просту структуру і добре вивчений, а його характеристики можна визначити за допомогою теоретичних концепцій і доступних літературних даних, доцільно застосовувати метод математичного моделювання. У такому випадку функціонування елементів системи можна представити у вигляді певних функціональних співвідношень, таких як алгебраїчні, диференціальні, інтегро-диференціальні чи скінченно-різницеві рівняння або за допомогою логічних виразів. Математична модель реальної системи або процесу є набором співвідношень – формул, рівнянь, нерівностей, логічних умов або операторів, що описують характеристики станів системи залежно від її параметрів, зовнішніх умов, початкових умов і часу.

Згідно з визначенням В.М. Глушкова, математична модель являє собою сукупність математичних об’єктів і відносин між ними. Водночас М.М. Амосов трактує її як систему, що відображає іншу систему. Для дослідження таких моделей застосовуються різні методи: аналітичні, які дозволяють отримати явні залежності для характеристик у загальному вигляді; чисельні, що забезпечують конкретні значення параметрів за певних початкових і крайових умов; якісні, що визначають властивості розв’язків (стійкість, асимптотичну поведінку, монотонність); та аналогові, які досліджують властивості систем через реальні об’єкти, як-от електричні схеми.

Для складних об’єктів математичні моделі часто створюють через ідентифікацію – вибір відповідної моделі з наявного класу та визначення її параметрів на основі емпіричних даних. Наприклад, при вивченні властивостей твердих тіл застосовують метод механічної спектроскопії, що дозволяє отримати релаксаційний спектр, представлений у вигляді суми максимумів, кожен із яких відповідає окремому процесу. Завдання

ідентифікації полягає у виборі рівняння, що описує окремий максимум, встановленні кількості таких максимумів і визначенні коефіцієнтів у рівняннях.

Математичні моделі класифікують за різними критеріями. За статичністю вони бувають статичними та динамічними. Статичні моделі описують стан системи в конкретний момент часу й підходять для моделювання рівноважних і стаціонарних станів. Наприклад, рівняння ідеального газу або закон Кулона. Динамічні моделі, навпаки, аналізують процеси зміни стану системи в часі, як-от рівняння руху або дифузії.

Інший критерій класифікації – природа моделей: детерміновані або стохастичні. Детерміновані моделі ґрунтуються на точних зв’язках між параметрами і дозволяють точно прогнозувати поведінку системи, як-от другий закон Ньютона. У стохастичних моделях враховують випадкові фактори чи невизначеності, тому їхній аналіз проводять із використанням теорії ймовірностей, що є актуальним для таких об’єктів, як валютні курси чи екосистеми.

Також моделі поділяються на дискретні й неперервні. Неперервні моделі використовуються для опису систем із неперервними параметрами, як-от температура чи час, тоді як дискретні моделі описують явища з чітко визначеними значеннями, як-от кількість об’єктів. Наприклад, диференціальні рівняння описують неперервні моделі, тоді як у задачах розподілу ресурсів або оптимізації використовують дискретні.

Моделі також розрізняються за лінійністю. Лінійні моделі, як-от закон Гука, характеризуються простими залежностями, що дозволяє їх легко аналізувати. Нелінійні моделі складніші й часто вимагають спеціальних методів аналізу. Нерідко такі моделі намагаються лінеаризувати, але це обмежує застосування отриманих результатів.

За кількістю вихідних змінних моделі поділяються на одновимірні та багатовимірні. Крім того, залежно від часу, вони можуть бути стаціонарними або нестаціонарними. Стаціонарні моделі описують системи, які змінюються

повільно, наприклад, у демографії чи економіці. Нестаціонарні моделі, як-от бюджетні моделі, враховують швидкі зміни.

Особливе значення має врахування просторової протяжності об'єкта. Для систем із зосередженими параметрами, як-от матеріальна точка, використовують звичайні диференціальні рівняння. У разі розподілених параметрів, коли характеристики змінюються у просторі, застосовуються диференціальні рівняння в частинних похідних, як-от закони Фіка і Фур’є.

Методи аналізу моделей також різноманітні. Аналітичні підходи, включаючи методи диференціювання й інтегрування, забезпечують загальні висновки. Чисельні методи дозволяють розв’язувати складніші задачі, проте обмежуються конкретними випадками. Обчислювальні технології, як-от Maple чи Maxima, значно розширили можливості математичного моделювання. Хоча аналітичні методи є універсальними, у багатьох випадках доводиться спрощувати моделі для їх реалізації, що зменшує їхню точність.

Таким чином, математичне моделювання є універсальним інструментом для вивчення та прогнозування поведінки складних систем у різних сферах науки та техніки [10].

2.2 Оцінка якості програмного забезпечення в вебтестуванні

Аналіз ризику є структурованим підходом до використання інформації для виявлення джерел небезпеки та кількісної оцінки ризику. Цей аналіз служить основою для подальшого оцінювання ризику і розробки стратегій управління ним, зокрема, для його мінімізації. Для цього можуть бути використані різноманітні дані, включаючи історичні відомості, результати теоретичних і експериментальних досліджень. Загалом, аналіз ризику включає дослідження, які зосереджені на виявленні небезпек і кількісному визначенні ризиків в контексті різних видів діяльності, проектів та інше.

Підприємець, ухвалюючи рішення, спочатку повинен оцінити масштаби ризику, які можуть загрожувати його бізнесу у випадку вибору певного напрямку дій. Оцінка ризику може розглядатися як інтегрований етап в процесі управління ризиком, що охоплює як якісний, так і кількісний аналіз, а також порівняння об'єктивних даних про ступінь ризику з суб'єктивним сприйняттям ризику тією особою, яка приймає рішення.

Процес аналізу ризику зазвичай включає такі етапи:

– виявлення невизначеностей у поведінці господарської системи;

– визначення показників, які характеризують ці невизначеності;

– ідентифікація ризикових факторів, пов'язаних з невизначеністю;

– оцінка ризиків за допомогою методів вимірювання;

– використання якісних підходів;

– використання кількісних методів;

– прогнозування рівня ризиків;

– управління ризиками, що передбачає запобіжні заходи.

Ступінь ризику визначається ймовірністю певного сценарію розвитку подій. Кількісно цей ступінь оцінюється суб'єктивно через прогнозовані максимальні та мінімальні доходи або збитки від інвестицій. Чим більший розрив між максимальним і мінімальним доходом (збитком) при однаковій ймовірності їх отримання, тим вищим вважається ступінь ризику.

Невизначеність ситуації пов'язана з фактором випадковості.

Випадковість – це явище, яке проявляється по-різному в аналогічних умовах, що ускладнює його передбачення і прогнозування. Проте, при наявності великої кількості спостережень можна виявити певні закономірності у випадкових подіях. Математичні інструменти, що використовуються для їхнього аналізу, надає теорія ймовірності. Ризик можна виразити в термінах ймовірності втрат, які можуть виникнути. На основі статистичних даних можна з високою точністю кількісно оцінити рівень ризику, а також всі можливі наслідки конкретних дій і ймовірності цих наслідків.

Ймовірність визначає можливість отримання певного результату.

$$P(A) = \frac{m}{n}, \quad (2.1)$$

де $P(A)$ – ймовірність настання події A;

m – кількість можливих ситуацій, при яких настає подія A;

n – загальна кількість ситуацій.

Коли йдеться про використання ймовірності для оцінки ризику, цю формулу можна інтерпретувати як відношення кількості випадків, що призводять до негативного результату (збитків), до загальної кількості можливих результатів. При аналізі економічного ризику залишаються актуальними правила додавання та множення ймовірностей, які використовуються для оцінки ризику при виникненні однієї з подій зі списку або при одночасному прояві кількох подій.

Багато сучасних вітчизняних та зарубіжних дослідників виділяють дві основні категорії методів оцінки економічного ризику:

– кількісні методи, які включають математичні, статистичні, ймовірнісні та аналітичні підходи;

– якісні методи, до яких належать моделювання, оптимізація, теорія ігор, а також стохастичне програмування та аналоги.

Якісний аналіз ризиків передбачає також отримання кількісних результатів. Це означає, що процес якісного аналізу має охоплювати не лише опис різних видів ризиків у маркетинговій діяльності, виявлення можливих причин їх виникнення, аналіз непередбачуваних наслідків і пропозиції щодо їх мінімізації, а й оцінку вартості потенційних збитків, а також всіх заходів, спрямованих на зменшення ризиків у маркетингу.

Одним з найбільш поширених методів, що застосовується під час якісного аналізу, є експертна оцінка, яка дозволяє визначити рівень ризику в умовах браку повної та достовірної інформації, необхідної для використання статистичних методів. Основу цього методу складають результати обробки думок досвідчених підприємців або фахівців. В практиці використовуються як

індивідуальні, так і групові експертні оцінки, переваги та недоліки яких зведені в табл. 2.1.

Таблиця 2.1 – Переваги й недоліки методу експертних оцінок в процесі якісного аналізу ризиків підприємства

Індивідуальна експертиза		Колективна (групова) експертиза	
Переваги	Недоліки	Переваги	Недоліки
Оперативність отримання інформації для прийняття рішень і невеликі витрати	Відсутність упевненості в достовірності оцінок, зумовленої високим рівнем суб'єктивності	Менш суб'єктивні і, як наслідок, більш достовірні	Відсутність гарантій вірогідності отриманих оцінок, труднощі в проведенні опитування експертів, обробці одержаних даних

Для колективного обговорення зазвичай застосовується метод генерації ідей у групі, відомий як «мозкова атака», який спрямований на створення великої кількості ідей. Цей підхід допомагає виявити джерела та причини ризику, визначити можливі загрози, окреслити напрямки та стратегії їх зменшення, а також формувати й оцінювати різні варіанти заходів для зниження ризиків.

Якісна оцінка ризику є досить складною, оскільки вона вимагає глибоких знань з теорії економіки, бізнесу та фінансів, а також знання специфічних предметів, необхідних для підготовки фахівців у певних галузях. Крім того, важливим є наявність практичного досвіду та інтуїції в обраній сфері економічної діяльності. Результати якісного аналізу ризиків слугують основою для подальшого кількісного аналізу.

Методи кількісної оцінки ризиків дозволяють визначити чисельні значення окремих ризиків, а також загальний ризик, пов'язаний з конкретними видами діяльності. У процесі кількісної оцінки ризиків підприємств використовують різноманітні підходи. Вибір конкретного методу залежить від типу та джерел ризику, особливостей сфери діяльності, фінансового стану та масштабів підприємства.

У загальному випадку ці методи можна класифікувати на:

– об’єктивні (які ґрунтуються на характеристиках випадкових процесів, що отримані з даних, незалежних від суб’єктивних думок конкретних осіб);

– суб’єктивні (які базуються на експертних оцінках ризиків).

Серед основних методів кількісної оцінки ризику можна виділити: статистичний метод, метод експертних оцінок, використання аналогів, метод критичних значень, оцінку ризику через «дерево рішень», аналіз чутливості, сценарний аналіз та імітаційне моделювання. У таблиці 2.2 представлені переваги та недоліки цих основних методів кількісної оцінки ризику.

Таблиця 2.2 - Кількісні методи оцінки ризику

Метод	Переваги	Недоліки
1	**2**	**3**
Статистичний	Можливість моделювання сценаріїв, висока точність розрахунків, часткова стандартизація, нескладність математичних розрахунків	Необхідність великої кількості спостережень; ризик відповідності обраної моделі; високі витрати на інформатизацію та аналіз інформації
Розрахунково-аналітичний	Широке застосування, можливість об’єктивної оцінки за невисоких витрат	Суб’єктивізм оцінок, відсутність стандартів
Метод експертних оцінок	Невисока вартість; відсутність необхідності в точних початкових даних і дорогих програмних засобах, можливість здійснювати оцінку до розрахунку ефективності проекту, простота розрахунку	Складність з отриманням фінансової оцінки ризику, висока залежність від суб’єктивної думки експертів; складність залучення незалежних експертів
«Дерево рішень»	Висока точність оцінки; детальний облік факторів ризику; можливість різних сценаріїв розвитку подій	Потребує багато часу на дослідження; високі витрати при великій кількості варіантів
Аналіз чутливості	Можливість за його допомогою вирішити проблему зіставлення впливу різних (натуральних, вартісних) характеристик проекту, що варіюються.	Врахування одного чинника проекту, що приводить до нестачі можливостей зв’язків між окремими чинниками або недообліку їхньої кореляції.
Аналіз сценаріїв	Можливий для різноманітних варіантів реалізації проекту; застосування програмних засобів (можливість збільшити кількість можливих сценаріїв і таким чином значно підвищиться ефективність оцінки ризику)	Для високої цінності сценарію з метою прогнозування необхідний великий об’єм вихідної інформації; низька можливість реалізації точного прогнозування при кожному сценарію; потребує масштабних підготовчих робіт

Продовження таблиці 2.2

1	2	3
Метод Монте-Карло	Дозволяє реалізувати моделі складних систем; можливість використання будь-яких розподілів; моделювання складної поведінки ринку.	Складність методу; необхідна велика кількість експериментальних даних необхідність потужних розрахунків; складність презентації.
Метод аналогій	Невисока вартість; простота розрахунку.	Невисока точність; проблематичність підбору аналогів; не враховується розвиток певного виду діяльності.

Статистичний метод ризику базується на принципі екстраполяції тенденцій зміни досліджуваного показника з минулих періодів у майбутнє. Цей підхід є корисним для оцінки економічних ризиків, особливо коли підприємство володіє значним обсягом аналітично-статистичних даних, що відображають результати господарських операцій. Основою статистичного методу є теорія ймовірності, що дозволяє, використовуючи наявні дані про ризики з минулого, визначити ймовірність їх прояву в майбутньому. Завдяки статистичному аналізу можливо оцінити не лише ризики окремих угод, а й загальні ризики підприємства, аналізуючи динаміку його прибутків за певний період.

Розрахунково-аналітичний метод ґрунтується на дисконтуванні грошових потоків.

Норма дисконтування є показником складного відсотка, що використовується для приведення вартості грошових потоків до конкретного моменту часу.

Цей метод передбачає використання традиційних фінансових показників, які найчастіше застосовуються при оцінці ефективності інвестиційних та інноваційних проектів:

− чиста теперішня вартість (NPV – net present value);

− індекс дохідності (PI – profitability index);

− період повернення інвестицій (PP – payback period);

− внутрішня норма рентабельності (IRR – internal rate of return).

Рівень ризику визначається за величиною відхилення розрахованих значень цих показників від рекомендованих або граничних значень, а також від аналогічних показників альтернативних проектів.

Чиста теперішня вартість визначається як сума дисконтованих фінансових результатів за всі роки реалізації проекту, починаючи з моменту інвестування.

$$NPV = \sum_{t=1}^{n} \frac{CF_t}{(1+r)^t} = \sum_{t=0}^{n} \frac{l_t}{(1+r)^t}, \quad (2.2)$$

де CF_t – це грошові надходження, отримані від інвестицій у період *t*;

l_t – це грошові потоки інвестицій у період *t*;

r – це норма дисконту, що відповідає необхідному рівню прибутковості інвестицій;

t – це період часу;

n – це тривалість інвестиційного проекту.

Чим ближче значення чистої теперішньої вартості проекту до нуля, тим вищий ризик, пов’язаний з його реалізацією.

Індекс дохідності визначається як відношення дисконтованих доходів до дисконтованих витрат на інвестиції:

$$PI = \sum_{t=1}^{n} \frac{CF_t}{(1+r)^t} \Big/ \sum_{t=0}^{n} \frac{I_t}{(1+r)^t}. \quad (2.3)$$

Проте цей показник має суттєвий недолік: він не може використовуватися як міра прибутковості, оскільки не враховує грошові потоки, які надходять після терміна окупності, а також вартість капіталу проекту. Це призводить до того, що при його використанні надається перевага короткостроковим проектам.

Внутрішня норма рентабельності є розрахунковою процентною ставкою, при якій проект стає безризиковим і не приносить прибутку:

$$IRR = \frac{NPV_{r1} + NPV_{r2}}{NPV_{r1}}(r_2 - r_1). \quad (2.4)$$

Методи експертних оцінок дозволяють залучити досвід фахівців для аналізу діяльності та врахування впливу різноманітних якісних факторів. Найчастіше застосовуються такі експертні техніки:

– SWOT-аналіз;

– «Зірка (троянда) ризиків» (порівняння різних факторів, створення «спіралі ризиків», що відображає ранжування ризиків);

– метод Дельфі (виключає взаємний вплив експертів один на одного, що знижує ризик психологічного дискомфорту, пов'язаного з персоніфікацією оцінок).

Метод оцінки ризику за допомогою «дерева рішень» передбачає створення гілок, за якими оцінюються різні шляхи і обирається менш ризикований, застосовуючи спеціальні методи для розрахунку ймовірностей.

Аналіз чутливості дозволяє відслідковувати варіації ключових припущень під час прогнозування грошових потоків, визначаючи їхній вплив на прогнозовану вигоду. У цьому випадку розраховується еластичність (відносна величина, яка відображає зміну критеріального показника при одиничній зміні ризикової змінної). Результати цього аналізу виявляють найбільш чутливі змінні, що потребують подальшого вивчення.

Процес проведення аналізу чутливості складається з наступних етапів:

– визначається математичний зв'язок між вихідними та результуючими показниками;

– визначаються найбільш імовірні значення для вихідних показників та можливі діапазони їх варіацій;

– розраховується найбільш ймовірне значення результуючого показника;

– кожен з вихідних параметрів змінюється на певну величину в межах допустимого діапазону, що дозволяє отримати нове значення результуючого критерію (для точності розрахунків рекомендується змінювати вихідні параметри на один і той же відсоток);

– всі вихідні параметри ранжуються за впливом на зміну результуючого критерію, що дає змогу групувати їх за ступенем ризику.

Після завершення розрахунків проводиться експертне ранжування змінних за їхньою важливістю та прогнозованістю.

Потім будують матрицю чутливості, що дозволяє виділити найменш та найбільш ризикові для проєкту зміни (табл 2.3).

Таблиця 2.3 – Матриця чутливості

Передбачуваність змін	Чутливість змін		
	Висока	Середня	Низька
Низька	I	I	II
Середня	I	II	III
Висока	II	III	III

Перша зона, розташована в лівому верхньому куті матриці, є територією для подальшого аналізу факторів, оскільки зміни в цих показниках мають найзначніший вплив на NPV проекту і демонструють низький рівень прогнозованості.

Друга зона, що включає елементи основної діагоналі матриці, потребує особливої уваги до змін факторів, які тут представлені (для цього були проведені розрахунки критичних значень для кожного показника).

Третя зона, або зона «максимального благополуччя», включає ті фактори, які, згідно з усіма іншими зробленими припущеннями і розрахунками, потрапили в правий нижній кут таблиці. Ці фактори є найменш ризикованими і не вимагають подальшого розгляду.

У процесі сценарного аналізу відхилення параметрів розраховуються з урахуванням їхніх взаємозалежностей (кореляцій), що є однією з переваг цього методу. Найчастіше аналізуються три можливі сценарії: песимістичний, оптимістичний та найбільш ймовірний.

Для кожного сценарію розраховуються середні значення результуючих показників, враховуючи ймовірність реалізації кожного з них, а також визначається діапазон їх варіацій або середньоквадратичне відхилення. Серед двох проектів, що порівнюються, більш ризикованим вважається той, у якого більший діапазон варіації критеріального показника або більше значення середньоквадратичного відхилення.

Метод сценарного аналізу має широке практичне застосування, і одним з яскравих прикладів є стрес-тестування.

Стрес-тестування активно використовується для оцінки кредитного ризику, ризику ліквідності, валютного ризику, ризику зміни процентних ставок і вартості активів. Його метою є оцінка ризиків та визначення здатності системи витримувати потрясіння на фінансових та інших ринках.

Серед найбільш поширених об’єктів стрес-тестування для банків можна виділити: різкі зміни процентних ставок за внутрішніми або зовнішніми запозиченнями, кредитами, цінними паперами тощо; суттєві коливання валютних курсів; кредитний ризик у портфелях; раптові зміни в обсягах і структурі капіталу фінансової установи, вартості застави при іпотеці; зниження ліквідності та ймовірність дефолту банку; а також ризик виникнення системного ризику через різке зниження ліквідності або втрату капіталу.

Метод Монте-Карло, також відомий як метод статистичних випробувань або імітаційного моделювання, застосовується в ситуаціях, коли необхідно врахувати велику кількість факторів ризику, що впливають на різні аспекти підприємницької діяльності.

Для моделювання за цим методом спочатку фіксуються параметри, що підлягають аналізу, визначаються їх можливі діапазони варіювання, і кожному параметру надається певна ймовірність. Після цього з заданого діапазону

випадковим чином вибираються значення параметрів, на основі яких розраховується інтегральний ризик.

Імітаційне моделювання дозволяє генерувати випадкові сценарії, результати яких представляються у формі ймовірнісного розподілу всіх можливих значень результуючих показників. Цей підхід часто використовується в найбільш складних для прогнозування ситуаціях. Аналітики зазначають, що він може надавати більш оптимістичні оцінки в порівнянні зі сценарним методом, оскільки включає більшу кількість проміжних варіантів.

У межах методу Монте-Карло ризик аналізується через моделі потенційних результатів. При створенні цих моделей фактори з невизначеністю замінюються діапазоном значень – розподілом ймовірностей. Потім здійснюються багаторазові розрахунки, при цьому кожен раз використовується новий набір випадкових значень для функцій ймовірності. Іноді для завершення моделювання потрібно провести тисячі, а інколи й десятки тисяч розрахунків, що залежить від кількості невизначеностей та їх діапазонів. Моделювання за методом Монте-Карло дозволяє отримати розподіл можливих наслідків.

При використанні ймовірнісних розподілів змінні можуть мати різні ймовірності настання різних наслідків, що робить цей підхід більш реалістичним для опису невизначеності змінних в аналізі ризику. Найпоширеніші типи розподілів ймовірностей включають:

– нормальний розподіл (або «крива Гауса»). Для опису відхилень від середнього значення дослідник визначає його середнє (очікуване) значення та стандартне відхилення. Значення, що розташовані ближче до середнього, мають найвищу ймовірність. Цей розподіл симетричний і відображає багато звичайних явищ, наприклад, розподіл населення за віком або вагою. Прикладами змінних, що підпадають під нормальний розподіл, є темпи інфляції та ціни на енергоносії;

– логнормальний розподіл. Цей розподіл має позитивну асиметрію і, на відміну від нормального, не є симетричним. Він підходить для величин, які не можуть бути меншими за нуль, але можуть набувати необмежених додатних значень. Прикладами змінних є ціни на акції, вартість нерухомості та запаси нафти;

– рівномірний розподіл. В усіх величинах є однакова ймовірність прийняття різних значень. Аналітик визначає лише мінімальне і максимальне значення. Приклади змінних з рівномірним розподілом – виробничі витрати або доходи від продажів нового продукту;

– трикутний розподіл. Аналітик встановлює мінімальне, найбільш імовірне та максимальне значення. Значення, що розташовані близько до найбільш ймовірного, мають найвищу ймовірність. Цей розподіл може описувати, наприклад, обсяги продажу за певний період;

– PERT-розподіл. Аналогічно трикутному розподілу, аналітик визначає мінімальне, найбільш імовірне та максимальне значення. Однак у PERT-розподілі величини між найбільш ймовірним і граничними значеннями мають вищу ймовірність прояву, що робить його менш зосередженим на крайніх значеннях. Цей розподіл часто використовується для оцінки тривалості виконання проектних завдань;

– дискретний розподіл. Користувач визначає конкретні значення з можливих, а також ймовірності для кожного з них.

Таким чином, метод Монте-Карло дозволяє отримати більш детальне уявлення про можливі події, оцінюючи не лише ймовірності їх настання, а й конкретні результати.

Метод аналогій дозволяє створити так звану криву ризику, порівнюючи дані, отримані раніше. Це включає використання баз даних та знань про ризики, пов’язані з аналогічними об’єктами або угодами. Цей метод є доцільним, коли інші способи оцінки ризику не можуть бути застосовані. Часто в якості аналогів використовуються підприємства з різними потужностями, і необхідно перерахувати вартість одного підприємства на

вартість іншого. Аналогічно, це стосується й оцінки характеристик ризиків. При такому перерахунку слід враховувати ефект концентрації виробництва, який впливає на питомі капіталовкладення підприємства [11].

2.3 Аналіз оптимізації тестування

Генетичні алгоритми – це еволюційні методи, які моделюють процес природного відбору. Вони можуть бути використані для оптимізації тестових випадків, вибору найефективніших наборів тестів з великої кількості можливих варіантів. Основні етапи включають:

– ініціалізація: створення початкової популяції тестових наборів;

– оцінка: вимірювання ефективності кожного набору тестів за певними критеріями;

– селекція: вибір найкращих наборів для подальшої мутації та схрещування;

– мутація та схрещування: модифікація наборів для створення нових комбінацій, що можуть бути більш ефективними [12].

Генетичні алгоритми використовують поняття еволюції, і їх можна формалізувати наступним чином:

Функція придатності $f(x)$: визначає, наскільки хороший певний набір тестів x.

$$f(x) = \sum_{i=i}^{n} w_i * p_i, \quad (2.5)$$

де n – кількість тестів;

w_i – вага i-того тесту (важливість);

p_i – результат виконання тесту (1, якщо тест пройшов, 0, якщо ні).

Операція селекції – відбір набір x' тестів на основі функції придатності.

$$x' = Select(x). \quad (2.6)$$

Мутація – зміна деяких тестів у наборі.

$$x'' = Mutate(x'). \quad (2.7)$$

Схрещування – комбінування двох наборів тестів [13].

$$x'' = Crossover(x_1, x_2). \quad (2.8)$$

Кластеризація допомагає групувати тестові випадки на основі їхніх характеристик, що дозволяє зосередитися на найбільш критичних зонах вебсайту. Алгоритми, такі як K-середніх або ієрархічна кластеризація, можуть бути використані для:

– виявлення подібних тестів, щоб зменшити повторювані тести;

– оптимізації ресурсів, оскільки не всі тести потрібно виконувати для кожного релізу.

Використання алгоритмів кластеризації, таких як K-середніх, можна описати так:

Функція відстані – визначає, наскільки близькі дані до центру кластера.

$$d(x, c) = \sqrt{\sum_{j=1}^{m} \left(x_j - c_j\right)^2}, \quad (2.9)$$

де d – відстань між точкою x і центром кластера c;

m – кількість характеристик (факторів).

Оновлення центру кластера – новий центр визначається як середнє всіх точок, що належать кластеру.

$$C_k = \frac{1}{|C_k|} \sum_{x \in C_k} x, \quad (2.10)$$

де C_k – набір точок у *k*-му кластері [14].

Методи навчання з підкріпленням – допомогають розробити стратегії тестування на основі минулого досвіду [15]. Наприклад, алгоритм може навчатися на результатах попередніх тестів і розробити стратегії для виконання найефективніших тестів у майбутньому [16].

Оптимізація на основі обмежень – використовує математичні моделі для визначення найбільш важливих тестових випадків, які слід виконувати в умовах обмежених ресурсів (часу, людей, фінансів). Моделі можуть включати:

– лінійне програмування для визначення оптимального набору тестів;

– використання формул для визначення найкращих варіантів на основі критеріїв, таких як ризик та ймовірність виявлення дефектів [17];

– алгоритми градієнтного спуску – цей підхід може бути використаний для мінімізації витрат часу на виконання тестів. Він дозволяє ідентифікувати найбільш критичні шляхи в тестах, скорочуючи їх тривалість за рахунок видалення малоефективних тестових випадків [18];

– використання математичних алгоритмів для оптимізації тестування вебсайтів може значно підвищити ефективність і зменшити витрати. Вибір конкретних алгоритмів залежить від специфіки проекту, обсягів тестування та доступних ресурсів.

3 ПРАКТИЧНІ АСПЕКТИ ТА ПОРІВНЯННЯ МЕТОДІВ ВЕБТЕСТУВАННЯ

3.1 Огляд методів вебтестування

У сучасному вебтестуванні існує безліч методів і підходів, які дозволяють перевіряти якість вебсайтів та вебзастосунків. Вибір методу залежить від цілей тестування, специфікацій продукту, а також від доступних інструментів і ресурсів. Однак основними категоріями методів можна вважати функціональне тестування, тестування продуктивності, тестування безпеки та тестування сумісності.

Функціональне тестування орієнтоване на перевірку того, чи працюють усі функціональні компоненти вебзастосунку згідно з технічними вимогами. Воно включає такі види тестування:

Тестування інтерфейсу користувача – перевіряються всі елементи інтерфейсу, такі як кнопки, поля вводу, меню, на правильність виконання.

Тестування взаємодії з базою даних – перевіряються операції з базою даних (зчитування, запис, оновлення, видалення) через вебінтерфейс.

Тестування API – перевіряється правильність роботи всіх інтерфейсів програмування застосунків, що забезпечують взаємодію з іншими системами.

Методи функціонального тестування:

– чорна скринька (Black-box testing). Тестувальник не має доступу до внутрішньої структури системи і перевіряє лише зовнішні функціональні характеристики;

– біла скринька (White-box testing). Тестувальник має доступ до коду і перевіряє, як правильно функціонує логіка застосунку, чи правильно обробляються всі дані.

Тестування продуктивності фокусується на тому, як вебсайт або вебзастосунок поводиться під навантаженням. Це важливо для оцінки

здатності системи витримати великий потік користувачів без значних затримок або збоїв. Продуктивність можна тестувати за такими параметрами:

– навантаження. Перевірка, як система працює при нормальних умовах використання;

– стрес-тестування. Оцінка того, як система поводиться при максимально високому навантаженні, перевірка її здатності витримати екстремальні умови;

– тестування стабільності. Визначення, чи може система працювати стабільно протягом тривалого часу під різними рівнями навантаження.

Методи тестування продуктивності:

– Load Testing вимірювання часу відгуку при визначеному навантаженні;

– Stress Testing визначення межі навантаження, при якому система вийде з ладу;

– Spike Testing оцінка реакції системи на різке збільшення навантаження.

Безпека вебсайтів і вебзастосунків є одним із найважливіших аспектів сучасного тестування. Тестування безпеки має на меті виявити уразливості, які можуть бути використані зловмисниками. Це включає перевірку наступних аспектів:

– аутентифікація та авторизація – перевірка надійності системи для обмеження доступу до даних користувачів;

– шифрування даних – перевірка наявності шифрування при передачі чутливої інформації;

– атаки на застосунок – перевірка на вразливості типу SQL-ін’єкцій, XSS, CSRF тощо.

Методи тестування безпеки:

– Penetration Testing – перевірка системи з точки зору хакера, спроби проникнути в систему через її уразливості;

– Static and Dynamic Analysis – перевірка коду та поведінки вебзастосунку на предмет безпеки.

3.2 Опис та обґрунтування обраних технологій

Обраними технологіями для розробки вебсайту є HTML, CSS, PHP, MYSQL, JAVASCRIPT.

HTML (від англ. HyperText Markup Language – мова розмітки гіпертексту) є стандартною мовою для створення вебсторінок. Документи HTML обробляються веббраузерами та відображаються у зручному для користувача вигляді [19]. HTML надає інструменти для:

– формування структури документа через визначення елементів тексту, таких як заголовки, абзаци, списки, таблиці, цитати та інші;

– створення гіперпосилань для зв’язку між різними вебсторінками;

– розробки інтерактивних форм;

– включення мультимедійних елементів, таких як зображення, відео, звук та інші [20].

Розмітка HTML базується на певних правилах, що визначають, як потрібно позначати елементи документа. Вона включає чотири основні компоненти:

– елементи основні складові розмітки, що позначають різні частини документа;

– типи даних визначають типи вмісту атрибутів (символьні дані, числа тощо);

– мнемоніки спеціальні позначення, які дозволяють вставляти символи, що відсутні в таблиці кодування;

– декларація типу документа визначає версію HTML, яку використовують у документі [21].

Елементи HTML є основними одиницями розмітки. Кожен елемент має своє ім’я і описується через теги, які записуються в кутових дужках [22].

Загальна форма:

– відкриваючий тег: <назва елемента>;

– закриваючий тег: </назва елемента>.

Вміст елемента записується між відкриваючим і закриваючим тегами:

 Lorem ipsum [23]

Елементи можуть мати атрибути, що задаються всередині відкриваючого тегу. Атрибути описуються через назву та значення, яке записується в подвійних лапках:

 Lorem ipsum [24]

Окремі елементи можуть не мати вмісту і використовуються через одиночні теги:

– у HTML:
;

– у XHTML:
 [25].

Елементи HTML можна поділити на три основні категорії:

– структурні елементи – відповідають за визначення семантики та структури документа [26]. Вони включають елементи для тексту, списків, таблиць, гіперпосилань та інших об’єктів (зображення, аплети тощо) [27]. Ці елементи не впливають на зовнішнє оформлення тексту, але браузери використовують стандартні стилі для їх відображення [28]. Для більш детального стилізування рекомендується застосовувати CSS;

– презентаційні елементи – визначають вигляд документа, але не впливають на його зміст [29]. Багато таких елементів не рекомендується використовувати у сучасних вебсторінках. До цієї категорії належать елементи для стилізації тексту (шрифт, колір, насиченість тощо), додавання рамок та інших візуальних ефектів [30];

– елементи для інтерактивної взаємодії – дозволяють користувачам взаємодіяти з вебсторінкою. Це включає форми для введення даних, а також

скрипти для динамічних змін на сторінці. Вони забезпечують активні функції, які залежать від дій користувача [31].

Ці елементи забезпечують основу для розробки сучасних вебзастосунків, де кожен тип елементів відіграє важливу роль у забезпеченні функціональності та вигляду сторінок.

Каскадні таблиці стилів (CSS, від англ. Cascading Style Sheets) – це мова, яка використовується для опису вигляду вебсторінок, написаних за допомогою мов розмітки, таких як HTML або XML. CSS дозволяє визначати стильові характеристики таких елементів, як кольори, шрифти, верстка, а також інші аспекти візуального відображення сторінок. В основному CSS застосовується для представлення HTML та XHTML-сторінок, проте також може використовуватися для форматування інших типів XML-документів.

CSS замінив старішу технологію табличної верстки вебсторінок, що дозволило відокремити зміст документа від його візуальної презентації. Однією з основних переваг такої технології є можливість чітко розділяти структуру контенту (яка формується через HTML) та його зовнішнє оформлення (яке описується в CSS). Це дає змогу досягти більшої гнучкості при створенні та обслуговуванні вебсторінок, забезпечує кращу доступність контенту і дозволяє адаптувати його для різних пристроїв і умов перегляду.

Основні переваги використання CSS:

– адаптивність: Контент може бути адаптований для перегляду на різних пристроях (моніторах, мобільних пристроях, принтерах, пристроях для людей з вадами зору та ін.);

– економія простору: Вебсторінки стають менш об’ємними та більш структурованими завдяки відокремленню стилів від основного вмісту;

– покращення швидкості завантаження: Завдяки кешуванню стилів, що зберігаються браузером, CSS дозволяє швидше завантажувати сторінки, зменшуючи обсяги переданої інформації та навантаження на сервер.

– можливість централізованого управління стилями: Усі стилі для вебсторінок можуть бути зібрані в одному файлі, що дозволяє легко вносити зміни у вигляд сайту.

CSS має три основні типи стилі:

– стилі автора – стилі, надані розробником сайту, зазвичай вказуються в зовнішніх або внутрішніх таблицях стилів;

– стилі користувача – користувач може налаштувати свій файл стилів (наприклад, для покращення доступності чи комфорту);

– стилі браузера – стандартні стилі, які застосовуються за замовчуванням, якщо автор не визначив інші.

CSS використовує принцип каскадності, що означає, що стилі для елементів сторінки застосовуються залежно від їх пріоритету і послідовності визначення. Цей підхід дозволяє ефективно управляти стилями та забезпечувати їх гнучке застосування в різних умовах [32].

JavaScript (скорочено JS) – це потужна динамічна мова програмування, яка додає інтерактивні можливості вебсторінкам. Розроблена Бренданом Ейхом, співзасновником Mozilla, JavaScript був створений для забезпечення динамічності HTML-сторінок.

Це скриптова мова, що дозволяє вбудовувати код безпосередньо в HTML-документи. Сценарії на JavaScript є текстовими файлами, що дає змогу створювати їх за допомогою простого текстового редактора. Для виконання коду необхідно лише відкрити його в браузері, і він запрацює, надаючи сторінці динамічні функції, такі як обробка подій, зміна вмісту без перезавантаження сторінки, анімації та інші інтерактивні ефекти.

За допомогою JavaScript можна перетворити статичні HTML-документи на динамічні та інтерактивні:

– створення візуальних ефектів, таких як слайдери, галереї зображень або динамічний текст;

– перевірка даних користувача у формах перед їх надсиланням на сервер;

– автоматичне відкриття нових вікон для виведення інформації;

– зміна вмісту вікна браузера в реальному часі, залежно від дій користувача.

Дані в JavaScript представляють собою інформацію, яку зберігають програми. Основними типами даних є числа, рядки і булеві значення. В JavaScript значення можна зберігати в змінних, яким надаються імена. Для створення змінної використовують ключове слово var, після якого вказується ім’я змінної, наприклад: var name [33].

PHP – це популярна мова сценаріїв загального призначення з відкритим вихідним кодом. Аббревіатура PHP розшифровується як «Hypertext Preprocessor» (Препроцесор Гіпертексту). Однією з головних переваг PHP порівняно з такими мовами, як Perl та C, є здатність генерувати HTML-документи з вбудованими командами PHP [34].

PHP по суті є чимось середнім між компілятором і інтерпретатором і обробляє сценарії за таким принципом:

– спочатку подається сценарій на вхід PHP;

– потім він компілюється в байт-код (внутрішнє представлення);

– байт-код виконується, при цьому не створюється виконуваний файл.

Оскільки байт-код є компактнішим за звичайний код програми, його виконання здійснюється швидше.

Правила визначення змінних у PHP:

– ім’я змінної повинно починатися з знака долара $;

– ім’я змінної може містити лише латинські літери, цифри та знак підкреслення;

– імена змінних у PHP чутливі до регістру;

– змінну можна оголосити в будь-якому місці програми, але перед першим її використанням.

Змінні в PHP можуть містити дані будь-якого типу, за винятком констант, які можуть містити лише числа або рядки. Тип змінної визначається

інтерпретатором, однак іноді PHP може помилково присвоїти змінній неправильний тип. У таких випадках необхідно явно вказати тип змінної [35].

Інформаційна система (ІС) – це комплекс, який об’єднує інформаційні, програмні, технічні, мовні та організаційні компоненти, що призначені для збору, зберігання, обробки, передачі та пошуку даних. Залежно від функціональної значущості, ІС поділяються на інформаційно-пошукові системи та системи обробки даних [36].

Основною метою інформаційно-пошукових систем є знаходження необхідних даних і їх подальше виведення. Водночас системи обробки даних займаються оновленням даних та їх наданням користувачеві.

Між фізичною базою даних та користувачами знаходиться рівень програмного забезпечення, що включає систему управління базами даних (СУБД) [37].

СУБД – це набір мовних та програмних інструментів, призначених для створення, підтримки та ефективного використання бази даних багатьма користувачами одночасно.

Існують різні типи СУБД:

– промислові універсальні СУБД;

– спеціалізовані промислові СУБД;

– СУБД, розроблені для конкретних замовників.

Основні функції СУБД включають:

– управління даними, що зберігаються в зовнішній пам’яті;

– контроль оперативної пам’яті для підвищення швидкодії бази даних;

– управління транзакціями, що дозволяє забезпечити цілісність даних, захист від збоїв і підтримку багатокористувацького доступу;

– ведення журналу змін для підвищення надійності зберігання даних;

– підтримка сучасних мов баз даних.

Переваги використання СУБД:

– контроль за надмірністю та несуперечливістю даних;

– забезпечення спільного використання даних кількома користувачами;

– підтримка цілісності та безпеки даних;

– зростаюча ефективність при масштабуванні системи;

– підвищення доступності і готовності даних до використання;

– поліпшення продуктивності системи;

– спрощення технічного обслуговування та резервне копіювання.

Однак існують і недоліки СУБД:

– складність програмного забезпечення та висока вартість;

– додаткові витрати на апаратне забезпечення та перетворення даних;

– зниження продуктивності у разі неправильної настройки;

– серйозні наслідки при відмовах системи.

Банк даних є видом інформаційної системи, що побудований на основі концепції бази даних та СУБД. Варто зазначити, що поняття «банк даних» не є синонімом «бази даних», оскільки база даних є лише інформаційною складовою частиною банку даних [38].

Для розробки вебсайту були обрані такі технології: HTML, CSS, PHP, MySQL, JavaScript. Кожна з них була вибрана з огляду на свої переваги та можливості в контексті веброзробки.

HTML (HyperText Markup Language) є основною мовою розмітки для створення вебсторінок. Вона дозволяє створювати структуру документа та організовувати контент, включаючи текст, зображення, таблиці, списки та гіперпосилання. HTML дає змогу формувати структуру вебсторінки, яка потім може бути відображена користувачеві у зручному вигляді. Завдяки розмітці HTML можна легко інтегрувати мультимедійні елементи та форми, що є важливими для створення інтерактивних ігор. Важливою особливістю HTML є те, що вона є основою для взаємодії з іншими технологіями, такими як CSS та JavaScript.

CSS (Cascading Style Sheets) – це мова стилів, яка дозволяє визначати вигляд вебсторінок. Використання CSS дає змогу відокремити структуру контенту (HTML) від його візуального оформлення, що забезпечує більшу гнучкість і зручність у підтримці дизайну. За допомогою CSS можна

створювати адаптивний дизайн, що підлаштовується під різні пристрої, такі як мобільні телефони або планшети. CSS також покращує швидкість завантаження сторінок і дозволяє централізовано керувати стилями на всіх сторінках вебсайту, що є важливим при розробці великих ігрових платформ.

JavaScript – це мова програмування, яка дозволяє додавати інтерактивність і динамічні ефекти на вебсторінки. Вона дає змогу створювати такі функції, як слайдери, галереї, анімації, перевірка форм, а також зміни контенту без перезавантаження сторінки. JavaScript особливо корисний для створення інтерактивних ігор, оскільки дозволяє обробляти події користувача та змінювати елементи на сторінці в реальному часі. Це забезпечує високу динамічність і покращує взаємодію з користувачем.

PHP (Hypertext Preprocessor) – серверна мова програмування, яка дозволяє створювати динамічні вебсторінки. PHP може генерувати HTML-документи з вбудованими скриптами, що дає змогу обробляти дані користувачів, взаємодіяти з базами даних і виконувати інші серверні функції. Однією з ключових переваг PHP є його здатність працювати з формами, а також здійснювати обробку запитів і збереження даних на сервері. PHP активно використовується для розробки складних вебсайтів з багатьма користувачами, де важлива обробка та зберігання даних у реальному часі.

MySQL – це система управління базами даних, яка є однією з найпопулярніших для використання в веброзробці [39]. MySQL дозволяє зберігати та ефективно обробляти великі обсяги даних, що необхідно для підтримки ігор з великою кількістю користувачів. Завдяки інтеграції з PHP можна реалізувати зберігання даних про користувачів, курси, інструкторів, а також інші важливі аспекти системи [40].

На рис. 3.1 наведено схему використаних технологій веброзробки.

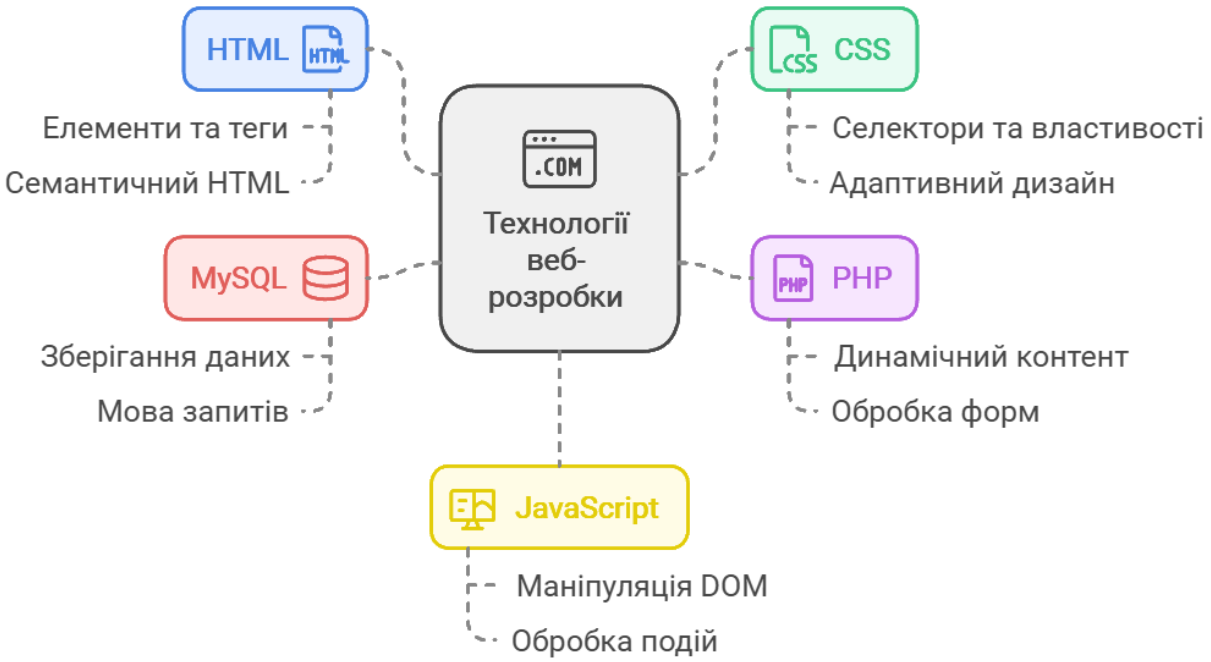

Рисунок 3.1 – Технології веброзробки

У цьому підпункті розглянуто основні технології, які були обрані для розробки вебсайту, зокрема HTML, CSS, PHP, MySQL та JavaScript. Кожна з цих технологій має свою важливу роль у забезпеченні функціональності та зручності використання вебресурсу.

HTML, як основна мова розмітки, дозволяє створювати структуру вебсторінок, що є основою для подальшої роботи з контентом. Вона дає змогу організувати текст, зображення, таблиці, списки та гіперпосилання, а також забезпечує інтеграцію з мультимедійними елементами та формами. Це робить HTML незамінним інструментом для побудови базової структури вебсторінок.

CSS дозволяє відокремити візуальне оформлення від структури контенту, забезпечуючи гнучкість і зручність у підтримці дизайну. Завдяки CSS можна створювати адаптивний дизайн, що підлаштовується під різні пристрої, а також покращувати швидкість завантаження сторінок і керувати стилями на всіх сторінках сайту.

JavaScript додає інтерактивності на вебсторінки, що є важливим для створення динамічних ігор та інтерактивних функцій. За допомогою цієї мови програмування можна створювати різноманітні ефекти, перевірку форм, а

також забезпечувати зміни контенту без необхідності перезавантаження сторінки.

PHP є серверною мовою програмування, яка дозволяє працювати з формами, зберігати дані на сервері та здійснювати взаємодію з базами даних. Це важливо для забезпечення динамічності вебсайту та обробки запитів від користувачів у реальному часі.

MySQL є потужною системою управління базами даних, що дозволяє зберігати та ефективно обробляти великі обсяги даних. Це є критичним для розробки великих ігрових платформ і вебсайтів з високим навантаженням.

Таким чином, поєднання цих технологій забезпечує створення функціонального, динамічного та адаптивного вебсайту, що відповідає вимогам сучасних вебресурсів та забезпечує зручність користування для кінцевих користувачів.

ВИСНОВКИ

У процесі проведеного дослідження були вивчені основні методи та технології вебтестування, а також проведений детальний аналіз їх застосування для технологій HTML, CSS, PHP, MySQL та JavaScript. Розробка та забезпечення високої якості вебсайтів та вебзастосунків стали критично важливими аспектами у сучасній розробці програмного забезпечення, оскільки вони забезпечують надійність, безпеку та продуктивність систем.

Наукова новизна роботи полягає в інтеграції генетичних алгоритмів, кластеризації, методів навчання з підкріпленням та математичних моделей для оптимізації тестування програмного забезпечення. Зокрема, запропоновано новий підхід до вибору ефективних тестів, зменшення їх кількості та адаптивного планування тестових стратегій на основі попереднього досвіду, що дозволяє значно підвищити ефективність і знизити витрати на тестування.

Вебтестування є важливим етапом у життєвому циклі розробки вебзастосунків, оскільки дозволяє виявити і виправити помилки на ранніх етапах, що може значно знизити витрати на виправлення дефектів у майбутньому. Вебтестування включає різні типи тестів, такі як функціональне, безпекове, тестування продуктивності та сумісності, які дозволяють забезпечити стабільність і коректну роботу вебсистеми в умовах реального навантаження. Для кожної з технологій, таких як HTML, CSS, PHP, MySQL, JavaScript, є специфічні методи тестування, що необхідно враховувати в процесі розробки.

Під час аналізу методів вебтестування було розглянуто як традиційні підходи, так і сучасні інструменти для автоматизації тестування. Окремо слід відзначити такі методи, як юніт-тестування для серверних технологій, тестування інтеграції для взаємодії різних компонентів системи, а також інструменти для тестування інтерфейсів користувача (UI) та навантаження на сервери. Методи автоматизації тестування, зокрема з використанням інструментів як Selenium, JUnit, PHPUnit, значно спрощують тестування

великих вебзастосунків і дозволяють скоротити час на виконання рутинних перевірок.

Вебтестування неможливо уявити без використання математичних підходів, зокрема моделей для оцінки ризиків та ймовірностей виникнення помилок. Технології тестування, які використовують математичні моделі, дозволяють оцінити ймовірність помилок в окремих частинах вебсистеми, прогнозувати її поведінку при різних навантаженнях і визначати критичні точки, що можуть потребувати особливої уваги. Наприклад, застосування теорії ймовірностей для моделювання ризиків допомагає визначити найбільш уразливі ділянки коду, які можуть спричинити серйозні збої в роботі вебсистеми.

У ході дослідження було показано, що ефективне тестування вебсайтів та вебзастосунків повинно проводитися на всіх етапах їхнього створення. Це включає як автоматизовані тести на етапі розробки, так і тести, що виконуються вручну, для перевірки поведінки вебсистеми в умовах реального використання. Важливим аспектом є тестування сумісності на різних пристроях, браузерах та операційних системах, що дозволяє забезпечити коректну роботу вебсайтів у різноманітних середовищах.

Вебтестування значно виграє від використання інструментів автоматизації. Зокрема, такі інструменти, як Selenium, JUnit, PHPUnit, а також спеціалізовані платформи для тестування продуктивності (наприклад, Apache JMeter), дозволяють автоматизувати великий обсяг тестів і проводити їх на постійній основі, що суттєво скорочує час на тестування та дозволяє виявляти помилки на ранніх етапах розробки. Інтеграція автоматизованих тестів в середовище безперервної інтеграції (CI/CD) дозволяє здійснювати тести на кожному етапі розробки, що гарантує постійну перевірку якості продукту.

Вебсистеми, особливо ті, що працюють з великими обсягами даних або забезпечують фінансові транзакції, потребують особливої уваги до безпеки. Для тестування безпеки необхідно використовувати спеціалізовані методи, такі як тестування на вразливості (SQL-ін’єкції, XSS-атаки), а також

інструменти для оцінки надійності шифрування та авторизації. Крім того, важливим аспектом є тестування продуктивності, яке включає перевірку здатності системи витримувати високе навантаження, а також оцінку часу відгуку і стабільності системи.

Розвиток нових технологій, таких як прогресивні вебзастосунки (PWA), а також інтеграція з мобільними застосунками, створює нові виклики для тестувальників. У майбутньому можна очікувати подальший розвиток інструментів автоматизації, які дозволять ще швидше і ефективніше проводити тести для сучасних вебсистем. Крім того, варто приділяти увагу тестуванню систем на основі штучного інтелекту та машинного навчання, що відкриває нові можливості для покращення якості тестування.

Загалом, вебтестування є невід’ємною частиною процесу розробки та підтримки вебсистем, яке забезпечує не лише високу якість продукту, але й безпеку та ефективність його роботи. Вибір методів тестування залежить від специфіки вебзастосунку та використовуваних технологій. Автоматизація тестування, математичні моделі для аналізу ризиків і оптимізації тестів, а також використання сучасних інструментів для безпеки та продуктивності дозволяють значно покращити ефективність і швидкість процесу тестування, забезпечуючи надійність і стійкість кінцевого продукту.

Результати дослідження апробовано у вигляді 1 тези доповіді під час VII Міжнародної студентської наукової конференції «Цифровізація науки та сучасні тренди її розвитку».

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Я. В. Іванчук, О. К. Галка Інформаційна технологія автоматичного тестування вебзастосунків. Серверна частина / в Матеріали конференції «LI Науково-технічна конференція підрозділів Вінницького національного технічного університету (2022)», Вінниця, 2022.

2. Fewster M. Software Test Automation / M. Fewster, D. Graham. – ACM Press, 1999. – 600 p.

3. Web site about Tests Automation. URL: http://automated-testing.info/tools (дата звернення: 06.10.2024)

4. Automation Testing Tutorial: What is Automated Testing URL: https://www.guru99.com/automationtesting.html (дата звернення: 06.10.2024)

5. Samanta, S. K., Achilleos, A., Moiron, S. R. F., Woods,J., and Ghanbari, M. (2010) Automatic languagetranslation for mobile SMS, International Journal of Information Communication Technologies and HumanDevelopment (IJICTHD) 2,43-58.12.

6. Sikuli: using GUI screenshots for search and automation / [Tom Yeh, Tsung-Hsiang Chang, Robert C. Miller] // In Proceedings of the 22nd annual ACM symposium on User interface software and technology (UIST '09). – ACM, NY, USA. – P. 183-192.

7. D. S. Rosenblum and E. J. Weyuker. Predicting the cost-effectiveness of regression testing strategies. In Proceedings of the ACMSIGSOFT '96 Fourth Symposium on the Foundations of Software Engineering, Oct. 1996.

8. H. Leung and L. White. A cost model to compare regression test strategies. In Proc. of Conf. on Software. Maint. Pages 201-208, Oct.1991.

9. J. Schoukens and R. Pintelon, Identification of Linear Systems: A Practical Guideline to Accurate Modeling, Pergamon Press, London (U.K.), 1991

10. Семенова І.Ю. Математичні моделі МСС. навчальний посібник: Київ, 2014 – с.82.

11. Аналіз та методи оцінки ризиків. URL: https://elib.lntu.edu.ua/sites/default/files/elib_upload/%D0%9D%D1%83%D0%B6%D0%BD%D0%B0%20%D0%B3%D0%BE%D1%82%D0%BE%D0%B2%D0%B8%D0%B9%202024/page10.html (дата звернення: 06.10.2024)

12. Особливості формування генетичних алгоритмів. URL: https://learn.ztu.edu.ua/mod/resource/view.php?id=113284 (дата звернення: 15.10.2024)

13. Навчання з підкріпленням у машинному навчанні. URL: https://evergreens.com.ua/ua/articles/reinforcement-learning.html (дата звернення: 15.10.2024)

14. Основні поняття кластеризації та постановка задачі. URL: http://csc.knu.ua/media/study/asp/mod_probl_inf_tech_sys_analysis_ivohin/lecture/lec11.pdf (дата звернення: 15.10.2024)

15. Штучний інтелект, машинне навчання та нейронні мережі: в чому різниця і для чого їх використовують. URL: https://evergreens.com.ua/ua/articles/machine-learning-overview.html

16. Класичне машинне навчання: завдання класифікації, узагальнення, кластеризації даних. URL: https://evergreens.com.ua/ua/articles/classical-machine-learning.html

17. Що таке концепція оптимізацій з обмеженнями? https://zpls.in.ua/shho-take-koncepciya-optimizacii-z-obmezhennyami/ (дата звернення: 15.10.2024)

18. Градієнтний спуск. URL: https://robotdreams.cc/uk/blog/331-gradiyentniy-spusk-algoritm-ta-priklad-na-python (дата звернення: 15.10.2024)

19. Брила А.Ю., Глебена М.І., Ломага М.М., Млавець Ю.Ю. Методичні вказівки для самостійної роботи студентів математичного факультету з дисципліни «Вступ до web-програмування». Вступ до web-програмування. Основи HTML. Ужгород, 2018. 72 с.

20. Мулеса О.Ю. Основи HTML та CSS. Лабораторний практикум. Ужгород, 2019. 53 с.

21. Цеслів О.В. Основи програмування та вебдизайн: Навч. посіб. К.,2020. 149с.

22. Баран С.В. Основи web-програмування: навч. посіб. Кривий Ріг, 2023. –316 с.

23. Паламарчук Є.А. Web-програмування. Методичні вказівки для виконання лабораторних та самостійних робіт бакалаврами напряму 6.050100– «Економічна кібернетика. Вінниця: ВНАУ, 2011. 77с.

24. Хайрова Н. Ф. Сучасні технології Web-програмування : навч. посібник.Нац. техн. ун-т «Харків. політехн. ін-т». Харків : Панов А. М., 2020. 112 с.

25. Проценко О.Б. Web-програмування та web-дизайн. Технологія XML: Навчальний посібник. - Суми: Видавництво СумДУ, 2009. 127 с.

26. Босько В.В. Web-програмування. Частина 1 (frontend) : навч. посіб. Кропивницький : ЦНТУ, 2022. 208 с.

27. Мельник Р. Програмування вебзастосувань (фронт-енд та бек-енд) Видавництво: Львівська політехніка, 2018, 248с.

28. Пасічник О.Г., Пасічник О.В., Стеценко І.В. Основи вебдизайну Видавництво: Вид група BHV, 2009, 336с.

29. Манако В., Манако Д., Данилова О., Войченко О.П. Основи будування сайтів Видавництво: Шкільний світ, 2006, 120с.

30. Зубик Л.В., Карпович. І.М., Степанченко О.М. Основи сучасних web-технологій. Частина 1. Навчальний посібник. Рівне: НУВГП, 2016. 290 с.

31. Романюк О.Н., Катєльніков Д.І., Косовець О. П. Вебдизайн і комп’ютерна графіка. Навчальний посібник. Вінниця: ВНТУ, 2007. 142 с.

32. Молчанов В. П. Основи проектування WEB-видань : навчальний посібник. Харків : ХНЕУ ім. С. Кузнеця, 2017. 159 с.

33. Молчанов В.П. Технології WEB-дизайну : конспект лекцій / В. П. Молчанов. Харків : Вид. ХНЕУ, 2011. 212 с.

34. Цвіркун Л.І. Глобальні комп’ютерні мережі. Програмування мовою PHP: навч. посібник. Д.: Національний гірничий університет, 2013. 239 с.

35. Титенко С. В. СКБД MySQL і доступ до БД в PHP. URL: http://www.znannya.org/labs/?view=mysqli-intro (дата звернення: 15.10.2024)

36. Огурцов В. В. Вебпрограмування на боці сервера за допомогою мови PHP : лабораторний практикум з навчальної дисципліни «Вебтехнології та вебдизайн» для студентів напряму підготовки 6.050101 «Комп'ютерні науки». Харків : ХНЕУ ім. С. Кузнеця, 2016. 132 с.

37. PHP + MySQL. URL: https://sites.znu.edu.ua/webprog/lect/1222.ukr.html (дата звернення: 15.10.2024)

38. Лосєв М. Ю. Бази даних : навчально-практичний посібник для самостійної роботи студентів. Харків : ХНЕУ ім. С. Кузнеця, 2018. 233 с.

39. Сидоренко В.В. Організація баз даних: Методичні вказівки до самостійної роботи студентів за спеціальностями 6.050102/123 «Комп'ютерна інженерія», 125 «Кібербезпека». Кропивницький: ЦНТУ, 2017. – 88 с.

40. Методичні вказівки до самостійної роботи (мова SQL) з дисципліни «Системи керування базами даних» для студентів спеціальностей 151 - «Автоматизація та комп'ютерно - інтегровані технології» всіх форм навчання. / Запоріжжя: НУ Запорізька політехніка, 2020.-28 с.