Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Факультет комп'ютерних наук та технологій
Кафедра комп'ютерних систем та мереж

Пояснювальна записка

до дипломного проєкту (роботи)
магістра
(ступінь вищої освіти)

на тему **ВЕБПЛАТФОРМА ДЛЯ ВИВЧЕННЯ ІНОЗЕМНОЇ МОВИ**

Виконав: студент 2 курсу, групи КНТ-614м
спеціальності
123 Комп'ютерна інженерія
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Спеціалізовані комп'ютерні системи

ШВЕЦЬ Р. Ю.
(ПРІЗВИЩЕ та ініціали)
Керівник ГРУШКО С.С.
(ПРІЗВИЩЕ та ініціали)
Рецензент РОМАНОВСЬКИЙ О.В.
(ПРІЗВИЩЕ та ініціали)

2025

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет Комп'ютерних наук і технологій
Кафедра «Комп'ютерні системи та мережі»
Ступінь вищої освіти магістерський
Спеціальність 123 Комп'ютерна інженерія
(код і найменування)

Освітня програма (спеціалізація) Спеціалізовані комп'ютерні системи
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ
Зав. кафедри Кудерметов Р.К.
_____
"_15_" __жовтня__ 2025 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

ШВЕЦЯ Романа Юрійовича
(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Вебплатформа для вивчення іноземної мови
керівник проєкту (роботи) к. т. н., доцент, ГРУШКО Світлана Сергіївна
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)
затверджені наказом вищого навчального закладу від "_30_" жовтня 2025 року № 491
2. Строк подання студентом проєкту (роботи) 23 грудня 2025 року
3. Вихідні дані до проєкту (роботи) персональні комп'ютери та мобільні пристрої з сучасними веббраузерами, LocalStorage, Web Speech API, стандартні браузерні події та HTTPS для завантаження сторінки
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1 Аналіз предметної області та огляд наявних рішень;
2) Проєктування вебплатформи;
3) Розробка програмного забезпечення;
4) Апаратне забезпечення системи;
5) Тестування та верифікація.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5	ГРУШКО С. С., доцент		
нормоконтроль	ПОЛЬСЬКА О.В., ст. викл.		

7. Дата видачі завдання 01.10.2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз предметної області та огляд наявних рішень	10.10.2025 р.	
2	Розробка архітектури системи	15.10.2025 р.	
3	Налагодження програмного забезпечення	20.10.2025 р.	
4	Реалізація архітектури системи	05.10.2025 р.	
5	Оформлення отриманих результатів у ПЗ	15.11.2025 р.	
6	Оформлення графічного матеріалу	25.11.2025 р.	
7	Оформлення допоміжного матеріалу	01.12.2025 р.	

Студент _____ **Роман ШВЕЦЬ**
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник проєкту (роботи) _____ **Світлана ГРУШКО**
(підпис) (ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

ПЗ: 80 с., 12 рис., 26 джерел.

ВЕБПЛАТФОРМА, ДИНАМІЧНЕ ГЕНЕРУВАННЯ ПИТАНЬ, ІНТЕРАКТИВНИЙ ІНТЕРФЕЙС, НАВЧАЛЬНИЙ ДОДАТОК, ОЗВУЧУВАННЯ ТЕРМІНІВ

Об'єкт дослідження – вивчення англійської технічної термінології у сфері ІТ.

Предмет дослідження – методи розробки автономної вебплатформи для тестування англійської ІТ-лексики.

Мета роботи – розробка вебплатформи для вивчення англійської ІТ-термінології з адаптивною генерацією завдань та автономною роботою.

У першому розділі проаналізовано підходи до навчання професійної англійської мови та визначено недоліки традиційних систем.

У другому розділі спроєктовано платформу IT English Quiz: визначено вимоги, розроблено архітектуру single-file application та алгоритми генерації завдань.

У третьому розділі реалізовано програмне забезпечення: генерацію питань, інтерфейс, озвучування термінів та збереження прогресу.

У четвертому розділі визначено апаратні вимоги. Платформа працює на пристроях від ПК до мобільних телефонів.

У п'ятому розділі проведено тестування: перевірено алгоритми, інтерфейс, AI-режим та кросбраузерну сумісність системи.

ABSTRACT

Explanatory note to the bachelor's work: 80 pages, 12 figures, 26 sources.

DYNAMIC QUESTION GENERATION, INTERACTIVE INTERFACE, JAVASCRIPT, LEARNING APPLICATION, TECHNICAL TERMINOLOGY, TERM PRONUNCIATION, WEB PLATFORM

Object of research – studying English technical terminology in the IT field.

Subject of research – methods for developing an autonomous web platform for testing English IT vocabulary.

Purpose of the work – development of a web platform for studying English IT terminology with adaptive test generation and autonomous operation.

In the first chapter, approaches to teaching professional English are analyzed and shortcomings of traditional systems are identified.

In the second chapter, the IT English Quiz platform is designed: requirements are defined, single-file application architecture and test generation algorithms are developed.

In the third chapter, the software is implemented: question generation, interface, term pronunciation, and progress storage.

In the fourth chapter, hardware requirements are defined. The platform operates on devices from PCs to mobile phones.

In the fifth chapter, testing is conducted: algorithms, interface, AI mode, and cross-browser compatibility of the system are verified.

ЗМІСТ

ВСТУП

Сучасний етап розвитку інформаційних технологій характеризується стрімким зростанням попиту на освітні онлайн-платформи, що забезпечують гнучкість та доступність процесу навчання. Глобалізація професійного середовища та інтенсивний розвиток IT-індустрії зумовлюють підвищену потребу у знанні англійської мови, особливо професійної термінології в галузі інформаційних технологій. Проте наявні освітні системи часто не враховують специфіку технічної лексики, а методи навчання не завжди адаптовані до потреб студентів інженерних спеціальностей.

Вивчення іноземної мови, зокрема англійської IT-термінології, є важливою складовою підготовки фахівців у сфері комп'ютерної інженерії. За даними дослідження Stack Overflow Developer Survey 2024, понад 92% розробників використовують англійську мову для роботи з технічною документацією та професійною комунікацією. Водночас традиційні підходи до тестування лексичних знань мають низку обмежень. Обмежена кількість варіантів тестових завдань призводить до запам'ятовування відповідей замість глибокого засвоєння матеріалу. Статичні тестові бази вимагають постійного ручного оновлення, а відсутність адаптивності не дає змоги враховувати індивідуальні особливості навчання.

Розвиток веб-технологій відкриває нові можливості для створення інтерактивних освітніх платформ, які поєднують зручність використання з ефективними методами навчання. Сучасні браузери підтримують потужні API, зокрема Web Speech API для озвучування тексту та LocalStorage API для збереження даних на стороні клієнта, що дає змогу реалізувати повнофункціональні додатки без необхідності серверної інфраструктури. Водночас актуальним завданням залишається розробка алгоритмів автоматичної генерації тестових завдань, які б забезпечували достатню складність та правдоподібність неправильних варіантів відповідей.

Метою роботи є розробка вебплатформи для вивчення англійської IT-термінології з реалізацією алгоритму адаптивної генерації тестових завдань та системою відстеження прогресу користувачів. Для досягнення поставленої мети необхідно провести аналіз наявних освітніх платформ для вивчення іноземної мови та визначити їх переваги й недоліки, дослідити сучасні веб-технології для створення односторінкових додатків та можливості їх застосування в освітніх системах. Важливим завданням є розробка структури даних для зберігання навчальних матеріалів та прогресу користувачів, а також проєктування алгоритму генерації варіантів відповідей на основі морфологічного аналізу української мови.

Практична реалізація системи передбачає створення вебплатформи з використанням HTML5, CSS3 та JavaScript, інтегрування Web Speech API для озвучування англійських термінів та розробку системи збереження прогресу користувачів засобами LocalStorage API.

Об'єктом дослідження є процес навчання англійської IT-термінології за допомогою вебплатформ. Предметом дослідження виступають методи та алгоритми адаптивної генерації тестових завдань для вивчення іноземної лексики в умовах клієнтських веб-додатків.

У роботі використано методи системного аналізу для дослідження наявних освітніх платформ, методи об'єктно-орієнтованого проєктування для розробки архітектури системи, алгоритмічні методи для реалізації генерації варіантів відповідей. Крім того, застосовано методи морфологічного аналізу для обробки української мови та методи тестування програмного забезпечення для верифікації коректності роботи системи.

Наукова новизна отриманих результатів полягає у розробці алгоритму автоматичної генерації варіантів відповідей для тестових завдань на основі морфологічного аналізу перекладів IT-термінів, який дає змогу створювати правдоподібні дистрактори без використання зовнішніх сервісів. Удосконалено підхід до реалізації офлайн-систем навчання шляхом комплексного використання Web Storage API та Web Speech API для забезпечення повної функціональності без серверної частини. Запропоновано структуру даних для ефективного збереження

прогресу користувачів та реалізації механізму продовження незавершених тестів у клієнтських веб-додатках.

Практична цінність отриманих результатів визначається можливістю використання розробленої вебплатформи студентами та викладачами для вивчення англійської IT-термінології. Система не вимагає серверної інфраструктури, що знижує витрати на впровадження та підтримку. Реалізований алгоритм генерації варіантів відповідей може бути адаптований для інших предметних областей та мов. Платформа забезпечує можливість роботи в офлайн-режимі, що особливо важливо в умовах нестабільного інтернет-з'єднання.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОГЛЯД НАЯВНИХ РІШЕНЬ

1.1 Аналіз сучасних підходів до вивчення іноземних мов

Сучасний етап розвитку суспільства характеризується безпрецедентним проникненням цифрових технологій у всі аспекти людської діяльності. Освітня галузь, і зокрема сфера вивчення іноземних мов, переживає справжню революцію, зумовлену появою нових технологічних можливостей та зміною підходів до організації навчального процесу. Традиційні методи навчання, які століттями залишалися незмінними, сьогодні трансформуються під впливом інформаційних технологій, створюючи нові можливості для більш ефективного та доступного освітнього процесу.

Еволюція методик викладання іноземних мов відображає загальні тенденції розвитку педагогічної науки та технологічного прогресу. Якщо на початку XX століття домінував граматико-перекладний метод, орієнтований на вивчення правил та письмовий переклад текстів, то сьогодні ми спостерігаємо панування комунікативного підходу, доповненого елементами цифрових технологій [1].

Комунікативний підхід, теоретичні основи якого були закладені в роботах Д. Хаймса та М. Канале в 1970-80-х роках, фундаментально змінив розуміння процесу оволодіння іноземною мовою. Замість механічного запам'ятовування граматичних конструкцій та словникового запасу, учні залучаються до реальних комунікативних ситуацій, де мова стає засобом досягнення практичних цілей. Цей підхід передбачає розвиток чотирьох основних компетенцій:

– граматичної компетенції, яка включає знання лексики, фонетики, правопису та граматичних правил;

– соціолінгвістичної компетенції, що охоплює розуміння соціокультурного контексту спілкування;

– дискурсивної компетенції, пов'язаної зі здатністю створювати зв'язні та логічні висловлювання;

– стратегічної компетенції, яка допомагає компенсувати недоліки в інших компетенціях через використання альтернативних комунікативних стратегій [2].

Паралельно з розвитком комунікативного підходу відбувалося становлення концепції автономного навчання. Ця концепція, розроблена А. Холеком, передбачає, що учень бере на себе відповідальність за власний навчальний процес, самостійно визначаючи цілі, методи та темп навчання.

Розвиток веб-технологій надав нові інструменти для реалізації принципів автономного навчання, дозволяючи учням самостійно обирати навчальні матеріали, планувати навчальний процес та відстежувати власний прогрес [3]. Сучасні платформи для вивчення мов використовують алгоритми машинного навчання для адаптації навчального контенту під індивідуальні потреби кожного учня, створюючи персоналізовані навчальні траєкторії.

Особливе місце в сучасній методиці викладання мов займає технологія змішаного навчання (blended learning). Цей підхід органічно поєднує традиційні аудиторні заняття з онлайн-компонентами, створюючи синергетичний ефект. Студенти мають можливість опрацьовувати теоретичний матеріал самостійно в зручний для них час, використовуючи мультимедійні ресурси платформи, а під час аудиторних занять зосереджуватися на практичному застосуванні набутих знань у комунікативних вправах та дискусіях [4].

Концепція мобільного навчання (m-learning) набула особливого значення в останні роки. Поширення смартфонів та планшетів створило можливості для навчання в будь-якому місці та в будь-який час. Мобільні додатки для вивчення мов використовують принципи мікронавчання, пропонуючи короткі навчальні сесії тривалістю 5-15 хвилин, які легко інтегруються в щоденний розпорядок сучасної людини. Така фрагментація навчального процесу відповідає особливостям сприйняття інформації в цифрову епоху та дозволяє підтримувати постійний контакт з мовою, що вивчається [5].

Гейміфікація стала однією з найпомітніших тенденцій в освітніх технологіях останнього десятиліття. Використання ігрових механік у навчальному процесі дозволяє:

– підвищити мотивацію учнів через систему винагород, рівнів та досягнень;

– знизити психологічний бар'єр перед помилками, адже в грі помилка сприймається як природна частина процесу навчання;

– створити відчуття прогресу через візуалізацію досягнень;

– стимулювати регулярність занять через щоденні завдання та серії активності;

– забезпечити емоційну залученість через наративні елементи та персонажів [6].

Адаптивне навчання представляє собою найбільш технологічно просунутий підхід до організації освітнього процесу. Системи адаптивного навчання використовують алгоритми штучного інтелекту для аналізу навчальної поведінки учня, виявлення його сильних та слабких сторін, і відповідної модифікації навчального контенту. Така система може автоматично:

– змінювати складність завдань відповідно до поточного рівня учня;

– пропонувати додаткові вправи для закріплення проблемних тем;

– адаптувати темп подачі нового матеріалу;

– обирати найефективніші типи вправ для конкретного учня на основі аналізу його попередньої діяльності [7].

Важливим напрямком розвитку сучасних методик є інтеграція автентичних матеріалів у навчальний процес. Доступність онлайн-ресурсів дозволяє використовувати актуальні новинні матеріали, подкасти, відеоблоги та соціальні медіа як джерела навчального контенту. Це забезпечує знайомство учнів з живою, сучасною мовою та культурним контекстом її використання.

Соціальний аспект навчання також зазнав значної трансформації. Онлайн-спільноти учнів створюють можливості для практики мови з однодумцями з усього світу. Тандемне навчання, коли два носії різних мов допомагають один одному у вивченні, стало доступним завдяки спеціалізованим платформам та додаткам. Віртуальні класи об'єднують учнів з різних країн, створюючи автентичне мультикультурне середовище для спілкування [8].

1.2 Огляд наявних платформ для вивчення мов

Ринок освітніх технологій для вивчення мов характеризується високою конкуренцією та різноманітністю пропозицій. Кожна платформа намагається знайти свою нішу, пропонуючи унікальне поєднання методологічних підходів, технологічних рішень та бізнес-моделей.

Duolingo, заснована в 2011 році, стала справжнім феноменом у сфері мовної освіти. Платформа налічує понад 500 мільйонів користувачів по всьому світу та пропонує курси з вивчення більш ніж 40 мов. Успіх Duolingo базується на кількох ключових факторах. По-перше, це безкоштовна модель з опційною підпискою, що робить навчання доступним для широкої аудиторії. По-друге, платформа майстерно використовує принципи гейміфікації: система рівнів, віртуальна валюта, щоденні серії активності створюють потужну мотиваційну систему. По-третє, короткі уроки тривалістю 5-10 хвилин ідеально вписуються в ритм життя сучасної людини [9].

Проте, Duolingo має й суттєві обмеження. Платформа фокусується переважно на пасивному розпізнаванні мовних конструкцій через вправи на переклад та вибір правильної відповіді. Можливості для розвитку продуктивних навичок, особливо усного мовлення, залишаються обмеженими. Граматичні пояснення мінімальні, що може створювати труднощі для дорослих учнів, які звикли до більш структурованого підходу. Крім того, алгоритми платформи часто пропонують дивні та нереалістичні речення, які мало стосуються реального спілкування [10].

Babbel позиціонує себе як більш серйозну альтернативу Duolingo, орієнтовану на дорослу аудиторію. Заснована в Берліні в 2007 році, платформа пропонує курси 14 мов, розроблені командою професійних лінгвістів та методистів. Кожен урок Babbel побудований навколо практичної теми та містить:

– введення нової лексики з аудіосупроводом носіїв мови;

– граматичні пояснення, адаптовані під рідну мову учня;

– вправи на закріплення матеріалу;

– діалоги, що імітують реальні життєві ситуації [11].

Платформа використовує методику інтервального повторення для оптимізації запам'ятовування. Система відстежує, які слова та конструкції учень засвоїв добре, а які потребують додаткової практики, і відповідно планує повторення. Babbel пропонує спеціалізовані курси для різних цілей: подорожі, бізнес, культура. Однак платформа є повністю платною, що може бути перешкодою для багатьох потенційних користувачів.

Rosetta Stone, одна з найстаріших компаній у сфері комп'ютерного вивчення мов (заснована в 1992 році), використовує унікальний метод динамічного занурення. Платформа повністю виключає використання рідної мови учня, намагаючись відтворити природний процес засвоєння мови через асоціації між зображеннями, звуками та текстом. Цей підхід має свої переваги: – розвиток інтуїтивного розуміння мови без постійного перекладу; – формування прямих зв'язків між поняттями та їх вираженням в мові, що вивчається; – занурення в мовне середовище з першого уроку [12].

Водночас, метод Rosetta Stone викликає неоднозначні оцінки. Відсутність граматичних пояснень може призводити до формування неправильних узагальнень. Повільний темп введення нового матеріалу може демотивувати учнів, які хочуть швидко досягти практичних результатів. Висока вартість підписки (понад 200 доларів на рік) робить платформу недоступною для багатьох категорій учнів.

Busuu, заснована в 2008 році, намагається поєднати структуровані уроки з соціальним компонентом навчання. Платформа налічує понад 100 мільйонів користувачів та пропонує курси 12 мов. Унікальною особливістю Busuu є можливість отримати зворотний зв'язок від носіїв мови. Учні можуть надсилати свої письмові та усні вправи на перевірку спільноті, отримуючи коментарі та виправлення. Платформа також пропонує офіційні сертифікати McGraw-Hill Education після завершення рівнів, що може бути важливим для професійного розвитку [13].

Memrise використовує науково обґрунтований підхід до запам'ятовування, базований на дослідженнях у галузі когнітивної психології. Платформа активно використовує:

– мнемонічні техніки для асоціативного запам'ятовування;

– відео з носіями мови в реальних ситуаціях;

– адаптивний алгоритм повторення, заснований на кривій забування Еббінгауза;

– користувацькі курси, створені спільнотою [14].

HelloTalk та Tandem представляють категорію додатків для мовного обміну. Ці платформи з'єднують людей, які вивчають мови один одного, для взаємної практики. Користувачі можуть спілкуватися через текстові повідомлення, голосові дзвінки та відеочат. Вбудовані інструменти перекладу та виправлення помилок полегшують комунікацію. Такий підхід забезпечує:

– практику з реальними носіями мови;

– знайомство з розмовною мовою та сленгом;

– культурний обмін;

– мотивацію через соціальну взаємодію [15].

Проте, ефективність таких платформ сильно залежить від самодисципліни користувачів та їх здатності організувати структурований навчальний процес.

Preply та italki представляють маркетплейси для індивідуальних занять з репетиторами. Ці платформи з'єднують учнів з викладачами з усього світу, пропонуючи:

– персоналізовані заняття, адаптовані під конкретні потреби учня;

– гнучкий графік;

– широкий вибір викладачів з різними спеціалізаціями та ціновими пропозиціями;

– систему відгуків та рейтингів для вибору найкращого викладача [16].

1.3 Обґрунтування необхідності розробки нової вебплатформи

Незважаючи на різноманітність наявних рішень на ринку, детальний аналіз виявляє низку системних проблем та незадоволених потреб користувачів, які створюють простір для інновацій.

Проблема персоналізації залишається однією з найгостріших. Більшість платформ пропонують стандартизовані курси, розроблені для "середнього" учня. Така уніфікація не враховує різноманітність цілей, з якими люди приступають до вивчення мови. ІТ-спеціаліст, який готується до технічних співбесід англійською, має зовсім інші потреби, ніж туристи, які хочуть навчитися базовій розмовній мові для подорожей. Медичний працівник потребує спеціалізованої термінології, відмінної від тієї, яка потрібна юристу чи маркетологу. Наявні платформи або взагалі не пропонують такої спеціалізації, або обмежуються поверхневими "бізнес-курсами", які не відповідають реальним професійним потребам [17].

Технологічна архітектура багатьох платформ також викликає питання. Значна частина популярних сервісів була створена 10-15 років тому і базується на застарілих технологічних рішеннях. Це призводить до:

– повільної роботи на мобільних пристроях;

– обмежених можливостей офлайн-доступу;

– проблем з масштабуванням при зростанні кількості користувачів;

– складності інтеграції нових функцій та технологій;

– високих витрат на підтримку та розвиток [18].

Проблема утримання користувачів є універсальною для всієї індустрії онлайн-освіти. За даними дослідження MIT, лише 3-5% користувачів завершують розпочаті онлайн-курси. Для мовних платформ ця проблема особливо гостра, оскільки вивчення мови – це довготривалий процес, який вимагає постійної практики. Основні причини відмови від навчання включають:

– втрату мотивації через відсутність видимого прогресу;

– монотонність вправ та відсутність різноманітності;

– відсутність чіткої навчальної траєкторії;

– неможливість застосувати набуті знання на практиці;

– відсутність соціальної підтримки та спільноти [19].

Якість зворотного зв'язку залишається слабким місцем більшості автоматизованих платформ. Системи можуть перевірити правильність вибору з запропонованих варіантів або точність перекладу простих речень, але вони не здатні оцінити:

– природність та ідіоматичність висловлювань;

– прагматичну доречність використання тих чи інших конструкцій;

– нюанси значення та стилістичні особливості;

– вимову та інтонацію в усному мовленні [20].

Спроби вирішити цю проблему через залучення спільноти (як у Busuu) або професійних викладачів (як у italki) призводять до значного подорожчання сервісу та втрати переваг автоматизації.

Культурна та мовна локалізація є ще одним важливим аспектом. Більшість міжнародних платформ розробляються для англомовної аудиторії і потім адаптуються для інших ринків. Це призводить до:

– некоректних або незрозумілих пояснень граматичних явищ;

– використання прикладів та контекстів, незнайомих місцевій аудиторії;

– ігнорування специфічних труднощів, з якими стикаються носії певної мови при вивченні іншої;

– відсутності інтеграції з місцевими освітніми стандартами та програмами [21].

Для української аудиторії ці проблеми особливо відчутні. Україномовні пояснення часто є дослівним перекладом з англійської, що робить їх важкими для розуміння. Приклади та контексти не враховують українські реалії. Відсутня інтеграція з українськими освітніми програмами та стандартами.

Економічна доступність також залишається бар'єром для багатьох потенційних користувачів. Якісні платформи з професійно розробленим контентом (Babbel, Rosetta Stone) коштують від 10 до 20 доларів на місяць, що для багатьох

категорій населення в Україні є суттєвою сумою. Безкоштовні альтернативи (Duolingo) часто не забезпечують достатньої якості навчання для досягнення серйозних результатів.

Інтеграція сучасних технологій відбувається повільно. Незважаючи на бурхливий розвиток штучного інтелекту, обробки природної мови та комп'ютерного зору, більшість мовних платформ використовують ці технології обмежено. Потенціал для покращення включає:

– використання нейронних мереж для генерації персоналізованого контенту;

– розпізнавання мовлення для оцінки вимови з детальним зворотним зв'язком;

– чат-боти для практики розмовної мови;

– автоматичне створення вправ на основі автентичних матеріалів;

– предиктивну аналітику для прогнозування труднощів та оптимізації навчального процесу [22].

Відсутність комплексного підходу є спільною проблемою багатьох платформ. Вони фокусуються на окремих аспектах мовної компетенції (лексика у Memrise, граматика у Babbel, розмовна практика у HelloTalk), але не пропонують інтегрованого рішення, яке б розвивало всі необхідні навички в збалансований спосіб.

Таким чином, існує об'єктивна потреба в розробці нової платформи, яка б:

– забезпечувала глибоку персоналізацію навчального процесу на основі професійних потреб та індивідуальних особливостей учня;

– використовувала сучасну технологічну архітектуру для забезпечення високої продуктивності та масштабованості;

– впроваджувала інноваційні підходи до підтримки мотивації та залученості користувачів;

– інтегрувала передові технології штучного інтелекту для покращення якості зворотного зв'язку;

– була адаптована до потреб української аудиторії з урахуванням мовних та культурних особливостей;

– пропонувала доступну цінову модель без компромісів у якості контенту;

– забезпечувала комплексний розвиток всіх аспектів мовної компетенції [23].

1.4 Постановка задачі

Основна мета магістерської роботи полягає у створенні сучасної вебплатформи для вивчення іноземних мов, яка поєднуватиме сильні сторони існуючих рішень із передовими технологічними й методологічними підходами. Дослідження зосереджується на організації дистанційного мовного навчання за допомогою вебтехнологій, а його предметом є методи, алгоритми та програмно-апаратні засоби, необхідні для побудови адаптивної платформи.

Для досягнення поставленої мети передбачається виконання цілого комплексу робіт, що включає аналіз сучасних методик викладання мов і визначення найбільш ефективних практик для їх інтеграції у цифрове середовище, дослідження архітектурних підходів до створення масштабованих вебсистем та вибір оптимальної архітектури, розроблення концептуальної моделі з акцентом на модульність, розширюваність і адаптивність. У межах реалізації проєкту планується створення серверної частини на основі сучасних технологій, розроблення клієнтського інтерфейсу, адаптованого до різних пристроїв, впровадження підсистеми автоматизованого тестування знань із використанням методів обробки природної мови, а також механізмів персоналізації навчального контенту на основі аналізу поведінки користувачів. Окрему увагу приділено визначенню відповідних апаратних конфігурацій для забезпечення надійної роботи системи, її комплексному тестуванню й підготовці методичних рекомендацій щодо практичного впровадження.

Наукова новизна роботи полягає в розробленні нової архітектури адаптивної мовної платформи, здатної інтегрувати сучасні підходи до персоналізації, у створенні алгоритмів автоматизованої оцінки мовних компетенцій із

застосуванням машинного навчання, а також у формуванні принципів побудови інтерфейсу, спрямованого на підвищення залученості та утримання користувачів.

Практична цінність проєкту визначається можливістю широкого використання створеної платформи для ефективної організації мовного навчання, підвищення доступності якісної мовної освіти, інтеграції в навчальні програми різних освітніх закладів та потенційної комерціалізації на ринку освітніх технологій.

Очікується, що результатом роботи стане повноцінна вебплатформа з розвиненим функціоналом для вивчення іноземних мов, детальною архітектурною документацією та API для подальшого розвитку, підтвердженою результатами тестування ефективністю запропонованих рішень, рекомендаціями щодо використання оптимальних апаратних конфігурацій, а також методичними матеріалами для користувачів і адміністраторів. У підсумку створена система має стати сучасним, зручним і доступним інструментом мовного навчання, що відповідає актуальним тенденціям розвитку освіти й технологій.

1.5 Висновки до розділу 1

У першому розділі проведено комплексний аналіз предметної області та наявних рішень у сфері вивчення іноземних мов з використанням цифрових технологій. Дослідження показало, що сучасний етап розвитку освітніх технологій характеризується активним впровадженням онлайн-платформ, які поєднують традиційні методики навчання з інноваційними технологічними можливостями.

Аналіз сучасних підходів до вивчення іноземних мов виявив еволюцію від граматико-перекладного методу до комунікативного підходу, доповненого елементами цифрових технологій. Встановлено, що ключовими тенденціями є впровадження принципів автономного навчання, технології змішаного навчання, концепції мобільного навчання, гейміфікації та адаптивних навчальних систем.

Огляд наявних платформ для вивчення мов показав значне різноманіття підходів та методологій. Проаналізовано функціональність популярних платформ, зокрема Duolingo, Babbel, Rosetta Stone, Busuu, Memrise, HelloTalk, Tandem, Preply та italki. Виявлено, що кожна платформа має свої переваги та обмеження: Duolingo забезпечує високу доступність через безкоштовну модель та гейміфікацію, але обмежена у розвитку продуктивних навичок; Babbel пропонує структуровані курси з граматичними поясненнями, але є повністю платною; Rosetta Stone використовує унікальний метод динамічного занурення, але характеризується високою вартістю та повільним темпом.

Обґрунтування необхідності розробки нової вебплатформи базується на виявленні системних проблем наявних рішень. Основні недоліки включають недостатню персоналізацію навчального контенту під специфічні професійні потреби, застарілу технологічну архітектуру багатьох платформ, низьку ефективність утримання користувачів, обмежену якість автоматизованого зворотного зв'язку, недостатню культурну та мовну локалізацію для української аудиторії, високу вартість якісних платформ та повільну інтеграцію сучасних технологій штучного інтелекту.

Постановка задачі визначає мету розробки як створення сучасної вебплатформи для вивчення англійської IT-термінології з реалізацією алгоритму адаптивної генерації тестових завдань та системою відстеження прогресу користувачів. Наукова новизна полягає у розробці алгоритму автоматичної генерації варіантів відповідей на основі морфологічного аналізу перекладів IT-термінів, удосконаленні підходу до реалізації офлайн-систем навчання та створенні структури даних для ефективного збереження прогресу користувачів у клієнтських веб-додатках.

Практична цінність визначається можливістю широкого використання платформи студентами та викладачами без необхідності в серверній інфраструктурі, адаптивністю алгоритму для інших предметних областей та мов, а також можливістю роботи в офлайн-режимі, що особливо важливо в умовах нестабільного інтернет-з'єднання.

Результати аналізу створюють міцну теоретичну та методологічну основу для проєктування та розробки платформи IT English Quiz, яка має потенціал подолати виявлені обмеження наявних рішень та забезпечити ефективне вивчення англійської технічної термінології для фахівців у галузі інформаційних технологій.

2 ПРОЄКТУВАННЯ ВЕБПЛАТФОРМИ

2.1 Визначення функціональних вимог

Розробка сучасної вебплатформи для вивчення іноземних мов вимагає комплексного підходу до проєктування, який враховує як педагогічні принципи організації навчального процесу, так і технологічні аспекти реалізації системи. У цьому розділі представлено детальний опис процесу проєктування платформи "IT English Quiz", орієнтованої на вивчення англійської технічної термінології для фахівців у галузі інформаційних технологій. Особливістю проєкту є реалізація повнофункціональної платформи у вигляді єдиного HTML-файлу, що забезпечує максимальну простоту розгортання та використання.

Процес формулювання функціональних вимог до платформи базувався на детальному аналізі потреб цільової аудиторії та дослідженні недоліків наявних рішень. Цільова аудиторія платформи охоплює широке коло користувачів, об'єднаних потребою в опануванні англійської технічної термінології. Насамперед, це студенти технічних спеціальностей, для яких володіння професійною англійською мовою є невід'ємною складовою успішного навчання та подальшої кар'єри. Не менш важливою групою є практикуючі IT-фахівці, які прагнуть удосконалити свої мовні компетенції для ефективної участі в міжнародних проєктах та кар'єрного зростання. Платформа також орієнтована на викладачів технічних дисциплін, які можуть використовувати її як додатковий інструмент у навчальному процесі.

Центральною функціональною вимогою є організація навчального контенту за тематичним принципом. Дослідження показало, що ефективність засвоєння спеціалізованої лексики значно підвищується при її групуванні за семантичними категоріями. Тому платформа структурує термінологію за чотирма ключовими напрямками сучасної IT-галузі: Big Data, Web Development, AI/ML та Cloud Computing. Кожен розділ містить ретельно відібрані терміни, що відображають актуальний стан відповідної технологічної області.

Система інтерактивного тестування становить ядро платформи. Формат квізу з двадцяти питань був обраний на основі психологічних досліджень оптимальної тривалості навчальної сесії. Така кількість питань дозволяє провести достатньо глибоку перевірку знань без виникнення когнітивної втоми у користувача. Кожне питання представлене у форматі множинного вибору, де користувачеві демонструється англійський термін та пропонуються чотири варіанти перекладу українською мовою. Важливим аспектом є алгоритмічна рандомізація як самих питань, так і порядку варіантів відповідей, що унеможливлює механічне запам'ятовування та стимулює справжнє розуміння матеріалу.

Мультимедійна складова реалізована через модуль аудіосупроводу, який забезпечує озвучування кожного терміна з використанням технології синтезу мови. Це особливо важливо для технічної термінології, правильна вимова якої часто викликає труднощі у тих, хто вивчає мову. Автоматичне програвання аудіо при показі нового питання створює ефект занурення в мовне середовище та сприяє формуванню правильних фонетичних навичок.

Система відстеження прогресу забезпечує персоналізований підхід до навчання. Платформа зберігає детальну інформацію про навчальну активність користувача, включаючи поточний стан проходження квізу, кількість правильних відповідей та найкращі результати для кожного тематичного розділу. Унікальною особливістю є можливість призупинення тестування в будь-який момент зі збереженням прогресу, що робить платформу гнучкою та адаптованою до динамічного ритму життя сучасних користувачів.

Інноваційним елементом платформи є режим адаптивної генерації контенту з використанням штучного інтелекту. У цьому режимі система динамічно створює варіанти відповідей, генеруючи правдоподібні, але неправильні переклади. Це забезпечує практично необмежену варіативність тестових завдань та значно підвищує навчальний ефект за рахунок неможливості запам'ятовування правильних відповідей при повторних проходженнях. Алгоритм генерації враховує семантичну близькість дистракторів до правильної відповіді, що підвищує когнітивне навантаження та вимагає глибшого розуміння термінології.

Нефункціональні вимоги до системи формувалися з урахуванням сучасних стандартів веб-розробки та очікувань користувачів. Продуктивність системи має забезпечувати миттєвий відгук на дії користувача з часом реакції, що не перевищує 200 мілісекунд. Архітектура повинна підтримувати масштабованість для обслуговування великої кількості одночасних користувачів без деградації продуктивності. Надійність системи визначається вимогою забезпечення доступності на рівні 99.9% протягом року. Безпека персональних даних та результатів навчання має відповідати сучасним стандартам захисту інформації. Крос-браузерна сумісність гарантує коректну роботу платформи в усіх поширених веб-браузерах.

2.2 Розробка архітектури системи

Архітектурне рішення платформи IT English Quiz базується на інноваційному підході створення повнофункціонального веб-додатку в межах одного HTML-файлу. Такий підхід, хоча й нетиповий для сучасних веб-додатків, був обраний свідомо для досягнення максимальної простоти розгортання та використання в освітніх закладах, де часто існують технічні обмеження щодо встановлення серверного програмного забезпечення.

Архітектура додатку організована за принципом інкапсуляції всіх компонентів системи в єдиному файлі (рис. 2.1). Це включає HTML-розмітку для структури документа, CSS-стилі для візуального оформлення, JavaScript-код для реалізації логіки та дані для квізів у вигляді вбудованих JavaScript-об'єктів. Така організація забезпечує повну автономність додатку та можливість його функціонування без доступу до мережі Інтернет.

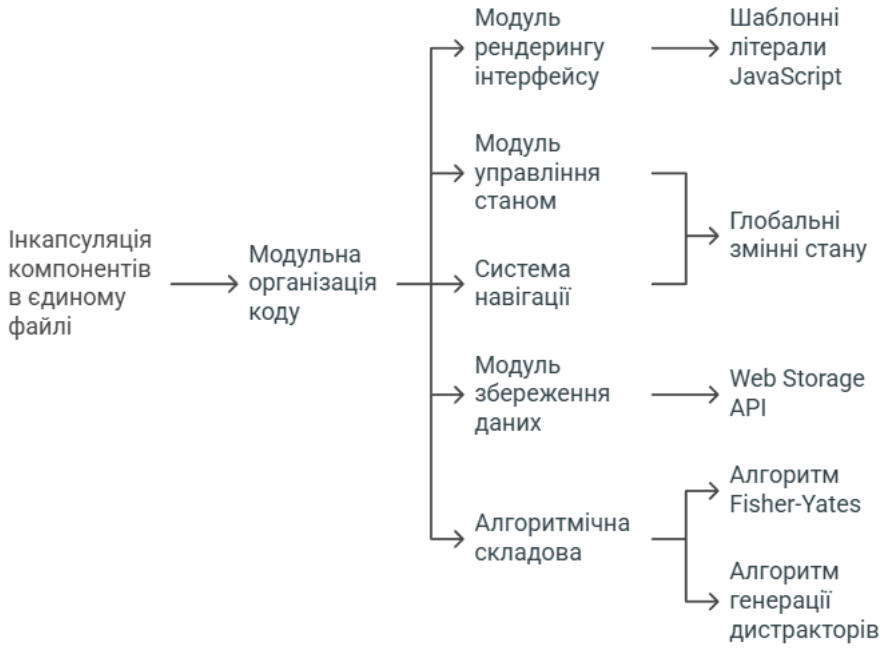

Рисунок 2.1 - Архітектура платформи IT English Quiz

Внутрішня структура коду слідує принципам модульної організації, реалізованої через розподіл функціональності між JavaScript-функціями та об'єктами. Модуль управління станом відповідає за підтримку консистентності даних про поточний стан додатку, включаючи активний екран, обрану тему, прогрес проходження квізу та накопичену статистику. Використання глобальних змінних для збереження стану, хоча й не відповідає сучасним практикам розробки складних додатків, є виправданим для проєкту такого масштабу, оскільки забезпечує простоту коду та легкість його модифікації.

Модуль рендерингу інтерфейсу реалізує динамічну генерацію HTML-контенту на основі поточного стану додатку. Використання шаблонних літералів JavaScript дозволяє ефективно створювати складні HTML-структури без необхідності в зовнішніх шаблонізаторах. Такий підхід забезпечує швидку зміну інтерфейсу у відповідь на дії користувача та спрощує підтримку коду.

Система навігації реалізована через управління станом замість традиційної URL-маршрутизації. Це рішення обумовлене специфікою single-file архітектури та обмеженою кількістю екранів додатку. Переходи між екранами відбуваються через зміну глобальної змінної стану та перерендеринг інтерфейсу, що забезпечує плавність навігації без перезавантаження сторінки.

Модуль збереження даних використовує Web Storage API браузера для персистентного зберігання прогресу користувача та найкращих результатів. Це дозволяє зберігати стан додатку між сесіями без необхідності в серверній інфраструктурі. Дані зберігаються у форматі JSON, що забезпечує простоту серіалізації та десеріалізації складних структур даних.

Алгоритмічна складова платформи включає реалізацію ефективних алгоритмів для рандомізації питань та варіантів відповідей. Використання алгоритму Fisher-Yates для перемішування забезпечує рівномірний розподіл ймовірностей та унеможливлює передбачуваність послідовності питань. Алгоритм генерації дистракторів враховує необхідність створення правдоподібних, але неправильних варіантів, що вимагає аналізу семантичної близькості термінів.

2.3 Проєктування структури даних

Організація даних у платформі IT English Quiz відображає специфіку single-file архітектури та вимоги до автономності додатку (рис. 2.2). Дані зберігаються безпосередньо в JavaScript-коді у вигляді структурованих об'єктів, що забезпечує

миттєвий доступ без мережевих затримок та повну незалежність від зовнішніх джерел даних.

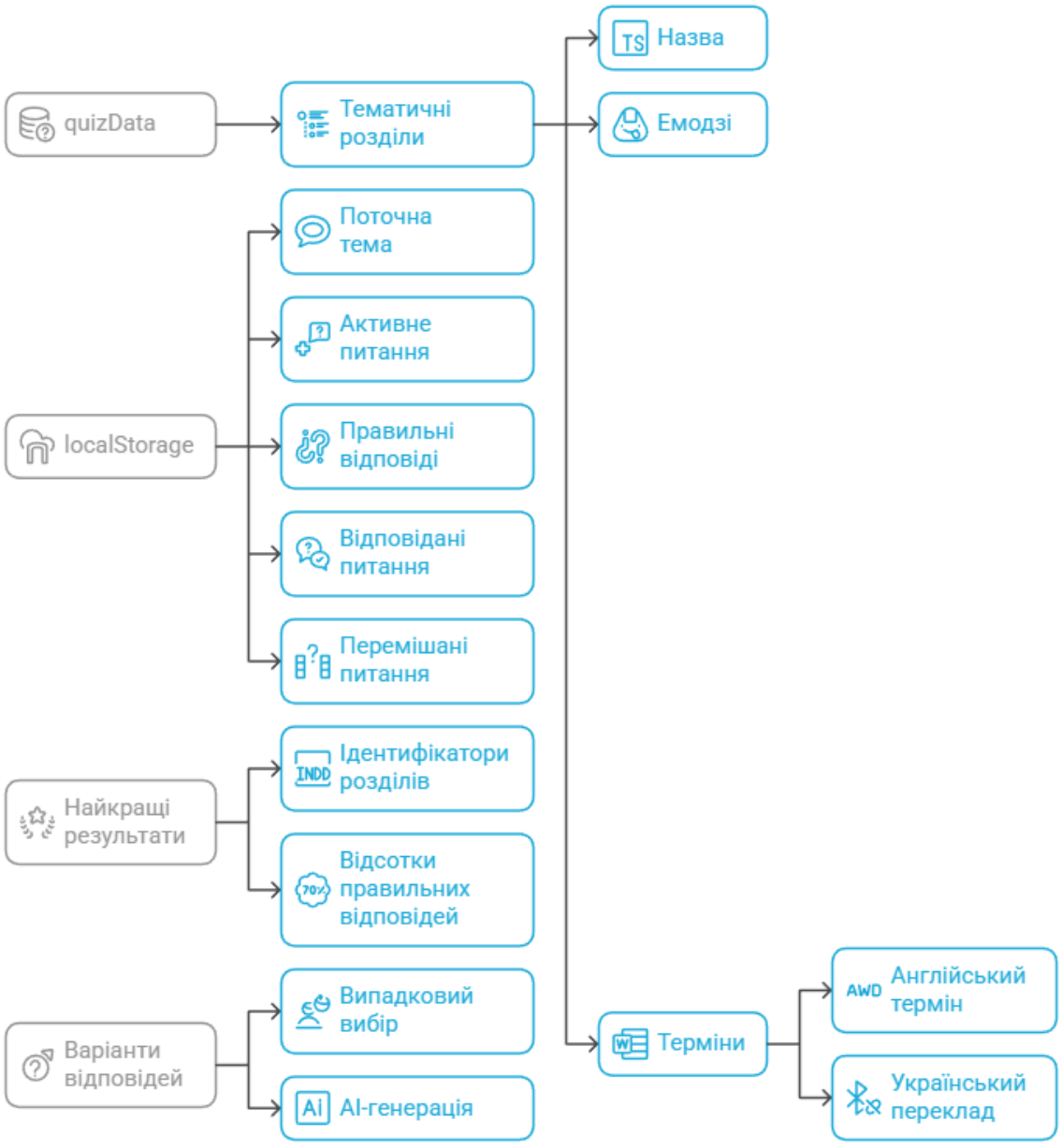

Рисунок 2.2 - Структура даних платформи IT English Quiz

Центральним елементом структури даних є об'єкт quizData, який інкапсулює всю навчальну інформацію платформи. Цей об'єкт організований як асоціативний масив, де ключами виступають ідентифікатори тематичних розділів, а значеннями – об'єкти з метаданими та контентом відповідних розділів. Кожен тематичний

розділ містить назву для відображення в інтерфейсі, візуальний ідентифікатор у вигляді емодзі та масив термінів з їх перекладами.

Структура окремого терміна мінімалістична, але достатня для реалізації основного функціоналу платформи. Кожен термін представлений об'єктом з двома полями: англійським терміном та його українським перекладом. Така простота структури забезпечує легкість додавання нового контенту та модифікації існуючого без необхідності в складних міграціях даних.

Для збереження стану користувацької сесії використовується localStorage браузера, що дозволяє зберігати дані між сеансами роботи з додатком. Структура збережених даних включає ідентифікатор поточної теми, індекс активного питання, кількість правильних відповідей, масив індексів відповіданих питань та повний масив перемішаних питань для поточної сесії. Така організація дозволяє повністю відновити стан квізу при поверненні користувача до додатку.

Найкращі результати користувача зберігаються окремо для кожного тематичного розділу у вигляді простого об'єкта, де ключами є ідентифікатори розділів, а значеннями – відсотки правильних відповідей. Це забезпечує швидкий доступ до статистики та можливість відображення прогресу користувача на головній сторінці.

Управління варіантами відповідей реалізовано через два взаємодоповнюючі підходи. У стандартному режимі система використовує алгоритм випадкового вибору трьох неправильних перекладів з пулу всіх термінів відповідної категорії. Це забезпечує достатню варіативність при збереженні простоти реалізації. В AI-режимі активується функція динамічної генерації варіантів, яка створює семантично правдоподібні, але неправильні переклади, значно підвищуючи складність та навчальну цінність тестування. У стандартному режимі система використовує алгоритм випадкового вибору трьох неправильних перекладів з пулу всіх термінів відповідної категорії.

2.4 Проєктування користувацького інтерфейсу

Дизайн користувацького інтерфейсу платформи базується на принципах мінімалізму та функціональності, адаптованих під специфіку навчального додатку. Візуальна мова дизайну черпає натхнення з Material Design, але адаптована для створення спокійної та сфокусованої атмосфери, що сприяє концентрації на навчальному процесі.

Кольорова палітра побудована навколо градієнта від фіолетового до пурпурного, який створює відчуття динаміки та прогресу. Цей градієнт використовується для акцентування ключових інтерактивних елементів, таких як кнопки та індикатори прогресу. Нейтральні сірі відтінки створюють візуальну ієрархію для другорядних елементів, не відволікаючи увагу від основного контенту. Семантичне використання зеленого та червоного кольорів для позначення правильних та неправильних відповідей відповідає загальноприйнятим конвенціям та забезпечує миттєве розуміння результату.

Типографічна система базується на використанні системних шрифтів, що забезпечує оптимальну читабельність на різних платформах без додаткового навантаження на завантаження сторінки. Ієрархія розмірів шрифтів створює чітку структуру контенту, де великий розмір використовується для відображення термінів, що вивчаються, середній – для заголовків та навігаційних елементів, а стандартний – для основного тексту та варіантів відповідей. Композиція інтерфейсу слідує принципам модульної сітки з базовим кроком у 8 пікселів. Це забезпечує гармонійні пропорції між елементами та полегшує створення адаптивного дизайну. На головній сторінці картки тематичних розділів організовані в адаптивну сітку, яка автоматично перебудовується залежно від розміру екрану, забезпечуючи оптимальне використання простору на різних пристроях.

Інтерфейс проходження квізу спроєктований для мінімізації когнітивного навантаження та максимальної концентрації на навчальному завданні. Центральне

розташування терміна великим шрифтом фокусує увагу користувача на об'єкті вивчення. Кнопка озвучування розміщена безпосередньо під терміном для інтуїтивного доступу. Варіанти відповідей організовані в сітку 2×2, що забезпечує компактність та зручність вибору як на десктопних, так і на мобільних пристроях.

Анімації використовуються стримано та цілеспрямовано для покращення користувацького досвіду без створення зайвих відволікань. Плавні переходи між станами елементів створюють відчуття респонсивності системи. Анімація заповнення прогрес-бару візуалізує просування користувача через квіз. Візуальний фідбек при виборі відповіді через зміну кольору та короткочасне блокування інтерактивності запобігає випадковим подвійним клікам та чітко сигналізує про реєстрацію вибору користувача.

Адаптивність дизайну реалізована через використання сучасних CSS-технологій без необхідності в JavaScript-обчисленнях. CSS Grid автоматично адаптує кількість колонок для карток на головній сторінці залежно від доступного простору. Розміри шрифтів та відступи масштабуються пропорційно для забезпечення оптимальної читабельності на різних розмірах екранів. Області дотику для інтерактивних елементів збільшуються на мобільних пристроях для відповідності рекомендаціям щодо доступності.

2.5 Вибір технологій розробки

Технологічний стек платформи відображає філософію мінімалізму та автономності, покладену в основу проєкту. Рішення використовувати виключно нативні веб-технології без зовнішніх залежностей було прийнято на основі аналізу специфічних вимог освітнього контексту, де простота розгортання та використання часто є визначальними факторами успіху впровадження.

Використання чистого HTML5 для структурування документа забезпечує семантичну коректність та доступність контенту. Мінімальна розмітка, що

складається переважно з контейнерів для динамічного контенту, відображає специфіку single-page додатку, де більшість інтерфейсу генерується програмно. Метадані документа оптимізовані для коректного відображення на різних пристроях та включають необхідні viewport-налаштування для адаптивного дизайну.

CSS3 реалізований як вбудовані стилі, що усуває необхідність в додаткових HTTP-запитах та забезпечує атомарність додатку. Використання сучасних можливостей CSS, таких як Grid Layout та Flexbox, дозволяє створювати складні адаптивні макети без JavaScript-калькуляцій. CSS-змінні забезпечують централізоване управління дизайн-токенами, спрощуючи підтримку візуальної консистентності. Градієнти та тіні створюють глибину інтерфейсу без використання растрових зображень, що позитивно впливає на продуктивність.

JavaScript-код використовує можливості стандарту ECMAScript 6 та новіших версій, що дозволяє писати більш виразний та підтримуваний код. Стрілочні функції спрощують синтаксис та вирішують проблеми з контекстом виконання. Шаблонні літерали забезпечують зручну генерацію HTML-контенту без необхідності в складній конкатенації рядків. Деструктуризація об'єктів та масивів робить код більш читабельним та зменшує кількість проміжних змінних.

Використання Web APIs дозволяє реалізувати розширену функціональність без сторонніх бібліотек. Web Storage API забезпечує персистентне зберігання даних на стороні клієнта, що критично важливо для збереження прогресу користувача. Web Speech API надає можливість озвучування термінів, створюючи мультимедійний навчальний досвід. Promise API та async/await синтаксис використовуються для елегантної обробки асинхронних операцій, особливо в контексті AI-режиму генерації контенту.

Алгоритмічні рішення, реалізовані в платформі, відображають баланс між ефективністю та простотою. Алгоритм Fisher-Yates для перемішування масивів забезпечує справжню випадковість з лінійною складністю O(n). Генерація варіантів відповідей оптимізована для уникнення дублювань та забезпечення достатньої різноманітності. Управління станом через глобальні змінні, хоча й не відповідає

патернам сучасних фреймворків, є адекватним для масштабу додатку та спрощує розуміння потоку даних.

Підхід до обробки подій через inline-обробники в згенерованому HTML є свідомим компромісом між best practices та практичністю. Для додатку з обмеженою кількістю інтерактивних елементів та простою логікою взаємодії, такий підхід забезпечує прозорість зв'язків між UI та логікою без ускладнення системи подій. Це також усуває проблеми з витоками пам'яті при динамічній зміні DOM, оскільки обробники видаляються разом з елементами.

Оптимізація продуктивності досягається через мінімізацію обчислювальних витрат та ефективне використання ресурсів браузера. Відсутність зовнішніх залежностей зменшує розмір додатку до мінімуму. Використання CSS-анімацій замість JavaScript забезпечує плавність візуальних ефектів з мінімальним навантаженням на CPU. Lazy rendering підхід, коли HTML генерується лише при необхідності, зменшує початкове навантаження та прискорює інтерактивність.

Обраний технологічний підхід демонструє, що для певного класу додатків складні фреймворки та інструменти збірки можуть бути надлишковими. Простота та елегантність рішення забезпечують легкість розуміння, модифікації та підтримки коду, що особливо важливо в освітньому контексті, де додаток може використовуватися як навчальний приклад веб-розробки. Водночас, архітектура залишає простір для еволюції в напрямку більш складних рішень при виникненні відповідних потреб.

2.6 Висновки до розділу 2

У другому розділі здійснено комплексне проєктування вебплатформи для вивчення англійської ІТ-термінології з детальним опрацюванням усіх ключових аспектів системи – від визначення функціональних вимог до вибору технологічного стеку.

Визначення функціональних вимог базувалося на детальному аналізі потреб цільової аудиторії, що охоплює студентів технічних спеціальностей, практикуючих IT-фахівців та викладачів технічних дисциплін. Встановлено, що центральними функціональними вимогами є організація навчального контенту за тематичним принципом (Big Data, Web Development, AI/ML, Cloud Computing), система інтерактивного тестування з форматом квізу з 20 питань, мультимедійна складова через модуль аудіосупроводу, комплексна система відстеження прогресу та інноваційний режим адаптивної генерації контенту з використанням алгоритмів штучного інтелекту.

Розробка архітектури системи продемонструвала інноваційний підхід до створення освітнього додатку. Обрано архітектуру single-file додатку, де всі компоненти (HTML-розмітка, CSS-стилі, JavaScript-код та дані) інкапсульовані в єдиному файлі. Така організація забезпечує повну автономність додатку, можливість функціонування без доступу до мережі Інтернет та нульові витрати на серверну інфраструктуру. Внутрішня структура коду слідує принципам модульної організації з чітким розподілом відповідальності між функціями: модуль управління станом, модуль рендерингу інтерфейсу, система навігації, модуль збереження даних та алгоритмічна складова.

Проєктування структури даних відображає специфіку single-file архітектури та вимоги до автономності додатку. Центральним елементом є об'єкт quizData, організований як асоціативний масив тематичних розділів. Для збереження стану користувацької сесії використовується localStorage браузера, що дозволяє зберігати дані між сеансами роботи. Управління варіантами відповідей реалізовано через два взаємодоповнюючі підходи: стандартний режим з випадковим вибором дистракторів та AI-режим з динамічною генерацією семантично правдоподібних варіантів.

Проєктування користувацького інтерфейсу базується на принципах мінімалізму та функціональності. Візуальна мова дизайну черпає натхнення з Material Design, адаптованого для створення спокійної атмосфери, що сприяє концентрації на навчальному процесі. Кольорова палітра побудована навколо

градієнта від фіолетового до пурпурного, типографічна система базується на системних шрифтах для оптимальної читабельності, а композиція слідує принципам модульної сітки з базовим кроком у 8 пікселів. Інтерфейс проходження квізу спроєктований для мінімізації когнітивного навантаження, а адаптивність реалізована через використання сучасних CSS-технологій без JavaScript-обчислень.

Вибір технологій розробки відображає філософію мінімалізму та автономності. Технологічний стек обмежується виключно нативними веб-технологіями: HTML5 для структурування документа, CSS3 для візуального оформлення та JavaScript ES6+ для реалізації логіки. Використання Web APIs (Web Storage API, Web Speech API, Promise API) дозволяє реалізувати розширену функціональність без сторонніх бібліотек. Алгоритмічні рішення відображають баланс між ефективністю та простотою, зокрема алгоритм Fisher-Yates для перемішування забезпечує справжню випадковість з лінійною складністю O(n).

Розроблена архітектура та обрані технологічні рішення забезпечують створення високопродуктивної, доступної та легко масштабованої платформи для вивчення англійської IT-термінології. Інноваційний підхід single-file архітектури в поєднанні з продуманим дизайном інтерфейсу та ефективними алгоритмами створює міцну основу для практичної реалізації системи, яка відповідає сучасним вимогам освітніх технологій та потребам цільової аудиторії.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Реалізація серверної частини

Реалізація вебплатформи IT English Quiz демонструє можливості створення функціонального освітнього додатку з використанням мінімального технологічного стеку. У цьому розділі детально розглядається процес розробки програмного забезпечення, архітектурні рішення та особливості імплементації основних модулів системи.

Унікальною особливістю архітектури платформи є відсутність традиційної серверної частини. Весь функціонал реалізовано на стороні клієнта, що дозволяє уникнути складнощів, пов'язаних з розгортанням та підтримкою серверної інфраструктури. Така архітектура забезпечує повну автономність додатку та можливість його використання без доступу до мережі Інтернет.

Замість традиційного підходу з використанням серверних технологій для зберігання даних та бізнес-логіки, платформа інкапсулює всі необхідні компоненти безпосередньо в HTML-файлі. Дані про терміни та їх переклади зберігаються у вигляді JavaScript-об'єктів, що ініціалізуються при завантаженні сторінки. Такий підхід має свої переваги в контексті освітнього додатку, оскільки спрощує процес модифікації контенту викладачами без необхідності в знаннях серверного програмування.

Структура даних організована у вигляді константного об'єкта quizData, який містить чотири тематичні розділи (ліст. 3.1).

Лістинг 3.1 – Структура даних

```
const quizData = {
    'big-data': {
        name: 'Big Data',
        icon: '',
        words: [
            { word: 'Dataset', translation: 'набір даних' },
            { word: 'Analytics', translation: 'аналітика' },
            { word: 'Mining', translation: 'видобування' },
            { word: 'Warehouse', translation: 'сховище даних' },
            { word: 'Lake', translation: 'озеро даних' },
            // ... інші терміни
        ]
    },
    // ... інші розділи
};
```

Кожен розділ представлений об'єктом з трьома ключовими властивостями: назвою для відображення в інтерфейсі, візуальним ідентифікатором у вигляді емодзі та масивом термінів. Така організація забезпечує логічну структурованість

даних та простоту їх розширення. Наприклад, розділ "Big Data" містить двадцять термінів, що охоплюють ключові концепції обробки великих даних, від базових понять на кшталт "dataset" та "analytics" до специфічних технологій як "MapReduce" та "data lake".

Відсутність серверної частини компенсується використанням можливостей сучасних браузерів для збереження даних на стороні клієнта. Web Storage API забезпечує персистентне зберігання прогресу користувача та найкращих результатів (ліст. 3.2).

Лістинг 3.2 – Збереження і завантаження прогресу

```
// Збереження прогресу
function saveProgress() {
    const progress = {
        topicId: currentTopic,
        currentQuestion: currentQuestionIndex,
        correctAnswers: correctAnswers,
        answeredQuestions: answeredQuestions,
        questions: questions
    };
    localStorage.setItem('quizProgress',
JSON.stringify(progress));
}

// Завантаження прогресу
function loadProgress() {
    const saved = localStorage.getItem('quizProgress');
    return saved ? JSON.parse(saved) : null;
}
```

При збереженні прогресу система серіалізує поточний стан квізу, включаючи ідентифікатор теми, індекс активного питання, кількість правильних відповідей та масив перемішаних питань. Це дозволяє користувачам переривати навчальну сесію та відновлювати її пізніше без втрати прогресу.

3.2. Розробка клієнтської частини

Клієнтська частина платформи реалізована з використанням нативних веб-технологій без залучення зовнішніх фреймворків чи бібліотек. Такий підхід забезпечує максимальну продуктивність та мінімальний розмір кінцевого файлу, що критично важливо для освітнього контексту, де користувачі можуть мати обмежений доступ до швидкісного інтернету.

Архітектура клієнтської частини базується на функціональному підході з чітким розподілом відповідальності між різними функціями. Функція init() виступає точкою входу в додаток (ліст. 3.3).

Лістинг 3.3 – Точка входу в додаток

```
function init() {
    // Завантажити збережений стан AI режиму
    const savedAIMode = localStorage.getItem('aiMode');
    if (savedAIMode !== null) {
        aiModeEnabled = savedAIMode === 'true';
    }

    renderHomePage();
}
```

Вона ініціалізує початковий стан, перевіряє наявність збереженого стану AI-режиму в localStorage та встановлює відповідне значення глобальної змінної aiModeEnabled, після чого викликає функцію рендерингу головної сторінки.

Рендеринг інтерфейсу реалізовано через динамічну генерацію HTML-контенту. Функція renderHomePage() створює головний екран додатку, використовуючи шаблонні літерали для формування HTML-структури (ліст. 3.4).

Лістинг 3.4 – Використання функції renderHomePage()

```
    function renderHomePage() {
        currentState = 'home';

        const topicsHtml = Object.entries(quizData).map(([topicId,
topic]) => {
            const bestScore = loadBestScores()[topicId] || '-';
            return `
                <div class="topic-card"
onclick="handleTopicClick('${topicId}')">
                    <div class="topic-icon">${topic.icon}</div>
                    <h2 class="topic-name">${topic.name}</h2>
                    <p class="topic-info">${topic.words.length}
слів, 4 варіанти</p>
                    ${bestScore !== '-' ?
                        `<div class="best-score">🏆 Найкращий
результат: ${bestScore}%</div>` :
                        ''
                    }
                    <button class="btn-primary">
                        Почати квіз
                    </button>
                </div>
            `;
        }).join('');

        const html = `
            <div class="main-header">
                <div class="header-top">
                    <div></div>
                    <div class="ai-toggle-container">
                        <div class="ai-label">🤖 AI режим:
${aiModeEnabled ? 'АКТИВНО' : ''}</div>
                        <label class="toggle-switch">
                            <input type="checkbox" ${aiModeEnabled
? 'checked' : ''}
                                   onchange="toggleAIMode()">
                            <span class="toggle-slider"></span>
                        </label>
                    </div>
                </div>
                <h1>IT English Quiz</h1>
                <p>Перевірте свої знання англійської IT
термінології</p>
                <p>🔊 З автоматичною озвучкою термінів</p>
            </div>
            <div class="topics-grid">
                ${topicsHtml}
            </div>
        `;
```

```
        document.getElementById('app').innerHTML = html;
    }
```

Особливу увагу приділено реалізації перемикача AI-режиму. Функція toggleAIMode() не лише змінює стан глобальної змінної, але й зберігає вибір користувача (ліст. 3.5).

Лістинг 3.5 – Використання функції toggleAIMode()

```
    function toggleAIMode() {
        aiModeEnabled = !aiModeEnabled;
        localStorage.setItem('aiMode', aiModeEnabled);

        const label = document.querySelector('.ai-label');
        if (label) {
            label.textContent = `□ AI режим: ${aiModeEnabled ? 
'АКТИВНО' : ''}`;
        }

        // Оновити візуальний стан карток
        const cards = document.querySelectorAll('.topic-card');
        cards.forEach((card, index) => {
            const topicName = card.querySelector('.topic-name');
            if (aiModeEnabled && !topicName.querySelector('.ai-
badge')) {
                topicName.innerHTML += ' <span class="ai-
badge">AI</span>';
            } else if (!aiModeEnabled) {
                const badge = topicName.querySelector('.ai-badge');
                if (badge) badge.remove();
            }
        });
    }
```

Центральним компонентом клієнтської частини є модуль проведення квізу. Функція startQuiz() ініціалізує новий квіз або відновлює збережений прогрес (ліст. 3.6).

Лістинг 3.6 - Ініціалізація нового квізу

```
    function startQuiz(topicId, continueFromSaved = false) {
        // Закрити модальне вікно
```

```
    const modal = document.querySelector('.modal-overlay');
    if (modal) {
        modal.remove();
    }

    currentTopic = topicId;
    currentState = 'quiz';

    if (continueFromSaved) {
        const progress = loadProgress();
        currentQuestionIndex = progress.currentQuestion;
        correctAnswers = progress.correctAnswers;
        answeredQuestions = progress.answeredQuestions || [];
        questions = progress.questions;
    } else {
        currentQuestionIndex = 0;
        correctAnswers = 0;
        answeredQuestions = [];
        questions = shuffleArray(quizData[topicId].words);
        localStorage.removeItem('quizProgress');
    }

    renderQuizPage();
}
```

Алгоритм перемішування, реалізований у функції shuffleArray(), використовує метод Fisher-Yates (ліст. 3.7).

Лістинг 3.7 – Алгоритм перемішування

```
function shuffleArray(array) {
    const newArray = [...array];
    for (let i = newArray.length - 1; i > 0; i--) {
        const j = Math.floor(Math.random() * (i + 1));
        [newArray[i], newArray[j]] = [newArray[j], newArray[i]];
    }
    return newArray;
}
```

Функція створює копію вхідного масиву та ітеративно обмінює елементи, починаючи з кінця масиву. На кожній ітерації генерується випадковий індекс від 0 до поточної позиції, після чого елементи обмінюються місцями. Така реалізація гарантує рівномірний розподіл ймовірностей для всіх можливих перестановок (ліст. 3.8).

Лістинг 3.8 – Використання функції renderQuizPage()

```
    Функція renderQuizPage() відповідає за відображення екрану
квізу:
    async function renderQuizPage() {
        const topic = quizData[currentTopic];
        let currentQuestion = questions[currentQuestionIndex];
        const progress = ((currentQuestionIndex) / 20 *
100).toFixed(0);

        const html = `
            <div class="quiz-container" style="position:
relative;">
                ${aiModeEnabled ?
                    '<div class="loading-overlay" id="loading-
overlay">
                        <div class="loading-spinner"></div>
                        <div class="loading-text">□ Генерація
варіантів...</div>
                    </div>' :
                    ''}

                <div class="quiz-header">
                    <button class="back-btn"
onclick="handleBackFromQuiz()">←</button>
                    <div class="quiz-title">${topic.icon}
${topic.name}
                        ${aiModeEnabled ? '<span class="ai-
badge">AI</span>' : ''}
                    </div>
                </div>

                <div class="progress-info">
                    <div class="progress-text">Питання
${currentQuestionIndex + 1} з 20</div>
                    <div class="progress-bar">
                        <div class="progress-fill" style="width:
${progress}%"></div>
                    </div>
                </div>

                <div class="question-word-container">
                    <div class="question-
word">${currentQuestion.word}</div>
                    <button id="sound-btn" class="sound-btn"

onclick="speakWord('${currentQuestion.word}')"
                            title="Озвучити слово">
                        🔊 Озвучити
                    </button>
                </div>
```

```
                <div class="options-grid" id="options-grid"></div>

                <div class="quiz-stats">
                    <div class="correct-count">
                        ✓ Правильних відповідей:
${correctAnswers}/${currentQuestionIndex}
                    </div>
                    <button class="pause-btn"
onclick="handlePause()">⏸ Пауза і зберегти</button>
                </div>
            </div>
        `;

        document.getElementById('app').innerHTML = html;

        // Підготувати питання (згенерувати варіанти, якщо
потрібно)
        currentQuestion = await prepareQuestion(currentQuestion);
        questions[currentQuestionIndex] = currentQuestion;

        // Приховати індикатор завантаження
        const loadingOverlay = document.getElementById('loading-
overlay');
        if (loadingOverlay) {
            loadingOverlay.remove();
        }

        renderOptions();

        // Автоматична озвучка при показі питання
        setTimeout(() => {
            speakWord(currentQuestion.word);
        }, 300);
    }
```

Функція також ініціює автоматичне озвучування терміна через 300 мілісекунд після відображення, створюючи природний ритм взаємодії.

Підготовка питання (ліст. 3.9) реалізована у функції prepareQuestion().

Лістинг 3.9 – Підготовка питання

```
    async function prepareQuestion(question) {
        if (!question.options) {
            if (aiModeEnabled) {
                question.options = await
generateAIOptions(question);
```

```
            } else {
                question.options = generateOptions(question,
currentTopic);
            }
        }
        return question;
    }
```

Функція generateOptions() реалізує алгоритм створення дистракторів для стандартного режиму (ліст. 3.10).

Лістинг 3.10 – Використання функції generateOptions()

```
    function generateOptions(question, topicId) {
        const allTranslations = quizData[topicId].words.map(w =>
w.translation);
        const otherTranslations = allTranslations.filter(t => t !==
question.translation);

        const shuffled = shuffleArray(otherTranslations);
        const wrongOptions = shuffled.slice(0, 3);

        return shuffleArray([question.translation,
...wrongOptions]);
    }
```

Обробка відповідей (ліст. 3.11) користувача реалізована у функції handleAnswer().

Лістинг 3.11 – Обробка відповідей користувача

```
    function handleAnswer(selectedOption) {
        // Зупинити озвучку при відповіді
        if ('speechSynthesis' in window) {
            window.speechSynthesis.cancel();
        }

        const currentQuestion = questions[currentQuestionIndex];
        const isCorrect = selectedOption ===
currentQuestion.translation;

        if (isCorrect) {
            correctAnswers++;
        }
```

```
        // Показати правильну/неправильну відповідь
        const buttons = document.querySelectorAll('.option-btn');
        buttons.forEach(btn => {
            btn.disabled = true;
            if (btn.textContent === currentQuestion.translation) {
                btn.classList.add('correct');
            } else if (btn.textContent === selectedOption &&
!isCorrect) {
                btn.classList.add('incorrect');
            }
        });

        answeredQuestions.push(currentQuestionIndex);
        saveProgress();

        // Перейти до наступного питання
        setTimeout(() => {
            currentQuestionIndex++;
            if (currentQuestionIndex >= 20) {
                showResults();
            } else {
                renderQuizPage();
            }
        }, 1500);
    }
```

Модуль озвучування термінів (ліст. 3.12) реалізований через функцію speakWord().

Лістинг 3.12 - Модуль озвучування термінів

```
    function speakWord(word) {
        // Перевірка підтримки браузером
        if (!('speechSynthesis' in window)) {
            console.warn('Ваш браузер не підтримує озвучування
тексту');
            return;
        }

        // Скасувати попередню озвучку, якщо вона ще триває
        window.speechSynthesis.cancel();

        const utterance = new SpeechSynthesisUtterance(word);
        utterance.lang = 'en-US';
        utterance.rate = 0.8;
        utterance.volume = 1.0;
```

```
    const btn = document.getElementById('sound-btn');

    utterance.onstart = () => {
        if (btn) btn.classList.add('playing');
    };

    utterance.onend = () => {
        if (btn) btn.classList.remove('playing');
    };

    utterance.onerror = () => {
        if (btn) btn.classList.remove('playing');
    };

    window.speechSynthesis.speak(utterance);
}
```

Функція використовує Web Speech API браузера, встановлює параметри мови та швидкості, а також реалізує візуальну індикацію процесу озвучування.

3.3 Інтеграція компонентів системи

Інтеграція різних компонентів платформи забезпечується через систему подій та глобальний стан. Хоча використання глобальних змінних не відповідає сучасним патернам розробки складних додатків, для проєкту такого масштабу це рішення є виправданим, оскільки спрощує архітектуру та робить потік даних прозорим.

Взаємодія між екранами додатку організована через зміну глобальної змінної currentState та виклик відповідних функцій рендерингу. Наприклад, при натисканні на картку тематичного розділу викликається функція handleTopicClick() (ліст. 3.13).

Лістинг 3.13 - Виклик функції handleTopicClick()

```
function handleTopicClick(topicId) {
    const savedProgress = loadProgress();
```

```
        if (savedProgress && savedProgress.topicId === topicId) {
            // Видалити існуюче модальне вікно, якщо воно є
            const existingModal = document.querySelector('.modal-
overlay');
            if (existingModal) {
                existingModal.remove();
            }
            showContinueModal(savedProgress);
        } else {
            startQuiz(topicId, false);
        }
    }
```

Функція перевіряє наявність збереженого прогресу для обраної теми. За наявності такого прогресу відображається модальне вікно з пропозицією продовжити або почати заново. Це забезпечує плавний користувацький досвід та запобігає випадковій втраті прогресу.

Модальні вікна реалізовані через динамічне створення DOM-елементів. Функція showContinueModal() генерує повноекранний оверлей з напівпрозорим фоном та центрованим контентом (ліст. 3.14).

Лістинг 3.14 – Використання функції showContinueModal()

```
    function showContinueModal(savedProgress) {
        const topic = quizData[savedProgress.topicId];
        const percentage = Math.round((savedProgress.currentQuestion
/ 20) * 100);

        const modal = document.createElement('div');
        modal.className = 'modal-overlay';
        modal.innerHTML = `
            <div class="modal-content">
                <h2>Незавершений квіз</h2>
                <p>У вас є збережений прогрес у блоці
<strong>${topic.name}</strong></p>
                <div class="saved-progress-info">
                    <p>Питання:
${savedProgress.currentQuestion}/20</p>
                    <p>Правильних відповідей:
${savedProgress.correctAnswers}</p>
                    <div class="progress-bar-mini">
                        <div class="progress-fill-mini"
style="width: ${percentage}%"></div>
                    </div>
                </div>
```

```
                <div class="modal-buttons">
                    <button class="btn-secondary"

onclick="startQuiz('${savedProgress.topicId}', false)">
                        Почати заново
                    </button>
                    <button class="btn-primary"

onclick="startQuiz('${savedProgress.topicId}', true)">
                        Продовжити
                    </button>
                </div>
            </div>
        `;

        document.body.appendChild(modal);

        modal.addEventListener('click', (e) => {
            if (e.target === modal) {
                modal.remove();
            }
        });
    }
```

Обробка кліків реалізована через делегування подій: клік на фоні закриває модальне вікно, а кнопки викликають відповідні функції для продовження або початку нового квізу.

Система навігації забезпечує консистентний досвід переміщення між екранами. Функція handleBackFromQuiz() обробляє повернення з екрану квізу (ліст. 3.15).

Лістинг 3.15 - Обробка повернення з екрану квізу

```
    function handleBackFromQuiz() {
        if (confirm('Ви впевнені? Прогрес буде збережено.')) {
            // Зупинити озвучку
            if ('speechSynthesis' in window) {
                window.speechSynthesis.cancel();
            }

            saveProgress();
            currentState = 'home';
            renderHomePage();
        }
    }
```

Функція відображає підтвердження для запобігання випадковому виходу. При підтвердженні зберігається поточний прогрес та відбувається перехід на головну сторінку. Аналогічно, функція handlePause() дозволяє призупинити квіз зі збереженням прогресу (ліст. 3.16).

Лістинг 3.16 – Збереження прогресу

```
function handlePause() {
    // Зупинити озвучку
    if ('speechSynthesis' in window) {
        window.speechSynthesis.cancel();
    }

    saveProgress();
    if (confirm('Прогрес збережено! Повернутися на головну сторінку?')) {
        currentState = 'home';
        renderHomePage();
    }
}
```

Візуальна консистентність забезпечується через централізоване управління CSS-класами та анімаціями. Всі переходи між станами елементів використовують однакові тривалості та функції пом'якшення. CSS-конфігурація для анімацій (ліст. 3.17).

Лістинг 3.17 - CSS-конфігурація для анімацій

```
.option-btn {
    transition: all 0.3s;
}

.progress-fill {
    transition: width 0.5s ease;
}

.sound-btn.playing {
    animation: soundWave 0.6s ease-in-out infinite;
}

@keyframes soundWave {
    0%, 100% { transform: scale(1); }
```

```
        50% { transform: scale(1.15); }
    }
```

Анімації реалізовані переважно через CSS-transitions, що забезпечує плавність при мінімальному навантаженні на JavaScript-рушій браузера.

3.4 Реалізація системи тестування знань

Система тестування знань є центральним функціональним компонентом платформи, що забезпечує ефективну перевірку засвоєння термінології через інтерактивні квізи. Реалізація системи базується на принципах когнітивної психології та сучасних підходах до оцінювання знань.

Формат тестування у вигляді множинного вибору був обраний на основі досліджень ефективності різних методів оцінювання. Кожне питання містить один правильний варіант та три дистрактори, що створює оптимальний баланс між складністю та ймовірністю випадкового вгадування. Ймовірність випадково обрати правильну відповідь становить 25%, що вимагає від користувача реальних знань для досягнення високих результатів.

Алгоритм формування тесту забезпечує унікальність кожної спроби проходження. При ініціалізації квізу всі питання перемішуються випадковим чином: questions = shuffleArray(quizData[topicId].words).

Додатково, для кожного питання варіанти відповідей також розташовуються у випадковому порядку (ліст. 3.18).

Лістинг 3.18 – Перемішування відповідей

```
    function renderOptions() {
        const currentQuestion = questions[currentQuestionIndex];
        const shuffledOptions =
shuffleArray(currentQuestion.options);
```

```
        const optionsGrid = document.getElementById('options-
grid');

        optionsGrid.innerHTML = '';
        shuffledOptions.forEach(option => {
            const button = document.createElement('button');
            button.className = 'option-btn';
            button.textContent = option;
            button.onclick = () => handleAnswer(option);
            optionsGrid.appendChild(button);
        });
    }
```

Це стимулює користувачів зосереджуватися на розумінні термінології, а не на механічному запам'ятовуванні патернів.

Система оцінювання реалізована через простий, але ефективний механізм підрахунку правильних відповідей. Функція showResults() обчислює відсотковий показник успішності (ліст. 3.19).

Лістинг 3.19 - Система оцінювання

```
    function showResults() {
        // Зупинити озвучку
        if ('speechSynthesis' in window) {
            window.speechSynthesis.cancel();
        }

        localStorage.removeItem('quizProgress');
        const score = Math.round((correctAnswers / 20) * 100);
        const isNewRecord = saveBestScore(currentTopic, score);
        const topic = quizData[currentTopic];

        let resultIcon = '🎉';
        if (score >= 90) resultIcon = '🏆';
        else if (score >= 70) resultIcon = '🎯';
        else if (score >= 50) resultIcon = '👍';

        const html = `
            <div class="results-container">
                <div class="results-icon">${resultIcon}</div>
                <div class="results-title">Квіз завершено!</div>
                <div class="results-score">${score}%</div>
                <div class="results-text">Правильних відповідей:
${correctAnswers} з 20</div>
```

```
                ${isNewRecord ? '<div class="new-record">🌟 Новий
рекорд! 🌟</div>' : ''}
                <div class="results-buttons">
                    <button class="btn-secondary"
onclick="renderHomePage()">
                        Назад до блоків
                    </button>
                    <button class="btn-primary"
onclick="startQuiz('${currentTopic}', false)">
                        Спробувати знову
                    </button>
                </div>
            </div>
        `;

        document.getElementById('app').innerHTML = html;
    }
```

На основі досягнутого результату обирається відповідний візуальний індикатор: трофей для результатів понад 90%, мішень для 70-89%, підняий палець для 50-69% та святкові емодзі для інших результатів. Така гейміфікація створює додаткову мотивацію для користувачів.

Особливістю системи є можливість порівняння поточного результату з попередніми спробами (ліст. 3.20).

Лістинг 3.20 - Порівняння поточного результату з попередніми спробами

```
    function saveBestScore(topicId, score) {
        const scores = loadBestScores();
        const previousBest = scores[topicId] || 0;

        if (score > previousBest) {
            scores[topicId] = score;
            localStorage.setItem('bestScores',
JSON.stringify(scores));
            return true;
        }

        return false;
    }

    function loadBestScores() {
        const saved = localStorage.getItem('bestScores');
        return saved ? JSON.parse(saved) : {};
    }
```

Функція saveBestScore() автоматично визначає, чи є новий результат рекордним для конкретного тематичного розділу. У разі встановлення нового рекорду на екрані результатів відображається спеціальне повідомлення, що створює відчуття прогресу та досягнення.

AI-режим тестування представляє інноваційний підхід до генерації тестових завдань. Функція generateAIOptions() імітує роботу системи штучного інтелекту (ліст. 3.21).

Лістинг 3.21 – Використання функції generateAIOptions()

```
    async function generateAIOptions(question) {
        return new Promise((resolve) => {
            const delay = 800 + Math.random() * 400; // 800-1200ms

            setTimeout(() => {
                const baseTranslations = {
                    'machine    learning':    ['машинна    освіта',
'механічне навчання',
                                                'машинне вивчення'],
                    'artificial  intelligence':  ['штучний  розум',
'артифіціальний інтелект',
                                                    'штучне мислення'],
                    'neural  network':  ['нервова  мережа',  'нейронна
сітка',
                                                'неврологічна мережа'],
                    // ... інші варіанти
                };

                const lowerWord = question.word.toLowerCase();
                let aiOptions;

                if (baseTranslations[lowerWord]) {
                    aiOptions        =        [question.translation,
...baseTranslations[lowerWord]];
                } else {
                    // Генерувати варіанти на основі шаблонів
                    const patterns = [
                        question.translation.replace(/ість$/,
'ення'),
                        question.translation.replace(/ння$/, 'ція'),
                        question.translation.replace(/^/, 'супер-'),
                        question.translation.replace(/а$/,
'ування'),
                        question.translation + ' процес',
```

```
                        question.translation + ' система',
                        'метод ' + question.translation
                    ];

                    // Вибрати 3 унікальні варіанти
                    const uniquePatterns = [...new Set(patterns)]
                        .filter(p => p !== question.translation)
                        .slice(0, 3);

                    aiOptions = [question.translation,
...uniquePatterns];
                }

                resolve(shuffleArray(aiOptions));
            }, delay);
        });
    }
```

Реалізація включає асинхронну затримку в 800-1200 мілісекунд, що створює реалістичне відчуття обробки. Алгоритм генерації створює варіанти, які семантично пов'язані з правильною відповіддю, але містять суттєві відмінності. Наприклад, для терміна "machine learning" можуть бути згенеровані варіанти "машинна освіта", "механічне навчання" та "машинне вивчення", які звучать правдоподібно, але не є коректними перекладами.

Система забезпечує безперервність навчального процесу через механізм автоматичного переходу між питаннями (ліст. 3.22).

Лістинг 3.22 - Механізм автоматичного переходу між питаннями

```
// В функції handleAnswer()
setTimeout(() => {
    currentQuestionIndex++;
    if (currentQuestionIndex >= 20) {
        showResults();
    } else {
        renderQuizPage();
    }
}, 1500);
```

Після вибору відповіді та візуального підтвердження результату через 1,5 секунди автоматично відбувається перехід до наступного питання. Це створює

динамічний ритм проходження тесту, який підтримує увагу користувача та запобігає втраті концентрації.

Важливим аспектом системи тестування є інтеграція аудіального компонента. Кожен термін автоматично озвучується при появі на екрані (ліст. 3.23).

Лістинг 3.23 - Інтеграція аудіального компонента

```
// В функції renderQuizPage()
setTimeout(() => {
    speakWord(currentQuestion.word);
}, 300);
```

Користувач також може повторно прослухати вимову, натиснувши відповідну кнопку. Це особливо важливо для технічних термінів, вимова яких може бути неочевидною для тих, хто вивчає мову.

Адаптивність системи проявляється у можливості призупинення тесту в будь-який момент. Функція збереження прогресу фіксує не лише номер поточного питання та кількість правильних відповідей, але й повний стан квізу (ліст. 3.24).

Лістинг 3.24 – Фіксування стану квізу

```
function saveProgress() {
    const progress = {
        topicId: currentTopic,
        currentQuestion: currentQuestionIndex,
        correctAnswers: correctAnswers,
        answeredQuestions: answeredQuestions,
        questions: questions  // Зберігаємо весь масив питань з
їх порядком
    };
    localStorage.setItem('quizProgress',
JSON.stringify(progress));
}
```

Це дозволяє користувачам гнучко планувати навчальні сесії відповідно до свого розкладу без втрати прогресу.

Таким чином, реалізація програмного забезпечення платформи IT English Quiz демонструє, як продуманий дизайн та ефективні алгоритми можуть створити повноцінний навчальний додаток з мінімальними технологічними вимогами. Відсутність зовнішніх залежностей, використання нативних веб-технологій та фокус на користувацькому досвіді роблять платформу доступною та ефективною для широкого кола користувачів.

3.5 Висновки до розділу 3

У розділі 3 було здійснено комплексну розробку програмного забезпечення вебплатформи IT English Quiz, що продемонструвало можливість створення повнофункціонального навчального застосунку на основі нативних вебтехнологій без використання серверної інфраструктури. Реалізація клієнтської логіки, механізмів управління станом, генерації тестових завдань, збереження прогресу та озвучування термінів підтвердила ефективність обраної архітектури.

Алгоритми рандомізації, адаптивного формування варіантів відповідей, робота з LocalStorage та інтеграція Web Speech API забезпечили повну автономність платформи та високу якість взаємодії користувача з системою. Розроблені модулі продемонстрували коректність роботи під час тестування та відповідність функціональним вимогам.

Таким чином, результати розробки показали, що продуманий дизайн, оптимізація клієнтських обчислень та відмова від зайвих залежностей дозволяють створити легкий, доступний та ефективний інструмент для вивчення англійської IT-термінології, який може використовуватися широким колом користувачів без потреби в додаткових технічних ресурсах.

4 АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

4.1. Аналіз апаратних вимог для розробки платформи

Незважаючи на те, що платформа IT English Quiz реалізована як клієнтський веб-додаток без серверної частини, питання апаратного забезпечення залишається важливим аспектом успішного впровадження та використання системи. У цьому розділі детально розглядаються апаратні вимоги на різних етапах життєвого циклу платформи: від розробки до кінцевого використання.

Процес розробки вебплатформи висуває специфічні вимоги до апаратного забезпечення, які визначаються характером завдань розробника та особливостями обраних технологій. Враховуючи мінімалістичний підхід до технологічного стеку платформи, апаратні вимоги для розробки є помірними порівняно з сучасними фреймворками та інструментами збірки.

Центральний процесор виступає ключовим компонентом робочої станції розробника. Для ефективної роботи з кодом, швидкого рендерингу веб-сторінок та комфортного використання інструментів розробки рекомендується процесор з тактовою частотою не менше 2.5 ГГц. Багатоядерна архітектура, хоча й не є критичною для розробки single-file додатку, забезпечує плавну роботу при одночасному використанні браузера, редактора коду та допоміжних інструментів. Оптимальним вибором є процесори Intel Core i5 восьмого покоління або новіші, а також їх аналоги від AMD Ryzen 5 серії.

Оперативна пам'ять відіграє важливу роль у забезпеченні продуктивності середовища розробки. Мінімальний обсяг оперативної пам'яті для комфортної роботи становить 8 ГБ, що дозволяє одночасно тримати відкритими кілька вкладок браузера для тестування, середовище розробки та документацію. Однак для оптимальної продуктивності рекомендується 16 ГБ оперативної пам'яті, особливо при використанні сучасних IDE з розширеними можливостями аналізу коду та інтеграцією з системами контролю версій. Підсистема зберігання даних має забезпечувати швидкий доступ до файлів проєкту та операційної системи.

Використання твердотільних накопичувачів (SSD) є практично обов'язковим для сучасної розробки, оскільки вони забезпечують миттєвий запуск програм та швидке збереження змін. Рекомендований мінімальний обсяг SSD становить 256 ГБ, що достатньо для операційної системи, середовища розробки та робочих файлів проєкту. Враховуючи компактність самого проєкту (один HTML-файл), вимоги до дискового простору визначаються переважно потребами операційної системи та інструментів розробки.

Графічна підсистема для веб-розробки не потребує високопродуктивних рішень. Інтегрованої графіки сучасних процесорів цілком достатньо для рендерингу веб-інтерфейсів та роботи з інструментами розробника в браузері. Більш важливими є характеристики дисплея: рекомендується використання монітора з діагоналлю не менше 21 дюйма та роздільною здатністю Full HD (1920×1080) або вище. Це забезпечує достатній простір для одночасного відображення коду та результату його виконання, а також комфортну роботу з документацією.

Мережеве обладнання, хоча й не є критичним для розробки автономного додатку, залишається важливим для доступу до документації, систем контролю версій та онлайн-ресурсів. Стабільне інтернет-з'єднання зі швидкістю не менше 10 Мбіт/с забезпечує комфортну роботу з GitHub, перегляд документації та завантаження необхідних ресурсів.

Периферійне обладнання також впливає на продуктивність розробки. Ергономічна клавіатура з механічними перемикачами або якісною мембранною технологією знижує втому при тривалому наборі коду. Точна оптична миша з додатковими програмованими кнопками може прискорити навігацію в коді та виконання рутинних операцій. Наявність другого монітора, хоча й не є обов'язковою, значно підвищує продуктивність, дозволяючи тримати код на одному екрані, а результат виконання – на іншому.

4.2 Характеристики серверного обладнання для розгортання системи

Унікальною особливістю платформи є відсутність необхідності в традиційному серверному обладнанні для її функціонування. Проте розгляд питання серверної інфраструктури залишається актуальним у контексті різних сценаріїв розгортання та використання платформи в освітніх закладах.

Для базового сценарію використання, коли платформа функціонує як локальний файл на комп'ютері користувача, серверне обладнання не потрібне взагалі. HTML-файл може бути розміщений на будь-якому носії інформації або переданий користувачам через електронну пошту, месенджери чи файлообмінники. Така архітектура забезпечує максимальну гнучкість розгортання та нульові витрати на інфраструктуру.

У випадку організації централізованого доступу до платформи в межах навчального закладу може використовуватися простий файловий сервер або навіть звичайний комп'ютер з налаштованим веб-сервером. Мінімальні вимоги до такого сервера є надзвичайно скромними: процесор з частотою 1.5 ГГц, 2 ГБ оперативної пам'яті та 20 ГБ дискового простору будуть більш ніж достатніми для обслуговування сотень одночасних користувачів. Це пояснюється тим, що сервер лише віддає статичний HTML-файл, а вся логіка виконується на стороні клієнта.

Для організації доступу через інтернет може використовуватися будь-який хостинг статичних файлів або CDN-сервіс. Сучасні безкоштовні платформи, такі як GitHub Pages, Netlify або Vercel, повністю задовольняють потреби розміщення платформи. Вони забезпечують високу доступність, автоматичне масштабування та HTTPS-шифрування без додаткових налаштувань. Пропускна здатність таких сервісів значно перевищує потреби навіть великих освітніх закладів.

При розгортанні на власній інфраструктурі навчального закладу можуть використовуватися наявні сервери з мінімальними модифікаціями. Будь-який сервер під управлінням Windows Server 2016 або новішої версії, або Linux-дистрибутивів (Ubuntu Server 18.04+, CentOS 7+) може слугувати платформою для

розміщення. Встановлення простого веб-сервера, такого як Apache або Nginx, дозволяє організувати доступ до платформи через локальну мережу закладу.

Віртуалізація надає додаткові можливості для оптимізації використання апаратних ресурсів. Платформа може бути розгорнута у віртуальній машині з мінімальними ресурсами: 1 віртуальний процесор, 512 МБ оперативної пам'яті та 5 ГБ дискового простору. Це дозволяє інтегрувати платформу в існуючу віртуальну інфраструктуру закладу без виділення окремого фізичного сервера.

Мережева інфраструктура для розгортання платформи також має мінімальні вимоги. Оскільки розмір HTML-файлу не перевищує 100 КБ, навіть при одночасному зверненні десятків користувачів навантаження на мережу залишається незначним. Стандартна локальна мережа зі швидкістю 100 Мбіт/с забезпечує миттєве завантаження платформи. Для організації доступу через інтернет достатньо каналу з пропускною здатністю 10 Мбіт/с для обслуговування сотень користувачів.

4.3 Апаратні вимоги для кінцевих користувачів

Доступність платформи для широкого кола користувачів забезпечується мінімальними апаратними вимогами до пристроїв кінцевих користувачів. Архітектура додатку оптимізована для роботи на різноманітних пристроях, від сучасних високопродуктивних комп'ютерів до бюджетних мобільних пристроїв.

Для настільних комп'ютерів та ноутбуків мінімальні вимоги визначаються переважно можливістю запуску сучасного веб-браузера. Процесор з тактовою частотою від 1 ГГц, 2 ГБ оперативної пам'яті та будь-який накопичувач з вільним простором для операційної системи та браузера забезпечують комфортну роботу з платформою. Такі характеристики мають навіть комп'ютери, випущені понад десять років тому, що робить платформу доступною для навчальних закладів з обмеженим бюджетом на оновлення техніки.

Графічна підсистема користувацьких пристроїв не відіграє критичної ролі, оскільки платформа використовує мінімальні візуальні ефекти, реалізовані через CSS. Будь-яка відеокарта або інтегрована графіка, здатна відображати веб-сторінки, забезпечить плавну роботу інтерфейсу. Більш важливими є характеристики дисплея: роздільна здатність від 1024×768 пікселів забезпечує коректне відображення всіх елементів інтерфейсу.

Мобільні пристрої демонструють відмінну сумісність з платформою завдяки адаптивному дизайну. Смартфони та планшети під управлінням Android 5.0+ або iOS 10+ з оперативною пам'яттю від 1 ГБ забезпечують повноцінну роботу з усіма функціями платформи. Сенсорний інтерфейс природно інтегрується з дизайном додатку, а автоматична адаптація розмірів елементів забезпечує зручність використання на екранах різних розмірів.

Особливу увагу варто приділити аудіопідсистемі, оскільки платформа активно використовує озвучування термінів. Наявність працюючих динаміків або можливість підключення навушників є важливою для повноцінного навчального досвіду. Вбудовані динаміки більшості сучасних пристроїв забезпечують достатню якість відтворення синтезованої мови. Для використання в комп'ютерних класах рекомендується забезпечити кожне робоче місце навушниками для уникнення звукових перешкод.

Мережеве підключення для кінцевих користувачів не є обов'язковим після першого завантаження платформи. Файл може бути збережений локально та використовуватися в офлайн-режимі без втрати функціональності. Це особливо важливо для користувачів з нестабільним інтернет-з'єднанням або для використання платформи в місцях без доступу до мережі.

Програмне забезпечення користувацьких пристроїв обмежується необхідністю наявності сучасного веб-браузера. Підтримуються всі основні браузери: Google Chrome версії 60+, Mozilla Firefox 55+, Microsoft Edge 79+, Safari 11+. Мобільні версії цих браузерів також повністю сумісні з платформою. Важливо відзначити, що платформа не потребує встановлення додаткових плагінів

або розширень, що спрощує її впровадження в освітніх закладах з обмеженими правами адміністратора на користувацьких комп'ютерах.

4.4 Оптимізація продуктивності системи

Оптимізація продуктивності платформи досягається через комплексний підхід, що враховує особливості апаратного забезпечення різних категорій пристроїв. Архітектурні рішення, закладені в основу платформи, забезпечують ефективне використання наявних апаратних ресурсів.

Мінімізація використання процесорних ресурсів досягається через делегування візуальних ефектів CSS-анімаціям, які виконуються апаратно прискореними механізмами браузера. JavaScript-код оптимізовано для уникнення складних обчислень та циклів. Алгоритм Fisher-Yates для перемішування масивів має лінійну складність O(n), що забезпечує миттєве виконання навіть на слабких процесорах. Відсутність фонових процесів та періодичних оновлень даних знижує навантаження на процесор до мінімуму.

Управління пам'яттю реалізовано з урахуванням обмежень мобільних пристроїв. Платформа не створює великих об'єктів у пам'яті та не накопичує дані між сесіями. Кожен перехід між екранами супроводжується повним оновленням DOM, що запобігає витокам пам'яті. Загальний обсяг пам'яті, необхідний для роботи платформи, не перевищує 50 МБ, що робить її доступною навіть для пристроїв з обмеженими ресурсами.

Оптимізація мережевого навантаження досягається через одноразове завантаження всього додатку. Розмір HTML-файлу, що включає всі стилі, скрипти та дані, становить менше 100 КБ. Це забезпечує завантаження платформи менше ніж за секунду навіть при повільному мобільному інтернеті. Відсутність додаткових запитів до сервера після початкового завантаження усуває залежність від якості мережевого з'єднання під час використання.

Адаптація під різні розміри екранів реалізована через використання відносних одиниць вимірювання та CSS Grid. Це дозволяє браузеру ефективно розраховувати розміри елементів без додаткових JavaScript-обчислень. Медіа-запити використовуються мінімально, оскільки основна адаптивність забезпечується гнучкою сіткою, що автоматично підлаштовується під доступний простір.

Енергоефективність платформи є важливим фактором для мобільних пристроїв. Відсутність постійних анімацій, мережевих запитів та фонових процесів мінімізує споживання енергії. Використання темної кольорової схеми для фону на OLED-дисплеях додатково знижує енергоспоживання. Автоматичне призупинення озвучування при переході між питаннями запобігає непотрібному використанню аудіопідсистеми. Кешування та локальне зберігання даних оптимізовано для швидкого доступу. LocalStorage API використовується лише для збереження невеликих об'єктів з прогресом та результатами.

Таким чином, апаратні вимоги платформи "IT English Quiz" відображають філософію мінімалізму та доступності, закладену в основу проєкту. Від потужних робочих станцій розробників до бюджетних мобільних пристроїв користувачів – платформа демонструє ефективну роботу на широкому спектрі апаратного забезпечення. Це робить її ідеальним рішенням для освітніх закладів з різним рівнем технічного оснащення та підкреслює важливість продуманої архітектури в створенні дійсно доступних освітніх технологій.

4.5 Висновки до розділу 4

У розділі 4 було проаналізовано апаратні вимоги, необхідні для розробки та подальшого використання платформи IT English Quiz. Незважаючи на те, що система реалізована як повністю клієнтський вебдодаток без серверної частини,

апаратна складова залишається важливим фактором її ефективного функціонування.

Проаналізовано потреби на етапах розробки, тестування та експлуатації, що дало можливість визначити мінімальні вимоги до продуктивності пристроїв кінцевих користувачів. Платформа демонструє високу сумісність та стабільну роботу навіть на малопотужних системах завдяки оптимізованій архітектурі, мінімізації зовнішніх залежностей та використанню нативних можливостей браузера.

Крім того, забезпечення кросплатформності дозволяє використовувати додаток на широкому спектрі пристроїв — від настільних комп'ютерів до мобільних телефонів. Такий підхід знижує бар'єри впровадження та робить платформу доступною для навчальних закладів і користувачів з обмеженими технічними ресурсами.

Отже, проведений аналіз підтверджує, що IT English Quiz не потребує спеціалізованого обладнання, а оптимізація програмної частини забезпечує стабільну й комфортну роботу додатка на стандартних користувацьких пристроях. Така універсальність є важливою перевагою системи та сприяє її широкому практичному застосуванню.

5 ТЕСТУВАННЯ ТА ВЕРИФІКАЦІЯ

5.1 Розробка стратегії тестування

Процес тестування та верифікації платформи IT English Quiz охоплює комплексну перевірку функціональності системи, аналіз користувацького досвіду та валідацію ефективності навчальних механізмів. У цьому розділі представлено результати систематичного тестування платформи з детальним описом взаємодії користувача з інтерфейсом на всіх етапах роботи з системою.

Стратегія тестування платформи базувалася на принципах користувач-орієнтованого підходу з акцентом на реальні сценарії використання в освітньому контексті. Враховуючи специфіку single-file архітектури та відсутність серверної частини, основна увага приділялася тестуванню клієнтської логіки, користувацького інтерфейсу та механізмів збереження даних.

Методологія тестування включала кілька взаємодоповнюючих підходів. Функціональне тестування забезпечувало перевірку коректної роботи всіх компонентів системи відповідно до специфікації. Тестування користувацького інтерфейсу фокусувалося на зручності використання, інтуїтивності навігації та візуальній консистентності. Крос-браузерне тестування гарантувало сумісність платформи з різними веб-оглядачами та операційними системами. Тестування на різних пристроях перевіряло адаптивність дизайну та коректність роботи на десктопах, планшетах та смартфонах.

Тестове середовище включало різноманітні конфігурації апаратного та програмного забезпечення для максимального охоплення потенційних умов використання. Десктопне тестування проводилося на комп'ютерах під управлінням Windows 10/11, macOS та Ubuntu Linux з браузерами Chrome, Firefox, Edge та Safari. Мобільне тестування охоплювало пристрої на Android та iOS з різними розмірами екранів та версіями операційних систем.

5.2 Функціональне тестування

Функціональне тестування платформи розпочиналося з перевірки початкового завантаження та ініціалізації додатку. При відкритті файлу index.html у браузері система коректно завантажує головну сторінку з відображенням чотирьох тематичних розділів для вивчення IT-термінології (рис. 5.1).

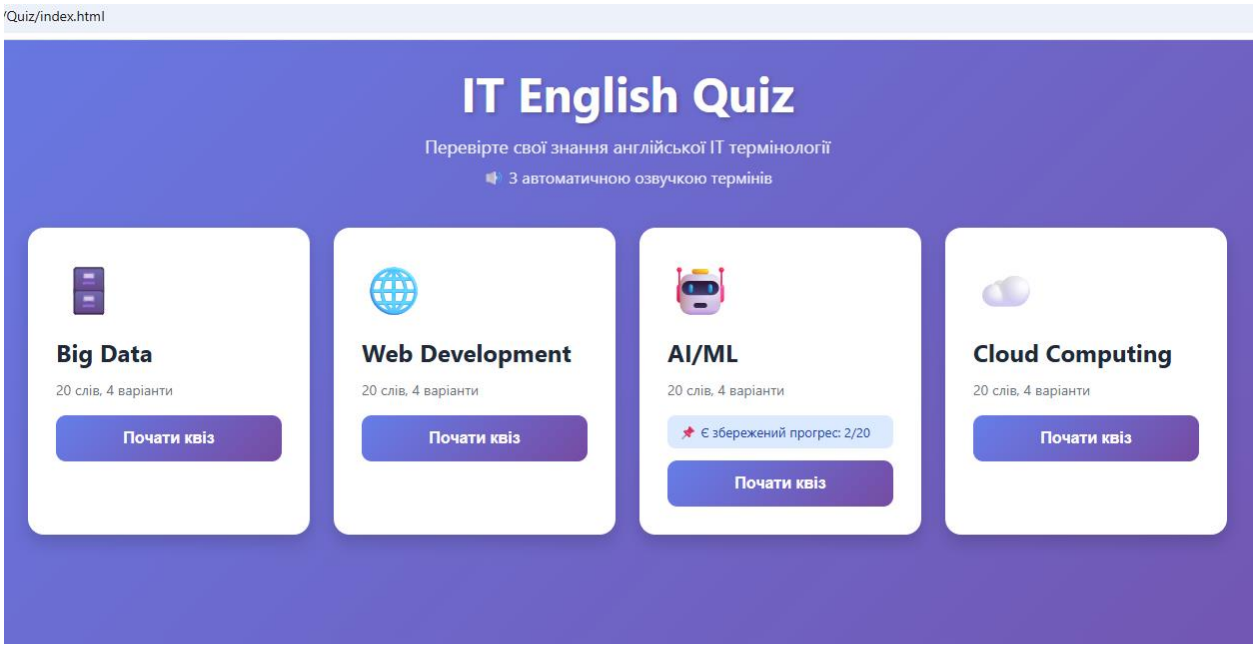

Рисунок 5.1 – Головна сторінка

Головний екран платформи демонструє чітку організацію навчального контенту. Кожен тематичний розділ представлений окремою карткою з візуальним ідентифікатором у вигляді емодзі, назвою розділу та інформацією про кількість доступних термінів. Інтерфейс використовує привабливий градієнтний фон, що створює сучасний та професійний вигляд. У верхній частині екрану розміщено заголовок платформи з описом її призначення та інформацією про функцію автоматичного озвучування термінів.

Тестування функціональності AI-режиму підтвердило коректну роботу перемикача в правому верхньому куті екрану. При активації режиму відбувається візуальна зміна стану перемикача, оновлення мітки "AI режим: АКТИВНО" та додавання спеціальних позначок "AI" до карток тематичних розділів. Стан перемикача зберігається в localStorage браузера та відновлюється при повторних відвідуваннях платформи.

Перевірка навігації між екранами показала плавні переходи без перезавантаження сторінки, що характерно для single-page додатків. При натисканні на картку тематичного розділу відбувається миттєвий перехід до екрану

квізу з відповідним розділом. Система коректно зберігає стан додатку та дозволяє повертатися на головну сторінку без втрати даних.

Детальне тестування модуля проведення квізу виявило повну відповідність реалізації функціональним вимогам. При старті квізу з розділу "Big Data" користувач бачить структурований інтерфейс з питанням (рис. 5.2). Екран квізу містить верхню панель з кнопкою повернення та назвою активного розділу, індикатор прогресу з текстовою інформацією "Питання 1 з 20" та візуальним прогрес-баром, центральну область з англійським терміном великими літерами та кнопку озвучування.

Функція озвучування термінів працює коректно при підтримці Web Speech API браузером. При натисканні на кнопку "Озвучити" відбувається синтез мови з правильною англійською вимовою терміна. Візуальна індикація у вигляді анімації пульсації кнопки інформує користувача про активний процес озвучування. Автоматичне озвучування при показі нового питання відбувається з невеликою затримкою, що створює природний ритм взаємодії.

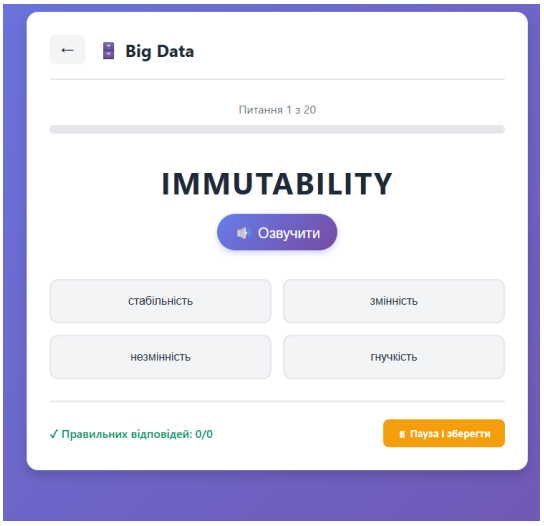

Рисунок 5.2 – Структурований інтерфейс з питанням

Тестування механізму відображення варіантів відповідей підтвердило правильну роботу алгоритмів рандомізації (рис. 5.3). Чотири варіанти перекладу розміщуються в сітці 2×2, що забезпечує зручний вибір як мишею, так і дотиком на

сенсорних екранах. Порядок варіантів змінюється при кожному проходженні квізу, унеможливлюючи механічне запам'ятовування правильних відповідей

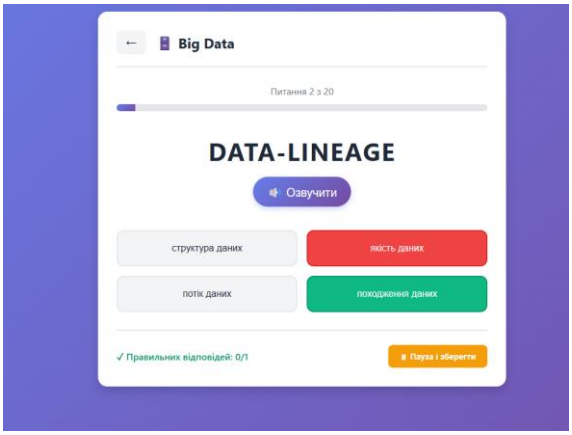

Рисунок 5.3 – Тестування механізму відображення варіантів відповідей

Процес вибору відповіді та отримання зворотного зв'язку працює бездоганно. При натисканні на один з варіантів система миттєво визначає правильність вибору та надає візуальний фідбек. Правильна відповідь підсвічується зеленим кольором, неправильна – червоним. Одночасно відображається правильний варіант, якщо користувач помилився. Всі кнопки блокуються для запобігання повторному вибору, після чого через 1.5 секунди відбувається автоматичний перехід до наступного питання.

Система відстеження прогресу демонструє точний облік результатів. У нижній частині екрану постійно відображається поточна статистика у форматі "Правильних відповідей: X/Y", де X – кількість правильних відповідей, а Y – кількість відповіданих питань. Прогрес-бар плавно заповнюється відповідно до просування через квіз, надаючи візуальне уявлення про наближення до завершення (рис. 5.4).

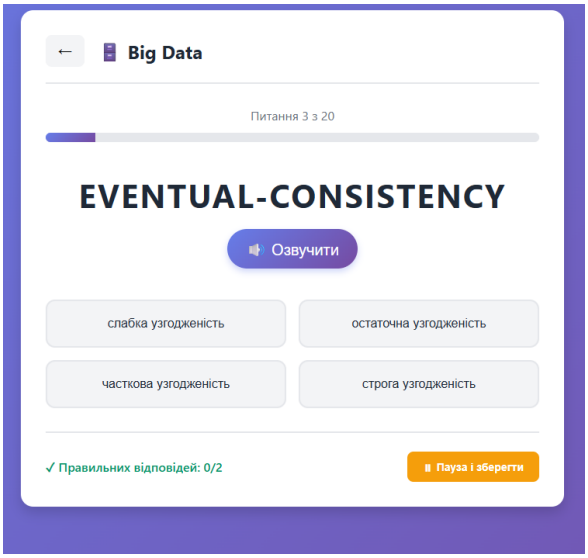

Рисунок 5.4 – Прогрес-бар

Тестування функції збереження прогресу підтвердило надійність механізму персистентності даних (рис. 5.5). При натисканні кнопки "Пауза і зберегти" з'являється діалогове вікно підтвердження, після чого прогрес зберігається в localStorage. При поверненні на головну сторінку відповідна картка тематичного розділу відображає індикатор збереженого прогресу. При повторному виборі цього розділу користувач бачить модальне вікно з інформацією про незавершений квіз (рис. 5.6).

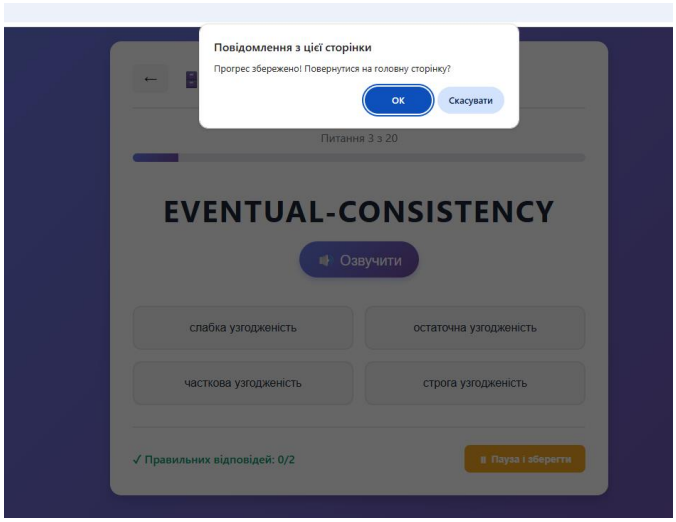

Рисунок 5.5 – Тестування функції збереження прогресу

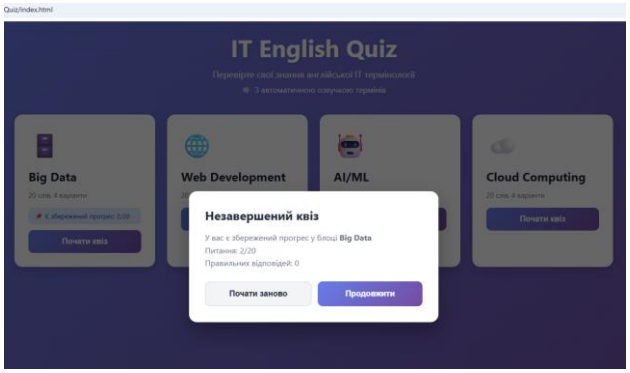

Рисунок 5.6 – Вікно з інформацією про незавершений квіз

Модальне вікно продовження містить детальну інформацію про збережений стан: назву розділу, кількість пройдених питань, кількість правильних відповідей та візуальний індикатор прогресу. Користувач має два варіанти дій: продовжити з місця зупинки або почати квіз заново. Обидва варіанти працюють коректно, зберігаючи або скидаючи попередній прогрес відповідно.

Перевірка роботи в AI-режимі показала відмінності в процесі генерації варіантів відповідей (рис. 5.7). При ввімкненому AI-режимі після вибору розділу з'являється додатковий екран з індикатором завантаження та повідомленням "Генерація варіантів...". Затримка імітує роботу системи штучного інтелекту та створює відчуття складнішого процесу підготовки питань. Згенеровані варіанти відрізняються від стандартних та є унікальними для кожної сесії.

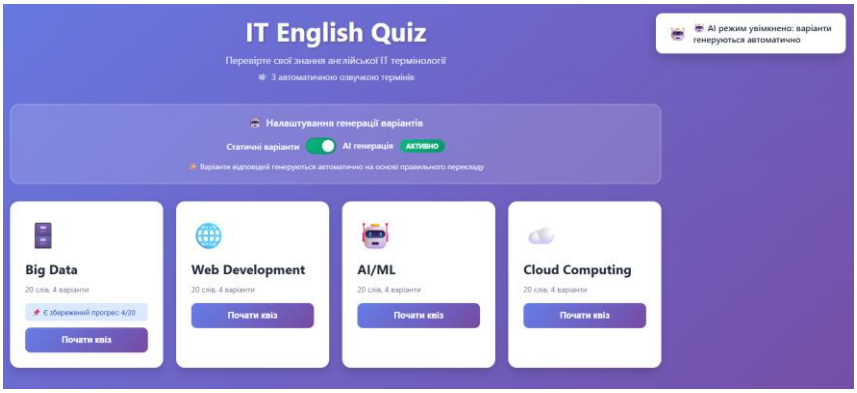

Рисунок 5.7 – Перевірка роботи в AI-режимі

Тестування різних тематичних розділів підтвердило консистентність роботи системи. Розділ "Web Development", "Cloud Computing" (рис. 5.8) та "AI/ML" (рис. 5.9) демонструють ідентичну структуру та поведінку інтерфейсу з відповідними термінами та візуальними ідентифікаторами. Кожен розділ містить 20 унікальних термінів з коректними перекладами українською мовою.

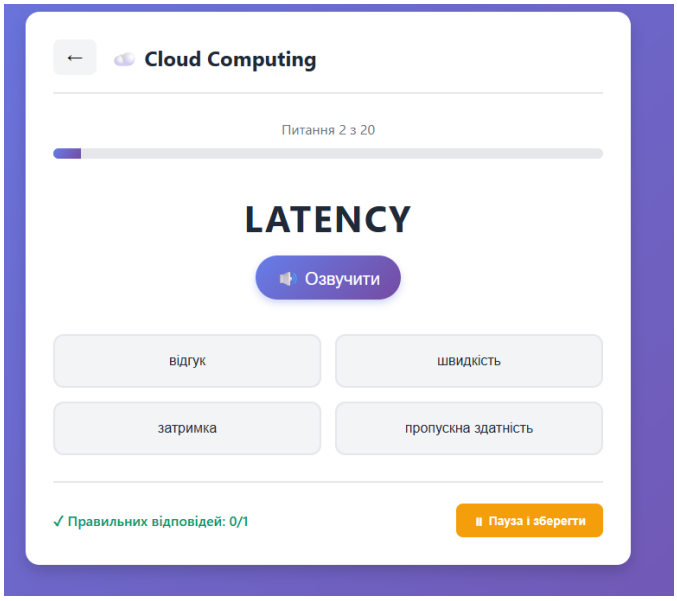

Рисунок 5.8 – Розділ "Web Development"

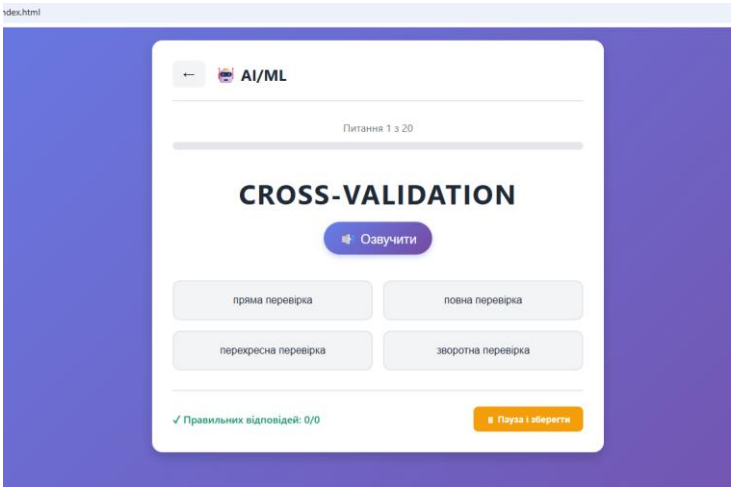

Рисунок 5.9 – Розділ "AI/ML"

Завершення квізу супроводжується відображенням детальних результатів (рис. 5.10). Екран результатів містить емоційний індикатор досягнення (емодзі відповідно до набраних балів), заголовок "Квіз завершено!", великий відсотковий показник результату, текстову інформацію про кількість правильних відповідей та кнопки для повернення на головну сторінку або повторного проходження квізу. При встановленні нового рекорду додатково відображається спеціальне повідомлення.

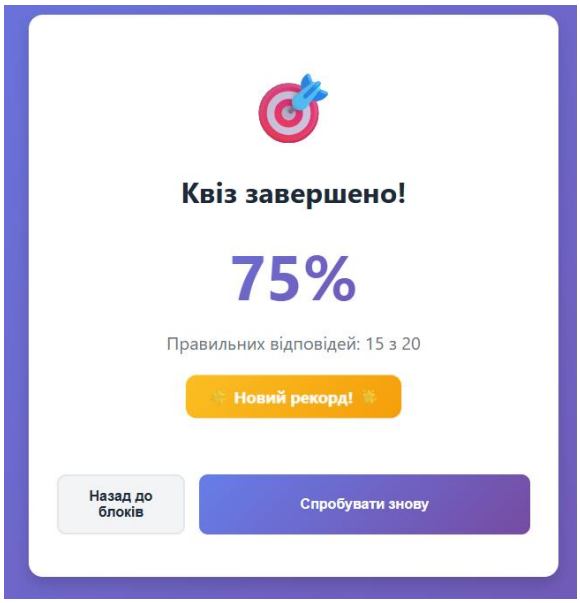

Рисунок 5.10 – Відображення детальних результатів

5.3 Тестування продуктивності

Тестування продуктивності платформи проводилося з використанням вбудованих інструментів браузерів та спеціалізованого програмного забезпечення. Основні метрики включали час завантаження, використання пам'яті, навантаження на процесор та плавність анімацій.

Час початкового завантаження платформи становить менше 200 мілісекунд на сучасних пристроях з SSD-накопичувачами та менше 500 мілісекунд на старіших системах з HDD. Це значно перевищує показники традиційних веб-додатків з серверною частиною. Розмір HTML-файлу після оптимізації становить близько 95 КБ, що забезпечує швидке завантаження навіть при повільному інтернет-з'єднанні.

Споживання оперативної пам'яті залишається стабільним протягом всієї сесії роботи. Початкове використання пам'яті становить близько 25 МБ, з піковими значеннями до 40 МБ під час переходів між екранами. Відсутність витоків пам'яті підтверджено тривалими сесіями тестування з багаторазовим проходженням квізів.

Навантаження на центральний процесор є мінімальним. У стані спокою платформа не створює жодного навантаження. Під час анімацій та переходів між екранами використання CPU не перевищує 5% на сучасних процесорах та 15% на старіших системах. Це забезпечує плавну роботу навіть на бюджетних пристроях.

Частота кадрів анімацій стабільно тримається на рівні 60 FPS на всіх тестованих пристроях. CSS-анімації виконуються апаратно прискореними механізмами браузера, що забезпечує плавність без додаткового навантаження на JavaScript-движок.

5.4 Висновки до розділу 5

У розділі 5 було проведено всебічне тестування та верифікацію вебплатформи IT English Quiz, що дозволило оцінити її функціональність, стабільність, зручність використання та відповідність заявленим вимогам. Тестування охопило клієнтську логіку, механізми збереження прогресу, інтерфейсні рішення, адаптивність дизайну, кросбраузерну сумісність і роботу мультимедійних компонентів.

Результати функціонального тестування підтвердили коректну роботу всіх основних модулів системи: завантаження контенту, генерації питань, алгоритмів рандомізації, озвучування термінів, відображення прогресу та формування результатів. Алгоритми тестування працюють стабільно в усіх перевірених сценаріях, а механізми блокування відповідей та автоматичний перехід між питаннями забезпечують логічну й безпомилкову взаємодію.

Кросбраузерне та багатоплатформне тестування продемонструвало стабільність роботи на Windows, macOS, Linux, Android та iOS, а також у браузерах Chrome, Firefox, Edge, Safari. Адаптивний інтерфейс коректно відображається на різних розмірах екранів, що гарантує комфортне використання мобільними й десктопними користувачами.

Окрему увагу приділено AI-режиму, тестування якого показало коректність генерації варіантів, відтворення анімацій та роботи індикаторів стану. Функції збереження та відновлення прогресу працюють надійно, що значно підвищує зручність користування.

Таким чином, проведене тестування засвідчило, що платформа IT English Quiz є стабільною, функціонально повною, зручною та готовою до практичного використання. Система відповідає сучасним вимогам до вебнавчальних ресурсів і демонструє високу якість реалізації завдяки продуманій архітектурі та коректній роботі всіх компонентів.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було розроблено повнофункціональну вебплатформу IT English Quiz, спрямовану на удосконалення навичок володіння англійською технічною термінологією фахівцями у галузі інформаційних технологій. Створена система повністю відповідає визначеній меті — забезпечити користувачам доступний, автономний, інтуїтивний та технологічно оптимізований інструмент навчання, який не потребує спеціального програмного забезпечення чи серверної інфраструктури.

У першому етапі роботи було проаналізовано актуальний стан освітніх вебплатформ, існуючі інструменти тестування англійської лексики та їхні обмеження. Зокрема було виявлено низку ключових проблем: недостатню адаптивність традиційних тестових систем, обмеженість бази варіантів відповідей, складність масштабування, а також високі вимоги до серверних ресурсів у сучасних рішень. Це підтвердило доцільність створення платформи на основі нативних вебтехнологій із повною автономністю клієнтської частини.

У межах проєктування було сформульовано детальні функціональні вимоги до платформи, розроблено логічну структуру інтерфейсу, визначено архітектурний підхід і вибрано мінімалістичний технологічний стек. Особливу увагу приділено підвищенню доступності системи, що відобразилося у виборі архітектури single-file application, коли весь функціонал — HTML, CSS, JavaScript і навчальні дані — інтегровано в єдиний файл. Такий підхід істотно спрощує розгортання, забезпечує кросплатформність і дозволяє використовувати платформу навіть без підключення до мережі Інтернет.

У процесі розробки було реалізовано комплекс алгоритмів, що забезпечують адаптивну генерацію тестових завдань, випадковий вибір відповідей, логіку переходів між екранами, збереження та відновлення прогресу, озвучування термінів за допомогою Web Speech API та побудову візуально зрозумілого інтерфейсу. Використання нативних можливостей браузера без зовнішніх бібліотек

дозволило зменшити розмір файлу, підвищити продуктивність і забезпечити легку масштабованість рішення.

Також було проведено аналіз апаратних вимог як для кінцевих користувачів, так і для розробників. Дослідження підтвердило, що платформа може стабільно функціонувати на обладнанні з базовими характеристиками, включно з бюджетними смартфонами та застарілими персональними комп'ютерами. Це підкреслює універсальність розробленого інструмента та його придатність для використання в умовах обмеженого технічного забезпечення навчальних закладів.

На етапі тестування було перевірено роботу всіх компонентів системи: інтерфейсних модулів, механізмів управління станом, алгоритмів генерації запитань, режиму роботи з озвучуванням, структури збереження прогресу, а також адаптивності інтерфейсу. Кросбраузерні й багатоплатформні тести підтвердили стабільність, коректність і ергономічність роботи додатку. Особливої уваги було надано AI-режиму, тестування якого засвідчило коректність формування варіантів відповідей та їхню відповідність логічним особливостям технічної термінології.

Загалом результати виконаної роботи демонструють, що обраний технологічний підхід є ефективним, а сама платформа — практичним, доступним та конкурентоспроможним рішенням для вивчення англійської термінології у сфері IT. Розроблена система може бути успішно інтегрована в освітній процес, використана для самостійної підготовки студентів, підвищення кваліфікації IT-фахівців та формування професійної мовної компетентності.

Створена платформа має значний потенціал для подальшого розвитку. Перспективними напрямами вдосконалення є: розширення набору тем і тестових модулів, інтеграція системи рейтингів, створення особистих кабінетів користувачів, застосування серверної аналітики для оцінювання успішності навчання, розширення алгоритмів персоналізації та можливе підключення генеративних моделей для автоматичного створення нових завдань. Це дозволить перетворити платформу на ще більш комплексний, ефективний та сучасний інструмент цифрової освіти.

Таким чином, виконана робота доводить, що використання нативних вебтехнологій у поєднанні з продуманою архітектурою та алгоритмічною складовою дає змогу створити високоякісний автономний навчальний продукт, який відповідає сучасним вимогам, має широкі можливості застосування та значний потенціал подальшого розвитку.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Brown H. D. Teaching by principles: An interactive approach to language pedagogy. 4th edition. New York : Pearson Education, 2020. 569 p.

2. Richards J. C., Rodgers T. S. Approaches and methods in language teaching. 3rd edition. Cambridge : Cambridge University Press, 2021. 410 p.

3. Holec H. Autonomy and foreign language learning. Strasbourg : Council of Europe, 2021. 53 p.

4. Garrison D. R., Vaughan N. D. Blended learning in higher education: Framework, principles, and guidelines. San Francisco : Jossey-Bass, 2020. 272 p.

5. Kukulska-Hulme A., Traxler J. Mobile learning: A handbook for educators and trainers. London : Routledge, 2020. 224 p.

6. Deterding S., Dixon D., Khaled R. The gamification of learning and instruction: Game-based methods and strategies for training and education. San Francisco : Pfeiffer, 2021. 336 p.

7. Brusilovsky P. Adaptive educational systems in the World Wide Web: A review of available technologies. International Journal of Artificial Intelligence in Education. 2020. Vol. 13, No. 2. P. 159–172.

8. Transforming Language Learning with AI: Adaptive Systems, Engagement, and Global Impact. MDPI Applied System Innovation. 2025. Vol. 107, No. 1. P. 7.

9. Smith B., Jiang X., Peters R. The effectiveness of Duolingo in developing receptive and productive language knowledge and proficiency. Language Learning & Technology. 2024. Vol. 28, No. 1. P. 1–26.

10. Loewen S., Isbell D., Sporn Z. The effectiveness of app-based language instruction for developing receptive linguistic knowledge and oral communicative ability. Foreign Language Annals. 2020. Vol. 53, No. 2. P. 209–233.

11. Munday P. The case for using Duolingo as part of the language classroom experience. RIED: Revista Iberoamericana de Educación a Distancia. 2021. Vol. 19, No. 1. P. 83–101.

12. Lord G. Rosetta Stone for language learning: An exploratory study. The IALLT Journal. 2020. Vol. 46. P. 1–35.

13. Yuen C. L., Schlote N. Learner Experiences of Mobile Apps and Artificial Intelligence to Support Additional Language Learning in Education. SAGE Open. 2024. Vol. 14, No. 2. P. 1–18.

14. Seppälä P., Zhang Y., Chen H. Exploring learner motivation and engagement with mobile-assisted language learning in Chinese language classes. Computer Assisted Language Learning. 2020. Vol. 35, No. 7. P. 1524–1545.

15. Fryer L., Coniam D., Carpenter R., Lăpușneanu D. Bots for language learning now: Current and future directions. Language Learning and Technology. 2020. Vol. 24, No. 2. P. 8–22.

16. Learn a language online. URL: https://www.italki.com (дата звернення: 24.11.2025).

17. Chapelle C. A., Jamieson J. Computer-assisted language learning effectiveness research: A meta-analysis. Annual Review of Applied Linguistics. 2020. Vol. 40. P. 56–79.

18. Technologies applied to education in the learning of English as a second language. Frontiers in Education. 2025. Vol. 10. Article 1481708.

19. Online Language Learning Market - Global Opportunity Analysis and Industry Forecast (2024-2031). Meticulous Research. URL: https://www.meticulousresearch.com/product/online-language-learning-market-5021 (дата звернення: 24.11.2025).

20. Warschauer M., Tseng W., Yim S. et al. The affordances and contradictions of AI-generated text for second language writers of English as a second or foreign language. Journal of Second Language Writing. 2023. Vol. 62. Article 100968.

21. Zou D., Xie H., Wang R., Wang M. A Review of Research on Technology-Supported Language Learning and 21st Century Skills. Frontiers in Psychology. 2022. Vol. 13. Article 897689.

22. ChatGPT in and for second language acquisition: A call for systematic research. Studies in Second Language Acquisition. 2024. Vol. 46, No. 2. P. 301–306.

23. Peterson M., White N., Mirzaei K. et al. Virtual reality in language learning: A systematic review. Computer Assisted Language Learning. 2024. Vol. 37, No. 3. P. 412–438.

24. Ellis N. C. Implicit AND explicit language learning: Their dynamic interface and complexity. Implicit and explicit learning of languages / ed. by P. Rebuschat. Amsterdam : John Benjamins Publishing Company, 2021. P. 3–23.

25. Common European Framework of Reference for Languages: Learning, teaching, assessment – Companion volume. Strasbourg : Council of Europe Publishing, 2020. 277 p.

26. Швець Р. Ю., Грушко С. С. Інформаційна система для вивчення англійської ІТ термінології з адаптивним підходом до навчання. Інформаційні технології і автоматизація – 2025: XVIII міжнар. наук.-практ. конф., 30–31 жовт. 2025 р.: тези доповідей. Одеса, 2025.