МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
УМАНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
Факультет економіки, підприємництва та інформаційних технологій
Кафедра інформаційних технологій

Допущено до захисту
"_____" _____ 2026 р.
Зав. кафедри, к.т.н., доцент
_____ **Ліщук Р.І.**

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

РОЗРОБКА ВЕБ-ДОДАТКУ ДЛЯ ДИСТАНЦІЙНОГО НАВЧАННЯ ОСНОВАМ ВЕБ-ПРОГРАМУВАННЯ

Виконав - студент 4 курсу, групи 41-кн
освітньої програми «Комп'ютерні науки»
спеціальності F3 Комп'ютерні науки
Чернієнко Віктор Олександрович

Науковий керівник - д.т.н., професор
Кулаков Павло Ігорович

Умань - 2026

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ……..... 7
ВСТУП…… 8
РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ…… 10
1.1. Об’єкт та предмет дослідження…… 10
1.2. Порівняльний аналіз аналогів розробки ……. 15
1.3. Постановка задачі..... 21
РОЗДІЛ 2. РОЗРОБКА МЕТОДУ, МОДЕЛІ ТА АЛГОРИТМІВ РОБОТИ ВЕБ-ДОДАТКУ …… 22
2.1. Аналіз навчального функціоналу веб-додатку……..…… 22
2.2. Розробка структури інтерфейсу веб-додатку …… 23
2.3. Розробка моделі, методу та алгоритмів роботи веб-додатку..... 28
РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ВЕБ-ДОДАТКУ…….. 34
3.1. Варіантний аналіз і обґрунтування вибору засобів для програмної реалізації веб-додатку ……... 34
3.2. Розробка програмних засобів для адаптивної верстки та навігації …….. 36
3.3. Розробка модуля тестування…… 38
ВИСНОВОК…… 45
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ…… 46
ДОДАТОК А. Лістинг програми……... 48

Вступ

Сфера веб-розробки сьогодні виступає одним із найпривабливіших та найбільш затребуваних містків для входу в індустрію інформаційних технологій. Стрімкі процеси глобалізації та невпинний технологічний прогрес диктують нові умови, за яких власні цифрові платформи стають життєвою необхідністю не лише для потужних корпорацій чи державних інституцій, а й для приватних осіб, включаючи незалежних контент-мейкерів, блогерів та фрілансерів. Сам процес створення таких систем є комплексним інженерним завданням, що традиційно охоплює розробку візуальної концепції та дизайну, кодування інтерфейсу на стороні клієнта, програмування серверної логіки, а також фінальне налаштування та конфігурацію веб-сервера [1].

Попри високу перспективність напрямку, поріг входження вимагає обов'язкового і ґрунтовного опанування фундаментального технологічного тріо, що складається з мови розмітки HTML, каскадних таблиць стилів CSS та динамічного JavaScript [2]. Головним бар'єром, через який чимало початківців відмовляються від цієї ідеї, є хаотичність інформаційного простору, де самостійний пошук якісних, структурованих і дійсно важливих знань перетворюється на серйозну проблему. Ефективним інструментом для подолання цих труднощів стають інтерактивні навчальні веб-платформи.

Аналіз наявного ринку онлайн-навчання виявив низку критичних недоліків у сучасних освітніх додатках. Більшість із них або обмежують користувача комерційними пейволами, приховуючи значну частину матеріалів за преміум-підписками, або ж ізолюють курси один від одного, змушуючи студента самостійно шукати та поєднувати розрізнені модулі по різних сайтах задля отримання комплексної спеціалізації. З огляду на це, проектування та програмна реалізація власної інтегрованої системи навчання набуває особливої актуальності. Створення платформи, яка гармонійно поєднує теоретичну базу, наочні практичні приклади з усіх трьох базових технологій в одному місці, а також містить вбудований інструментарій автоматизованого тестування для

оперативної самоперевірки знань, дозволяє вирішити ключові проблеми сучасної ІТ-освіти.

Актуальність

Актуальність кваліфікаційної роботи бакалавра зумовлена тим, що в умовах військової агресії, розробка програмних засобів з метою реалізації дистанційної навчальної системи для вивчення основ веб-програмування, яка поєднує теорію та практичні приклади базових технологій, а також можливістю самоперевірки шляхом тестування, є важливим завданням.

Об'єкт дослідження - процес розробки програмних продуктів для дистанційних навчальних веб-систем, які призначені для вивчення основ веб-програмування.

Предмет дослідження - методи та засоби розробки програмних продуктів для дистанційних навчальних веб-систем, які призначені для вивчення основ веб-програмування.

Мета дослідження

Метою кваліфікаційної роботи бакалавра є створення веб-додатку з метою підвищення ефективності дистанційного вивчення веб-програмування шляхом поєєднання теоретичної бази, практичних прикладів та контролю знань, що дозволить забезпечити потреби користувача під час дистанційного навчального процесу.

Відповідно до поставленої мети потрібно вирішити наступні задачі:

- розробити структурну схему та архітектуру розроблюваного веб-додатку;
- розробити метод та модель функціонування веб-додатку;
- розробити блоки теоретичного матеріалу та блоки з практичними прикладами для вивчення матеріалу;
- розробити блок самотестування користувача;
- розробити програмне забезпечення веб-додатку;
- здійснити тестування розробленого програмного забезпечення.

РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Об’єкт та предмет дослідження

Стрімкий розвиток глобального цифрового простору та перехід більшості сфер людської діяльності у віртуальне середовище сформували попит на кваліфікованих інженерів, здатних створювати, модернізувати та підтримувати складні мережеві платформи. Аналіз предметної області в галузі дистанційного навчання основам веб-програмування наочно демонструє, що шлях становлення майбутнього розробника є надзвичайно містким та багатогранним процесом, який вимагає від початківця послідовного засвоєння величезного масиву фундаментальних та прикладних знань [3, 4].

Старт у цій професії традиційно асоціюється з подоланням високого інформаційного бар'єра, оскільки індустрія інформаційних технологій еволюціонує з безпрецедентною швидкістю, щодня генеруючи нові інструменти, бібліотеки, підходи та методологічні парадигми. Попри появу численних вузькоспеціалізованих надбудов та фреймворків, концептуальним базисом і непорушним фундаментом для будь-якого розробника залишається класичне тріо технологій клієнтської сторони, до якого входять мова розмітки гіпертексту HTML, каскадні таблиці стилів CSS та інтерактивна мова сценаріїв JavaScript [5, 6]. Глибоке, всебічне та деталізоване освоєння цих трьох компонентів є безальтернативною та обов'язковою умовою для формування первинної професійної компетентності, оскільки саме вони відповідають за логічну структуру, візуальне позиціонування та динамічну поведінку будь-якого цифрового інтерфейсу в мережі інтернет.

Мова розмітки HTML відповідає за скелет і семантичну коректність документа, дозволяючи браузеру інтерпретувати елементи як заголовки, параграфи, форми чи мультимедійні блоки. Каскадні таблиці стилів CSS додають цьому скелету естетичний вигляд, керуючи колірною палітрою, типографікою, адаптивними сітками розташування та складними візуальними ефектами, які забезпечують комфортне сприйняття контенту користувачем. У свою чергу,

JavaScript надає життя статичним сторінкам, трансформуючи їх у повноцінні інтерактивні додатки, здатні обробляти події без перевантаження сервера, взаємодіяти з асинхронними запитами, модифікувати модель об'єктів документа DOM та керувати складною логікою на стороні клієнта [7].

У сучасному освітньому просторі парадигма самостійного онлайн-навчання зазнала кардинальних трансформацій, перетворившись із допоміжної форми здобуття знань на провідний і домінуючий тренд, особливо яскраво виражений у сфері комп'ютерних наук. Гнучкість графіку, відсутність жорсткої географічної прив'язки та можливість рухатися у власному індивідуальному темпі роблять дистанційну освіту магнітом для тисяч новачків, які прагнуть змінити професійний вектор або здобути актуальні технічні навички.

Ринок миттєво відреагував на цей запит, внаслідок чого в інтернет-середовищі сформувалася колосальна інфраструктура, що складається з тисяч комерційних курсів, мобільних застосунків, менторських програм та безкоштовних відеоуроків на платформах спільного перегляду відео. Проте детальний критичний аналіз існуючих дидактичних інструментів дозволяє виявити низку глибоких концептуальних протиріч і недоліків, які суттєво знижують загальну ефективність самостійної освіти [8]. Переважна більшість комплексних, професійно структурованих та інтерактивних онлайн-курсів та програмних симуляторів поширюються на платній основі, вимагаючи від студента значних фінансових інвестицій або регулярної дорогої підписки [9].

Такий стан речей виступає потужним психологічним та економічним бар'єром для потенційних початківців, школярів, студентів чи людей, які опинилися в складних життєвих обставинах і не мають фінансової можливості оплачувати навчання без гарантій успішного працевлаштування. З іншого боку, безкоштовні альтернативні джерела, на кшталт аматорських чи навіть професійних відеороликів на відеохостингах, страждають від тотальної фрагментації, хаотичності та повної відсутності системного педагогічного підходу. Навчальні відеоматеріали найчастіше створюються авторами відірвано

один від одного, не мають наскрізної логіки викладу, містять застарілу інформацію або фокусуються на вузьких прикладних завданнях, повністю ігноруючи фундаментальні теоретичні взаємозв'язки. У результаті початківець потрапляє в ситуацію інформаційного перевантаження, де він змушений самостійно виловлювати крупиці актуальних знань із хаотичного океану контенту, що призводить до швидкої втрати мотивації, формування хибних уявлень про технології та виникнення серйозних прогалин у базовій підготовці.

Оцінюючи архітектурні особливості сучасного програмного забезпечення, що використовується в освітньому процесі, неможливо оминути увагою глобальний зсув користувацьких уподобань від локальних, завантажуваних додатків до хмарних вебрішень. Станом на сьогоднішній день, завдяки широкому розповсюдженню високошвидкісного, безлімітного та стабільного доступу до мережі інтернет, необхідність попереднього скачування, встановлення та конфігурування важких програмних пакетів на локальні пристрої користувачів стала рудиментом минулого. Локальні додатки вимагають від початківця додаткових зусиль, пов'язаних із контролем сумісності операційних систем, виділенням вільного місця на жорсткому диску, постійним ручним оновленням версій та ризиком завантаження шкідливого програмного коду [10, 11].

На противагу цьому, ключовою, незаперечною та фундаментальною перевагою сучасних навчальних вебсистем є їхня абсолютна транскордонність та кросплатформеність. Користувачеві більше не потрібно виконувати жодних підготовчих дій на своєму комп'ютері: достатньо мати стандартний веб-браузер та стабільне з'єднання з мережею, щоб отримати моментальний доступ до всього масиву теоретичних знань, практичних завдань та інтерактивних тренажерів з будь-якої точки земної кулі. Це суттєво знижує поріг входження, дозволяючи залучати до процесу навчання ширші верстви населення, незалежно від технічних характеристик їхніх персональних пристроїв. Саме тому науково-практичне питання створення та програмної реалізації якісної, цілісної,

безкоштовної та систематизованої навчальної вебсистеми для вивчення основ вебпрограмування залишається надзвичайно актуальним, відкритим та затребуваним викликом для сучасної інженерної думки [12].

З технічної та архітектурної точок зору, сучасна навчальна вебсистема є складною сукупністю цифрових сторінок, розміщених під єдиним унікальним доменним ім’ям у глобальній мережі інтернет, які логічно пов’язані між собою наскрізною системою навігації, спільним стилістичним оформленням та єдиною базою даних. За кожною такою платформою закріплюється унікальна мережева адреса - уніфікований покажчик ресурсу, який дозволяє однозначно ідентифікувати її в кіберпросторі. Безпосередня взаємодія користувача з інтерфейсом вебсистеми здійснюється через спеціалізоване програмне забезпечення - браузер, який виконує роль інтерпретатора, трансформуючи отриманий із сервера програмний код у візуальні образи, інтерактивні елементи та текстові блоки. Фізично вся логіка додатка, сховища даних, медіафайли та текстові лекції розміщуються на одному або кількох віддалених серверах, які забезпечують цілодобову доступність ресурсу та обробку запитів від тисяч одночасних відвідувачів. Грамотно спроєктований інтерфейс такої системи повинен гармонійно поєднувати в собі простоту сприйняття, естетичну привабливість та ергономічність, надаючи студенту швидкий доступ до лекційного матеріалу, вікна практичного редактора коду, де він може відразу тестувати отримані знання, та аналітичних панелей, що відображають поточний прогрес навчання. Створення такого інструменту дозволяє ліквідувати розрив між теоретичною підготовкою та практичним застосуванням навичок, оскільки учень отримує можливість писати свій перший код безпосередньо у вікні браузера, миттєво бачачи результат своєї роботи та отримуючи автоматизований зворотний зв'язок від системи тестування.

Проведений аналіз предметної області чітко вказує на те, що популярність інтерактивних навчальних вебсайтів стрімко зростає, оскільки вони ідеально відповідають вимогам часу та запитам сучасного покоління студентів. Особливо

гостро ця потреба відчувається в ІТ-індустрії, де класичні підходи до написання паперових підручників чи тривалих лекційних курсів виявляються абсолютно нежиттєздатними через високу динаміку оновлення технологій.

Освітні вебплатформи нового покоління не просто надають сухий структурований текст, а виступають повноцінним інтерактивним середовищем, яке поєднує гейміфікацію, елементи штучного інтелекту для адаптації складності матеріалу під індивідуальні особливості учня та спільноти для взаємодопомоги. Завдяки тому, що доступ до вебсистем вимагає лише базового підключення до мережі інтернет, потенційна аудиторія таких проектів розширюється до глобальних масштабів, охоплюючи людей різного віку, соціального статусу та рівня попередньої підготовки. Студент може вивчати ази кодування, перебуваючи в дорозі, за допомогою смартфона, виконувати практичні вправи на планшеті або проектувати складні макети на настільному комп'ютері, що забезпечує безпрецедентний рівень свободи та інтеграції навчання в повсякденне життя. Розробка та масштабування безкоштовних спеціалізованих вебсистем для освоєння HTML, CSS та JavaScript дозволяє демократизувати сферу ІТ-освіти, відкриваючи шлях до високооплачуваної професії для мільйонів талановитих людей та створюючи надійну основу для розвитку світової цифрової економіки. Впровадження в такі системи вбудованих компіляторів та інтерпретаторів, які дозволяють виконувати JavaScript-код або рендерити HTML-розмітку в реальному часі безпосередньо на сторінці навчання, мінімізує технічні труднощі, з якими часто стикаються новачків при спробі налаштувати локальне середовище розробки на своєму ПК [13, 14].

Таким чином, фокусування зусиль на оптимізації архітектури навчальних веб-платформ, підвищенні інтерактивності їхнього контенту та забезпеченні суворої системності викладу матеріалу є головним та найбільш перспективним вектором розвитку сучасної дистанційної освіти в галузі комп'ютерних наук та програмування.

1.2. Порівняльний аналіз аналогів розробки

Аналіз сучасного стану розвитку індустрії дистанційної освіти у сфері інформаційних технологій, а особливо в секторі підготовки спеціалістів із базового фронтенд-інжинірингу, вимагає детального, комплексного та всебічного зіставлення існуючих програмних продуктів. Для проведення якісного порівняльного дослідження було обрано чотири репрезентативні веб-системи, які на сьогоднішній день мають значну користувацьку аудиторію та пропонують різноманітні методологічні підходи до вивчення кодування: W3schools, TeamtreeHouse, Codecademy та Css-tricks.

Кожна з цих технологічних платформ володіє своєю унікальною філософією побудови навчального процесу, специфічною архітектурою інтерфейсу, певними експлуатаційними перевагами та функціональними обмеженнями, які безпосередньо впливають на ефективність сприйняття складного технічного матеріалу початківцями. Ретельне вивчення цих аналогів дозволяє виявити вільні ніші на ринку освітніх послуг та сформувати надійну концептуальну основу для проектування та програмної реалізації власного інноваційного продукту, який покликаний усунути виявлені недоліки конкурентів та надати користувачам безкоштовний, збалансований та системний інструментарій для освоєння базового стеку вебтехнологій.

Перший розгляданий аналог, всесвітньо відома платформа W3schools (Рис. 1.1), традиційно позиціонується в інтернет-середовищі як один із найстаріших та найпопулярніших довідкових ресурсів і навчальних систем для програмістів-початківців. Головна концептуальна перевага цієї системи полягає в її колосальній багатопрофільності та величезному охопленні технологій, оскільки користувачам пропонуються теоретичні матеріали та найпростіші практичні завдання з величезної кількості мов програмування, розмітки та взаємодії з базами даних, включаючи JavaScript, Python, PHP, SQL та багато інших суміжних напрямків. Інтерфейс платформи побудований за принципом класичного інтерактивного підручника, де кожен теоретичний блок підкріплюється

вбудованим пісочним редактором коду, що дозволяє студенту експериментувати з синтаксисом безпосередньо у вікні браузера [15].



Рисунок 1.1 - Інтерфейс web-системи W3schools

Однак детальний критичний аналіз W3schools виявляє низку фундаментальних недоліків, які суттєво знижують її цінність як самостійної освітньої екосистеми для підготовки фахівців. Ключовою проблемою є те, що повний, глибокий та розгорнутий обсяг навчальних матеріалів, а також доступ до профільних сертифікаційних тестів, практичних проектів та розширених функцій відстеження прогресу відкривається лише після оформлення дорогої платної підписки преміум-рівня «Pro». Більше того, попри декларований широкий вибір мов, якість та глибина подачі матеріалу в безкоштовних модулях залишаються вкрай поверхневими та фрагментарними порівняно з іншими професійними аналогами. Матеріал подається у вигляді коротких ізольованих довідок, які чудово підходять як швидка шпаргалка для практикуючого розробника, але абсолютно не здатні сформувати у початківця цілісне, системне та архітектурне мислення, необхідне для створення самостійних комерційних проектів.

Другий об'єкт порівняльного аналізу, вебсистема TeamtreeHouse (Рис. 1.2), представляє зовсім інший, медійний та проектно-орієнтований підхід до

дистанційного навчання. Ця інтерактивна платформа фокусується на наданні користувачам доступу до високоякісних відеокурсів, розроблених професійними лекторами та практикуючими інженерами [16].

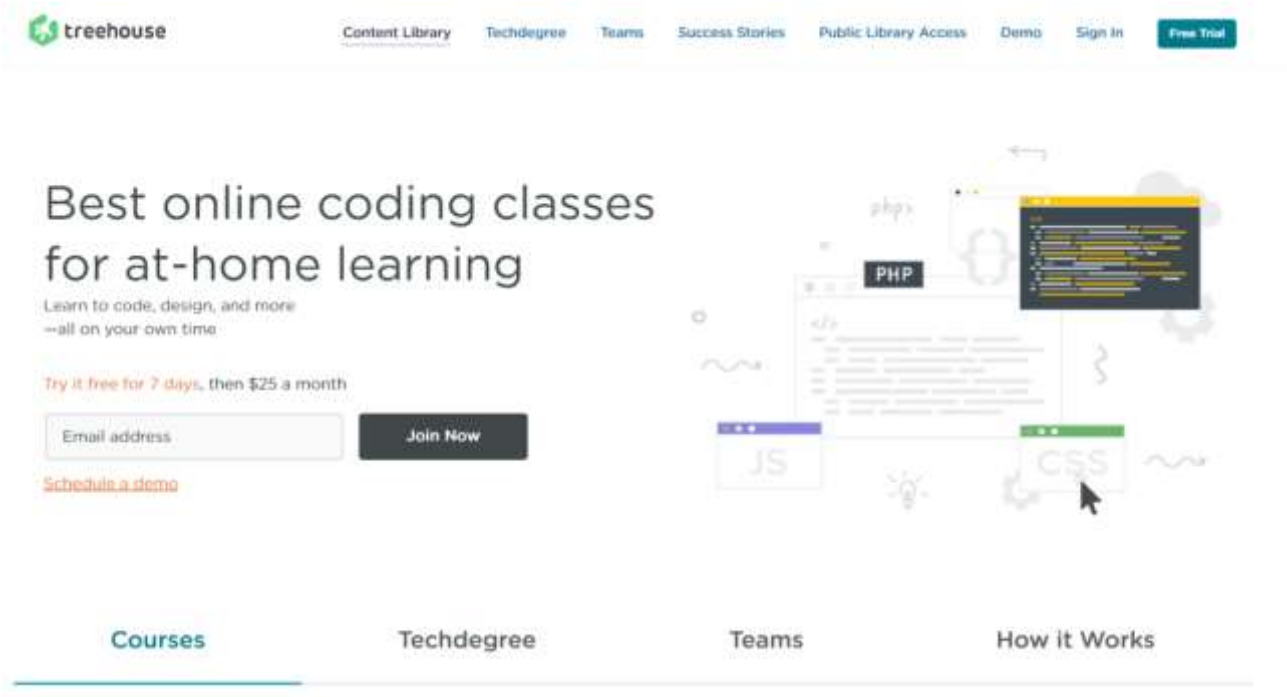


Рисунок 1.2 - Інтерфейс web-системи TeamtreeHouse

Основним ядром контенту тут виступають детальні програми вивчення мови розмітки гіпертексту HTML та каскадних таблиць стилів CSS, які супроводжуються покроковим розбором створення реальних макетів сайтів, адаптивною версткою та роботою з сучасними графічними інструментами. Візуальне оформлення та інтерфейс TeamtreeHouse відрізняються високою естетичністю, продуманою ергономікою та залучаючими елементами гейміфікації, що створює сприятливе перше враження та підвищує рівень утримання уваги студента. Проте з точки зору комплексної підготовки фронтенд-розробника дана система має архітектурні та економічні недоліки. З одного боку, у безкоштовному або базовому сегменті платформи спостерігається відчутний брак або повна відсутність структурованих та глибоких матеріалів для вивчення повноцінної мови програмування JavaScript, без якої неможливо уявити жоден

сучасний динамічний вебдодаток. З іншого боку, фінансова політика компанії є надзвичайно жорсткою та комерціалізованою: повністю безкоштовний доступ до всього масиву лекцій та практичних симуляторів надається користувачеві лише на дуже обмежений пробний період, який становить всього сім днів. Після завершення цього тижневого терміну подальше навчання стає абсолютно неможливим без обов'язкового щомісячного списання коштів за платну підписку, що повністю відсікає величезну аудиторію потенційних новачків, які не мають стабільного доходу або можливості оплачувати міжнародні освітні сервіси.

Третя досліджувана вебсистема, Codecademy (Рис. 1.3), здобула широке визнання завдяки своїй унікальній інтерактивній методиці, в основі якої лежить повна відмова від тривалих відеолекцій на користь негайної практичної діяльності [17].

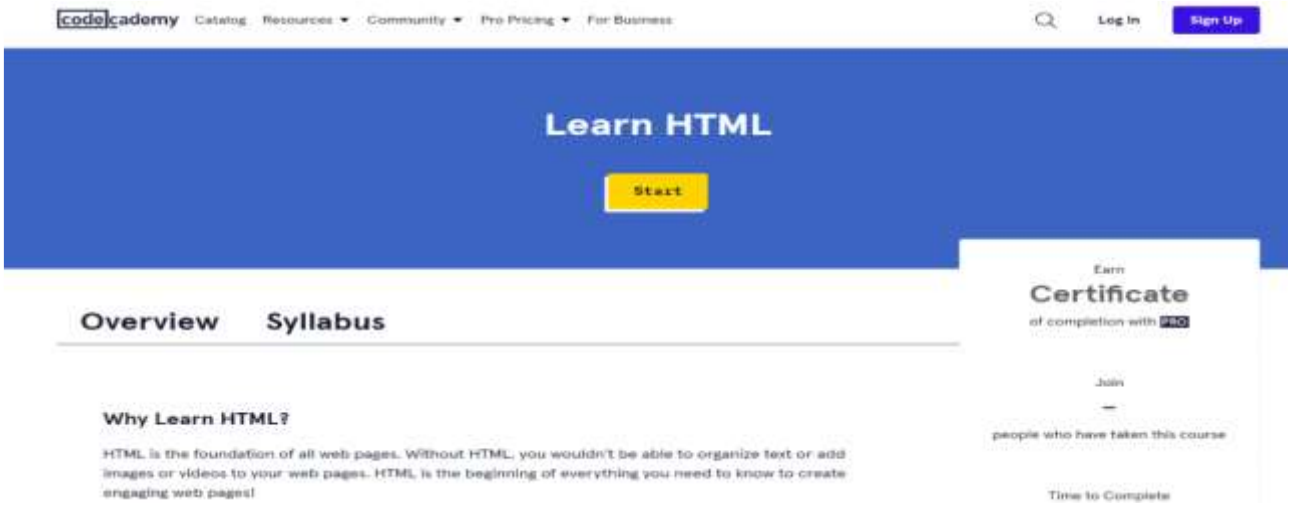


Рисунок 1.3 – Інтерфейс web-системи Codecademy

Інтерфейс цієї платформи розділений на кілька логічних зон: текстове пояснення концепції, вікно живого редактора коду та вбудована консоль для моментального відображення результату виконання скрипту. В межах аналізу предметної області було встановлено, що Codecademy надає чудові, детально пропрацьовані, логічно структуровані та послідовні матеріали для вивчення

мови розмітки гіпертексту HTML, дозволяючи студенту пройти шлях від написання найпростіших тегів до формування складних семантичних структур, а також пропонує офіційну сертифікацію після успішного складання фінальних іспитів. Проте головний і найбільш вагомий мінус цієї платформи в контексті базової підготовки вебпрограміста полягає у жорсткій сегментації та ізольованості курсів. У безкоштовній версії користувач часто стикається з відсутністю або серйозним обмеженням комплексних інтегрованих матеріалів для вивчення стилізації CSS та логіки JavaScript у зв'язці з HTML. Початківець змушений вивчати технології відірвано одна від одної, що створює серйозні труднощі при спробі об'єднати отримані знання для створення єдиного цілісного вебдодатка, перетворюючи навчальний процес на хаотичне виконання розрізнених синтаксичних вправ.

Четвертий аналог, вебсистема Css-tricks (Рис. 1.4)), кардинально відрізняється від трьох попередніх платформ, оскільки вона функціонує не як класичний покроковий онлайн-курс, а як масштабний, постійно оновлюваний експертний блог, ком'юніті-платформа та база практичних кейсів, орієнтована на глибоке вивчення та нестандартне застосування каскадних таблиць стилю CSS.



Рисунок 1.4 – Інтерфейс web-системи Css-tricks

Користувачі та гості сайту мають унікальну можливість безкоштовно ознайомитися з реальними складними прикладами роботи, фішками, трюками та

оптимізаційними рішеннями, створеними досвідченими світовими веб-розробниками. Цей ресурс є незамінним джерелом натхнення та підвищення кваліфікації для верстальників, які прагнуть освоїти технології Flexbox, CSS Grid, складну анімацію чи специфіку кросбраузерної оптимізації. Проте як самостійний засіб дистанційного навчання для новачків з нуля Css-tricks виявляється малоефективним через повну відсутність системного, послідовного педагогічного курсу та повне ігнорування суміжних фундаментальних технологій, зокрема мови програмування JavaScript та базових принципів структурування документів HTML. Початківець, потрапляючи на цей сайт, тоне у величезній кількості розрізнених статей та просунутих лайфхаків, не маючи при цьому базового розуміння того, як взагалі працює клієнт-серверна архітектура та логіка програмування [18].

Підсумовуючи результати проведеного всебічного порівняльного аналізу існуючих світових аналогів, можна сформулювати узагальнені висновки та чітко обґрунтувати високу актуальність, доцільність та практичну необхідність проектування та створення власної навчальної веб-системи. Жоден із розглянутих глобальних сервісів не здатний надати сучасному українському початківцю повністю безкоштовне, комплексне, взаємопов'язане та інтерактивне середовище, яке б одночасно і на належному рівні глибини об'єднувало в собі теорію та практику з HTML, CSS та JavaScript без прихованих комерційних обмежень чи втрати системності викладу матеріалу. В результаті виконання роботи буде розроблено та впроваджено веб-додаток, який повністю врахує всі критичні недоліки та прорахунки існуючих конкурентів. Створювана платформа забезпечить користувачам не лише доступ до структурованої та послідовної теоретичної бази, а і запропонує розгалужену систему вбудованих інтерактивних розвиваючих вправ, механізмів гнучкої оцінки та моніторингу прогресу, а також повноцінне інтегроване середовище розробки, що дозволить безкоштовно, ефективно та якісно опанувати фундаментальні основи вебпрограмування з нуля будь-якому користувачу, який має доступ до мережі інтернет.

1.3. Постановка задачі

Метою кваліфікаційної роботи бакалавра є створення веб-додатку з метою підвищення ефективності дистанційного вивчення веб-програмування шляхом поєєднання теоретичної бази, практичних прикладів та контролю знань, що дозволить забезпечити потреби користувача під час дистанційного навчального процесу.

Актуальність роботи зумовлена тим, що в умовах військової агресії, розробка дистанційної навчальної системи для вивчення основ веб-програмування, яка поєднує теорію та практичні приклади базових технологій, а також має функцію самоперевірки шляхом тестування, є важливим завданням.

Відповідно до поставлених завдань необхідно вирішити наступні задачі:

- розробити структурну схему та архітектуру розроблюваного веб-додатку;
- розробити метод та модель функціонування веб-додатку;
- розробити блоки теоретичного матеріалу та блоки з практичними прикладами для вивчення матеріалу;
- розробити блок самотестування користувача;
- розробити програмне забезпечення веб-додатку;
- здійснити тестування розробленого програмного забезпечення.

РОЗДІЛ 2. РОЗРОБКА МЕТОДУ, МОДЕЛІ ТА АЛГОРИТМІВ РОБОТИ ВЕБ-ДОДАТКУ

2.1. Аналіз навчального функціоналу веб-додатку

Завдання розробки полягає в створенні інтерактивного та дидактично орієнтованого навчального веб-додатку, головною місією якого є надання всебічної допомоги та підтримки користувачам, що прагнуть з нуля опанувати фундаментальні базисні знання у сфері сучасної розробки інтернет-ресурсів. Під час проектування та технічної реалізації цього програмного продукту першочергову увагу необхідно спрямувати на педагогічний та методологічний аспект функціонування веб-додатку, оскільки кінцевою метою є забезпечення необхідного рівня продуктивності, швидкості та результативності засвоєння нової інформації здобувачами.

Для того щоб процес написання коду не викликав у початківців перевантаження, фахівці у галузі комп'ютерної дидактики радять застосовувати перевірену часом методологію поетапного та порційного розподілу інформаційних масивів на логічно завершені невеликі фрагменти. Не рекомендується намагатися охопити інструментарій розробки одномоментно або вивчати складні концепції хаотично, адже подібний підхід не просто демонструє повну неефективність на практиці. Оптимальним рішенням є ретельне структурування та каскадне членування всього масиву знань за окремими тематичними модулями, де траєкторія навчання вибудовується за лінійним принципом від найпростіших базових дефініцій з поступовим, плавним переходом до складніших алгоритмічних завдань та архітектурних рішень.

Необхідним є органічне поєднання теоретичних лекційних матеріалів із практичними заняттями, глибокий та детальний аналіз уже готових працюючих програмних кодів, розбір реальних кейсів та прикладів із практичної діяльності. Саме такий формат дозволяє здобувачу перейняти кращі індустріальні практики, наочно побачити архітектурні тонкощі написання програм, мінімізувати ризик виникнення типових помилок.

Окрім вищезазначеного, важливу роль у загальній структурі освітнього циклу відіграє регулярна та об'єктивна верифікація ступеня засвоєння пройденого матеріалу, а також моніторинг динаміки зростання професійної кваліфікації студента. З метою впровадження цієї функції у веб-додаток інтегрується спеціалізований інструментарій для комплексного тестування користувачів, регулярне вирішення контрольних завдань безпосередньо після завершення теоретичного блоку є найбільш продуктивною формою експрес-діагностики залишкових знань.

Зважаючи на те, що масив інформації функціонує в межах цифрового простору, для максимізації показників юзабіліті є потреба у проектуванні ергономічного, інтуїтивно зрозумілого та графічно привабливого інтерфейсу користувача, який слугуватиме надійним навігатором та дозволить здобувачу легко орієнтуватися в архітектурі розділів, не відволікаючись від основного завдання. Також вагомим чинником успіху розроблюваного додатку є інклюзивність та технічна доступність для кінцевого споживача, тому необхідно реалізувати веб-додаток на принципах повної адаптивності. Під адаптивною верстkою та дизайном у сучасній індустрії мають на увазі такий спосіб побудови інтерфейсу, завдяки якому кодова структура сайту здатна динамічно підлаштовуватися під технічні параметри екранів і коректно, без спотворень відображатися на будь-яких типах пристроїв, починаючи від великогабаритних стаціонарних моніторів та ноутбуків і закінчуючи планшетами чи малорозмірними смартфонами.

2.2. Розробка структури інтерфейсу веб-додатку

Графічна оболонка та візуальний інтерфейс розроблюваного веб-додатка за своєю архітектурою та логікою побудови зобов'язані бути максимально інтуїтивними і прозорими навіть для тих відвідувачів, які вперше взаємодіють із даною цифровою платформою, що дозволить повністю нівелювати поріг входження та забезпечити комфорт і простоту в повсякденному використанні

системи. У момент відкриття та завантаження головної сторінки веб-додатку користувачеві надається можливість здійснити процедуру створення нового персонального облікового запису та пройти авторизацію, для чого безпосередньо по центру екранного простору інтегровано спеціалізований функціональний блок реєстрації, який укомплектовано електронною формою із відповідними полями для введення персональних ідентифікаційних даних (Рис. 2.1).

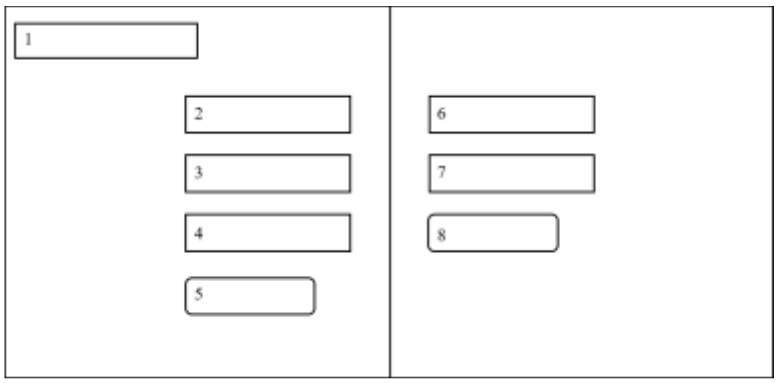


Рисунок 2.1 - Графічна схема блоку реєстрації

Блок реєстрації містить наступні елементи: заголовок блоку; поле «логін» для проведення реєстрації; поле «ім’я» для проведення реєстрації; поле «пароль» для проведення реєстрації; кнопка для проведення проведення реєстрації; поле «логін» для проведення авторизації; поле «пароль» для проведення авторизації; кнопка для проведення авторизації.

З метою забезпечення максимально комфортного, швидкого та інтуїтивно зрозумілого переміщення між інформаційними розділами та функціональними блоками розроблюваного веб-додатка, в його верхній частині спроектовано та інтегровано навігаційну панель, зовнішній вигляд та графічне виконання якої представлені на рис. 2.2. Цей керуючий елемент інтерфейсу є багатофункціональним і містить у собі як статичні локальні гіперпосилання для скролінгу та переходу до логічних блоків у межах відкритої сторінки, спеціальне динамічне випадаюче меню, наведене на рис. 2.3, яке при натисканні розгортає повний перелік лінків, що дозволяють відвідувачу переміщатися на інші

сторінки та внутрішні екрани веб-додатка.



Рисунок 2.2 - Навігаційна панель

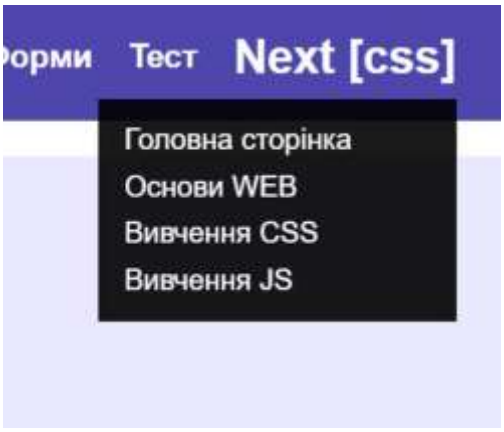


Рисунок 2.3 - Випадаюче меню

На цій же панелі стаціонарно розміщено текстовий або графічний логотип із назвою веб-додатка, який виконує роль глобального інтерактивного посилання, що дозволяє користувачеві з будь-якого місця миттєво повернутися на початкову головну сторінку веб-додатка. Для зручності навігації по навчальних матеріалах на сторінках додано зміст матеріалу (рис. 2.4). Навчальний матеріал розділено на блоки (рис. 2.5) для зручності користувача.



Рисунок 2.4 - Зміст матеріалу



Рисунок 2.5 - Блок з інформацією

Інтерактивні блоки, призначені для проведення контрольного тестування та перевірки рівня засвоєння знань користувачами, які наведені на рис. 2.6, оснащені набором стандартних радіокнопок-перемикачів, що дозволяють здобувачу швидко та безпомилково обрати один конкретний варіант відповіді, який він вважає правильним у межах поточного завдання. Блок тестування містить наступні елементи: заголовок блоку; питання; варіанти відповідей; індикатор номеру питання; кнопка для переходу до наступного питання.

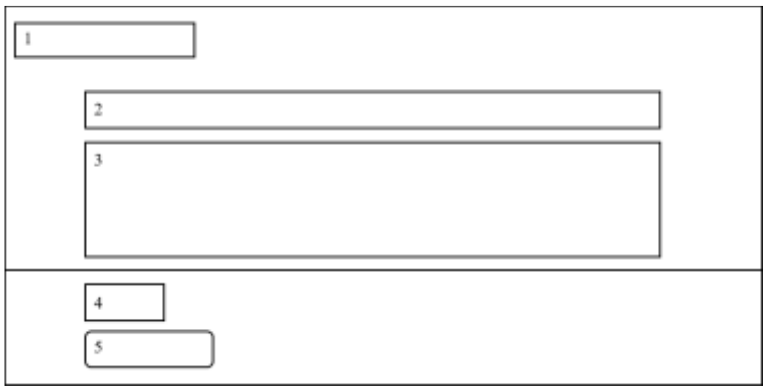


Рисунок 2.6 - Графічна схема блоку тестування

Для забезпечення динаміки навчального процесу в інтерфейсі передбачена функціональна кнопка з написом «Далі», при натисканні на яку система миттєво валідує поточний вибір, фіксує його у базі даних та здійснює автоматичний перехід екранного простору до наступного за чергою контрольного запитання

або практичного завдання. Окрім цього, у структурі програмного коду реалізовано керуючу кнопку «Заново», яка слугує для повного скидання поточних результатів та надання користувачеві можливості здійснити повторне проходження всього тестового модуля з самого початку. З метою оптимізації інтерфейсу та запобігання випадковому перериванню процесу оцінювання, дана кнопка перезапуску є динамічним елементом і стає видимою та доступною після того, як користувач повністю відповість на всі запитання і система зафіксує фінал тестування, вивівши підсумкові результати (Рис. 2.7).

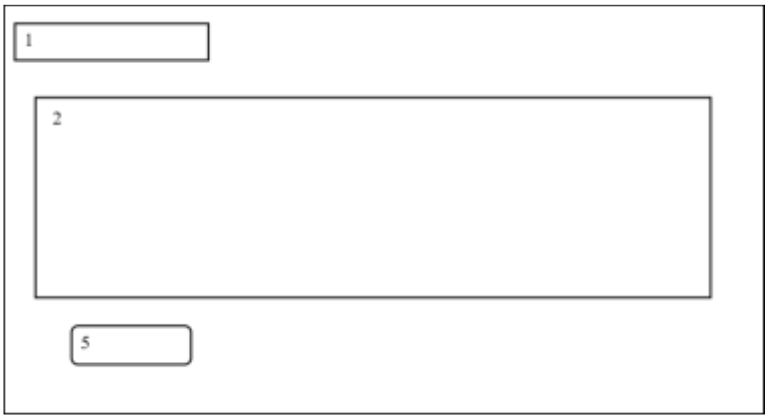


Рисунок 2.7 - Графічна схема блоку тестування при виведенні результатів

Оскільки на сенсорних дисплеях відсутня можливість класичного наведення курсора миші для розгортання вкладених елементів, безпосередньо поруч із основним текстовим посиланням було інтегровано додатковий самостійний графічний елемент у вигляді стрілки-маркера, спеціально призначений для виклику та розгортання прихованого випадаючого меню за допомогою звичайного дотику. Таке рішення дозволило повністю зберегти навігаційні властивості та функціональність самого базового посилання. Для забезпечення належного рівня ергономіки та графічного комфорту на пристроях, фізична ширина дисплея яких є меншою за граничне нормативне значення у сімсот шістдесят сім пікселів, спроектовано класичне компактне меню типу бургер (Рис. 2.8). Створення цього згортального компонента є важливим фактором для досягнення правильного та пропорційного відображення всіх

елементів головної навігаційної панелі на екранах смартфонів та малорозмірних мобільних телефонів.

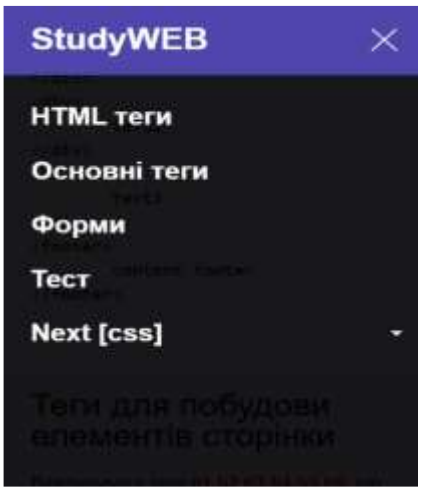


Рисунок 2.8 - Меню «Бургер»

Розробку графічного інтерфейсу веб-додатку проведено у середовищі Sublime Text.

2.3. Розробка моделі, методу та алгоритмів роботи веб-додатку

З метою надання користувачам веб-додатка можливості для оперативного самоконтролю, реалізовано інтерактивний метод тестування, архітектурна схема якого наведена на рис. 2.9.

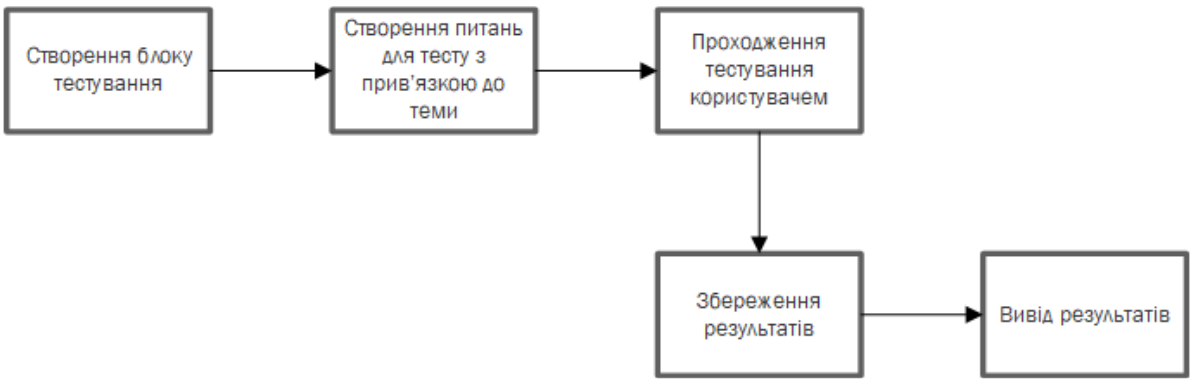


Рисунок 2.9 - Схема методу тестування для користувача

Даний метод передбачає конструювання та динамічну генерацію інтерфейсних блоків із тестами, візуальне виконання та розміщення яких наведено на рис. 2.10.

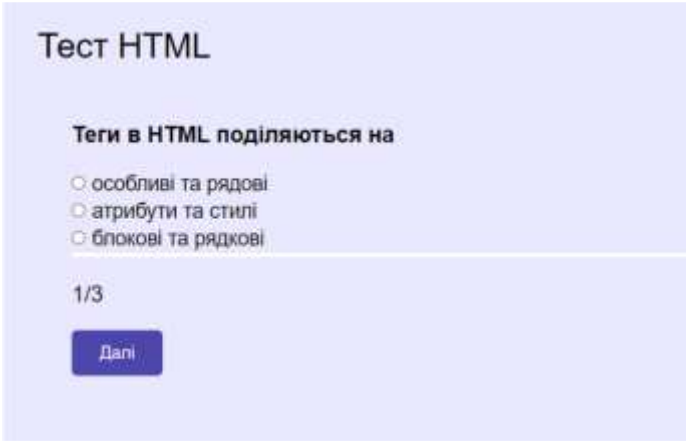


Рисунок 2.10 - Приклад блоку тестування

Ці модулі володіють високим ступенем універсальності і здатні містити в собі наперед задану кількість контрольних запитань різного рівня складності та довільний набір варіантів потенційних відповідей для кожного окремого питання. Відповівши на запропоновані системою завдання, здобувач отримує розгорнуту аналітику та свої кількісні результати успішності. Для закріплення вивченого матеріалу та опрацювання помилок, після повного завершення поточної перевірки користувач наділяється можливістю запустити цей процес повторно, для чого необхідно натиснути на інтерактивну керуючу кнопку з написом «Заново».

Запропонований та практично реалізований метод інтерактивного тестування надає користувачам чожливість оперативно здійснювати верифікацію поточного рівня своєї теоретичної підготовки безпосередньо в процесі активної взаємодії з веб-додатком. З технічної точки зору даний алгоритм контролю знань повноцінно використовує згенеровані та структуровані масиви даних, призначені для збереження текстових формулювань запитань, акумуляції відповідних їм варіантів відповідей, а також для оперативного запису та утримання поточних. Логічна модель і

функціональна архітектура роботи веб-додатку специфікована у формі уніфікованої UML діаграми діяльності [19] (рис. 2.11).

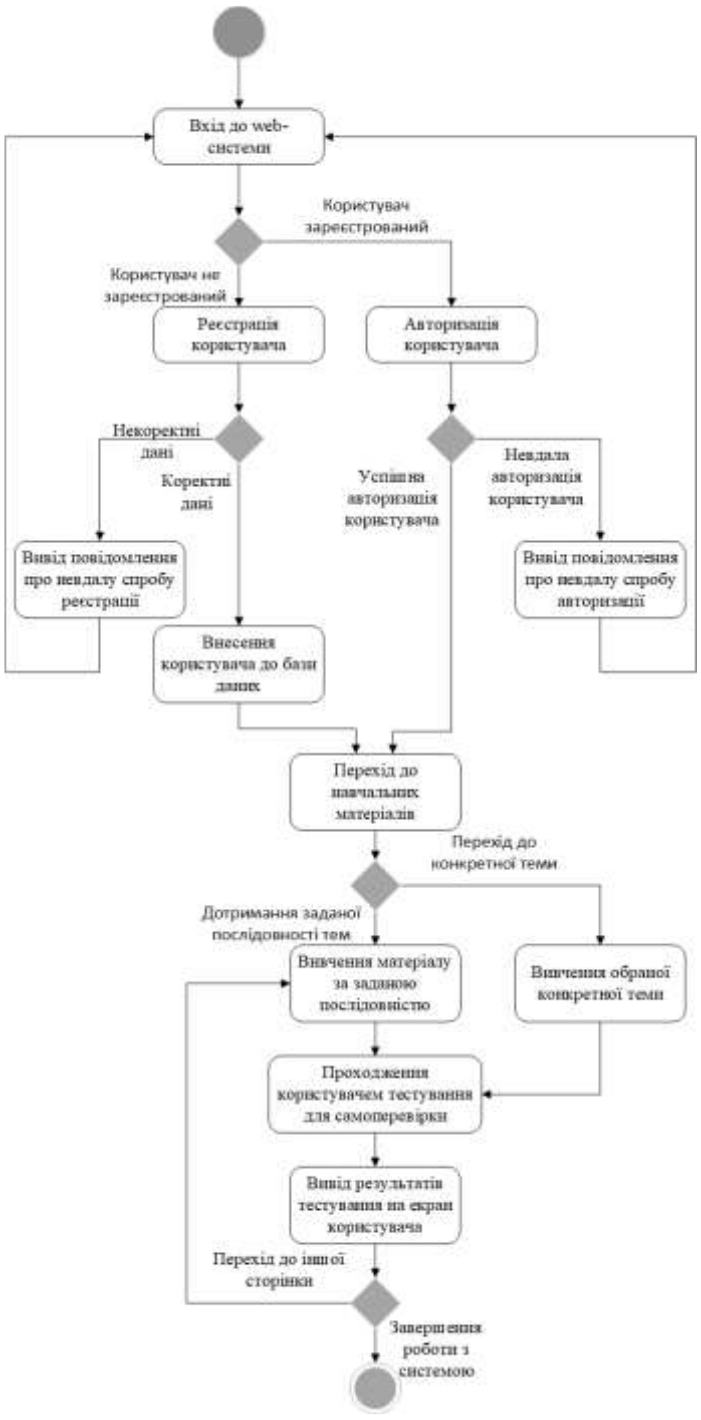


Рисунок 2.11 – UML-діаграма діяльності веб-додатку

Стандартна UML-діаграма діяльності є засобом візуального представлення орієнтованого графу діяльності, що є різновидом графу станів скінченного автомату, де ключовими вершинами виступають конкретні технологічні або

логічні дії, а системні переходи та зміна станів відбуваються строго по факту повного виконання та завершення цих безпосередніх дій.

Розроблено загальний алгоритм функціонування веб-додатку, відповідно до логіки якого здійснюється робота програмної платформи (рис. 2.12).

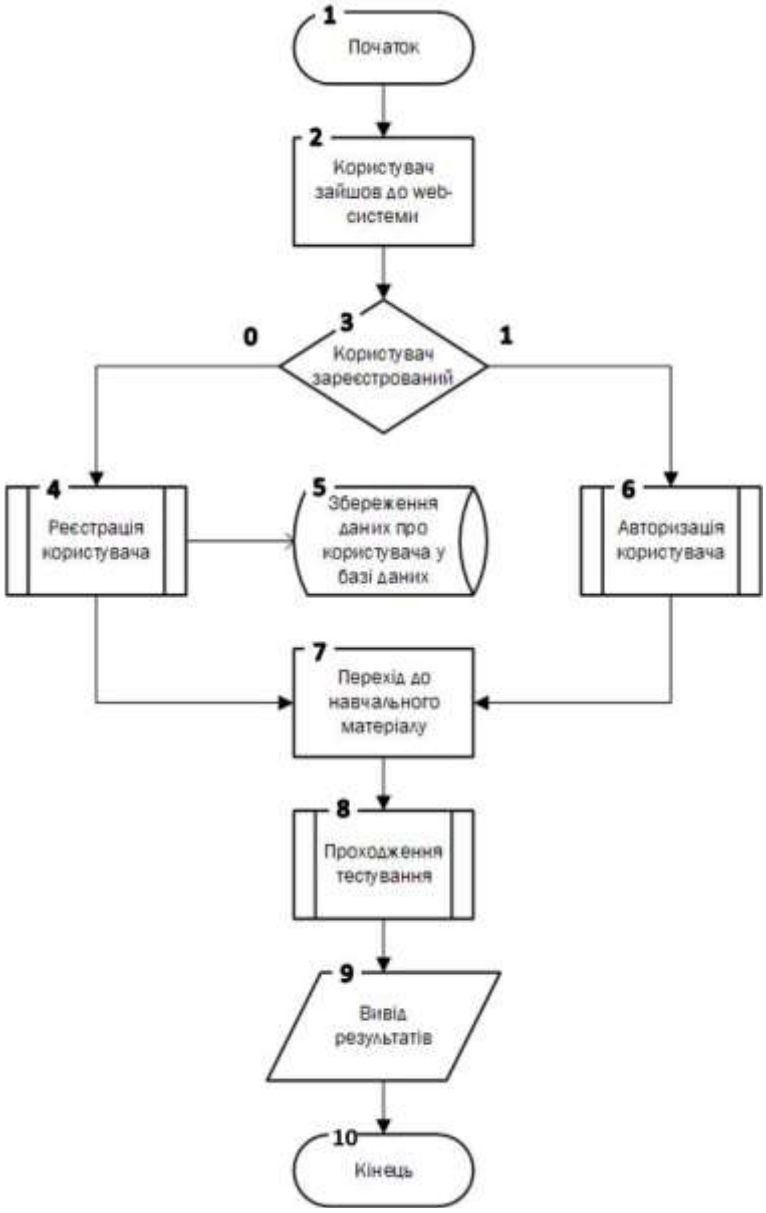


Рисунок 2.12 - Блок-схема загального алгоритму роботи веб-додатку

У момент реєстрації всі відомості про нового користувача автоматично піддаються валідації та заносяться до таблиць реляційної бази даних. Всі наявні дидактичні матеріали чітко розподілені на окремі самостійні веб-сторінки, а також додатково сегментовані на тематичні блоки за логічними напрямками.

Користувач володіє повною свободою дій і може або послідовно проходити весь навчальний курс від самого початку і до фінального розділу, або ж вибірково вивчати будь-яку конкретну цікаву для нього тему.

Також розроблено локальний алгоритм тестування користувачів, який наведено на рис. 2.13.

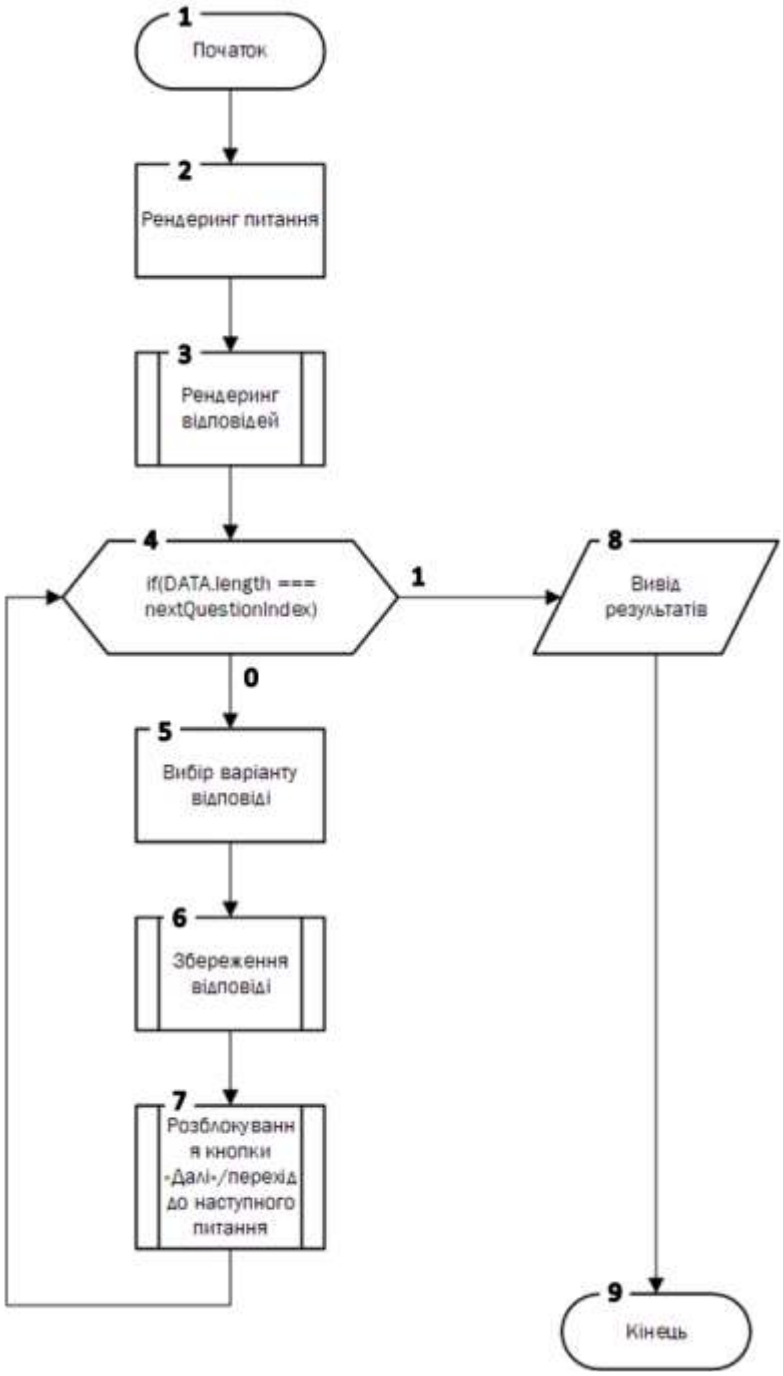


Рисунок 2.13 - Блок-схема алгоритму тестування користувача

Процес перевірки знань стартує автоматично безпосередньо з виведення на екран першого за чергою запитання, зафіксованого у блоці даних. З метою оптимізації інтерфейсу та запобігання випадковому пропуску завдань, для розблокування інтерактивної керуючої кнопки з написом «Далі» та здійснення подальшого переходу до наступного інформаційного завдання, користувачеві обов'язково необхідно позначити та зафіксувати свій варіант відповіді на поточне запитання. Кожна надана студентом відповідь перехоплюється програмними скриптами і акумулюється в оперативній пам'яті клієнта на весь час проходження поточного тесту, що є важливим для наступної коректної генерації та демонстрації фінальних аналітичних результатів.

Безпосередньо у момент ініціалізації переходу до кожного чергового запитання на стороні фронтенду відбувається автоматичний повторний рендеринг екранної форми, під час якого завантажується нове запитання та відповідні йому варіанти відповідей. Паралельно з цим процесом вбудований програмний лічильник виконує операцію інкрементування графічного індикатора порядкових номерів запитань, і якщо поточне запитання визначається алгоритмом як заключне у списку, то відразу після фіксації останньої відповіді користувач отримує повний доступ до перегляду підсумкових балів свого тестування. Одночасно з виведенням результатів на інтерфейсній панелі веб-додатка активується динамічна кнопка з написом «Заново», функціональне призначення якої полягає у наданні здобувачу можливості обнулити проміжні масиви даних та пройти тест повторно будь-яку кількість разів для закріплення навчального матеріалу.

РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ВЕБ-ДОДАТКУ

3.1. Варіантний аналіз і обґрунтування вибору засобів для програмної реалізації веб-додатку

Структурна основа розробленого веб-додатку побудована стандартними засобами класичної мови розмітки гіпертексту HTML, яка відповідає за каркас сторінок, каскадних таблиць стилів CSS, що забезпечують естетичний зовнішній вигляд і адаптивність інтерфейсу, а також динамічної мови програмування JavaScript, призначеної для написання гнучких клієнтських скриптів та реалізації всього інтерактивного функціоналу взаємодії з користувачем. Для ефективної роботи над технічною реалізацією даного веб-додатка обрано популярне текстове середовище для розробки Sublime Text [20], зовнішній інтерфейс якого наведений на рис. 3.1.



Рисунок 3.1 - Інтерфейс середовища для розробки Sublime Text

Даний програмний редактор дозволяє розробнику працювати з усіма необхідними для повноцінної реалізації веб-додатку мовами програмування, розмітки, стилізації та іншими технологіями. Окрім іншого, Sublime Text володіє надзвичайно високим ступенем індивідуальної персоналізації інтерфейсу під

потреби конкретного програміста, а також надає можливість завантажувати, підключати та комбінувати різноманітні додаткові плагіни, розширення та модулі для оптимізації процесу написання коду. У середовищі цього редактора за замовчуванням присутня вбудована панель швидкої навігації по структурі активного файлу, а також гнучка інтелектуальна система контекстного пошуку необхідних визначень, специфічних символів, функцій чи окремих рядків тексту, завдяки чому програміст може швидко орієнтуватися навіть у дуже великому за обсягом написаному коді. Задля досягнення найвищого рівня оптимізації та швидкості написання стильових правил, були використані інструменти та засоби популярного CSS препроцесора SASS, зокрема його найбільш розповсюдженого синтаксису SCSS. Для швидкої та безпомилкової компіляції вихідного коду препроцесора у стандартні стилі, було прийнято рішення скористатися спеціалізованою автономною програмою-збирачем проектів Prepros, загальний інтерфейс та панель керування якої наведено на рис. 3.2.

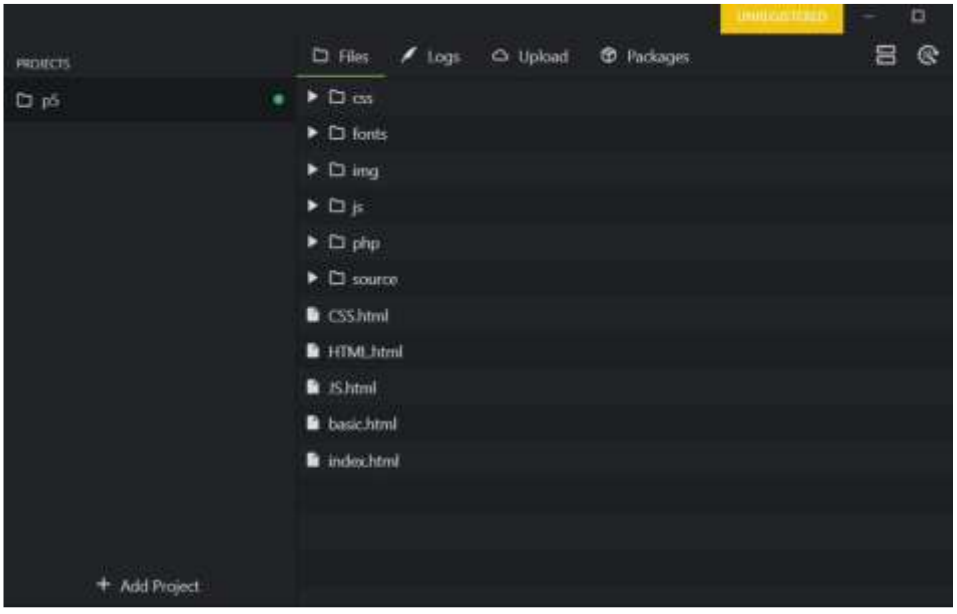


Рисунок 3.2 - Інтерфейс програми-збирача Prepros

Ця утиліта дозволяє в автоматичному режимі трансформувати та перетворювати складні вкладені формати сучасних препроцесорів у класичний, формат з розширенням .css, який є доступним для коректної інтерпретації будь-якими браузерами. Окрім того, інтегроване середовище Prepros за містить у собі

інформативну вбудовану консоль розробника, завдяки злагодженій та точній роботі якої програміст має можливість миттєво фіксувати, аналізувати та швидко виявляти будь-які потенційні помилки, синтаксичні збої чи неточності, що виникають у процесі написання програмного коду.

3.2. Розробка програмних засобів для адаптивної верстки та навігації

Для успішного та безпомилкового впровадження адаптивної верстки розроблюваного веб-додатка та правильного перенесення елементів інтерфейсу, у програмній логіці необхідно визначити поточний тип пристрою користувача, з якого безпосередньо здійснюється сесія, що покроково проілюстровано на рис. 3.3 та рис. 3.4.

```
1  "use strict"
2
3  const isMobile = {
4      Android: function () {
5          return navigator.userAgent.match(/Android/i);
6      },
7      BlackBerry: function () {
8          return navigator.userAgent.match(/BlackBerry/i);
9      },
```

Рисунок 3.3 - Визначення типу пристрою користувача

```
10     iOS: function () {
11         return navigator.userAgent.match(/iPhone|iPad|iPod/i);
12     },
13     Opera: function () {
14         return navigator.userAgent.match(/Opera Mini/i);
15     },
16     Windows: function () {
17         return navigator.userAgent.match(/IEMobile/i);
18     },
19     any: function () {
20         return (
21             isMobile.Android() ||
22             isMobile.BlackBerry() ||
23             isMobile.iOS() ||
24             isMobile.Opera() ||
25             isMobile.Windows());
26     }
27 };
```

Рисунок 3.4 - Визначення типу пристрою користувача

Після визначення типу пристрою користувача, необхідно адаптувати інтерфейс додатку під заданий тип. Для мобільних пристроїв, використовуючи метод classList.add, додається клас «_touch» (рис. 3.5).

```
if (isMobile.any()) {
    document.body.classList.add('_touch');
```

Рисунок 3.5 - Додання класу «_touch»

Даний клас визначено в каскадних таблицях стилів, він адаптує елементи інтерфейсу для пристроїв з тачскріном. Наступним кроком здійснюється адаптація під мобільні пристрої елементів з випадаючим меню (рис. 3.6).

```
let menuArrows = document.querySelectorAll('.menu__arrow');
if (menuArrows.length > 0){
    for (let index = 0; index < menuArrows.length; index++){
        const menuArrow = menuArrows[index];
        menuArrow.addEventListener("click", function(e){
            menuArrow.parentElement.classList.toggle('_active');
        });
    }
```

Рисунок 3.6 - Адаптація під мобільні пристрої елементів з випадаючим меню

Розміри дисплею мобільних пристроїв не дозволяють коректно відобразити панель навігації. Для вирішення цієї проблеми розроблено меню «Бургер» (рис. 3.7).

```
//Меню бургер

const iconMenu = document.querySelector('.menu__icon');
const menuBody = document.querySelector('.menu__body');
if(iconMenu){
    iconMenu.addEventListener("click", function(e){
        document.body.classList.toggle('_lock');
        iconMenu.classList.toggle('_active');
        menuBody.classList.toggle('_active');
    });
}
```

Рисунок 3.7 - Розробка меню «Бургер»

Задля забезпечення зручності та підвищення комфорту кінцевого користувача під час тривалого і послідовного вивчення структурованих освітніх матеріалів, розміщених усередині платформи, виникає необхідність забезпечити створюваний веб-додаток інтуїтивно зрозумілою системою навігації.

Основа цієї навігаційної системи реалізована стандартними інструментами та засобами класичної мови розмітки гіпертексту HTML, яка формує логічні зв'язки, та каскадних таблиць стилів CSS, які відповідають за візуальне позиціонування та зовнішній вигляд посилань. Безперешкодний динамічний перехід та автоматичне прокручування екрана технічно реалізовано на стороні фронтенду з активним використанням спеціалізованого користувацького атрибуту під назвою «data-goto», код якого наведено на рис. 3.8.

```
57  //Прокрутка при натисканні
58
59  const menuLinks = document.querySelectorAll('.menu__link[data-goto]');
60  if (menuLinks.length > 0){
61      menuLinks.forEach(menuLink => {
62          menuLink.addEventListener("click", onMenuLinkClick);
63      });
```

Рисунок 3.8 - Перехід за посиланням в межах однієї сторінки

3.3. Розробка модуля тестування

З метою контролю якості освоєння і закріплення теоретичного матеріалу користувачем веб-додатка, розроблено спеціалізовані інтерактивні блоки з контрольним тестуванням для кожної окремої тематики. Зовнішній користувацький інтерфейс та графічна оболонка даного освітнього блока реалізовані за допомогою стандартних засобів класичної мови розмітки гіпертексту HTML, яка формує структурний каркас завдань, а також каскадних таблиць стилів CSS, які відповідають за оформлення, колірну гаму та адаптивне розміщення елементів на екрані. Сам же інтерактивний тест та вся логіка його функціонування створені за допомогою гнучких засобів клієнтської мови програмування JavaScript, завдяки чому всі наявні екзаменаційні питання, підготовлені варіанти відповідей до них та правильні ключі зберігаються

безпосередньо всередині структурованого локального масиву даних у коді програми (рис. 3.9).

```
const DATA = [
    {
        question: 'Питання 1: питання',
        answers: [
            {
                id: '1',
                value: 'Відповідь 1',
                correct: true,
            },
            {
                id: '2',
                value: 'Відповідь 2',
                correct: false,
            },
            {
                id: '3',
                value: 'Відповідь 3',
                correct: false,
            },
        ]
    },
```

Рисунок 3.9 - Приклад елементу масиву

Відповідні одному питанню варіанти відповідей формують окремий масив, кожен варіант відповіді має унікальний ідентифікатор id. Оголошення констант методом getElementById наведено на рис. 3.10.

```
const quiz = document.getElementById('quiz');
const questions = document.getElementById('questions');
const indicator = document.getElementById('indicator');
const results = document.getElementById('results');
const btnNext = document.getElementById('btn-next');
const btnRestart = document.getElementById('btn-restart');
```

Рисунок 3.10 - Оголошення констант за допомогою методу getElementById

Функція renderQuestion (рис. 3.11), відповідає за рендеринг питань.

```
const renderQuestions = (index) => {
    renderIndicator(index + 1);

    questions.dataset.currentStep = index;

```

Рисунок 3.11 - Функція renderQuestion

Функція renderAnswers відповідає за рендеринг поточних варіантів відповідей (рис. 3.12, 3.13).

```
    const renderAnswers = () => DATA[index].answers
        .map((answer) => `
            <li>
                <label>
                    <input class="answer-input" type="radio" name=${index}
                    value=${answer.id}>
                    ${answer.value}
                </label>
            </li>
```

Рисунок 3.12 - Функція renderAnswers

```
        `)
        .join('');

    questions.innerHTML = `
        <div class="quiz-questions-item">
          <div class="quiz-questions-item__question">${DATA[index].question}
          </div>
          <ul class="quiz-questions-item__answers">${renderAnswers()}</ul>
        </div>
    `;
};
```

Рисунок 3.13 - Функція renderAnswers

Спеціалізована програмна функція renderResults, детальний вихідний код та логіка роботи якої наведені на рис. 3.14, використовується в архітектурі фронтенду для фінальної демонстрації користувачу його підсумкових результатів проходження контрольного тестування. За допомогою іншої допоміжної функції під назвою getClasname, внутрішні алгоритми системи автоматично визначають, який саме стильовий клас необхідно застосувати до

кожного конкретного запитання під час динамічного виведення підсумків на екран пристрою.

```
const renderResults = () => {
    let content = '';

    const getClassname = (answer, questionIndex) => {
        let classname = '';

        if (!answer.correct && answer.id === localResults[questionIndex]) {
            classname = 'answer--invalid';
        } else if (answer.correct && answer.id === localResults[questionIndex]
        {
            classname = 'answer--valid';
        } else if (answer.correct ) {
            classname = 'answer--invalid--correct';
        }

        return classname;
    };
```

Рисунок 3.14 - Функція renderResults

Службові класи «answer--invalid», «answer--valid» та «answer--invalid--correct» заздалегідь чітко описані та зафіксовані в каскадних таблицях стилів CSS задля забезпечення правильної стилізації, графічного підсвічування та кольорового виділення вірних, помилкових, а також пропущених чи скоригованих відповідей. Паралельно з цим, окрема функція getAnswers дозволяє додатку оперативно отримувати повний перелік відповідних варіантів відповідей з локального масиву для їхнього подальшого виводу на екран, тоді як за допомогою стандартного перебираючого методу forEach здійснюється послідовний обхід усіх елементів та остаточно формується загальний структурований результат для його відображення в інтерфейсі веб-додатка. Для рендерингу індикатора питань реалізовано функцію renderIndicator (рис. 3.15).

```
const renderIndicator = (currentStep) => {
    indicator.innerHTML = `${currentStep}/${DATA.length}`;
};
```

Рисунок 3.15 - Функція renderIndicator

Спеціалізований метод addEventListener в автоматичному режимі реєструє та відстежує будь-які зміни, кліки та інтерактивні події користувача, що відбуваються всередині виділеного структурного блока «quiz». Безпосередньо після фіксації вибору будь-якого варіанта відповіді, програмний скрипт миттєво змінює стан інтерфейсу та успішно розблоковує інтерактивний елемент - керуючу кнопку з написом «Далі», яка за замовчуванням та з міркувань захисту від випадкового пропуску завдань є недоступною і заблокованою для натискання на початковому етапі завантаження кожного нового питання (рис. 3.16).

```
quiz.addEventListener('change', (event) =>{
    //логіка відповіді
    if(event.target.classList.contains('answer-input')) {
        localResults[event.target.name] = event.target.value;
        btnNext.disabled = false;
    }
});
```

Рисунок 3.16 - Метод для реєстрації змін для блоку «quiz»

Перед переходом до кожного наступного запитання користувач може необмежену кількість разів передумати і змінити свій поточний вибір, обравши будь-який інший варіант відповіді з представленого списку. За допомогою вбудованого методу відстежування подій addEventListener програмний скрипт розроблюваного веб-додатка в автоматичному режимі реєструє та аналізує кожне фізичне натискання миші, здійснений користувачем у межах виділеного інтерактивного блоку «quiz» (рис. 3.17). У випадку, якщо поточний цільовий елемент, на який безпосередньо припав клік користувача, містить у собі специфічний системний клас «btn-next», внутрішня логіка програми зчитує цю подію та оперативно визначає точний порядковий індекс поточного питання для коректного розрахунку подальшого кроку тестування.

```
148  quiz.addEventListener('click', (event) =>{
149
150      if(event.target.classList.contains('btn-next')) {
151          const nextQuestionIndex = Number(questions.dataset.currentStep) + 1;
152
153          if(DATA.length === nextQuestionIndex) {
154              //перехід до результатів
155              questions.classList.add('questions--hidden');
156              indicator.classList.add('indicator--hidden');
157              results.classList.add('indicator--visible');
158              btnNext.classList.add('btn-next--hidden');
159              btnRestart.classList.add('btn-restart--visible');
160              renderResults();
```

Рисунок 3.17 - Метод для реєстрації натискань в межах блоку «quiz»

У випадку, якщо поточний визначений індекс стає рівним числу загальної кількості елементів, зафіксованих у локальному масиві екзаменаційних питань, внутрішня логіка системи автоматично завершує цикл опитування та переходить до фінального етапу генерації та виводу підсумкових аналітичних результатів тестування користувача на екран пристрою. В іншому ж випадку, коли лічильник ще не досяг ліміту, програмою здійснюється стандартний послідовний перехід до відображення наступного за чергою запитання. Одразу після виконання цього переходу інтерактивний елемент - керуюча кнопка з написом «Далі» - знову примусово переводиться в неактивний стан і блокується для натискання, залишаючись у такому положенні допоки користувач самостійно не визначиться і не обере конкретний варіант відповіді вже для поточного нового запитання. Якщо ж під час аналізу кліків виявляється, що поточний цільовий елемент містить у собі специфічний системний клас «btn-restart» (рис. 3.18), керуючий скрипт розроблюваного веб-додатка запускає процедуру повного скидання даних, обнуляючи масив збережених раніше відповідей користувача та всі сформовані проміжні результати, призначені для виводу. Паралельно з цим, за допомогою стандартного методу classList.remove, програма в автоматичному режимі повністю видаляє всі динамічно додані під час активного процесу тестування стильові класи елементів інтерфейсу, а головна функція рендерингу

та відображення питань повертає стан системи до початкової точки. Увесь цей алгоритм очищення та перезапуску дозволяє користувачу безперешкодно проходити контрольне тестування заново будь-яку необхідну кількість разів для максимального закріплення вивченого матеріалу.

```
165
166         btnNext.disabled = true;
167     }
168
169     if(event.target.classList.contains('btn-restart')) {
170         localResults = {};
171         results.innerHTML = '';
172
173         questions.classList.remove('questions--hidden');
174         indicator.classList.remove('indicator--hidden');
175         results.classList.remove('indicator--visible');
176         btnNext.classList.remove('btn-next--hidden');
177         btnRestart.classList.remove('btn-restart--visible');
178
179         renderQuestions(0);
180     }
181 });
```

Рисунок 3.18 - Обнулення масиву збережених відповідей

Висновок

Протягом виконання кваліфікаційної роботи бакалавра проведено проектування та програмна реалізація інтерактивної інтернет-платформи, націленої на інтенсифікацію та оптимізацію процесу опанування навичок веб-розробки. Головна цінність створеного ресурсу полягає в синергетичному поєднанні теоретичного базису, наочних практичних кейсів та модулів контролю знань з фундаментальних ІТ-дисциплін, які тепер зосереджені в єдиному цифровому просторі, усуваючи потребу звертатися до сторонніх джерел. На підготовчому етапі було здійснено аналітичний огляд сучасного ринку електронного навчання та виконано порівняльну характеристику наявних рішень. Особливу увагу в роботі було приділено дидактичній складовій, зокрема вивченню психолого-педагогічних підходів до ефективної подачі навчального матеріалу аудиторії початківців. На основі цього аналізу було спроектовано інтуїтивно зрозумілу архітектуру інтерфейсу, яка мінімізує навантаження на нових користувачів під час навігації сайтом. З метою автоматизації оцінювання результатів навчання було формалізовано тестування, а також розроблено загальну логічну модель і математичний алгоритм проходження інформаційних потоків усередині системи.

Практичний етап розробки завершився створенням та впровадженням низки взаємопов'язаних програмних модулів. До них належать гнучкі інструменти адаптивної верстки, що гарантують коректне відображення інтерфейсу на екранах будь-якого формату, ергономічні засоби навігації, захищені блоки реєстрації та автентифікації користувачів, а також інтерактивна система тестування. Фінальний етап верифікації софту, що включав стрес-тести та перевірку користувацьких сценаріїв, підтвердив стабільність архітектури, працездатність усіх заявлених функцій та повну відповідність вихідним вимогам технічного завдання.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Jennifer Campbell Web Design: Introductory / J. Campbell – Cengage Learning 2017. – 27 ст.
2. Панель інструментів розробника [Електронний ресурс] – Режим доступу до ресурсу: https://training.qatestlab.com/blog/technical-articles/developer-tools-in-the-browser-console-tab/.
3. Website [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Website.
4. W3schools [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3schools.com/.
5. TeamtreeHouse [Електронний ресурс] – Режим доступу до ресурсу: https://teamtreehouse.com/.
6. Codecademy [Електронний ресурс] – Режим доступу до ресурсу: https://www.codecademy.com/learn/learn-html.
7. Css-tricks [Електронний ресурс] – Режим доступу до ресурсу: https://css-tricks.com/.
8. Методологія БЕМ в дії [Електронний ресурс] – Режим доступу до ресурсу: https://avivi.pro/ua/blog/metodologiya-bem-v-deystvii/.
9. Dave Raggett Getting started with HTML / D. Raggett – W3C, may 2005.
10. CSS [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/CSS.
11. SASS [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Sass.
12. JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/JavaScript.
13. AcademyUA [Електронний ресурс] – Режим доступу до ресурсу: https://academyua.com/ua/stati/78-vchimosya-programuvati-6-porad-novachkam.
14. Адаптивна верстка сайту [Електронний ресурс] – Режим доступу до

ресурсу: https://webtune.com.ua/statti/web-rozrobka/adaptyvna-verstka-sajtu/.

15. Linda Heaton Unified Modeling Language Superstructure Specification, version 2.1.1. Document formal/2007-02-05. / L. Heaton – Object Management Group, February 2007. – 295 ст.

16. PHP [Електронний ресурс] – Режим доступу до ресурсу: http://programming.in.ua/web-design/allphp/30-about-php.html.

17. PhpMyAdmin [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/PhpMyAdmin.

18. BUSINESS SITE [Електронний ресурс] – Режим доступу до ресурсу: https://www.site2b.ua/web-blog/mobile-first-website-design.html.

19. Mobile First Design [Електронний ресурс] – Режим доступу до ресурсу: https://esputnik.com/uk/blog/mobile-first-design-pry-stvorenni-saitiv-shcho-vin-oznachaie-i-chomu-ie-nastilky-aktualnym.

20. Sam Lightstone Physical Database Design: the database professional's guide to exploiting indexes, views, storage, and more / Lightstone, S. Teorey T. Nadeau T. – Morgan Kaufmann Press 2007.