Міністерство освіти і науки України
Університет митної справи та фінансів

Факультет інноваційних технологій
Кафедра комп'ютерних наук та інженерії програмного забезпечення

Кваліфікаційна робота бакалавра

на тему: «Проектування та розробка вебсайту для тренування математичних навичок»

Виконала: студентка групи К21-1

Спеціальність: 122 Комп'ютерні науки

Кузіна Є. В.
(прізвище та ініціали)

Керівник: доц., к. фіз.-мат. н, доц. Лебідь О. Ю.
(науковий ступінь, вчене звання, прізвище та ініціали)

Рецензент: ДМСУ, Спеціалізоване управління розробки та супроводження програмного забезпечення, Департамент з питань цифрового розвитку, цифрових трансформацій та цифровізації
(місце роботи)

головний державний інспектор відділу розробки програмного забезпечення
(посада)

Бахтін О. В.
(науковий ступінь, вчене звання, прізвище та ініціали)

Дніпро
2025

АНОТАЦІЯ

Кузіна Є. В. Проектування та розробка вебсайту для тренування математичних навичок.

Кваліфікаційна робота на здобуття освітнього ступеня бакалавр за спеціальністю 122 «Комп'ютерні науки» – Університет митної справи та фінансів, Дніпро, 2025.

Об'єктом дослідження є процес проектування та розробки кросплатформеного вебсайту математичного тренажеру, що використовує стандартний стек веб-технологій (HTML, CSS, JavaScript).

Предмет дослідження – особливості реалізації алгоритмів для тренування математичних навичок усного рахунку в кросплатформеному застосунку з використанням веб-технологій.

Метою роботи є створення веб-платформи, що забезпечує ефективне та зрозуміле тренування математичних навичок з наведених тем: таблиця множення, дії з десятковими дробами, відсотки.

У процесі розробки проведено дослідження актуальності використання інтерактивних платформ у освіті, проведено аналіз вже існуючих програмних продуктів. Обґрунтовано вибір технологічного стеку (HTML, CSS, JavaScript) та розглянуто архітектурні рішення клієнтської сторони веб-застосунку.

Практична цінність роботи полягає у створенні сучасної інтерактивної платформи математичного тренажеру для тренування навичок усного рахунку, який можна використовувати як на заняттях так і для самостійної роботи. Веб-застосунок є кросплатформним, що дозволяє використовувати на будь-якому пристрої, для розробки використовується базовий стек веб-технологій.

Ключові слова: вебсайт, математичний тренажер, HTML, CSS, JavaScript.

ABSTRACT

Kuzina Ye.V. Design and Development of a Website for Training Mathematical Skills.

Bachelor's Thesis for the Educational Degree of Bachelor in Specialty 122 "Computer Science" – University of Customs and Finance, Dnipro, 2025.

The object of research is the process of designing and developing a cross-platform mathematical trainer website using a standard web technology stack (HTML, CSS, JavaScript).

The subject of research is the peculiarities of implementing algorithms for training mental math skills in a cross-platform application using web technologies.

The aim of the work is to create a web platform that provides effective and clear training of mathematical skills in the following topics: multiplication tables, operations with decimal fractions, percentages.

During the development process, a study on the relevance of using interactive platforms in education was conducted, and an analysis of existing software products was performed. The choice of the technology stack (HTML, CSS, JavaScript) was justified, and architectural solutions for the client side of the web application were reviewed.

The practical value of the work lies in the creation of a modern interactive mathematical trainer platform for training mental math skills, which can be used both in lessons and for self-study. The web application is cross-platform, allowing its use on any device, and a basic web technology stack is used for its development.

Keywords: web site, mathematical trainer, HTML, CSS, JavaScript.

ЗМІСТ

ВСТУП

Актуальність дослідження. Швидкий розвиток цифрових технологій, охоплює всі сфери життя, в тому числі і освіту, що дає можливість запровадження нових методів навчання. Використання інформаційних технологій та інтерактивних платформ є корисним та важливим елементом урізноманітнення традиційних методів навчання, що базуються на пасивному засвоєнні інформації та поступово втрачає свою ефективність.

Починаючи з 2020 року, із запровадженням дистанційного навчання, заклади освіти мусили адаптувати до використання онлайн-платорм у навчанні, саме тоді був побачений величезний потенціал використання інформаційних технологій, задля забезпечення якісного та ефективного навчального процесу.

Сучасне покоління дітей виросло в цифровому середовище та звикло до швидкого отримання інформації, взаємодії з привабливим інтерфейсом та елементами гри в повсякденному житті. Це створює попит на розробку та подальше використання різноманітних освітніх платформ чи вебсайтів, що забезпечать високий рівень залученості учнів та будуть підвищувати мотивацію до навчання.

Такі умови забезпечують актуальність та важливість створення конкурентоспроможного вебсайту, що буде надавати можливість вдосконалення та тренування важливих у навчанні навичок, таких, як усний рахунок.

Використання інтерактивних платформ або вебсайтів надає можливість персоналізації навчання, адаптації контенту до індивідуальних потреб учня. Однак, частина вже існуючих платформ не завжди використовують інтерактивні елементи та гейміфікацію.

Складнощі освітнього процесу, що зумовлені повномасштабним вторгненням, значною мірою впливають на погіршення результатів навчання та освітні втрати учнів, зокрема при вивчені математики.

Математика є основою для розвитку критичного та логічного мислення та навичок вирішення проблем. Хоча для багатьох математика є однією найскладніших дисциплін.

Ефективне засвоєння математичних знань вимагає систематичного виконання завдань та вправ, для вдосконалення та запам'ятовування певних тем. Для можливості повторення та вдосконалення навичок усного рахунку при будь-якій можливості, варто мати вебсайт, що можна використовувати на будь-якому пристрої.

Отже, розробка інтерактивного вебсайту – математичного тренажеру є надзвичайно актуальною задачею, що дозволяє підвищити ефективність навчання та засвоєння пройденого матеріалу з математики. Сприяє підвищенню мотивації та зацікавленості учнів цим предметом. Дозволяє використання за будь-яких умов при підключення до мережі Інтернет.

Мета дослідження. Розробка та реалізація освітнього вебсайту – математичного тренажеру для вдосконалення навичок усного рахунку, яка дозволить підвищити ефективність освітнього процесу використовуючи інтерактивні елементи та принципи геміфікації, такі як налічування балів та змінення рівнів складності.

Для досягнення поставленої мети слід визначити завдання дослідження:

– провести аналіз актуальності використання інтерактивних розробок у сучасній освіті;

– ознайомитися та проаналізувати вже готові рішення в предметній області;

– сформувати вимоги до вебсайту;

– провести аналіз цільової аудиторії та визначити основні потреби базуючись на проведеному опитування;

– вибрати та обґрунтувати вибір технологій для реалізації проекту;

– розробити структуру та архітектуру проекту;

– розробити та реалізувати дизайн інтерфейсу користувача;

– розробка та реалізація основного функціоналу платформи, що включає генерацію завдань, можливість введення та перевірки відповіді, надання зворотного зв'язку користувачеві;

– проведення тестування основних функціональних модулів та інтерфейсу і його адаптивності.

Об'єктом дослідження. Процес проектування та розробки кросплатформеного вебсайту математичного тренажеру, що використовує стандартний стек веб-технологій (HTML, CSS, JavaScript).

Предмет дослідження. Особливості реалізації алгоритмів для тренування математичних навичок усного рахунку в кросплатформеному застосунку з використанням веб-технологій.

Практичне значення роботи. Розробка функціонального та легкого у використанні вебсайту для вдосконалення математичних навичок усного рахунку, який можна використовувати у закладах освіти або для самостійної роботи. Вебсайт має допомогти підвищити зацікавленість та мотивацію учнів при вивченні математики, використовуючи елементи гейміфікації, такі, як нарахунок балів та складність, що поділяється на рівні. Велика кількість прикладів, що генеруються, надає можливість відпрацювання навички обчислення певних видів завдань, що допомагає закрити прогалини у зазначеній темі.

Структура та обсяг кваліфікаційної роботи. Кваліфікаційна робота складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків. Обсяг складає 65 сторінок. Робота містить 1 таблицю, 22 рисунки, додатки А, Б, В. Список використаних джерел складає 27 найменувань.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Актуальність використання інтерактивних платформ в сучасній освіті

Швидкий розвиток цифрових технологій, за останні 5 років створив всі умови для впровадження цифровізації в усіх сферах життя, особливо у навчанні. Кожний навчальний заклад, що хоче позиціонувати себе, як прогресивна та сучасна установа, має використовувати інтерактивні платформу в освітньому процесі.

У зв'язку із зовнішнім впливом, а саме пандемією COVID-19 з 2019 року та до тепер, в наслідок повномасштабного вторгнення, використання інтерактивного онлайн навчання є невід'ємною частиною здобування освіти українцями. Хоча деякі люди досі відносяться до цього скептично, неможливо заперечувати значні переваги такі, як можливість підтримки якісної освіти у нинішніх умовах.

Інтерактивні технології визначаються як інструменти та методи, що забезпечують активну участь студентів у процесі навчання. Вони можуть включати в себе віртуальну реальність, ігрові технології, мультимедійні інструменти тощо. Використання цифрових технологій у навчанні має на меті забезпечити студентам активне залучення до навчального процесу, створюючи сприятливі умови для ефективного засвоєння навчального матеріалу [27].

Використання інтерактивних технологій у сучасній освіті не лише покращує процес навчання, але й має значний вплив на розвиток студентів у різних аспектах їхнього життя. Однією з основних переваг є стимулювання розвитку критичного та проблемного мислення у здобувачів освіти.

Цифровізація забезпечує заохочення учнів до аналізу інформації та пошуку шляху вирішення проблемних ситуацій, що сприяє переходу від

пасивного сприйняття інформації до активної взаємодії з навчальним матеріалом. Отримані навички є важливими в наш час, де швидкий розвиток технологій потребує постійної адаптації та змоги вирішення складних завдань.

Інтерактивні технології – це інструменти, що дозволяють учням та викладачам взаємодіяти з навчальним матеріалом у режимі реального часу через цифрові пристрої, такі як інтерактивні панелі, віртуальні дошки, планшети, смартфони та ноутбуки [10]. Вони дозволяють створювати динамічний та персоналізований освітній простір, де інформація не лише передається від викладача до студента, але й активно обробляється та використовується.

Технологічний прогрес сприяв розвитку дистанційної освіти, яка забезпечує легкий доступ до всіх навчальних ресурсів і дозволяє учасниками освітнього процесу зручно взаємодіяти один з одним. Онлайн-освіта дозволяє здобувачам освіти працювати з широким спектром навчальних матеріалів, доступних у відкритому доступі в Інтернеті, таким чином створюючи середовище для самостійного навчання.

Інтерактивні платформи сприяють індивідуалізації навчання. Це дозволяє адаптувати навчальний матеріал до індивідуальних потреб та темпу навчання кожного учня [17]. Вони надають можливість відстежувати прогрес учнів та надавати їм персоналізований зворотний зв'язок, що дозволяє учням краще засвоювати матеріал і досягати вищих результатів.

Онлайн-платформи та інтерактивні інструменти можуть створювати можливості для спільної роботи над проектами та завданнями, навіть якщо студенти знаходяться на різних географічних розташуваннях. Це сприяє розвитку комунікативних та навичок співпраці, які є ключовими у сучасному ринку праці [17].

Використання інтерактивних платформ у сучасній освіті є не просто тенденцією, а необхідністю. Вони дозволяють зробити освітній процес більш ефективним, цікавим та доступним для кожного учня.

Постійний розвиток технологій та доступ до мережі Інтернет надає можливість використання нових інструментів та застосування інтерактивних платформ та вебсайтів для навчання. На сьогодні в мережі доступна велика кількість інтерактивних матеріалів, тренажерів та інших освітніх ресурсів, які можна використати при вивченні різних дисциплін. Це дозволяє викладачам та вчителям використовувати вже готові матеріали, що значно економить час підготовки уроку.

Щоб оцінити важливість цих проблем та дізнатися думку зацікавлених осіб, а саме вчителів та репетиторів математики, було проведено анонімне онлайн-опитування. Результати визначені при вибірці – 20 людей.

Провівши аналіз отриманих результатів можемо зазначити, що в середньому рівень навичок усного рахунку в дітей на низькому-середньому рівні (рис. 1.1). 8 людей проголосувало за низький та середній рівень, та 2 за високий і ще 2 не змогли визначитися.

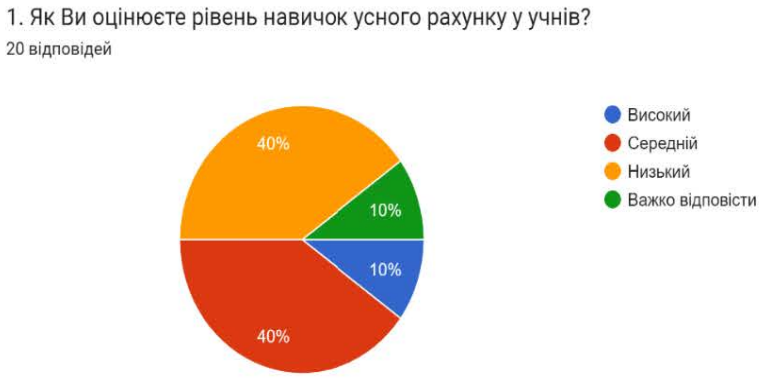

Рисунок 1.1 – Результат опитування

Наступне питання: «Наскільки важливим Ви вважаєте розвиток навичок усного рахунку в учнів?». Відповіді учасників розділилися навпіл: 50% (10 людей) вважають, що навички усного рахунку дуже важливі та 50% – скоріше важливі. Тобто з точністю можна стверджувати, що вміння усно рахувати є важливою та корисною навичкою у навчанні.

Для більшого розуміння проблематики наступним питанням було: «Чи відчувають учні труднощі з усним обчисленням під час навчання математики?» (рис. 1.2.).

Рисунок 1.2 – Результати опитування

В результаті можемо побачити, що проблеми пов'язані з навичками усного рахунку, виникають не часто (65% – 13 людей обрали відповідь «Іноді»). Тобто можна припустити, що проблеми виникають лише у деяких темах, що потребують усного рахунку.

Для розуміння чи актуальне створення інтерактивної платформи – математичний тренажер. Поставлено питання: «Чи використовуєте Ви додаткові ресурси або тренажери для вдосконалення усного рахунку учнів?» (рис. 1.3.).

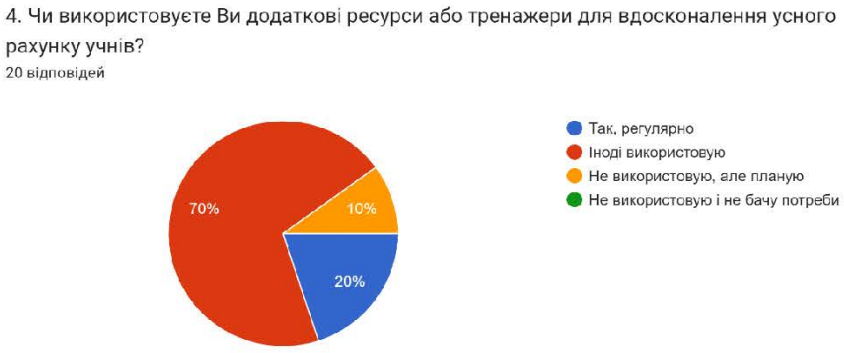

Рисунок 1.3 – Результати опитування

Отже, за наведеною діаграмою можна зазначити, що 70% опитуваних, тобто 14 з 20 людей, іноді використовують інтерактивні платформи на власних уроках для покращення навичок усного рахунку. Тобто можна стверджувати, що розроблений продукт – вебсайт математичний тренажер матиме попит.

Наступне питання: «Наскільки корисним, на вашу думку, є онлайн-тренажер з усного рахунку для школярів?» (рис. 1.4.). В загальному більшість опитуваних (90% – 50% скоріше корисним, 40% дуже корисним) вважає, що використання онлайн-тренажеру з усного рахунку дійсно може покращити навички усного рахунку.

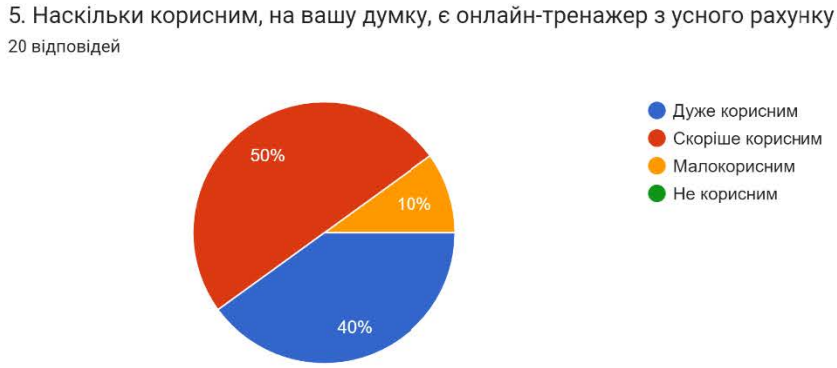

Рисунок 1.4 – Результати опитування

На питання: «Чи зацікавилися б Ви або Ваші учні сайтом, який допомагає тренувати усний рахунок у зручному інтерактивному форматі?» Відповіді були: 50% – так, обов'язково, та 50% можливо, спробую. Отже можна зазначити що розробка такого вебсайту однозначно зацікавила цільову аудиторію.

Наступне питання допоможе визначити для якої вікової категорії доречно розробляти тренажер. (рис. 1.5.).

За наявною діаграмою можемо визначити, що 40% (8 людей) опитуваних вважають доречною розробку для учнів 5-6 класів, 35% (7 людей) – для всіх вікових категорій та 25% (5 людей) – для початкової школи 1-4 класи. Отже,

можемо зазначити, що цей тренажер може бути корисних для учнів будь-якого віку.

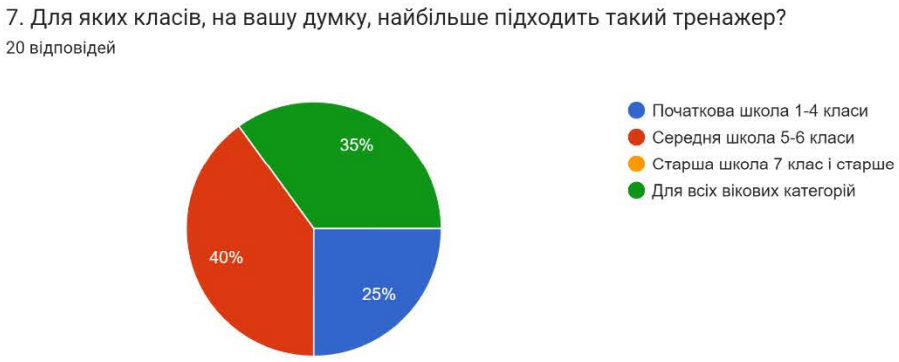

Рисунок 1.5 – Результати опитування

Наступне питання допоможе визначити, які теми варто обрати для тренажера, що принесуть найбільше користі. На розгляд опитуваних було представлено такі теми з математики, як: таблиця множення, дії з натуральними числами, дії з дробами, дії з раціональними числами, відсотки та перетворення одиниць вимірювання (рис. 1.6.). Опитуванні могли обрати декілька відповідей.

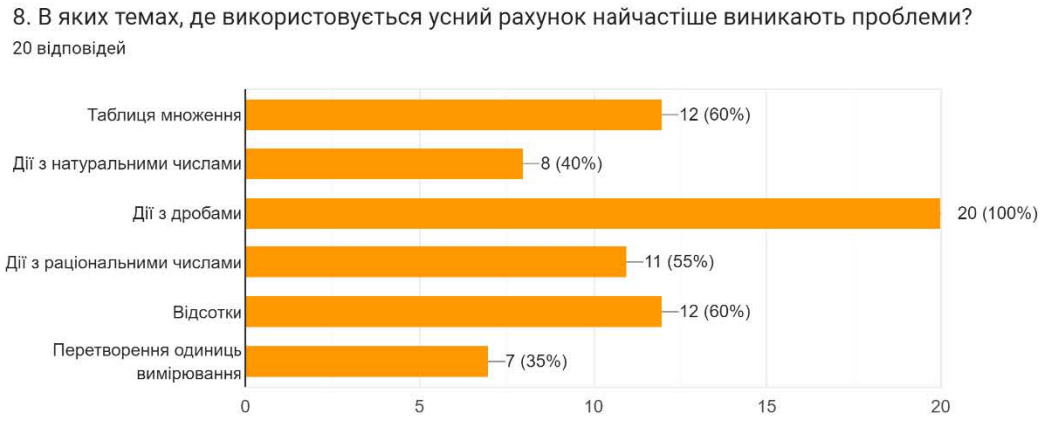

Рисунок 1.6 – Результати опитування

За поданою гістограмою можемо визначити, що одноголосно (20 голосів) найбільш проблематичною темою є дії з дробами, на другому місці

(по 12 голосів) таблиця множення та відсотки, далі 11 голосів дії з раціональними числами, 8 голосів – дії з натуральними числами та 7 голосів перетворення одиниць вимірювання.

Отже можна визначити, що за результатами опитування тренажер міститиме такі теми, як дії з дробами, таблиця множення та відсотки.

Останнім питанням опитування, було: «Які функції Ви б хотіли бачити в такому тренажері?». Отримані відповіді:

- швидкий усний рахунок з таймером;
- можливість використання на будь якому девайсі;
- щоб він передбачав різні завдання для різних класів;
- швидкісний тренажер на таблицю множення, на дії з дробами;
- великий вибір тем та типів завдань;
- можливість підрахунку балів та відстеження прогресу;
- велика кількість різноманітних завдань на одну тему;
- дії з великими числами;
- збереження результатів та статистика;
- можливість додавати власні завдання.

Деякі із запропонованих функцій будуть реалізовані у створюваному вебсайті, а інші для майбутніх оновлень.

Отже, інтерактивні технології відкривають широкі можливості для подальшого розвитку освіти, що сприятиме покращенню якості та ефективності навчання та розвитку навичок, необхідних для успішної кар'єри в сучасному світі. Актуальність розробки є очевидною, оскільки використання вебсайтів зможе покращити зайкавленість та мотивацію при виконанні однотипних дії, таких як обчислення математичних прикладів.

1.2 Ігри та гейміфікація у навчанні

Культура цифрових ігор зростає, охоплюючи значну частину населення світу. Ігрова індустрія виробляє постійний потік ігор, які продовжують

розширюватися за своєю природою та впливом – вони можуть бути художніми, соціальними та спільними, дозволяючи великій кількості людей з усього світу брати участь одночасно. Дослідження Американської психологічної асоціації 2013 року підкреслює когнітивний, мотиваційний, емоційний та соціальний вплив відеоігор на поведінку людини; цей значний обсяг досліджень підкреслює величезний потенціал ігор для навчання новим формам мислення та поведінки [4].

Гейміфікація – це інтеграція інтерактивних елементів у неігровий контекст (зокрема в освіту) з метою вирішення завдання. Використання елементів гри, таких як бали, рівні, відзнаки та змагання, у діяльності, що не пов'язана безпосередньо з грою, сприяє підвищенню мотивації, залученню учнів та поліпшенню результатів навчання [17].

Гейміфікація в освіті – це використання ігрових технік та елементів у навчальному процесі, що створює інтерактивне та мотивуюче середовище. Вона використовується для покращення залученості та зацікавленості студентів, стимулювання їх до активної участі в навчанні та формування позитивного ставлення до навчального процесу [21]. Основні цілі гейміфікації полягають у покращенні певних здібностей, впровадженні цілей, що надають навчанню мети, залученні учнів, оптимізації навчання, підтримці зміни поведінки та соціалізації.

Основні елементи гри, що застосовуються в освітніх процесах:

1. Бали та відзнаки. Учні після виконання завдання можуть отримати бали або нагороди, це стимулює їх досягати кращих результатів. Одним з прикладів може бути впровадження накопичувальної системи балів, які можна буде обміняти на реальний приз.

2. Рівні та прогрес. Прогрес через рівні або етапи надає учням відчуття досягнення та розвиток. Відстеження прогресу дозволяє контролювати ефективність навчального процесу.

3. Змагання та лідерборди. Здорова конкуренція є зовнішньою мотивацією учнів, що породжує певний азарт та заохочує учнів до навчання.

4. Історії і сценарії. Використання сюжетних ліній перетворює освітній процес на більш цікаве, зрозуміле і залучене дійство. Перетворення учня зі спостерігача на учасника вигаданої чи реальної події створює ефект особливої значущості засвоєного матеріалу.

Гейміфікація довела свою ефективність у багатьох освітніх дослідженнях. Зокрема, дослідження, проведене Массачусетським технологічним інститутом (MIT), показало, що студенти, які навчаються за допомогою ігрових методик, показують на 34% кращі результати порівняно з тими, хто використовує традиційні методи навчання [3].

Психологічні аспекти гейміфікації пояснюються її здатністю активувати різні мотиваційні фактори. Зокрема, вона задовольняє потреби в компетентності, особистому рості, зміцненні самооцінки та визнання, соціальній взаємодії, що є основними компонентами внутрішньої мотивації. Гейміфікація використовує принцип позитивного підкріплення, коли учні отримують винагороди за досягнення. Це надає їм негайний зворотний зв'язок, який підсилює бажання продовжувати навчання. Прогресування через рівні та досягнення цілей створює відчуття успіху, що також додатково мотивує студентів.

Як окремий приклад, можна розглянути використання геймифікації на уроках з математики. Зазвичай навчання математики є сухим та формалістичним: учня мають завчити формули та теореми, розв'язують рівняння та задачі, виконують обчислення. В традиційному підході викладання бракує зв'язку навчання з реальним навколишнім світом. Учні в свою чергу не зацікавленні у процесі та в результаті мають не задовільні результати.

Можна зазначити, що використання ігрового процесу може змінити думку учнів стосовно вичення математики.

Використання ігрових платформ може знизити рівень стресу при вивченні математики, роблячи процес більш захопливим та менш загрозливим. Також інтерактивні ресурси дозволяють наочно продемонструвати певні

математичні поняття, наприклад фігури та аксіоми стереометрії, тригонометричні функції тощо.

Однією з невід'ємних частин вивчення математики є розв'язання однотипних прикладів для відпрацювання навичок та поліпшення швидкості виконання. Цей процес також можна модифікувати та зробити більш цікавим для учнів використовуючи математичні тренажери.

1.3 Огляд та аналіз вже існуючих програмних продуктів в сфері освіти

Програмні продукти в сфері освіти відіграють важливу роль, забезпечуючи нові можливості для організації навчального процесу, підвищення його ефективності та забезпечення доступності якісної освіти. Наразі у вільному доступі наявна велика кількість програмних продуктів, що значно ускладнює вибір.

Найбільш популярними інтерактивними платформами, на яких можна забезпечити повторення та закріплення пройденого матеріалу з математики є: Novatika, Learning.ua та Wordwall.

Novatika – це сайт, що пропонує математичні ігри, що доступні онлайн та робочі аркуші, які можна роздрукувати для вивчення математики. Вебсайт охоплює широкий спектр тем, включаючи математику 1-6 клас та алгебру і геометрію 7-8 клас.

Основною перевагою сайту є генерування завдань, тобто на одну тему можна отримати велику кількість однотипних прикладів, які будуть відрізнятися при кожному проходженні. Інтерфейс сайту простий і зрозумілий, але трохи застарілий. На платформі доступні різні формати завдання, з вводом відповіді або вибором з наявних. Також користування платформою є абсолютно безкоштовним.

Освітня платформа Новатіка є гарним інструментом для закріплення навичок по певній темі.

Learning.ua – це найбільша освітня онлайн-платформа в Україні, яка пропонує інтерактивні завдання, цікаві задачі, ігри, нагороди та сертифікати для дітей дошкільного та шкільного віку. Платформа охоплює різні предмети, включаючи математику, українську мову, читання, а також пропонує матеріали для підготовки до НМТ. Освітні програми повністю відповідають вимогам Міністерства освіти України та міжнародним стандартам Common Core.

На платформі доступні різноманітні завдання, включаючи приклади, задачі, рівняння та завдання на логіку. Всі задачі генеруються автоматично, також можливі різні види подання завдань з однієї теми. Інтерфейс сучасний та зрозумілий, кожна вправа представлена як рівень гри, за проходження якого учні отримують винагороду. Також є можливість відслідковування прогресу учнів, зазначення його слабких місць. Хоча повний функціонал платформи доступний лише за наявності платної підписки, у безкоштовному використання можна виконати 3-5 вправ.

Learning.ua є корисною платформою для інтерактивного навчання дітей, яка допомагає зробити навчання цікавим та ефективним.

Wordwall – це онлайн-платформа, яка дозволяє вчителям та іншим користувачам легко створювати власні інтерактивні навчальні ресурси та друковані матеріали. Вона пропонує широкий спектр шаблонів, які можна налаштувати під різні предмети та вікові групи, включаючи математику.

Ця платформа відрізняється від минулих тим, що завдання потрібно створювати самостійно, або використовувати вже створені та наявні у бібліотеці сайту. Інтуїтивно зрозумілий інтерфейс дозволяє швидко створювати вправи, без спеціальних технічних навичок. Велика кількість шаблонів дозволяє використовувати одну й ту саму вправу у різних варіаціях, але більшість доступні лише при платній підписці.

Загалом, Wordwall є цінним інструментом для вчителів, які хочуть створювати цікаві та інтерактивні навчальні матеріали з математики та інших предметів, а також використовувати готові ресурси від спільноти.

Проаналізувавши обрані програмні продукти можна зробити висновок, що розробка вебсайту, що забезпечить тренування навичок усного рахунку є актуальним. Оскільки за останні роки дистанційного навчання учні зазнають освітніх втрат, в тому числі з математики.

1.4 Постановка завдання кваліфікаційної роботи

Математичний тренажер з усного рахунку буде корисним для учнів усіх класів та, навіть, дорослих, оскільки розвиток навичок усного рахунку є важливим аспектом математичної підготовки. Традиційні методи навчання не завжди є ефективними через відсутність інтерактивності, гнучкості та індивідуального підходу. Ідеальною комбінацією може стати поєднання традиційних методів навчання з використанням інтерактивних ресурсів. Таким чином виконання одноманітних прикладів можна урізноманітнити використанням математичного тренажеру.

Основною метою даної кваліфікаційної роботи є розробка вебсайту математичного тренажера для вдосконалення навичок усного рахунку, який генерує завдання з обраної теми тим саме забезпечуючи повторення матеріалу.

Розроблений тренажер повинен мати наступний функціонал:

1. Генерація завдань – система повинна створювати математичні вирази з обраної теми (додавання, віднімання, множення, ділення, комбіновані завдання).

2. Адаптація складності – рівень складності завдань змінюється в залежності від успіхів користувача.

3. Тестовий режим – при виконанні завдання зараховуються правильні відповіді, а при помилці програма показує правильний результат.

4. Інтуїтивно зрозумілий інтерфейс – зручний дизайн, що дозволяє сконцентруватися на завданні та не відволікає сторонніми об’єктами. Таким чином покращуючи ефективність.

Основні вимоги до програмного продукту можна поділити на три основні категорії: функціональні, нефункціональні та технічні.

Функціональні вимоги описують поведінку системи, тобто дії, які повинна виконувати програма. До таких вимог можна віднести:

– генерація випадкових математичних виразів відповідно до заданих умов та правил;

– можливість вибору типів операцій (арифметичні дії, рівняння тощо);

– автоматична перевірка правильності відповідей;

– надання зворотного зв'язку, щодо правильності виконання завдання.

Нефункціональні вимоги:

– простий і зручний користувацький інтерфейс;

– можливість роботи на різних пристроях (ПК, планшети, смартфони);

– висока швидкодія генерації та перевірки завдань;

– мінімальні вимоги до апаратного забезпечення;

Технічні вимоги:

– реалізація програмного продукту на базі вебтехнологій (HTML, CSS, JavaScript) для кросплатформності;

– використання JavaScript для генерації та перевірки математичних виразів;

– можливість подальшого розширення функціоналу (наприклад, додавання нових типів завдань, графічного відображення прогресу тощо);

Виконання даних вимог дозволяє створити ефективний математичний тренажер, який сприятиме покращенню навичок усного рахунку у користувачів та забезпечить покращення ефективності навчання.

1.5 Висновки до першого розділу

Стрімкий розвиток цифрових технологій та вплив зовнішній чинників, роблять якісну онлайн освіту необхідністю для забезпечення безперервного та ефективного навчання. Використання інтерактивних платформ та геміфікаія

освіти сприяють активній участі здобувачів освіти, стимулюють критичне мислення та забезпечують персоналізований підхід до навчання.

Отже, використання інтерактивних ресурсів у сучасній освіті – це не просто тренд, а нагальна потреба. Вони роблять освітній процес ефективнішим, цікавішим і доступнішим для кожного. Постійний розвиток цифрових технологій і доступ до мережі Інтернет відкривають нові можливості для використання цифрових ресурсів у навчальному процесі.

Впровадження ігрових елементів, таких як бали, рівні, змагання та історії, робить процес навчання більш цікавим та захопливим, особливо у таких дисциплінах, як математика, яка традиційно вважається складною та формальною.

В розділі було проведено огляд та аналіз вже існуючих та часто використовуваних в сфері освіти програмних продуктів, а саме: Novatika, Learning.ua та Wordwall. Визначено переваги та недоліки кожного з продуктів.

Для підтвердження актуальності роботи було проведено анонімне опитування. В результаті було опитано 20 людей. За результатами опитування можна стверджувати, що розробка інтерактивної платформи, що дозволяє тренувати навички усного рахунку використовуючи елементи гейміфікації є дійсно важливим для розвитку сучасної освіти в сучасних умовах.

Метою кваліфікаційної роботи визначено розробку вебсайту математичного тренажера для відпрацювання навичок усного рахунку з базових тем математики. Основною функцією тренажера має стати генерація прикладів за обраною темою, автоматично перевірка відповідей та виведення результату.

Отже, розробка математичного тренажера є актуальною та обґрунтованою задачею. Існуючі інтерактивні платформи мають певні переваги, але також і обмеження, що створює простір для розробки власного.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБСАЙТУ МАТЕМАТИЧНОГО ТРЕНАЖЕРУ

2.1 Аналіз методів вирішення задачі

Ефективна розробка математичного тренажера вимагає ретельного вибору технологій, що відповідають цілям проєкту, потребам цільової аудиторії та наявним ресурсам. Сучасний ринок технологій дозволяє обрати з широкого спектру можливостей, саме, те що буде відповідати встановленим. критеріям розробки. Вибір платформи є одним з найважливіших стратегічних рішень, оскільки він безпосередньо впливає на доступність продукту, вартість його розробки та подальшої підтримки, а також на кінцевий користувацький досвід.

Серед основних варіантів платформ для розгортання подібного програмного забезпечення виділяються мобільні, десктопні та веб-застосунки. Кожен з цих підходів має свої унікальні характеристики, переваги та недоліки, які необхідно детально розглянути для прийняття обґрунтованого рішення. Такий комплексний аналіз дозволить визначити оптимальну архітектуру та інструментарій, що забезпечить максимальну ефективність та досяжність цілей проєкту.

Мобільна розробка – це процес створення програмного забезпечення, спеціально розробленого для використання на мобільних пристроях, таких як смартфони та планшети. Ці застосунки можуть бути встановлені безпосередньо на пристрій через магазини застосунків (наприклад, Google Play Store для Android та Apple App Store для iOS) або іншими способами.

Основними характеристиками мобільної розробки є орієнтація на мобільні пристрої, а саме інтерфейс та функціонал розробляється з урахуванням невеликих розмірів екрану, сенсорного екрану та специфічних можливостей [7].

Десктопна розробка – це процес створення програмного забезпечення, яке запускається на персональних комп'ютерах (ПК) або ноутбуках під певними операційними системами (Windows, macOS, Linux). Ці застосунки встановлюються безпосередньо на комп'ютер користувача та використовують ресурси операційної системи та апаратного забезпечення.

Інтерфейс десктопної розробки розробляється з урахуванням використання миші та клавіатури, великого розміру екрану та потужного апаратного забезпечення порівняно з мобільними пристроями.

Десктопні застосунки мають широкий доступ до ресурсів комп'ютера, таких як процесор, оперативна пам'ять, файлова система, а також підключене периферійне обладнання.

Веброзробка – це процес створення вебсайтів та вебзастосунків, які доступні користувачам через веббраузери (наприклад, Chrome, Firefox, Safari, Edge) без необхідності встановлення додаткового програмного забезпечення. Вебзастосунки можуть бути різної складності, від простих інформаційних сайтів до складних інтерактивних [12].

Доступ до веброзробки користувачі отримують через URL-адресу в браузері на будь-якому пристрої, підключеному до Інтернету. Отже, Вебзастосунок не потребує встановлення на пристрій користувача, що спрощує процес доступу. Але доступ залежить від стабільного Інтернет з'єднання. Вебзастосунки зазвичай є кросплатформними, оскільки вони працюють у будь-якому сучасному веббраузері незалежно від операційної системи.

Розуміння цих відмінностей є критично важливим для прийняття рішення щодо оптимального вибору платформи, яка найкраще задовольнить потреби цільової аудиторії та специфічні вимоги математичного тренажера. Таким чином, вибір технологічної основи базуватиметься на всебічному аналізі переваг та недоліків кожного підходу.

Для наочності та спрощення вибору між представленими основними платформами розробки складено порівняльну Таблицю 1.

Таблиця 1

Порівняльна характеристика основних платформ розробки

Характеристика	Мобільний застосунок	Десктопний застосунок	Веб-застосунок
Доступність	Завжди під рукою на мобільному пристрої	Доступний лише на компютері, де встановлений	Доступний з будь-якого пристрою, що має підключення до Інтернету
Встановлення	Потребує завантаження та встановлення	Потребує завантаження та встановлення	Не потребує встановлення, миттєвий доступ через браузер
Оновлення	Оновлення через магазин	Потребує ручного оновлення	Оновлюється автоматично на сервері
Продуктивність	Висока продуктивність завдяки оптимізації під платформу	Висока продуктивність завдяки потужності комп'ютера	Продуктивність залежить від інтернет-з'єднання
Зручність	Зручний для використання "на ходу", оптимізований під сенсорне керування.	Зручний для тривалої роботи з великим обсягом інформації, використання клавіатури та миші.	Легкий доступ з будь-якого пристрою, адаптивний дизайн.
Витрати	Розробка для різних платформ може бути дорожчою	Розробка для кожної ОС може бути окремим проєктом, що підвищує вартість	Одноразова розробка для всіх платформ може бути економічнішою

За вимогами вебсайт математичного тренажеру має бути швидко доступною на будь якому пристрої, на ПК для того, щоб вчитель міг демонструвати екран із завданнями для спільного використання на уроках, та на смартфонах для самостійної роботи учня. Можливість автоматичного оновлення, що відбуваються на сервері та стає доступним усім користувачам одночасно, спрощує підтримку та оновлення платформи.

Отже, для зазначених вимог доречніше всього використовувати веброзробку. Оскільки вебдодатки працюють та адаптуються під використання на будь-якому пристрої, що надає доступ ширшій аудиторії.

2.2 Вибір програмних засобів для реалізації проекту

Для розробки вебсайту математичного тренажеру для покращення навичок усного рахунку обрано використовувати базовий набір веб-технологій, а саме HTML, CSS та JavaScript. Цей вибір базується на ряді переваг, які визначають його оптимальним для досягнення мети проекту.

HTML (HyperText Markup Language) є стандартизованою мовою розмітки, що використовується для створення структури вебсторінок. CSS (Cascading Style Sheets) відповідає за візуальне представлення HTML-елементів, дозволяючи контролювати їхній зовнішній вигляд, розташування та інші стильові характеристики. JavaScript є потужною та гнучкою мовою програмування, яка додає інтерактивність та динамічну поведінку вебсторінкам [23].

У сукупності, ці три технології є фундаментом сучасних вебтехнологій. Їхня широка підтримка означає, що існує величезна кількість безкоштовних навчальних матеріалів, що значно спрощує процес розробки та знижує витрати на навчання та розробку.

Основною перевагою вебзастосунків, що створені на основі HTML, CSS та JavaScript, є їхня вбудована кросплатформність. Оскільки ці технології інтерпретуються веббраузерами, тренажер буде доступний через будь-який сучасний браузер незалежно від операційної системи, встановленої на пристрої користувача. Це означає, що користувачі зможуть отримати доступ до навчальної платформи як на своїх персональних комп'ютерах (Windows, macOS, Linux), так і на мобільних пристроях (Android, iOS) без необхідності встановлення окремих застосунків. Такий підхід значно розширює потенційну аудиторію проекту, що розробляється, та спрощує процес його розповсюдження.

Використання базового набору вебтехнологій забезпечує створення основного функціоналу розробки:

– забезпечення інтерфейсу користувача, відображення згенерованих завдань, за допомогою HTML та CSS;

– обробка дій користувача, реагування на взаємодію з елементами інтерфейсу, перевірка правильності відповідей, за допомогою JavaScript;

– надання зворотного зв'язку, візуальне інформування користувача про результати перевірки за допомогою динамічної зміни HTML-елементів через JavaScript та стилізації цих змін за допомогою CSS.

Тобто для реалізацій базових функцій платформи цілком достатньо використовувати HTML для структурування контенту, CSS для його оформлення та JavaScript для забезпечення інтерактивності [23].

HTML використовується як основа для створення логічної структури вебсторінок математичного тренажеру.

CSS використовується для візуального оформлення елементів інтерфейсу, що створює інтуїтивно зрозумілий та візуально привабливий досвід для користувача. Основними аспектами використання CSS є:

– розробка кольорової схеми та загального стилю тренажера, що створюють гармонійних дизайн;

– детальне оформлення окремих елементів, таких, як кнопки, поля виведення та введення інформації;

– впровадження макетування для створення адаптивної розмітки, яка буде коректно відображати зміст сторінок на екранах будь-яких розмірів;

– використання медіа запитів для налаштування стилів залежно від параметрів пристрою, таких як розмір екрану чи орієнтація.

JavaScript відіграє ключову роль у забезпеченні інтерактивності та реалізації логіки проекту. Використання JavaScript дозволяє виконувати основні функції математичного тренажеру такі, як:

– обробка події користувача, що забезпечує відстеження дій користувача: натискання кнопок, введення відповідей;

– динамічне оновлення контенту, створення функції відображення нових згенерованих математичних прикладів;

– реалізація алгоритмів генерації завдань, що працює за встановленими умовами;

– перевірка введених користувачем відповідей та надання зворотного зв'язку.

Основним інструментом для розробки використовується інтегроване середовище розробки Visual Studio Code (VS Code).

Тобто, використання базових інструментів веброзробки, може бути достатнім для створення ефективного вебсайту математичного тренажеру з повним функціоналом та доступністю для широкого кола користувачів.

2.3 Проектування архітектури платформи

Розробка ефективного та зручного вебсайту математичного тренажера вимагає не лише обґрунтованого вибору технологій, але й ретельного проектування його архітектури та функціональних компонентів.

Оскільки створюваний проект реалізується з використанням базогвого набору вебтехнологій: HTML, CSS та JavaScript, без залучення сторонніх фреймворків або серверної логіки він є клієнтським вебзастосунком, тобто вся програма виконується на стороні клієнта, безпосередньо у браузері, не потребуючи постійного обміну даних із сервером.

Для того щоб зробити використання сайту простим та ефективним, перед розробкою необхідно детально продумати структуру програмного забезпечення: які частини входять до системи та як вони взаємодіють між собою. Для візуалізації структури проекту можна використати Component Diagram (діаграму компонентів) [14]. Ця діаграма надає можливість наочно продемонструвати залежність між елементами проекту (рис. 2.1).

Такий підхід дозволяє забезпечити модульність та масштабованість застосунку, що спрощує його подальшу розробку та підтримку. Чітке розмежування компонентів сприяє легкому тестуванню та інтеграції нових функцій.

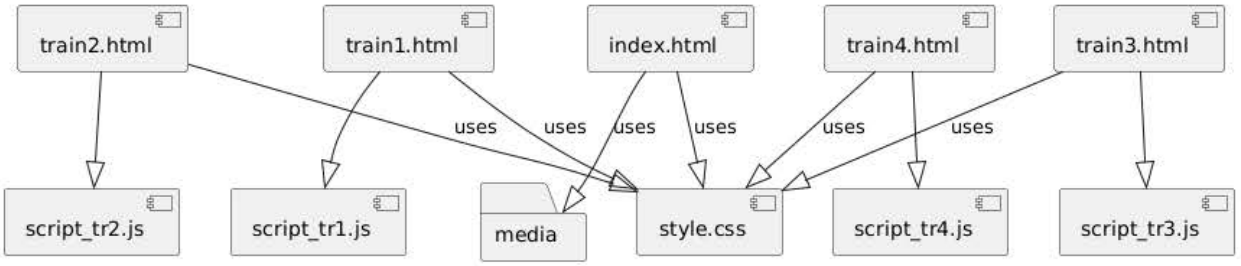

Рисунок 2.1 – Діаграма компонентів

За даною діаграмою можна зрозуміти, що проект налічує 5 основний сторінок: головну та 4 сторінки з тренажерами. Всі сторінки підключені до одного файлу зі стилями, що регулює їх візуальне оформлення. Сторінки тренажерів, також мають однаковий дизайн, але трохи різний функціонал. Саме тому кожна з них використовує окремий скрип, де описаний основний функціонал: генерація завдань, взаємодія з користувачем, обробник подій та надання зворотного зв'язку. Також одним з елементів, що використовується є папка media, яка містить зображення, які використовуються у проекті.

Наступним важливим елементом будь-якого вебсайту є добре продумана система навігації, що дозволяє користувачу на інтуїтивному рівні без зайвих перешкод користуватися сайтом. Але при розробці слід уникати тикий ряд помилок, а саме заплутаність меню, наявність великої кількості сторінок без чіткого взаємозв'язку, хаотичність структури подання даних, що ускладнює пошук певної інформації. Для запобіганню цих помилок заздалегідь розробляється карта навігації сайту, на якій зазначаються всі сторінки або розділи сайту, порядок та структура даних, що вони містять та можливості взаємодії між ними.

Як визначено раніше вебсайт має 5 сторінок. Основна взаємодія користувача відбувається через головну сторінку, на якій представлено 4 різні варіанти тренування навичок усного рахунку із зазначених тем. При виборі потрібної теми, користувач переходить на саму сторінку тренажера, де виконує поставлені завдання за певний проміжок часу. Після завершення виконання користувач отримує результат проходження, а саме набрану

кількість балів та має можливість повернутися на головний екран. Детально з картою навігації можна ознайомитися на рис. 2.2.

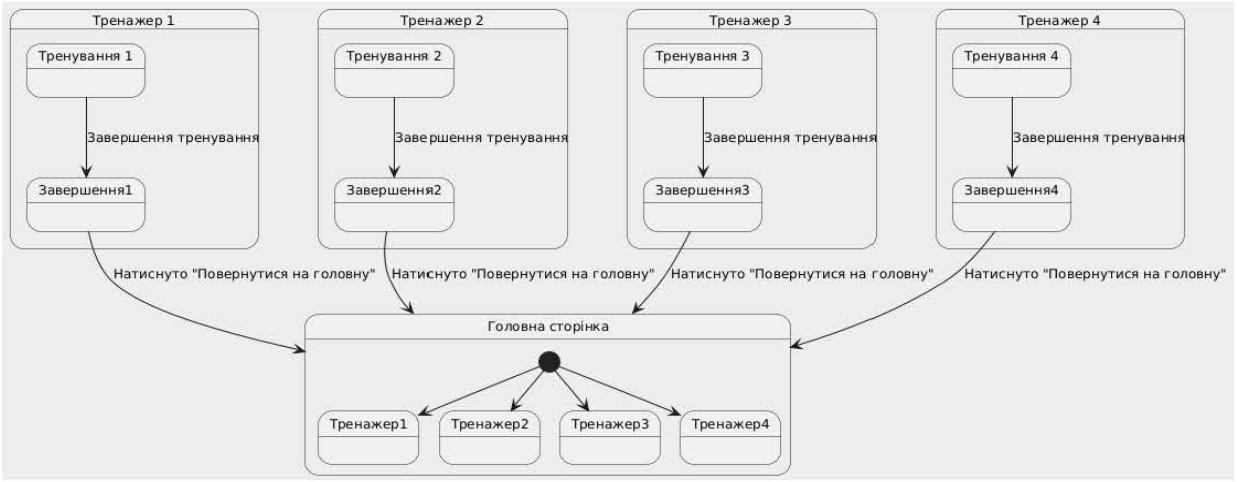

Рисунок 2.2 – Карта навігації

Після визначення основних елементів проекту та їх взаємодії. Можемо зазначити основні можливості платформи: функціональні та нефункціональні. Функціональні вимоги визначають основні аспекти взаємодії користувача із системою. До функціональних вимог проекту можна віднести:

– розробка алгоритму генерації математичних завдань, що відповідає поставленим умовам, щодо вигляду та складності завдання;

– відображення завдань користувачу;

– реалізація механізму введення відповідей, можливість використовувати для цього відповідні кнопки на екрані або клавіатуру пристрою;

– реалізація алгоритму перевірки відповідей та надання відповідного зворотного зв'язку. При правильній відповіді нараховуються бали та з'являється нове завдання, при неправильній – система зупиняє час та показує правильний результат;

Ці основні функціональні вимоги мають бути зазначені у проекті. Також можна визначити функціональні вимоги системи з точки зору користувача, тобто дії, які може виконувати користувач у системі. Для візуалізації цих

функцій використовується Use Case Diagram (Діаграма варіантів використання), яка показана на рис 2.3.

Рисунок 2.3 – Діаграма варіантів використання

На діаграмі можна побачити, Актора що є Користувачем, який взаємодіє з вебсайтом математичного тренажеру та варіанти використання самої платформи, на діаграмі у вигляді еліпса, описують конкретні дії, а саме: обрати один із запропонованих тренажерів; розпочати тренування; виконати завдання, що включає в себе введення та перевірку відповіді; завершення тренування; отримання результату; можливість повернутися на головну сторінку.

Отже, діаграма сприяє розумінню функціональності проекту з точки зору користувача та грає важливу роль при проектуванні розробки.

Також перед початком розробки можемо визначити нефункціональні вимоги до проекту. Нефункціональні вимоги описують загальні властивості програмного продукту, що визначають якість його функціонування, а також

обмеження, пов'язані з продуктивністю, сумісністю, безпекою, зручністю використання та іншими параметрами. Вони не описують конкретні функції, але мають вирішальне значення для користувацького досвіду та подальшої підтримки системи.

До нефункціональних вимог можна віднести:

– продуктивність, швидка реакція на дії користувача;

– кросплатформеність, можливість використовувати застосунок в будь-якому браузері та бути сумісною з різними операційними системами;

– інтерфейс користувача має бути інтуїтивно зрозумілий, навігація має бути простою та чіткою;

– вебсайт має містити можливість масштабованості, що дозволяє розширювати функціонал;

Отже, ці вимоги можуть гарантувати, що вебсайт математичного тренажеру для тренування навичок усного рахунку буде не лише функціональним, а й зручним та надійним у використанні.

2.4 Проектування інтерфейсу користувача

Важливим елементом розробки вебсайту математичного тренажеру є якісний інтерфейс користувача. Зручний та інтуїтивно зрозумілий інтерфейс забезпечує ефективну взаємодію користувача з математичним тренажером. Це також значно впливає на загальне враження користувача від вебзастосунку та полегшує виконання завдань.

Перед проектуванням інтерфейсу варто визначити основні вимоги до нього, а саме:

– простота використання: інтерфейс має бути інтуїтивно зрозумілий для користувача, не містити зайвих елементів, що ускладнюють використання;

– ефективність: за допомогою основних елементів інтерфейсу користувач має змогу швидко виконувати поставленні завдання;

– зрозуміла навігація: можливість легкого переміщення між сторінками сайту;

– взаємодія з користувачем: інтерфейс має забезпечити зворотній зв'язок на дії користувача;

– естетичність: візуальний дизайн має бути приємний та привабливий;

– адаптивність: інтерфейс має бути адаптивний до використання на будь-якому пристрої, не втрачаючи функціональності та візуального вигляду.

Наступним етапом є вибір стилю та візуальних елементів інтерфейсу. Важливими елементами вебдизайну є правильно обрані кольори та шрифти, оскільки вони впливають на сприйняття та взаємодію користувача з вебсайтом, а також додають унікальності, що дозволяє ідентифікувати його серед інших.

Важливо розуміти, що обрані кольори можуть по різному впливати на користувачів. Обираючи кольори для використання в освітній сфері варто зазначити, що краще обирати функціональні кольори, оскільки вони впливають на мозок різними способами, впливаючи на концентрацію, запам'ятовування та емоційну реакцію. Дослідження в галузі педагогічної психології показують, що певні кольори можуть покращити навчання, стимулюючи когнітивні процеси.

Наприклад, червоний колір символізує любов і пристрасть, але в освіті він вказує на важливість і діє як візуальний стимулятор. Синій символізує ясність і спокій, але в освіті він позначає точність і швидкість. Жовтий, колір сонця, символізує радість, а в освіті він покращує зорове сприйняття. Зелений, колір природи, символізує життєву силу та активність в освітньому контексті [11].

Основними кольорами розробки обрано синій та блакитний. Оскільки саме ці кольори, найчастіше асоціюються з математикою. Синій – це спокійний, інтелектуальний колір, він узгоджується з потребою математиків у критичному мисленні та навичках вирішення проблем [13].

Використання саме таких кольорів може зменшити напругу та допомогти краще сконцентруватися на завданнях. Спокійні кольори створюють більш сприятливу та підтримуючу атмосферу для навчання.

У сучасному світі, який наповнений великою кількістю платформ, дуже важко зачепити та запам'ятатися користувачеві. Формування впізнаваності, довіри та лояльності до користувачів відіграє вирішальну роль в розробці будь-якого проекту. Влучна та легка для запам'ятовування назва є першим кроком для встановлення комунікації з цільовою аудиторією та є основою для майбутнього сприйняття бренду.

Не менш важливим є логотип, що слугує візуальним якорем та миттєво ідентифікує продукт серед безлічі інших, формуючи його унікальний образ та емоційний зв'язок з користувачем.

Оскільки назва має влучно ідентифікувати продукт, то для вебсайту математичного тренажеру для вдосконалення навичок усного рахунку, було обрану назву «MathGym».

Така назва не лише вказує на предметну область розробки – математику, а й переосмислює процес навчання як систематичне тренування. Вивчення математики є потужним інструментом для тренування мозку, тобто вирішення математичних завдань настільки ж важливо для мозку, як силові тренування для тіла.

Так, само як регулярні тренування призводять до помітних фізичних змін: збільшення сили, покращення тонусу м'язів та загальної фізичної форми. Аналогічно до цього, розв'язування математичних вправ є своєрідним тренуванням для мозку. Кожна нова задача, кожна складніша концепція – це когнітивне навантаження, яке змушує нейронні зв'язки в мозку зміцнюватися та утворювати нові. Процес пошуку розв'язку активізує різні ділянки мозку, відповідальні за логічне мислення, аналіз, абстрагування, пам'ять та увагу. Ці навички будуть корисні не лише у вивченні математики, а й в повсякденному житті.

Для візуалізації обраної концепції було вирішено використовувати простий та зрозумілий логотип, який у собі буде відображати основну мету даної розробки (рис. 2.4.).

Рисунок 2.4 – Логотип платформи

Обраний малюнок для логотипу «MathGym» надзвичайно добре візуалізує обрано концепцію математичного тренажеру. Він одразу передає ідею що математичні навички потребують таких самих тренувань, як і м'язи у спортзалі.

Використання зображення мозку вказує, що поданий логотип використовується в інтелектуальній сфері та наголошує на тому, що створюваний проект – є інструментом для тренування розуму.

Зображення штанги символізує зусилля, наполегливість та прогрес, які необхідні для досягнення результату як в спорті так і в математиці. Поєднання мозку та штанги створює потужну метафору, яка легко запам'ятовується та викликає правильні асоціації. Вона підкреслює, що математика – це не лише сухі формули, а й активний процес розвитку когнітивних навичок.

Після визначення основних вимог можна перейти до проектування самого інтерфейсу. Як було зазначено в минулих розділах вебсайт міститиме 5 сторінок: головна та 4 сторінки з тренажерами.

На головній сторінці представлені всі створені тренажери, у вигляді карток, розміщених по центру екрану із зазначенням назви кожної теми, як показано на рис. 2.5.

Основним елементом сторінок-тренажерів є робоча область, яка представлена у вигляді калькулятора (рис. 2.6.). Робоча область зверху містить основну інформацію, таку, як рівень, кількість набраних балів, прогрес бар для візуального сприйняття балів та таймер зі зворотним відліком.

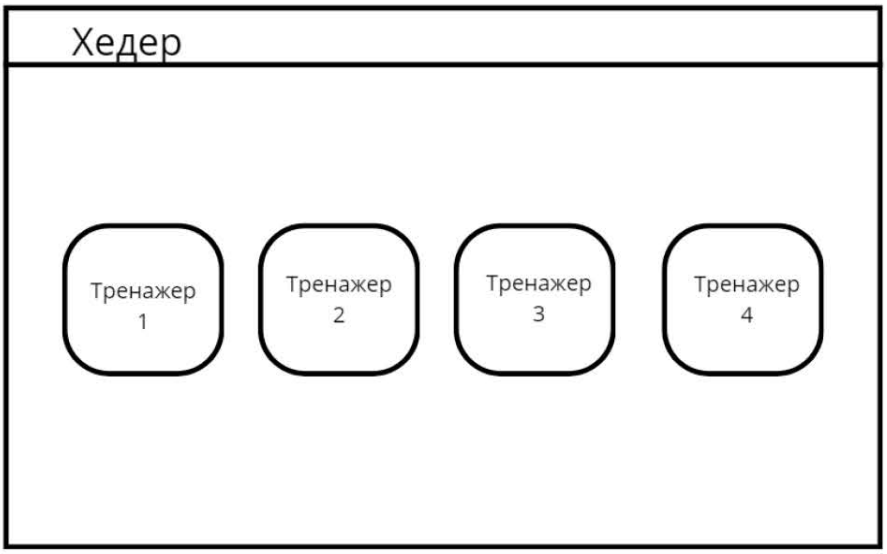

Рисунок 2.5 – Ескіз головної сторінки

Наступним елементом є поле з завданням, де видно згенерований приклад, нижче поле для введення відповіді. Останнім елементом є безпосередньо цифрові кнопки для введення відповіді. Серед кнопок представлені безпосередньо цифри, знак «.», що використовується при введенні у відповідь десяткового дробу, кнопка «С», яка виділяється червоним та надає можливість видалення останнього введеного елементу і кнопка «ОК», що виділена зеленим і підтверджує введену відповідь, відправляючи її на перевірку.

Рисунок 2.6 – Ескіз сторінки-тренажера

Візуальний дизайн відіграє ключову роль у формуванні першого враження користувача та впливає на його подальшу взаємодію з продуктом, вирішуючи чи затримається користувач на платформі. Враховуючи вище зазначений вибір кольорової гами, назви та логотипу, ескізів розміщення елементів інтерфейсу, створення готового дизайну інтерфейсу користувача – це завершальний етап формування інтуїтивно зрозумілого, привабливого та функціонального інтерфейсу, що відображає основну концепцію розробки.

Основною метою є не просто створення естетичного та привабливого інтерфейсу, а розробка інтуїтивно зрозумілого та функціонального середовища, що сприятиме ефективному процесу навчання математики, роблячи його більш різноманітним та цікавим.

Варто зазначити, що готовий дизайн має бути лаконічним, щоб не відволікати користувача від основної мети відвідування платформи, а саме вдосконалення навичок усного рахунку, та водночас бути достатньо динамічним та візуально привабливим, щоб підтримувати зацікавленість користувачів.

Обрана кольорова схема поєднує три відтінки синього кольору, які асоціюються з інтелектуальною діяльністю та концентрацією. Ці кольори було обрано спираючись на раніше проведений аналіз психологічного впливу кольору на навчальний процес. Синій колір створює відчуття спокою та зосередженості, що налаштовує на інтелектуальну роботу.

Візитівка платформи, а саме її логотип, що візуально втілює концепцію «тренування мозку», інтегровано у хедер кожної сторінки сайту, де займає помітне, але не нав'язливе місце.

Структура головної сторінки розроблена на етапі створення макетів без змін була перенесена у фінальний дизайн, оскільки підтвердила свою зручність та зрозумілість для користувача (рис 2.7.)

Обов'язковим елементом сторінки є хедер, що містить логотип та назву платформи, а також пункт меню «Тренажери», за допомогою якого можна буде повернутися на головну сторінку, де розміщений перелік математичних

тренажерів. Кольором для хедера було обрано синій, основним фоном сторінки є блакитний фон з білими елементами, що позначають математичні операції, щоб урізноманітнити вільний простір. Сам перелік можливих тренажерів представлений у вигляді карток, такого ж синього кольору, як і хедер. Таке розміщення є простим та зрозумілим для користувача, він може одразу побачити можливі теми для тренування і обрати відповідну до свого запиту, кожна картка містить назву теми та короткий опис завдань. Весь текст сторінки представлений білим кольором для підвищення читабельності, оскільки білий є достатньо контрастним для відтінків синього, що використовуються.

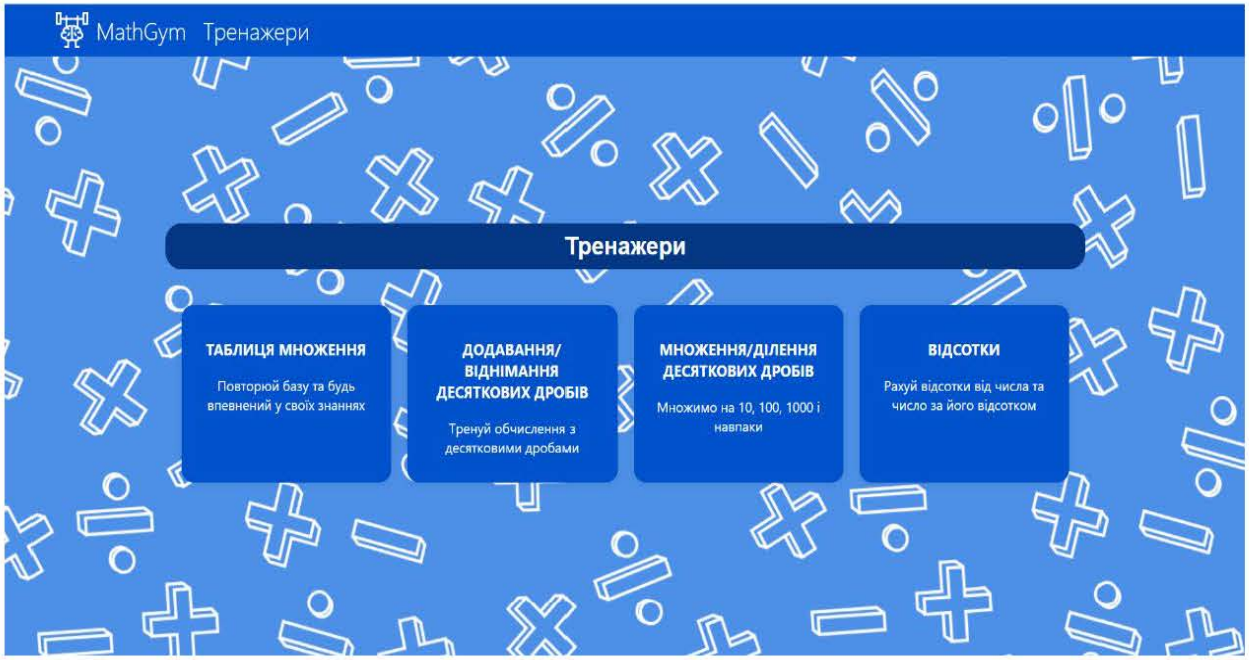

Рисунок 2.7 – Інтерфейс головної сторінки

Сторінки тренажерів мають однаковий дизайн. Оскільки формат тренажерів однаковий, користувач отримує завдання та має вводити відповідь, щоб спростити введення відповіді сам тренажер представлений у вигляді так званого калькулятора (рис 2.8.).

Як і на головній сторінці, сторінка тренажеру містить хедер з логотипом та назвою і пунктом меню «Тренажери», що повертає користувача на головну сторінку.

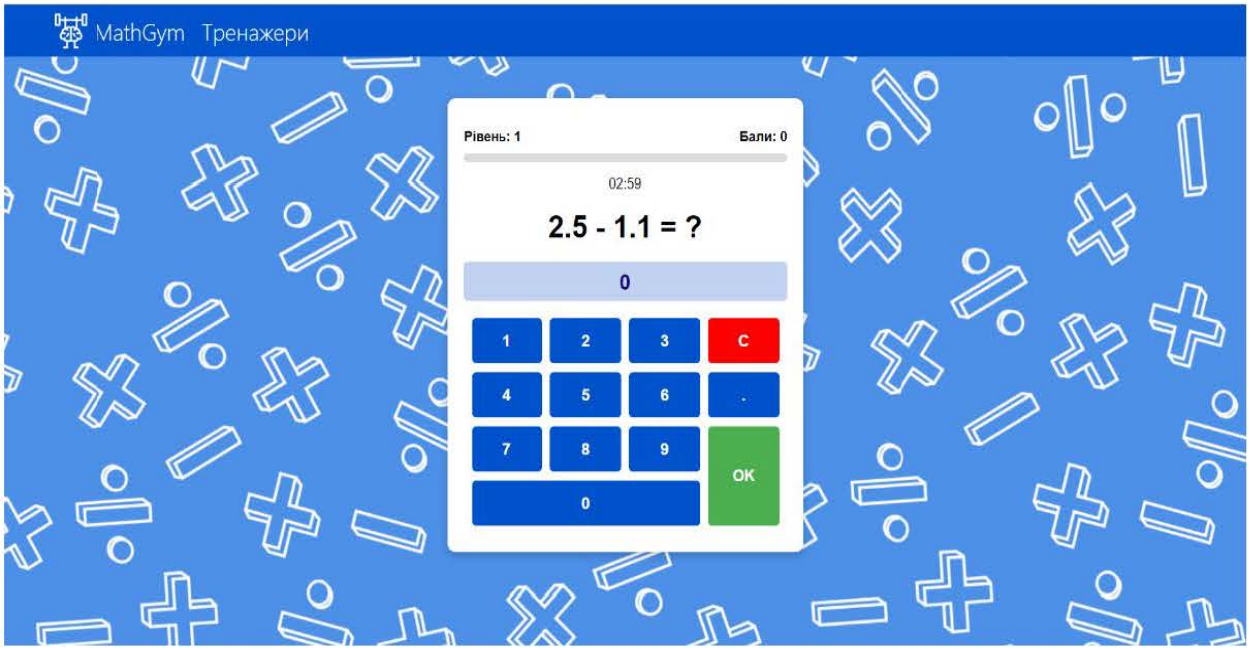

Рисунок 2.8 – Інтерфейс сторінки-тренажеру

Основним елементом сторінки є контейнер-калькулятор. Зверху користувач може ознайомитися з основною інформацією, а саме: на якому рівні знаходиться та скільки балів отримав за правильні відповіді, для більшої візуалізації отриманих балів нижче знаходиться прогрес-бар, який заповнюється поступово з кожною правильною відповіддю.

Наступним елементом є таймер, який відображає час, який залишився у користувача на виконання завдання, що спонукає тренуванню швидкості. Нижче розташоване поле на якому з'являється приклад або завдання, при введені відповіді завдання змінюється на інше, завдання виділено чорним кольором на білому фоні для забезпечення максимальної видимості. Далі є поле для введення відповіді, що виділено блакитним кольором.

Нижня частина контейнеру – це сама клавіатура для введення відповіді. Вона містить числа від 0 до 9 розташовані, як на калькуляторі, що спрощує використання та робить її більш знайомою для користувача. Також клавіатура містить кнопку «.» з крапкою, яку використовують для запису десяткових дробів у відповіді. Кнопку «С», що виділена червоний та виконує функцію очистки останнього введеного елементу. Та кнопка «ОК», виділена зеленим,

для підтвердження введеної відповіді, яка розташована у нижньому правому куті, візуально відокремлює її від інших, що полегшує її знаходження.

Отже, білий контейнер чітко виділяє основне завдання та елементи для його виконання. Розташування цифрових клавіш та кнопок «С» та «ОК» є зручним та інтуїтивно зрозумілим для користувача. Зелений колір кнопки «ОК» та червоний – «С» слугують візуальною підказкою, зелений асоціюється з правильним вибором, а червоний з очищенням або скасуванням.

Розміщення рівня та балів у верхній частині контейнеру, є стандартним для ігрових та навчальних інтерфейсів, що дозволяє користувачеві легко відстежувати свій прогрес. Розташування таймера безпосередньо над полем із завданням тримає час у полі зору користувача під час виконання завдання.

Також основною вимогою для створення платформи була можливість використання на різних девайсах таких, як: комп'ютер, смартфон, планшет. Таким чином створений дизайн має бути адаптивним до будь-якого пристрою (рис 2.9.).

У адаптованому дизайні на головній сторінці блоки з тренажерами розміщуються вертикально у стовпчик. На сторінці самого тренажеру весь екран займає основний контейнер із завданням та інструментами для його виконання. Такий дизайн є зручним при використанні екранів маленького розміру. Елементи залишаються комфортного для користувача розміру, що дозволяє ефективно та зручно використовувати вебсайт на мобільному пристрої. Незважаючи на зміну розташування та розмірів деяких елементів функціональність вебсайту залишається незмінним. Використовуючи картки з назвами тренажерів на головній сторінці можна перейти до тренування. При безпосередньому виконанні завдань на мобільному пристрої для зручності введення відповіді, використовується кнопки на екрані, що містять: цифри, знак «.» для введення десяткових дробів, кнопка «С» для видалення останнього введеного елементу та кнопку «ОК» для перевірки відповіді.

Це дозволяє зручно використовувати вебсайт на будь-якому пристрої при різному розширенні екрану.

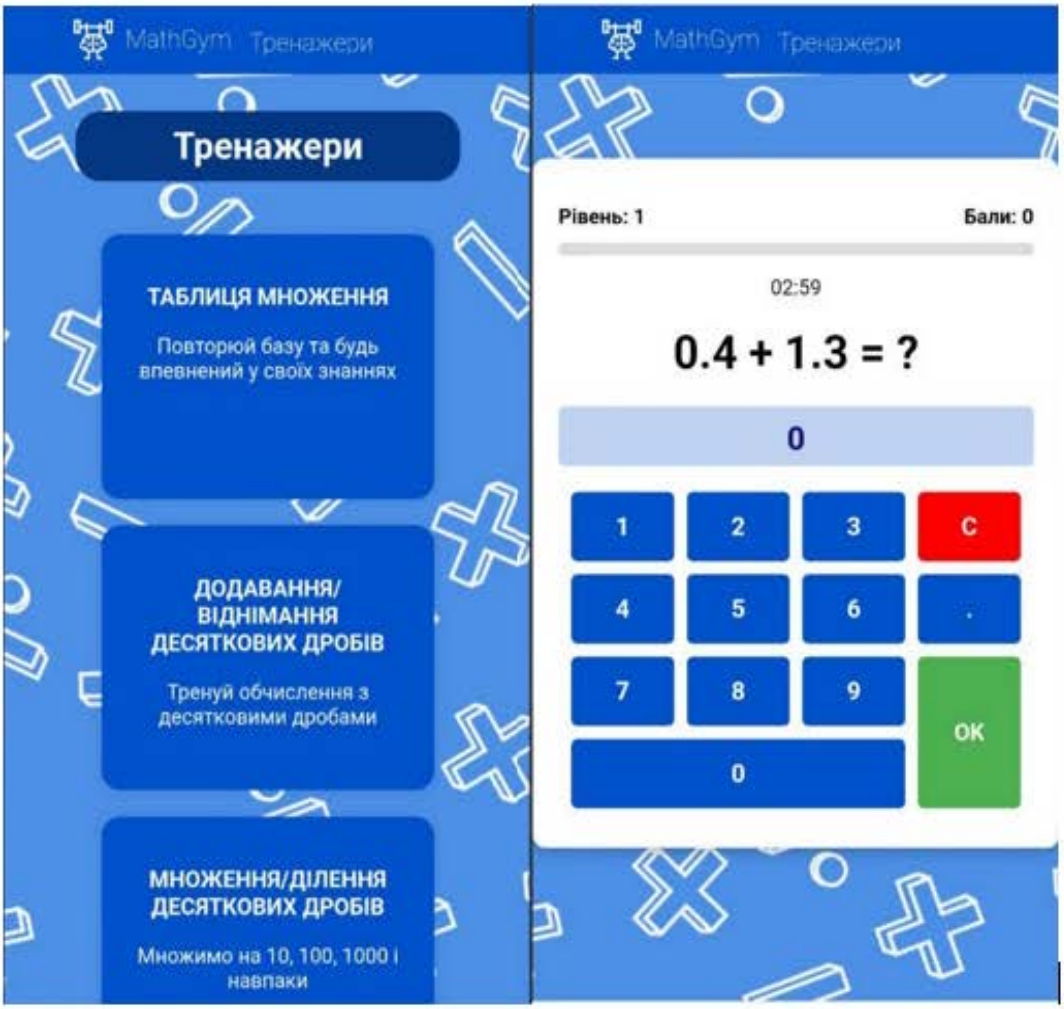

Рисунок 2.9 – Адаптація дизайну на смартфоні

Отже, готовий дизайн має продуману функціональність та візуальну привабливість, що спрямоване на створення ефективного та комфортного навчання. Дизайн створює позитивне враження, поєднуючи в собі естетичну привабливість та високу функціональність. Він сприяє підвищенню ефективності навчання, не відволікає від основного контенту та підтримує мотивацію користувачів завдяки чіткому відображенню прогресу та інтуїтивно зрозумілій взаємодії. Обрана концепція "тренування мозку" вдало візуалізується через логотип та загальний стиль, роблячи процес вивчення математики більш захоплюючим та цілеспрямованим. Цей підхід забезпечує оптимальну користувацьку взаємодію, що є важливим для використання вебсайту. Кожен елемент дизайну ретельно продуманий та має своє місце, що відповідає основній концепції розробки.

2.5 Висновки до другого розділу

У розділі було проведено аналіз методів вирішення поставленої задачі – розробка математичного тренажеру з усного рахунку. На основі порівняльного аналізу можливих платформ розробки програмного забезпечення – мобільних, десктопний та веб-застосунків, перевагу отримало використання веб-розробки, оскільки вона найбільше відповідає вимогам, щодо створення даної інтерактивної платформи.

Оскільки для реалізація проекту було обрано веб-розробки з використанням основних веб-технологій, а саме HTML для структурування контенту, CSS для його стилізації та JavaScript для забезпечення інтерактивності та основної функціональності на стороні клієнта. Цей набір є універсальним та забезпечує виконання усіх вимог до програми.

Також було проведено етап проектування платформи, під час нього було визначено клієнтську архітектуру. За допомогою діаграми компонентів було визначено структуру проекту, яка складається з головної сторінки та чотирьох сторінок-тренажерів, кожна з яких має свій скрипт для реалізації функціоналу кожного тренажера. Для візуалізації структури переміщення між сторінками платформи було розроблено карту навігації, яка показує можливі шляхи переходи між сторінками.

У розділі проектування були визначені функціональні можливості платформи, та зазначено функції з якими може взаємодіяти користувач, ці дані подані на діаграмі варіантів використання. Також були визначені нефункціональні вимоги, дотримання, яких забезпечить якісний користувацький досвід.

Було проведено проектування інтерфейсу користувача. Сформовано основні вимоги до інтерфейсу, спираючись на зручність та інтуїтивність використання, а також на відповідність потребам користувача.

Перед безпосередньою розробкою дизайну інтерфейсу було створено основні макети сторінок, на яких візуалізовано розміщення основних

елементів та логіку їх взаємодії між собою, що забезпечувало чітке розуміння структури створюваної системи.

У процесі проектування було враховано можливість масштабування та адаптивності інтерфейсу для різних типів пристроїв, що забезпечує використання, як на комп'ютерах так і на мобільних пристроях. Увагу було приділено питанням доступності, зокрема забезпеченню достатньої контрастності елементів, читабельності шрифтів та простоти навігації.

Проведено вибір та аналіз кольорової гами, спираючись на психологічні та візуальні аспекти сприйняття кольору. Обрана кольорова гама в синіх кольорах підкреслює мети створюваної платформи, сприяє концентрації уваги та створює приємну візуальну картинку.

Одними з основних елементів впізнаваності є назва та логотип, які створювалися безпосередньо для платформи відображаючи її основних зміст, як простір для тренування навичок усного рахунку, що є важливим для розвитку навичок критичного мислення.

При проектуванні враховувалися потреби цільової аудиторії, яка потребує простого, зручного, зрозумілого та легкого у використанні інтерфейсу.

Ці етапи призвели до заключного етапу проектування, а саме створення реалізація дизайну інтерфейсу, що поєднав у собі естетичність, функціональність та зрозумілість для користувача, забезпечуючи комфортне та ефективне використання веб-застосунку.

РОЗДІЛ 3. РОЗРОБКА ВЕБСАЙТУ ДЛЯ ТРЕНУВАННЯ МАТЕМАТИЧНИХ НАВИЧОК

3.1 Розробка програмного забезпечення платформи

Розробка програмного забезпечення математичного тренажеру "MathGym" являла собою комплексний процес, що включав створення клієнтської частини веб-додатку. Ключовими аспектами цього етапу були реалізація динамічної функціональності тренажеру за допомогою JavaScript, забезпечення структурованої та семантичної верстки сторінок з використанням HTML, а також створення візуально привабливого та інтуїтивно зрозумілого інтерфейсу засобами CSS.

Основний функціонал математичного тренажеру «MathGym» був розроблений з використанням чистого JavaScript, без використання сторонніх бібліотек чи фреймворків, щоб забезпечити максимально швидку взаємодію та мінімізувати залежності. Функціональні можливості реалізовані у вигляді окремих модулів, кожен з яких відповідає за певну частину функціоналу.

Кожна сторінка-тренажер має однаковий функціонал та складається з таких модулів:

– керування часом, відповідає за відлік часу;

– генерація завдань, відповідає за створення нових математичних завдань;

– введення відповіді користувача, обробляє введення відповіді користувачем за допомогою кнопок на екрані або клавіатури;

– перевірка відповіді та керування прогресом, перевіряє введену відповідь, оновляє рахунок та відображає результат, керує прогрес-баром та рівнем складності;

– ініціалізація, відповідає за запуск основних процесів при завантаженні сторінки.

Усі сторінки-тренажери складаються з однакових модулів, окрім модуля «Генерація завдань», що залежить від обраної теми тренажера.

Розберемо детальніше кожний модуль.

Основне призначення модуля «Керування часом» полягає в забезпеченні контролю за часом, що відведений на виконання завдань користувачем у кожному математичному тренажері. Це модуль відповідає за такі функції:

– ініціалізація таймеру, встановлення початкового значення часу на таймері;

– відлік часу, поступове зменшення значення таймеру;

– візуалізація таймера, оновлення відображення часу на екрані;

– обробка закінчення часу, визначення моменту завершення часу на таймері та початок виконання наступних зазначених дій, а саме вивід результату;

– реалізація паузи таймер, тимчасове призупинення відліку часу, у разі надання неправильної відповіді.

Для реалізації зазначеного функціоналу використовується декілька основних функцій: startTimer(), showFinalResult(), pauseTimer().

Функція startTimer() є точкою входу для початку відліку часу, використовуючи посилання на DOM-елемент timer, який використовується для відображення часу на екрані. Наступним кроком є визначення інтервалу зменшення часу (1 секунда) та подальше зменшення часу на задану одиницю, коли результат перевірки стану паузи є негативним (false).

Обов'язковим елементом даної функції є форматування часу, тобто виконується обчислення хвилин та секунд, що залишилися. Формат часу представлений для обох значень двома цифрами (наприклад «02:44»), для більш естетичного вигляду.

Останнім елементом зазначеної функції startTimer() є перевірка закінчення часу, тобто в кожній ітерації перевіряється значення змінної, яка відповідає за наявну кількість секунд, чи стала вона дорівнювати 0. Якщо це так, зупиняється процес відліку часу та запускається наступна функція

showFinalResult(), що в свою чергу показує результат користувача за проходження завдань даного тренажеру.

Функція showFinalResult() отримує значення набраних балів з DOM-елемента score та відображає прихований блок з кінцевими результатами.

Функція pauseTimer() змінює статус змінної, що відповідає за зупинку відліку часу та припиняє відлік на заданий інтервал. При призупиненні відліку основного таймера, запускається таймер, який рахує заданий інтервал та після закінчення цього інтервалу відновлює відлік часу у інтервалі функції startTimer().

Отже, модуль «Керування часом» є основним для створення обмеження часу при тренуванні, додаючи елемент виклику та необхідність швидко приймати рішення і виконувати обчислення.

Основною метою модуля "Генерація задач" є автоматизоване створення різноманітних математичних завдань з різних тем, а саме: таблиця множення, додавання/віднімання десяткових дробів, множення десяткового дробу на 10 і тд., знаходження відсотку від числа та числа за його відсотком.

Генерація завдань у математичному тренажері, що тренує знання таблиці множення, тобто множення і ділення цілих чисел складається з вибору типу операції, випадковим чином визначається чи буде згенерований приклад на множення чи на ділення, ймовірність створення однакова.

Для формування прикладу на множення генеруються два випадкових числа в зазначеному діапазоні, що залежить від рівня складності, на початкових рівнях такі числа, як від 1 до 5, на наступних діапазон розширюється. На основі обраних чисел формується сам приклад, що створюється у вигляді рядка (наприклад: «7*2=?»). Правильна відповідь обчислюється, як добуток заданих чисел.

Для створення прикладу на ділення генерується два випадкових цілих числа, одне з яких є дільником, а інше часткою. Щоб гарантовано згенерувати приклад на ділення без залишку, ділене обчислюється як добуток, попередньо

згенерованих двох чисел. Наступним етапом одне з раніше згенерованих чисел обирається як дільник, відповідно інше, як частка.

На основі обчислених даних формується приклад, що має вигляд «48/8=?». Правильна відповідь – це одне з початково згенерованих чисел, яке не було обрано у якості дільника, що гарантує цілий результат.

Таким чином, даний процес забезпечує створення коректних та зрозумілих прикладів на ділення та множення, що забезпечить якісне тренування обраної теми, а обраний алгоритм генерації дозволить використовувати тренажер постійно, оскільки забезпечується створення різноманітних прикладів, що будуть відрізнятися при кожному запуску, та адаптуватися під прогрес учня.

Для тренажеру на тренування навичок додавання та віднімання десяткових дробів також використовується розподілення по рівнях. За значенням змінної, що відповідає за рівень складності визначається максимальні значення для цілої частини десяткового дробу та кількості знаків після коми. Наприклад, для першого рівня максимальне число для цілої частини є 3, а кількість знаків після коми дорівнює 1. За визначеними даними в подальшому генерується два числа, між якими буде відбуватися випадково обрана операція (додавання чи віднімання), операція вибирається зі створеного масиву.

Створений приклад з'являється на екрані користувача у вигляді рядка. У прикладах на додавання числа розміщуються в довільному порядку, в той час як при відніманні важливо, щоб більше число було першим, займало місце зменшуваного, для того щоб у результаті було отримано додатне значення.

Обчислення правильної відповіді виконується в залежності від обраної дії, результат округляється до максимального значення знаків після коми. Після надання та перевірки відповіді завдання оновлюється.

Тренажер, що створений для тренування навичок множення або ділення десяткового дробу на числа: 10, 100, 1000, 0.1, 0.01, 0.001 і тд. Використовує випадково згенерований десятковий дріб, із відповідною кількістю знаків

після коми, що зазначається рівнем складності та фіксований множник чи дільник, які знаходяться в створеному масиві з якого випадково обираються.

В цьому процесі також використовується функція для генерування десяткових дробів із зазначенням параметрів складності, генерується випадкова ціла частина та дробова.

Формат прикладу такий самий як і в минулих тренажерах, тобто це рядок із зазначеним завданням на ділення або множення (наприклад: «4.25*0.1=?»).

Таким чином, створюються математичні завдання на множення або ділення десяткових дробів на фіксовані значення, забезпечуючи різноманітність та адаптацію складності до рівня користувача.

Останнім видом тренажера, що є на момент створення платформи – є тренування знання про відсотки, а саме знаходження відсотка від числа та числа за його відсотком.

Перед початком генерації самого прикладу випадковим чином обирається тип задачі: знаходження відсотка від числа або числа за його відсотком. В залежності від рівня складності встановлюються параметри для генерації задач: визначається діапазон чисел, можливі відсотки та складність обчислення. Тобто на початкових рівнях користувач буде мати менші числа та набір простіших відсотків (наприклад: 10%, 20%, 50%), на вищих рівнях відсотки ускладнюються (наприклад: 15%, 35% і тд.). Складність обчислень відображається в тому, що на початкових рівнях числа та відсотки генеруються таким чином, щоб у відповіді виходило ціле число, в подальших рівнях це може бути і дробове значення.

Завдання формується у вигляді питання (наприклад, "Знайти [вибраний відсоток]% від числа [випадкове число]") для завдань із знаходження відсотка від числа. Правильна відповідь буде обчислюватися за формулою: (базове число * вибраний відсоток) / 100.

Для завдань на знаходження числа за його відсотком випадково генерується значення, яке відповідає певному відсотку від невідомого числа. Тест задачі матиме вигляд: "Знайти число, якщо [вибраний відсоток]% від

нього дорівнює [згенероване значення]", а правильна відповідь обчислюється за формулою: (згенероване значення * 100) / вибраний відсоток.

Таким чином, даних тренажер дозволяє урізноманітнити завдання на відпрацювання теми відсотки, охоплюючи основні типи задач на відсотки.

Модуль "Введення відповіді користувачем" відповідає за забезпечення інтерактивності користувача з тренажером у процесі введення відповіді на запропоновану математичну задачу. Він обробляє різні способи введення даних та відображає їх у відповідному полі.

На створюваній платформі існує два види введення відповіді: використання кнопок на екрані або застосування клавіатури комп'ютера чи ноутбуку. Коли користувач натискає на кнопки, що представляють цифри від 0 до 9 або десяткову крапку, оновлюється вміст елемента, що містить значення введеної відповіді та виводить її на екран, так само відбувається при натисканні кнопок на клавіатури ПК. При використанні кнопки «С» - очистити на екрані, або Backspace відповідь можна стерти. Натискання кнопки «ОК» або Enter приводить до перевірки відповіді, наступного модуля.

Використання цих двох методів введення відповіді користувача забезпечує легке та ефективне користування тренажером на будь -якому пристрої.

Модуль «Перевірка відповіді та керування прогресом» відповідає за логіку перевірки введеної користувачем відповіді на поточну задачу, оновлення стану гри (рахунок, прогрес, рівень) та генерацію нової задачі.

Цей модуль починає роботу після підтвердження користувачем своєї відповіді, тобто натискання кнопки «ОК» або Enter. Отримане значення порівнюється з правильною, обчисленим програмою відповіддю.

Якщо відповідь правильна користувач отримує 5 балів, що додаються до його поточного рахунку, відбувається оновлення відображеного рахунку та оновлюється прогрес-бар, на екрані з'являється наступний згенерований приклад. При наборі 30 балів, рівень складності для користувача збільшується.

Якщо відповідь неправильна відображається повідомлення про помилку на екрані та правильна відповідь, цей текст виділено червоним для привертання уваги. Для того, щоб користувач мав можливість ознайомитися з правильною відповіддю викликається функції з модуля керування часом, а саме pauseTimer(), що зупиняє таймер на 5 секунд. Після закінчення паузи таймера очищується повідомлення про результат та генерується нове завдання.

Даний модуль, дозволяє користувачеві відслідковувати свій прогрес у зручному форматі та надає можливість проаналізувати свої результати.

Основний та найперший у виконанні є модуль "Ініціалізація", що відповідає за виконання початкових дій при завантаженні веб-сторінки, щоб підготувати гру до початку.

При запуску сторінки викликаються функції: startTimer() з модуля "Керування часом", для запуску відліку часу; generateNewProblem() модуля "Генерація задач" для створення та відображення першої математичної задачі; ініціалізує відображення рахунку на екрані, встановлюючи textContent DOM-елемента scoreDisplay на початкове значення score, тобто 0; Ініціалізує відображення прогрес-бара, встановлюючи його початкову ширину (progressBar.style.width) на "0%".

Ці модулі разом забезпечують основну функціональність математичного тренажера, від генерації завдань до обробки відповідей користувача та відображення прогресу гри. Кожен модуль відповідає за свою чітко визначену частину логіки, що полегшує розуміння та підтримку коду.

Метою стилізації інтерфейсу даного математичного тренажера засобами CSS є створення зручного, інтуїтивно зрозумілого та естетично приємного навчального середовища.

Для створення визначеного та створеного в попередньому підрозділі дизайну використовувалися техніки та підходи для створення адаптивного та візуально привабливого веб-додатку, включаючи адаптивний дизайн, гнучку розмітку та кастомне оформлення елементів.

Однією з найважливіших вимог до дизайну була його адаптивність під будь-який пристрій. При розробці для цього використовувалися медіа-запити. Медіа-запити в CSS є потужним інструментом для реалізації цієї концепції, дозволяючи застосовувати різні набори стилів залежно від характеристик пристрою, на якому відображається сторінка.

Медіа-запити дозволяють застосовувати певні стилі лише тоді, коли максимальна ширина пристрою дорівнює параметру медіа-запиту. Наприклад, @media (max-width: 480px), застосовує певні стилі лише тоді, коли максимальна ширина екрана пристрою становить 480 пікселів або менше. Це типовий брейкпойнт, який часто використовується для стилізації під невеликі мобільні телефони.

Також, у файлі стилів CSS присутні додаткові аспекти адаптивного дизайну, такі, як використання відносних одиниць (%, em, vh) замість фіксованих пікселів для розмірів шрифтів та елементів, що дозволяє їм краще масштабуватися на різних екранах. Для зображень використовується властивість max-width: 100%, що дозволяє зображенням не перевищувати ширину батьківського контейнеру та залишитися адаптивними.

Ще однією корисною властивістю, що використовується для стилів є Flexbox, що надає можливість будувати гнучкі структури інтерфейсу та керувати вирівнюванням елементів.

Flexbox є потужною системою розмітки в CSS, розробленою для створення одновимірних макетів (в ряд або в стовпець). Вона надає гнучкі можливості для розподілу простору між елементами всередині контейнера та керування їхнім вирівнюванням. Застосування Flexbox значно спрощує створення складних макетів, особливо тих, що повинні бути адаптивними.

Ця властивість використовується для центрування основного контенту головної сторінки, розміщення елементів у хедері, а саме розміщення логотипу, назви та пункту меню «Тренажери».

Для розміщення карток, за якими можна обрати тренажер також використовує властивість flex-wrap: wrap; що дозволяє flex-елементам

(карткам) переноситися на новий рядок, якщо вони не вміщаються по ширині контейнера. Це ключова властивість для створення адаптивних макетів, де кількість карток у рядку може змінюватися залежно від розміру екрана.

Використання Flexbox надає значні преваги у розробці дизайну математичного тренажеру. Дозволяє легко керувати розміщенням та вирівнюванням елементів, особливо коли розмір елементів є динамічним і залежить від розмірів екрану. Властивості flex-wrap та медіа-запити дозволяють створювати макети, що матимуть гарний вигляд на екранах будь-якого розміру. Спрощує багатьох завдань розмітки та надає потужні інструменти для вирівнювання елементів як по горизонталі, так і по вертикалі.

Тобто використання Flexbox значно спрощує процес створення сучасних веб-макетів, зробивши їх більш гнучкими, адаптивними та легкими у керуванні.

3.2 Тестування створеного математичного тренажеру

Етап тестування є важливою частиною процесу розробки програмного забезпечення. Він дозволяє оцінити якість створеного продукту, виявити потенційні помилки та недоліки, та переконатися у відповідності розробленої платформи математичного тренажера поставленим вимогам.

Метою проведення тестування є всебічне вивчення функціональних можливостей тренажера, перевірка коректності його роботи в різних сценаріях використання, оцінка стабільності та продуктивності, а також аналіз зручності інтерфейсу для кінцевого користувача. Для досягнення поставленої мети було застосовано комбінацію різних методологій, включаючи функціональне тестування, тестування зручності використання та тестування адаптивності.

Тестування спрямоване на досягнення наступних цілей:

– виявлення помилок, ідентифікація будь-яких неочікуваних або неправильних поведінок тренажера, що можуть призвести до некоректної роботи або негативного користувацького досвіду;

– перевірка функціональності, підтвердження того, що кожен функціональний модуль тренажера працює згідно зі специфікацією та виконує передбачені завдання (наприклад, коректна генерація математичних задач різних типів і рівнів складності, точна обробка введення відповідей користувачем, правильна перевірка відповідей та нарахування балів, адекватне керування прогресом навчання);

– оцінка зручності використання, аналіз того, наскільки інтуїтивно зрозумілим, легким у навігації та ефективним є інтерфейс тренажера для користувачів з різним рівнем підготовки;

– перевірка адаптивності, підтвердження коректного та адекватного відображення інтерфейсу тренажера на різних типах пристроїв з різними розмірами екранів (комп'ютери, ноутбуки, планшети, смартфони) та в різних орієнтаціях (горизонтальна, вертикальна).

Об'єктами тестування є основні функціональні модулі та ключові аспекти розробленого математичного тренажеру.

Метою тестування модуля «Генерація завдань» є перевірка коректності генерації математичних прикладів різних типів на різних рівнях складності згідно з заданими параметрами.

Для проведення тестування використовувався підхід ручного тестування, оскільки воно полягає у багаторазовій генерації задач для кожного типу операцій та на кожному визначеному рівні складності. Візуальна перевірка згенерованих прикладів є ключовим методом для оцінки їхньої коректності.

Для всебічної перевірки необхідно виконати наступні тестові сценарії:

– для кожного рівня складності перевірити генерацію завдань з кожної теми;

– перевірка діапазону генерованих чисел;

– перевірка коректності форматування тексту задачі;

– перевірка уникнення некоректних прикладів.

Для виконання перевірки генерації кожного типу завдань, було багаторазово запущено процес генерації різних завдань по різних темах, Отримані результати наведено на рис 3.1. За даними результатами можна зазначити, що функція генерування прикладів працює коректно та стабільно.

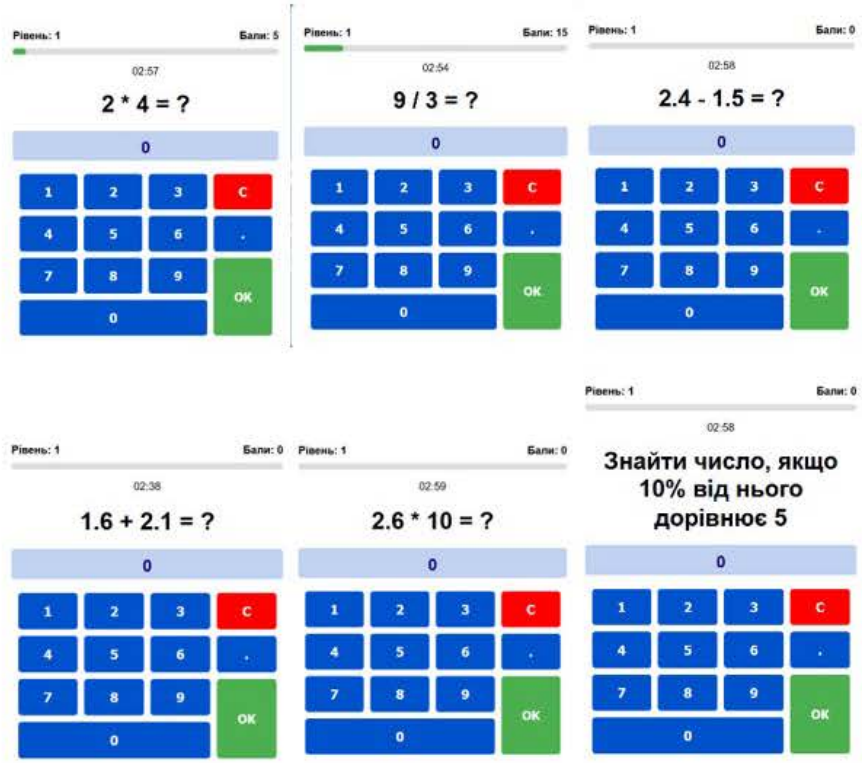

Рисунок 3.1 – Результати генерації завдань

Можна зазначити, що при виконанні багаторазової генерації було перевірено діапазон чисел, що генеруються для кожного завдання та рівня складності.

Формування тексту задач було коректним для всіх типів операцій, математичні символи використовувалися правильно, усі необхідні елементи були присутні.

Генерація задач з цілими та дробовими результатами відбувалася відповідно до очікувань. Наприклад, при додаванні десяткових дробів часто генерувалися приклади з дробовим результатом.

Зазначаючи отримані результати тестування модуля «Генерування задач» проведено успішно, модуль продемонстрував високу якість роботи.

Задачі всіх передбачених типів успішно генеруються на різних рівнях складності, діапазони чисел відповідають встановленим параметрам, а формат тексту задач є коректним.

Тестування модуля «Введення відповіді користувачем». Метою тестування є перевірка коректності та надійність функціональності введення відповідей користувачем за допомогою віртуальних числових кнопок та фізичної клавіатури, а також працездатність кнопки «Очистити» для редагування введеної відповіді.

Для тестування використовувалися такі тестові сценарії:

– перевірка введення відповідей числовими кнопками на екрані;
– введення відповіді за допомогою клавіатури;
– перевірка видалення символів кнопкою «Очистити»;
– перевірка поведінки при спробі ввести декілька крапок.

Після проведення тестування, отримали такі результати: введення цілих чисел за допомогою числових кнопок та клавіатури оброблялося коректно, цифри відображалися у правильній послідовності. При введенні десяткових дробів десяткова крапка відображалася правильно. Повторне введення крапки з клавіатури було заблоковано, але при використанні кнопок на екрані можна ввести декілька крапок, що є багом та потребує виправлення.

Клавіша «Backspace» на клавіатурі успішно видаляла останній введений символ. Натискання клавіші «Enter» ініціювало перевірку відповіді, як і натискання кнопки «ОК». Кнопка «С» коректно видаляла символи по одному при кожному натисканні та повністю очищала поле вводу після багаторазового натискання.

За результатами проведеного тестування можемо визначити, що загалом модуль «Введення відповіді користувачем» працює коректно, незважаючи на незначні помилки, такі як можливість введення декількох точок, використовуючи числові кнопки на екрані.

Метою тестування модулю «Перевірка відповіді та керування прогресом» є перевірка коректності логіки порівняння введеної користувачем

відповіді з правильною, а також функціональність оновлення рахунку, відображення результатів перевірки (правильно/неправильно), динаміку прогрес-бару та механізм зміни рівня складності залежно від набраних балів.

Для тестування використовувалися такі тестові сценарії:

– введення правильної відповіді;

– введення неправильної відповіді;

– перевірка відображення правильної відповіді після помилки;

– перевірка оновлення рівня;

Результати проведення тестування. При введенні правильної відповіді рахунок користувача успішно збільшувався на 5 балів, прогрес-бар відповідно заповнювався. У випадку введення неправильної відповіді рахунок та прогрес-бар залишалися без змін, відображалося повідомлення ‘Неправильно.’ червоним кольором. Після введення неправильної відповіді на екрані відображалася правильна відповідь на поточну задачу, що допомагає користувачеві у навчанні. (рис 3.2.). Можемо бачити, що в завдання на множення і ділення десяткових дробів, правильна відповідь виводиться в некоректному записі.

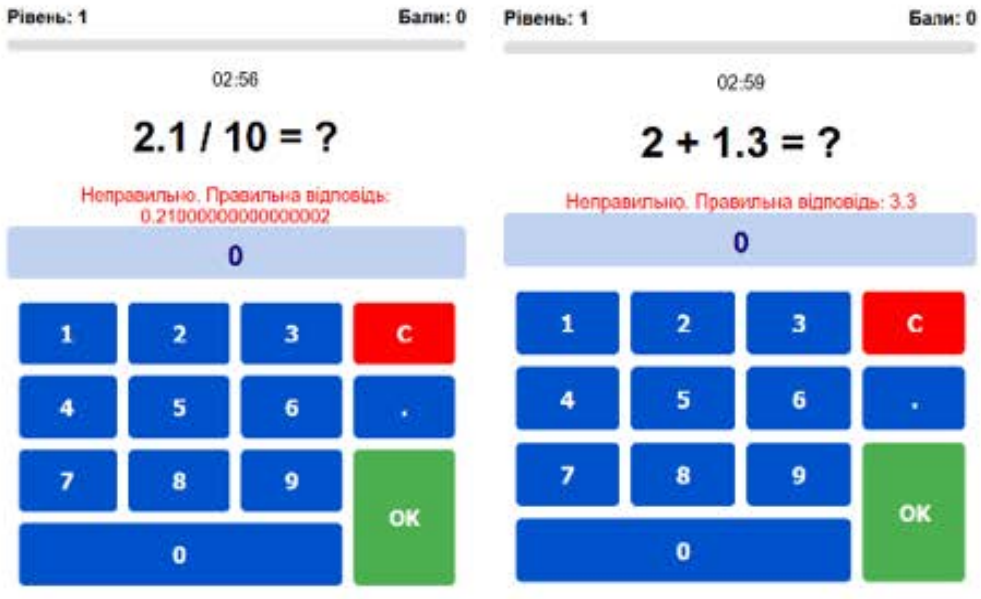

Рисунок 3.2 – Результат при введенні неправильної відповіді

При досягненні 30 балів, рівень складності автоматично збільшився на 1, що було відображено у відповідному елементі інтерфейсу. При наборі великої кількості балів, можна помітити, що рівень складності збільшується (більше 3), але складність самих завдань не міняються, що є помилкою.

В результаті тестування було виділено такі помилки, як: некоректне виведення правильної відповіді після помилки, в одному з тренажерів. Відсутність ускладнення завдань після 3 рівня складності, рівень оновлюється, а складність не змінюється.

Мета тестування модулю «Керування часом» – це перевірка коректності запуску, відліку та відображення таймера, а також його інтеграцію з процесом навчання.

Були використані такі тестові сценарії:

- перевірка запуску таймеру при початку тренування;
- перевірка коректності відображення часу;
- перевірка затримки часу під час помилки;
- перевірка дій після закінчення часу.

Після перевірки можемо визначити, що таймер успішно запускався автоматично одночасно з відображенням нової математичної задачі. Перевірка точності відліку часу показала, що таймер відображає час коректно, без помітних відхилень від реального часу. При введенні неправильної відповіді таймер успішно зупиняється на зазначений інтервал 5 секунд. Після закінчення часу виводиться повідомлення про результат тренування.

За результатами проведеного тестування, модуль «Керування часом» в цілому функціонує коректно та виконує свою роль у обмеженні часу на розв'язання задач. Запуск таймера відбувається своєчасно, відлік часу є точним, а дії після закінчення часу відповідають очікуваній поведінці. Зупинка таймера після введення неправильної відповіді також коректно працює.

Тестування інтерфейсу користувача є важливим процесом, щоб оцінити візуальний дизайн, розташування елементів, зрозумілість навігації та загальну зручність взаємодії користувача з математичним тренажером.

Під час цього тестування слід перевірити:

– відображення всіх елементів інтерфейсу;

– розташування елементів та їхньої логічної зв'язку;

– читабельності тексту та контрастності кольорів;

– зручність використання кнопок та поля вводу;

– візуальну реакцію на дії користувача.

В результаті тестування можемо спостерігати, що усі основні елементи інтерфейсу відображаються коректно та не перекривають один одного. Розташування елементів є логічним: поле вводу знаходиться безпосередньо над числовими кнопками. Розмір шрифтів є достатнім для комфортного читання як у тексті задач, так і в елементах інформації (рахунок, рівень). Контрастність кольорів між текстом та фоном забезпечує хорошу читабельність.

Кнопки для введення чисел та керування мають достатній розмір для зручного натискання на різних типах пристроїв (особливо на сенсорних екранах). При натисканні кнопок відбувається візуальна зміна їхнього стану (легке затемнення), що забезпечує зворотний зв'язок.

Загалом, інтерфейс користувача математичного тренажера є інтуїтивно зрозумілим, візуально привабливим та зручним для взаємодії. Розташування елементів є логічним, читабельність тексту забезпечена, а візуальний зворотний зв'язок на дії користувача є адекватним.

Метою тестування адаптивності дизайну є перевірка коректності відображення та функціональності інтерфейсу математичного тренажеру на різних розмірах екранів (десктоп, планшет, мобільний) та в різних веб-браузерах (Chrome, Firefox, Safari, Edge).

Основними тестовими сценаріями є:

– перевірка відображення на різних стандартних розмірах екранів;

– перевірка коректності розташування елементів при зміні розміру вікна браузера;

– перевірка функціональності всіх елементів на різних пристроях;

– перевірка читабельності тексту та масштабування елементів на малих екранах;

– візуальне порівняння відображення в різних браузерах.

Інтерфейс тренажера успішно адаптувався до різних стандартних розмірів екранів. На мобільних пристроях елементи розташовувалися вертикально, шрифти масштабувалися для кращої читабельності, а елементи керування були достатньо великими для натискання.

Реалізація адаптивного дизайну в математичному тренажері в цілому є успішною. Інтерфейс коректно відображається та функціонує на різних типах пристроїв та розмірах екранів, забезпечуючи зручний користувацький досвід.

3.3 Інструкція користувача

Легкому та якісному використанню вебсайту сприяє зручний інтерфейс користувача, а також детальна інструкція його використання.

При переході за посилання, користувача зустрічає «Головний екран» на якому у верхній частині міститься хедер, з назвою вебсайту та пунктом меню «Тренажери». По центру екрану розміщуються 4 картки, де зазначена назва та призначення кожного з тренажерів (рис. 3.3). На цьому етапі користувач може обрати потрібний тренажер для вдосконалення певної навички.

При виборі однієї з карток-тренажерів користувач опиняється безпосередньо на сторінці тренажеру (рис. 3.4). Верхня чамстина цієї сторінки ідентична, як у головної сторінки, при натисканні пункту меню «Тренажери» або при натисканні на логотип чи назву вебсайту у хедері користувач переходить на головну сторінку.

Основним елементом сторінки тренажеру є робоча область, що представлена калькулятором, у верхній частині міститься інформація про поточний рівень та кількість набраних балів, нижче розташовується прогрес-бар, що візуально демонструє користувачеві кількість набраних балів.

Наступним елементом є таймер, на якому відображається поточний час, що залишився на виконання завдань. Таймер запускається одразу при переході користувача на сторінку тренажеру. Загальний час, що виділяється дорівнює 3 хвилинам. Після закінчення відведеного на тренування часу, на екрані з'являється повідомлення, про кількість набраних балів та кнопка, що дозволяє користувачеві повернутися на головний екран.

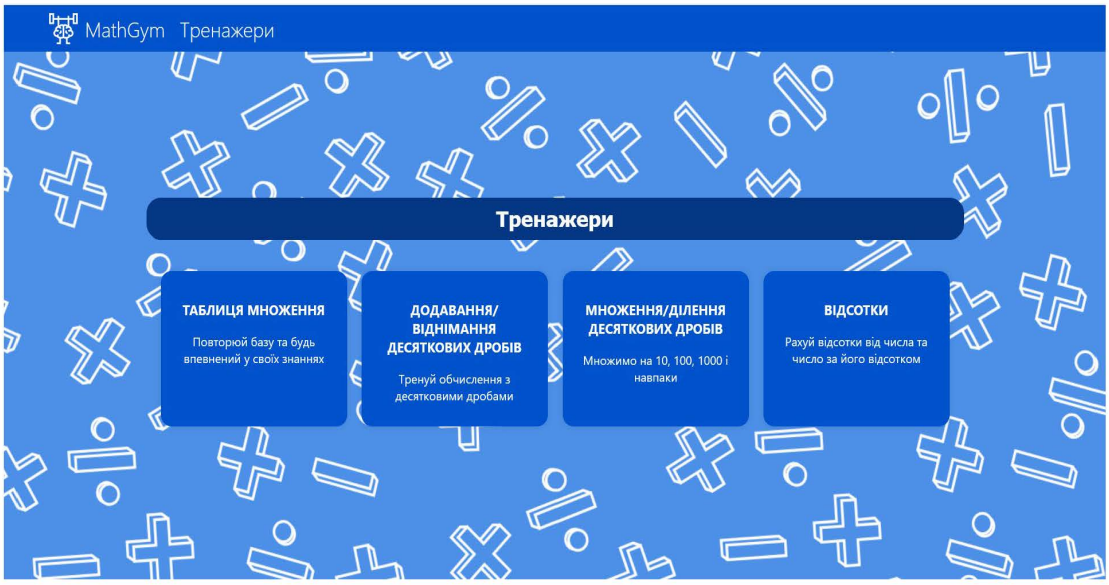

Рисунок 3.3 – Головний екран вебсайту

Після основної інформації такої, як: рівень, кількість балів та таймеру – розміщується поле на якому, з'являється згенерований приклад (завдання), що потребує введення користувачем правильної відповіді.

Для введення відповіді користувач може використовувати цифрові кнопки на екрані, що зручно при використання на мобільних пристроях чи планшетах. Користувачі, що використовують вебсайт безпосередньо за допомогою комп'ютера для введення відповіді можуть використовувати клавіатуру.

Уведені цифри з'являються у полі для введення відповідей, що представляє собою блакитний прямокутник. В залежності від завдань користувачу може знадобитися розділовий знак «.», для введення десяткових дробів. Для цього також можна використати кнопки на екрані чи клавіатуру.

Рисунок 3.4 – Інтерфейс сторінки тренажеру

Якщо при введені відповіді користувач допустив помилку, він може її виправити, використовуючи кнопку «С» на екрані, що виділена червоним кольором або кнопку «Backspace» на клавіатурі, це дозволить видалити останній введений елемент.

Коли користувач буде впевнений у наданій відповіді для її перевірки використовується кнопка «ОК» зеленого кольору або Enter. Для зручного використання та зрозумілості при натисканні на кнопки на екрані вони змінюють свій колір на більш темний, що дозволяє зрозуміти, що кнопка використовується (рис. 3.5).

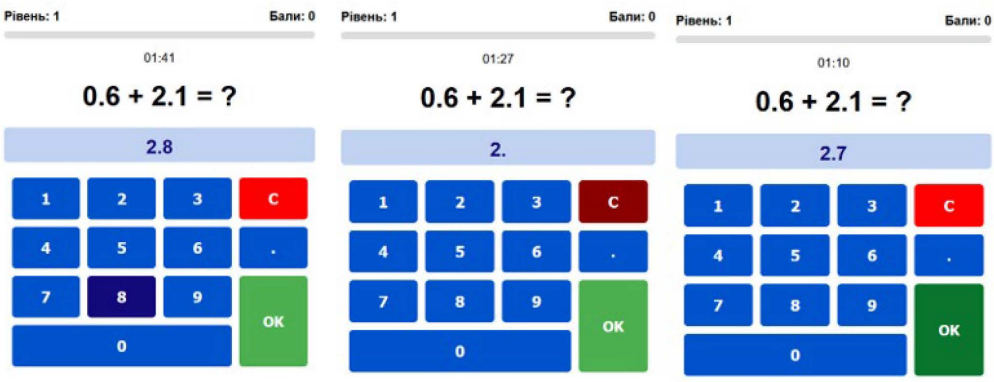

Рисунок 3.5 – Використання числових кнопок на екрані

Після натискання кнопки «ОК» або Enter введена відповідь автоматично перевіряється, якщо відповідь правильна нараховується 5 балів, у випадку

хибної відповіді, з'являється повідомлення на якому зазначається правильна відповідь, у цей момент таймер зупиняється на декілька секунд (рис. 3.6).

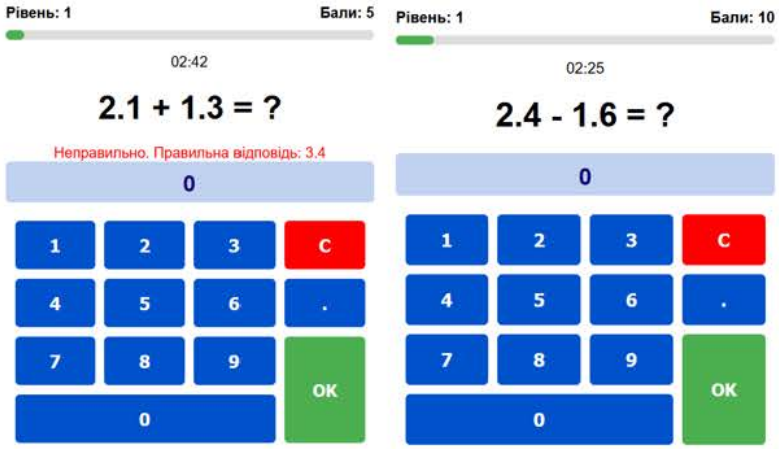

Рисунок 3.6 – Можливі результати перевірки відповіді

Після паузи таймеру генерація прикладів продовжується допоки не вичерпано весь час таймеру. При досягненні 30 балів, рівень складності збільшується, що дозволяє ще більше прокачати навички з певної теми.

Коли значення часу дорівнює 00:00 на екрані з'являється повідомлення про набрану кількість балів та кнопкою, що пропонує повернутися на головну сторінку (рис. 3.7).

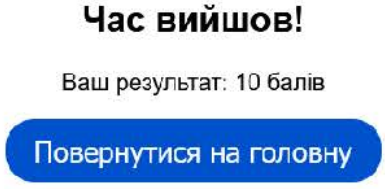

Рисунок 3.7 – Повідомлення після закінчення часу

Можна зазначити, що використання вебсайту є простим та зрозумілим для кожного, що дозволяє з легкістю користуватися будь-кому. Інтерфейс є доступним та ефективним на будь-якому пристрої.

3.4 Висновки до третього розділу

У поданому розділі було проведено безпосередню розробку та реалізацію інтерактивної навчальної платформи математичного тренажеру, що є основною метою кваліфікаційної роботи.

Основним етапом є безпосередня розробка платформи. Основними інструментами були базові технології веб-розробки: HTML, CSS, JavaScript. Основний функціонал реалізовано за допомогою JavaScript, що дозволило створити динамічний та інтерактивний користувацький досвід, реалізувати логіку математичних завдань, механізм генерації завдань, введення та перевірки відповідей, що є важливим елементом інтерактивної платформи.

У розділі детально описано розробку кожного функціонального модуля програми, а саме генерація завдань, керування часом, введення правильної відповіді та надання зворотного зв'язку.

Також було ознайомлено з основними елементами стилів CSS, які надавали вебсайту можливість адаптуватися під будь-який пристрій.

Наступним етапом розробки було тестування усіх модулів, що використовувалися. Проведення тестів дало змогу зрозуміти якість та коректність роботи функціоналу платформи. Загалом тестування показало добрі результати, але також були знайдені деякі помилки, що потребують усунення в майбутніх оновленнях.

Для зручного використання було зроблено інструкцію користувача, в якій зазначалось як саме користувач може взаємодіяти з вебсайтом. Описано основні сценарії використання та реакцію вебсайта на дії користувача.

Таким чином, у рамках третього розділу було успішно спроектовано, розроблено та протестовано повноцінний, функціональний та зручний вебсайт. Розроблений вебсайт є готовим інструментом для подальшого впровадження та використання в освітній діяльності.

ВИСНОВКИ

У сучасному освітньому просторі, де використання цифрових технологій відіграє важливу роль, створення якісних, доступних та ефективних навчальних платформ набуває неабиякої значущості.

Метою роботи була розробка та реалізація інтерактивної платформи магматичного тренажеру, що призначений для тренування навичок усного рахунку з таких тем: таблиця множення, дії з десятковими дробами, відсотки. Використання такої платформи дозволить підвищити ефективність освітнього процесу, залученість та мотивацію учнів, використовуючи інтерактивні елементи та елементи гейміфікації.

Для досягнення поставленої мети були вирішенні, поставлені на початку роботи завдання.

На початку роботи було проведено аналіз актуальності використання цифрових технологій у навчанні та виявлено основні вимоги до функціоналу. В основі першого розділу кваліфікаційної роботи було обґрунтовано актуальність впровадження інтерактивних ресурсів у навчання. Визначено, що традиційних методів вже замало для якісного навчання сучасних поколінь, що звикли до швидкого отримання інформації та постійного використання смартфонів. Створення інтерактивних вебсайтів, що завжди буде під рукою може збільшити залученість та мотивацію учнів.

Для впевненості у актуальності проекту було проведено анонімне опитування цільової аудиторії, а саме вчителів та репетиторів з математики, що показало важливість та необхідність розробки вебсайту, який дозволяє тренувати навички усного рахунку.

Наступним етапом було проведено аналіз вже існуючих рішень у сфері освіти таких, як: інтерактивний вебсайт тренажер Новатіка, платформа Learning, та сучасний сервіс створення інтерактивних завдань Wordwall. Цей аналіз дав змогу зрозуміти, що предметна область добре наповнена різноманітними продуктами, але не всі з них відповідають вище зазначеним

умовам. Тож незважаючи на велику конкуренцію кожна освітній вебсайт має своїх користувачів.

Перед безпосередньої розробки варто було визначитися з технологія розробки. Вибираючи між мобільним, десктопним та вебзастосунками, для виконання зазначених вимог найкращий підійшла саме веброзробка, оскільки вона є найбільш легкою у використанні на будь-якому пристрої за наявності Інтернету, що збільшує коло потенційних користувачів.

Для розробки клієнтської частини та забезпечення функціональності і інтерактивності було обрано стандартний набір веб технологій: HTML, CSS та JavaScript. Цей вибір забезпечує крос-платформність, високу швидкість завантаження, гнучкість у розробці користувацького інтерфейсу та легкість масштабування. JavaScript, зокрема, виявився ідеальним для реалізації динамічної логіки математичних завдань, перевірки відповідей у реальному часі та керування анімаціями гейміфікації. Вибір цих технологій дозволив створити сучасний, ефективний та легко підтримуваний програмний продукт.

Важливим етапом реалізації було створення логічної структури та архітектури, а також детальний дизайн інтерфейсу користувача.

Було детально опрацьовано дизайн кожного елементу інтерфейсу, починаючи від загальної композиції та розташування блоків, до дрібних інтерактивних елементів. Розроблено кольорову гаму, що сприяє концентрації та не викликає зорової втоми, а також візуально привабливий логотип та назву "MathGym", які підкреслюють спрямованість тренажеру.

В результаті реалізований дизайн інтерфейсу відповідає сучасним стандартам, роблячи процес навчання максимально комфортним та приємним.

Після розробки дизайну та основної структури платформи було здійснено програмну реалізацію функціоналу навчального середовища. Використовуючи JavaScript було створено алгоритми функціональних модулів, що відповідали за генерацію різноманітних математичних завдань, за обраною тематикою, можливість зручного введення відповідей на будь-якому

пристрої, перевірки введеної відповіді та система підрахунку балів та візуалізації прогресу.

Стилізація інтерфейсу за допомогою CSS забезпечила не лише естетичний вигляд, але й адаптивність до різних розмірів екранів, що робить тренажер доступним на будь-якому пристрої.

Останнім, але не менш головним етапом розробки було тестування кожного функціонального модулю та інтерфейсу користувача на зрозумілість, зручність та адаптивність.

Тестування охоплювало перевірку функціональності всіх модулів, коректність генерації та перевірки математичних завдань, стабільність роботи системи під різними навантаженнями та на різних браузерах, а також зручність користувацького інтерфейсу. За результатами тестування було підтверджено повну працездатність платформи, її відповідність всім початково встановленим вимогам та здатність ефективно виконувати свої функції як інтерактивного математичного тренажеру.

Для ознайомлення користувача з основним функціоналом вебсайту було створено інструкцію користувача, де розглядалися основні сценарії використання. Було зазначено, як дії користувача впливають на систему та, який зворотній зв'язок отримує користувач.

Отже, можна зазначити, що в рамках кваліфікаційної роботи було успішно розроблено та реалізовано вебсайт – математичний тренажер, який є сучасним, якісним та ефективним інструментом для поліпшення навичок усного рахунку та закріплення математичний навичок і знань з пройдених тем.

Таким чином, розроблений інтерактивний математичний тренажер є вагомим внеском у розвиток цифрових освітніх технологій та має значний потенціал для підвищення якості математичної освіти та формування стійкого інтересу до цієї дисципліни.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. 17 Important Data Visualization Techniques. URL: https://online.hbs.edu/blog/post/data-visualization-techniques/

2. Dagiene V., Stupuriene G., Vinikiene L. Implementation of Dynamic Tasks on Informatics and Computational Thinking. Baltic Journal of Modern Computing, 2017, 5(3), 306–316.

3. DigitalOcean. DigitalOcean | Cloud Infrastructure for Developers. URL: https://www.digitalocean.com/community/tutorial-series/how-to-code-in-javascript

4. Game-Based Learning in Universities and Lifelong Learning. (2004). URL: https://doi.org/10.3217/jucs-010-01-0014

5. Marcotte E. Responsive Web Design: Principles and Best Practices, Journal of Web Engineering, 2023, 15(3), 287–301.

6. Mazarakis A., Bräuer P. (2022). Gamification is Working, but Which One Exactly? International Journal of Human–Computer Interaction, 39(3), 612–627. https://doi.org/10.1080/10447318.2022.2041909

7. Mobile App vs. Website: What to Choose for the Business in 2025. Cleveroad Inc. URL: https://www.cleveroad.com/blog/mobile-app-vs-mobile-website/

8. Olsson M., Mozelius P., Collin J. Visualisation and Gamification of e-Learning and Programming Education. Electronic Journal of e-Learning, 2015, 13, 441–454. https://files.eric.ed.gov/fulltext/EJ1087309.pdf

9. Responsive vs. Adaptive Design. URL: https://careerfoundry.com/en/blog/uidesign/responsive-vs-adaptive-design/

10. The NMC Horizon Report: 2014 K-12 Edition. URL: https://files.eric.ed.gov/fulltext/ED559369.pdf

11. The Psychology of Colors in Education: How Colors Affect Students' Learning and Mood. URL: https://masarat-sy.org/en/the-psychology-of-colors-in-education-how-colors-affect-students-learning-and-mood/

12. Web vs. Desktop vs. Mobile: Why Web Platforms Are Often the Best Choice - 3 APPES. URL: https://www.3appes.com/web-vs-mobile-vs-desktop/

13. What Color Is Math? Shocking Learning Psychology in the US & Canada - Educify Blog. URL: https://learn.educify.org/what-color-is-math-shocking-learning-psychology-in-the-us-canada/

14. What Is Component Diagram? Ideal Modeling & Diagramming Tool for Agile Team Collaboration. URL: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/

15. Yilmaz E., Sahin M., Turgut M. Variables Affecting Student Motivation Based on Academic Publications. Journal of Education and Practice, 2017, 8(12), 112–120.

16. Бондар І. О. Теорія кольору: навчальний посібник для студентів напряму підготовки 6.051501 «Видавничо-поліграфічна справа». Харків: ХНЕУ ім. С. Кузнеця, 2016. 164 с.

17. Вплив платформ для онлайн-курсів на навчання в школах. Освіта.UA. URL: https://osvita.ua/news/89291/#google_vignette

18. Інформаційно-комунікаційні технології в освіті і науці: електронний навчально-методичний посібник / уклад. А. Грітченко. Умань: УДПУ, 2021. 122 с.

19. Ключові показники ефективності. Юзабіліті, UX, Залучення, Конверсія, CRO. URL: https://topuser.pro/pokazateli-usability-ux-konversiya-cro/

20. Кучерак І. В. Цифровізація та її вплив на освітній простір у контексті формування ключових компетентностей. Інноваційна педагогіка, 2020, Вип. 22, т.2. С. 91–94.

21. Майбутнє освіти: роль інтерактивних технологій в школах та університетах – INTBOARD. URL: https://intboard.ua/pres-sluzhba/blog/maybutnye-osvity-rol′-interaktyvnykh-tekhnolohiy-v-shkolakh-ta-universytetakh/

22. Немчук О. О., Стрельбіцький В. В. Переваги та недоліки дистанційного навчання. Distance Education in Ukraine: Innovative, Normative-

Legal, Pedagogical Aspects, 2021, № 1, С. 208–209. https://doi.org/10.18372/2786-5495.1.15784

23. Розробка веб-сайтів для початківців: HTML – CSS – JavaScript. Самвидав, 2022.

24. Сидоренко П. В. Веб-технології та розробка інтерактивних інтерфейсів. Київ: Техносер, 2021.

25. Силенко Ю. В., Іванова М. С. Інтерактивний дизайн у сучасному візуальному середовищі: тенденції та інновації. Грааль науки, 2024, (45), 697–700.

26. Expans. Адаптивна верстка сайту або мобільна версія? URL: https://expans.ua/blog/adaptyvnaverstka-saytu-abo-mobilna-versiya/

27. Яковлєва І. Використання освітніх платформ в освітньому середовищі. Ukrainian Educational Journal, 2022, № 3, С. 137–148. https://doi.org/10.32405/2411-1317-2022-3-137-148

ДОДАТОК А. ЛІСТИНГ HTML

Головна сторінка:

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link
href="https://fonts.googleapis.com/css2?family=Montserrat:wght@200;300&family=Open+Sans:ital,wgh
t@0,300..800;1,300..800&display=swap" rel="stylesheet">
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <header>
    <div class="logo"><a href="index.html">
      <img src="media/logo.png" alt="MathGym Logo" />
      <span>MathGym</span>
    </div></a>
  <nav>
      <a href="index.html#about" >Тренажери</a>
  </nav>
  </header>
  <div class="content">
    <button class="title">Тренажери</button>
    <div class="train_card">
      <a href="train3.html"><div class="card">
        <h3>Таблиця множення</h3>
        <p>Повторюй базу та будь впевнений у своїх знаннях</p>
      </div></a>
      <a href="train1.html"> <div class="card" >
        <h3>Додавання/віднімання десяткових дробів</h3>
        <p>Тренуй обчислення з десятковими дробами</p>
      </div></a>
      <a href="train2.html"><div class="card">
        <h3>Множення/ділення десяткових дробів</h3>
        <p>Множимо на 10, 100, 1000 і навпаки</p>
      </div></a>
      <a href="train4.html"><div class="card">
        <h3>Відсотки</h3>
        <p>Рахуй відсотки від числа та число за його відсотком</p>
      </div></a>
    </div>
  </div>
</body>
```

```
</html>
```

Сторінки тренажерів:

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
                                                                                <link
href="https://fonts.googleapis.com/css2?family=Montserrat:wght@200;300&family=Open+Sans:ital,wgh
t@0,300..800;1,300..800&display=swap" rel="stylesheet">
  <link rel="stylesheet" href="styles.css">
</head>

<body class="calculator-page">
  <header>
    <div class="logo"><a href="index.html">
      <img src="media/logo.png" alt="MathGym Logo" />
      <span>MathGym</span>
    </div></a>
  <nav>
      <a href="index.html#about" >Тренажери</a>
  </nav>
  </header>
  <div class="container">
    <div class="progress-container">
      <div class="top-bar">
        <span class="level">Рівень: <span id="level">1</span></span>
        <span class="score">Бали: <span id="score">0</span></span>
      </div>
      <div class="progress-bar">
        <div id="progress" class="progress"></div>
      </div>
    </div>
    <div class="timer">
      <i class="time"></i>
      <span id="timer">00:00</span>
    </div>
    <div class="problem-container">
      <span id="problem"></span>
    </div>
    <div id="result-message"></div>
    <div>
      <div class="answer-container">
        <span id="user-answer">0</span>
      </div>
```

```
        <div class="calculator">
            <button class="number-btn">1</button>
            <button class="number-btn">2</button>
            <button class="number-btn">3</button>
            <button class="clear-btn">C</button>
            <button class="number-btn">4</button>
            <button class="number-btn">5</button>
            <button class="number-btn">6</button>
            <button class="number-btn">.</button>
            <button class="number-btn">7</button>
            <button class="number-btn">8</button>
            <button class="number-btn">9</button>
            <button id="submit-btn">OK</button>
            <button class="number-btn zero">0</button>

        </div>
        <div id="result" class="hidden">
            <h2>Час вийшов!</h2>
            <p>Ваш результат: <span id="finalScore"></span> балів</p>
            <button class="back"><a href="index.html">Повернутися на головну</a></button>
        </div>
    </div>
  </div>
<script src="script_tr1.js"></script>
</body>
</html>
```

Тренажер 1. Таблиця множення.

```
// Timer functionality
let seconds = 180; // Початкове значення таймера в секундах
let timerInterval;
let timerPaused = false;

function startTimer() {
   const timerDisplay = document.getElementById('timer');
   timerInterval = setInterval(() => {
      if (!timerPaused) {
         seconds--;
         const minutes = Math.floor(seconds / 60);
         const remainingSeconds = seconds % 60;
         timerDisplay.textContent =
            `${minutes.toString().padStart(2, '0')}:${remainingSeconds.toString().padStart(2, '0')}`;

         if (seconds <= 0) {
            clearInterval(timerInterval);
            showFinalResult();
         }
      }
   }, 1000);
}
function showFinalResult() {
   const score = document.getElementById('score').textContent;
   document.getElementById('finalScore').textContent = score;
   document.getElementById('result').style.display = 'block';
   document.getElementById("answerInput").disabled = true;
   document.getElementById("submitButton").disabled = true;
}

function pauseTimer(duration) {
   timerPaused = true;
   setTimeout(() => {
      timerPaused = false;
   }, duration);
}

let currentAnswer = '';
const numberButtons = document.querySelectorAll('.number-btn');
const clearButton = document.querySelector('.clear-btn');
const userAnswerDisplay = document.getElementById('user-answer');
const submitButton = document.getElementById('submit-btn');
const problemDisplay = document.getElementById('problem');
```

```
const resultDisplay = document.getElementById('result-message');
const progressBar = document.getElementById('progress');
const levelDisplay = document.getElementById('level');
const scoreDisplay = document.getElementById('score');

let currentProblem = {
  num1: 0,
  num2: 0,
  operation: '',
  correctAnswer: 0
};
let score = 0;
let level = 1;
function generateNewProblem() {

  let min, max;

  // Встановлення діапазону чисел відповідно до рівня
  if (level === 1) {
    min = 1;
    max = 4;
  } else if (level === 2) {
    min = 5;
    max = 7;
  } else {
    min = 2;
    max = 10;
  }

  const num1 = Math.floor(Math.random() * (max - min + 1)) + min;
  const num2 = Math.floor(Math.random() * (max - min + 1)) + min;

  const operations = ['*', '/'];
  const operation = operations[Math.floor(Math.random() * operations.length)];

  let problemText = '';
  let correctAnswer;

  switch (operation) {
    case '*':
      problemText = `${num1} * ${num2} = ?`;
      correctAnswer = num1 * num2;
      break;
    case '/':
      const product = num1 * num2;
      problemText = `${product} / ${num1} = ?`;
      correctAnswer = num2;
      break;
  }
```

```
    problemDisplay.textContent = problemText;
    currentProblem = {
        num1,
        num2,
        operation,
        correctAnswer
    };
    resultDisplay.textContent = '';
    resultDisplay.style.color = '';
}


// обробка кліків по числових кнопках
numberButtons.forEach(button => {
    button.addEventListener('click', () => {
        const number = button.textContent;
        currentAnswer += number;
        userAnswerDisplay.textContent = currentAnswer || '0';
    });
});

// очищення поля відповіді
clearButton.addEventListener('click', () => {
    currentAnswer = currentAnswer.slice(0, -1);
    userAnswerDisplay.textContent = currentAnswer || '0';
});

// підтримка введення з клавіатури
document.addEventListener('keydown', (event) => {
    const key = event.key;

    if ((/^[0-9]$/).test(key)) {
        currentAnswer += key;
    } else if (key === '.' && !currentAnswer.includes('.')) {
        currentAnswer += key;
    } else if (key === 'Backspace') {
        currentAnswer = currentAnswer.slice(0, -1);
    } else if (key === 'Enter') {
        submitButton.click();
    }

    userAnswerDisplay.textContent = currentAnswer || '0';
});

// Обробка відповіді
submitButton.addEventListener('click', () => {
    console.log("currentAnswer перед перевіркою:", currentAnswer);
    console.log("Очікувана правильна відповідь:", currentProblem.correctAnswer);
    const userAnswer = parseInt(currentAnswer);  // читаємо як ціле число
```

```
    const correctAnswer = currentProblem.correctAnswer;

    // Перевірка правильної відповіді — точне порівняння цілих
    const isCorrect = Number.isInteger(userAnswer) && userAnswer === correctAnswer;

    if (isCorrect) {
        score += 5;
        scoreDisplay.textContent = `${score}`;
        resultDisplay.textContent = '';
        const currentWidth = parseInt(progressBar.style.width) || 0;
        const newWidth = Math.min(currentWidth + 5, 100);
        progressBar.style.width = `${newWidth}%`;
        generateNewProblem();
        currentAnswer = '';
        userAnswerDisplay.textContent = '0';

        if (score % 30 === 0) {
            level++;
            levelDisplay.textContent = `${level}`;
        }
    } else {
        resultDisplay.textContent = `Неправильно. Правильна відповідь: ${correctAnswer}`;
        resultDisplay.style.color = 'red';
        pauseTimer(5000);
        setTimeout(() => {
            resultDisplay.textContent = '';
            generateNewProblem();
            currentAnswer = '';
            userAnswerDisplay.textContent = '0';
        }, 5000);
    }

    if (!isCorrect) {
        currentAnswer = '';
        userAnswerDisplay.textContent = '0';
    }
});

// Запуск при завантаженні
window.onload = () => {
    startTimer();
    generateNewProblem();
    scoreDisplay.textContent = `${score}`;
    progressBar.style.width = '0%';
};
```

Тренажер 2. Додавання/віднімання десяткових дробів.

```
// функціонал таймеру
let seconds = 180; // Початкове значення таймера в секундах
let timerInterval;
let timerPaused = false;
// початок відліку
function startTimer() {
   const timerDisplay = document.getElementById('timer');
   timerInterval = setInterval(() => {
      if (!timerPaused) {
         seconds--;
         const minutes = Math.floor(seconds / 60);
         const remainingSeconds = seconds % 60;
         timerDisplay.textContent =
            `${minutes.toString().padStart(2, '0')}:${remainingSeconds.toString().padStart(2, '0')}`;

         if (seconds <= 0) {
            clearInterval(timerInterval);
            showFinalResult();
         }
      }
   }, 1000);
}
// функція показу результату
function showFinalResult() {
   const score = document.getElementById('score').textContent;
   document.getElementById('finalScore').textContent = score;
   document.getElementById('result').style.display = 'block';
   document.getElementById("answerInput").disabled = true;
   document.getElementById("submitButton").disabled = true;
}
// зупинка таймера
function pauseTimer(duration) {
   timerPaused = true;
   setTimeout(() => {
      timerPaused = false;
   }, duration);
}

let currentAnswer = '';
const numberButtons = document.querySelectorAll('.number-btn');
const clearButton = document.querySelector('.clear-btn');
const userAnswerDisplay = document.getElementById('user-answer');
const submitButton = document.getElementById('submit-btn');
const problemDisplay = document.getElementById('problem');
const resultDisplay = document.getElementById('result-message');
const progressBar = document.getElementById('progress');
const levelDisplay = document.getElementById('level');
```

```
const scoreDisplay = document.getElementById('score');

let currentProblem = {
  num1: 0,
  num2: 0,
  operation: '',
  correctAnswer: 0
};
let score = 0;
let level = 1;
// генерація десяткових дробів
function generateDecimal(maxIntegerPart, maxDecimalPlaces) {
  const integerPart = Math.floor(Math.random() * maxIntegerPart);
  const decimalPlaces = Math.floor(Math.random() * maxDecimalPlaces) + 1;
  const decimalPart = Math.floor(Math.random() * Math.pow(10, decimalPlaces));
  return parseFloat(`${integerPart}.${decimalPart.toString().padStart(decimalPlaces, '0')}`);
}
// функція генерації завдання
function generateNewProblem() {
  let maxInteger, maxDecimal;

  // Залежно від рівня складність росте
  if (level === 1) {
    maxInteger = 3;
    maxDecimal = 1;
  } else if (level === 2) {
    maxInteger = 5;
    maxDecimal = 2;
  } else {
    maxInteger = 10 + level*2;
    maxDecimal = Math.min(2, 1 + Math.floor(level / 2));
  }

  const num1 = generateDecimal(maxInteger, maxDecimal);
  const num2 = generateDecimal(maxInteger, maxDecimal);

  const operations = ['+', '-'];
  const operation = operations[Math.floor(Math.random() * operations.length)];

  let problemText = '';
  let correctAnswer;
  switch (operation) {
    case '+':
      problemText = `${num1} + ${num2} = ?`;
      correctAnswer = parseFloat((num1 + num2).toFixed(maxDecimal));
      break;
    case '-':
      const max = Math.max(num1, num2);
      const min = Math.min(num1, num2);
      problemText = `${max} - ${min} = ?`;
```

```
            correctAnswer = parseFloat((max - min).toFixed(maxDecimal));
            break;
    }

    problemDisplay.textContent = problemText;
    currentProblem = {
        num1,
        num2,
        operation,
        correctAnswer
    };
    resultDisplay.textContent = '';
    resultDisplay.style.color = '';
}

// обробка кліків по числових кнопках
numberButtons.forEach(button => {
    button.addEventListener('click', () => {
        const number = button.textContent;
        currentAnswer += number;
        userAnswerDisplay.textContent = currentAnswer || '0';
    });
});

// очищення поля відповіді
clearButton.addEventListener('click', () => {
    currentAnswer = currentAnswer.slice(0, -1);
    userAnswerDisplay.textContent = currentAnswer || '0';
});

// підтримка введення з клавіатури
document.addEventListener('keydown', (event) => {
    const key = event.key;

    if ((/^[0-9]$/).test(key)) {
        currentAnswer += key;
    } else if (key === '.' && !currentAnswer.includes('.')) {
        currentAnswer += key;
    } else if (key === 'Backspace') {
        currentAnswer = currentAnswer.slice(0, -1);
    } else if (key === 'Enter') {
        submitButton.click();
    }

    userAnswerDisplay.textContent = currentAnswer || '0';
});

// Обробка відповіді
submitButton.addEventListener('click', () => {
    const userAnswer = parseFloat(currentAnswer);
```

```
    const isCorrect = !isNaN(userAnswer) && Math.abs(userAnswer - currentProblem.correctAnswer) <
1e-9;

    if (isCorrect) {
        score += 5;
        scoreDisplay.textContent = `${score}`;
        resultDisplay.textContent = '';
        const currentWidth = parseInt(progressBar.style.width) || 0;
        const newWidth = Math.min(currentWidth + 5, 100);
        progressBar.style.width = `${newWidth}%`;
        generateNewProblem();
        currentAnswer = '';
        userAnswerDisplay.textContent = '0';
        if (score % 30 === 0) {
            level++;
            levelDisplay.textContent = `${level}`;
        }
    } else {
                        resultDisplay.textContent = `Неправильно. Правильна відповідь:
${currentProblem.correctAnswer}`;
        resultDisplay.style.color = 'red';
        pauseTimer(5000);
        setTimeout(() => {
            resultDisplay.textContent = '';
            generateNewProblem();
            currentAnswer = '';
            userAnswerDisplay.textContent = '0';
        }, 5000);
    }

    if (!isCorrect) {
        currentAnswer = '';
        userAnswerDisplay.textContent = '0';
    }
});

// Запуск при завантаженні
window.onload = () => {
    startTimer();
    generateNewProblem();
    scoreDisplay.textContent = `${score}`;
    progressBar.style.width = '0%';
};
```

Тренажер 3. Множення/ділення десяткових дробів.

```
// Timer functionality
let seconds = 180; // Початкове значення таймера в секундах
let timerInterval;
let timerPaused = false;

function startTimer() {
  const timerDisplay = document.getElementById('timer');
  timerInterval = setInterval(() => {
    if (!timerPaused) {
      seconds--;
      const minutes = Math.floor(seconds / 60);
      const remainingSeconds = seconds % 60;
      timerDisplay.textContent =
        `${minutes.toString().padStart(2, '0')}:${remainingSeconds.toString().padStart(2, '0')}`;

      if (seconds <= 0) {
        clearInterval(timerInterval);
        showFinalResult();
      }
    }
  }, 1000);
}
function showFinalResult() {
  const score = document.getElementById('score').textContent;
  document.getElementById('finalScore').textContent = score;
  document.getElementById('result').style.display = 'block';
  document.getElementById("answerInput").disabled = true;
  document.getElementById("submitButton").disabled = true;
}

function pauseTimer(duration) {
  timerPaused = true;
  setTimeout(() => {
    timerPaused = false;
  }, duration);
}

let currentAnswer = '';
const numberButtons = document.querySelectorAll('.number-btn');
const clearButton = document.querySelector('.clear-btn');
const userAnswerDisplay = document.getElementById('user-answer');
const submitButton = document.getElementById('submit-btn');
const problemDisplay = document.getElementById('problem');
const resultDisplay = document.getElementById('result-message');
const progressBar = document.getElementById('progress');
const levelDisplay = document.getElementById('level');
const scoreDisplay = document.getElementById('score');
```

```
let currentProblem = {
  num1: 0,
  num2: 0,
  operation: '',
  correctAnswer: 0
};
let score = 0;
let level = 1;

function generateDecimal(maxIntegerPart, maxDecimalPlaces) {
  const integerPart = Math.floor(Math.random() * maxIntegerPart);
  const decimalPlaces = Math.floor(Math.random() * maxDecimalPlaces) + 1;
  const decimalPart = Math.floor(Math.random() * Math.pow(10, decimalPlaces));
  return parseFloat(`${integerPart}.${decimalPart.toString().padStart(decimalPlaces, '0')}`);
}

function generateNewProblem() {
  let maxInteger, maxDecimal, multipliersAndDivisors;

  // Залежно від рівня складність росте
  if (level === 1) {
    maxInteger = 3;
    maxDecimal = 1;
    multipliersAndDivisors = [10, 0.1];  // Множники та дільники на рівні 1
  } else if (level === 2) {
    maxInteger = 5;
    maxDecimal = 2;
    multipliersAndDivisors = [100, 0.01];  // Множники та дільники на рівні 2
  } else {
    maxInteger = 10 + level * 2;
    maxDecimal = Math.min(2, 1 + Math.floor(level / 2));
    multipliersAndDivisors = [1000, 0.001];  // Множники та дільники на рівні 3+
  }

  // Генерація випадкових чисел з десятковими дробами
  const num1 = generateDecimal(maxInteger, maxDecimal);

  // Операції множення та ділення
  const operations = ['*', '/'];
  const operation = operations[Math.floor(Math.random() * operations.length)];

  // Вибір випадкового множника або дільника
      const multiplierOrDivisor = multipliersAndDivisors[Math.floor(Math.random() *
multipliersAndDivisors.length)];

  let problemText = '';
  let correctAnswer;

  switch (operation) {
```

```
        case '*':
            problemText = `${num1} * ${multiplierOrDivisor} = ?`;
            correctAnswer = parseFloat((num1 * multiplierOrDivisor));
            break;
        case '/':
            problemText = `${num1} / ${multiplierOrDivisor} = ?`;
            correctAnswer = parseFloat((num1 / multiplierOrDivisor));
            break;
    }

    problemDisplay.textContent = problemText;
    currentProblem = {
        num1,
        operation,
        multiplierOrDivisor,
        correctAnswer
    };
    resultDisplay.textContent = '';
    resultDisplay.style.color = '';
}

// обробка кліків по числових кнопках
numberButtons.forEach(button => {
    button.addEventListener('click', () => {
        const number = button.textContent;
        currentAnswer += number;
        userAnswerDisplay.textContent = currentAnswer || '0';
    });
});

// очищення поля відповіді
clearButton.addEventListener('click', () => {
    currentAnswer = currentAnswer.slice(0, -1);
    userAnswerDisplay.textContent = currentAnswer || '0';
});

// підтримка введення з клавіатури
document.addEventListener('keydown', (event) => {
    const key = event.key;

    if ((/^[0-9]$/).test(key)) {
        currentAnswer += key;
    } else if (key === '.' && !currentAnswer.includes('.')) {
        currentAnswer += key;
    } else if (key === 'Backspace') {
        currentAnswer = currentAnswer.slice(0, -1);
    } else if (key === 'Enter') {
        submitButton.click();
    }
```

```
  userAnswerDisplay.textContent = currentAnswer || '0';
});

// Обробка відповіді
submitButton.addEventListener('click', () => {
  const userAnswer = parseFloat(currentAnswer);
   const isCorrect = !isNaN(userAnswer) && Math.abs(userAnswer - currentProblem.correctAnswer) <
1e-9;

  if (isCorrect) {
    score += 5;
    scoreDisplay.textContent = `${score}`;
    resultDisplay.textContent = '';
    const currentWidth = parseInt(progressBar.style.width) || 0;
    const newWidth = Math.min(currentWidth + 5, 100);
    progressBar.style.width = `${newWidth}%`;
    generateNewProblem();
    currentAnswer = '';
    userAnswerDisplay.textContent = '0';
    if (score % 30 === 0) {
      level++;
      levelDisplay.textContent = `${level}`;
    }
  } else {
                    resultDisplay.textContent = `Неправильно. Правильна відповідь:
${currentProblem.correctAnswer}`;
    resultDisplay.style.color = 'red';
    pauseTimer(5000);
    setTimeout(() => {
      resultDisplay.textContent = '';
      generateNewProblem();
      currentAnswer = '';
      userAnswerDisplay.textContent = '0';
    }, 5000);
  }

  if (!isCorrect) {
    currentAnswer = '';
    userAnswerDisplay.textContent = '0';
  }
});

// Запуск при завантаженні
window.onload = () => {
  startTimer();
  generateNewProblem();
  scoreDisplay.textContent = `${score}`;
  progressBar.style.width = '0%';
};
```

Тренажер 4. Відсотки.

```
// Timer functionality
let seconds = 180; // Початкове значення таймера в секундах
let timerInterval;
let timerPaused = false;

function startTimer() {
  const timerDisplay = document.getElementById('timer');
  timerInterval = setInterval(() => {
    if (!timerPaused) {
      seconds--;
      const minutes = Math.floor(seconds / 60);
      const remainingSeconds = seconds % 60;
      timerDisplay.textContent =
        `${minutes.toString().padStart(2, '0')}:${remainingSeconds.toString().padStart(2, '0')}`;

      if (seconds <= 0) {
        clearInterval(timerInterval);
        showFinalResult();
      }
    }
  }, 1000);
}
function showFinalResult() {
  const score = document.getElementById('score').textContent;
  document.getElementById('finalScore').textContent = score;
  document.getElementById('result').style.display = 'block';
  document.getElementById("answerInput").disabled = true;
  document.getElementById("submitButton").disabled = true;
}

function pauseTimer(duration) {
  timerPaused = true;
  setTimeout(() => {
    timerPaused = false;
  }, duration);
}

let currentAnswer = '';
const numberButtons = document.querySelectorAll('.number-btn');
const clearButton = document.querySelector('.clear-btn');
const userAnswerDisplay = document.getElementById('user-answer');
const submitButton = document.getElementById('submit-btn');
const problemDisplay = document.getElementById('problem');
const resultDisplay = document.getElementById('result-message');
const progressBar = document.getElementById('progress');
const levelDisplay = document.getElementById('level');
const scoreDisplay = document.getElementById('score');
```

```
let currentProblem = {
    num1: 0,
    num2: 0,
    operation: '',
    correctAnswer: 0
};
let score = 0;
let level = 1;

function generateNewProblem() {
    const taskType = Math.random() < 0.5 ? 'findPercentage' : 'findNumber';

    let percentageOptions;
    let maxNumber;

    // Встановлюємо параметри залежно від рівня
    if (level === 1) {
        percentageOptions = [10, 20, 25, 50 ];
        maxNumber = 100;
    } else if (level === 2) {
        percentageOptions = [5, 10, 25, 50, 75];
        maxNumber = 200;
    } else if (level === 3) {
        percentageOptions = [5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100];
        maxNumber = 500;
    } else {
        percentageOptions = [5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100];
        maxNumber = 1000;
    }

    let percentage = percentageOptions[Math.floor(Math.random() * percentageOptions.length)];
    let baseNumber, result;
    let problemText, correctAnswer;

    if (taskType === 'findPercentage') {
        // Шукаємо n% від числа — підбираємо число так, щоб результат був цілим
        do {
            baseNumber = Math.floor(Math.random() * maxNumber) + 1;
            result = (baseNumber * percentage) / 100;
        } while (!Number.isInteger(result));

        problemText = `Знайти ${percentage}% від числа ${baseNumber}`;
        correctAnswer = result;

    } else {
        // Шукаємо число, якщо n% від нього = відоме значення
        do {
            result = Math.floor(Math.random() * maxNumber) + 1;
            baseNumber = (result * 100) / percentage;
```

```
    } while (!Number.isInteger(baseNumber));

    problemText = `Знайти число, якщо ${percentage}% від нього дорівнює ${result}`;
    correctAnswer = baseNumber;
  }

  problemDisplay.textContent = problemText;
  currentProblem = {
    num1: baseNumber,
    num2: result,
    operation: taskType,
    correctAnswer
  };
  resultDisplay.textContent = '';
  resultDisplay.style.color = '';
}


// обробка кліків по числових кнопках
numberButtons.forEach(button => {
  button.addEventListener('click', () => {
    const number = button.textContent;
    currentAnswer += number;
    userAnswerDisplay.textContent = currentAnswer || '0';
  });
});

// очищення поля відповіді
clearButton.addEventListener('click', () => {
  currentAnswer = currentAnswer.slice(0, -1);
  userAnswerDisplay.textContent = currentAnswer || '0';
});

// підтримка введення з клавіатури
document.addEventListener('keydown', (event) => {
  const key = event.key;

  if ((/^[0-9]$/).test(key)) {
    currentAnswer += key;
  } else if (key === '.' && !currentAnswer.includes('.')) {
    currentAnswer += key;
  } else if (key === 'Backspace') {
    currentAnswer = currentAnswer.slice(0, -1);
  } else if (key === 'Enter') {
    submitButton.click();
  }

  userAnswerDisplay.textContent = currentAnswer || '0';
});
```

```
// Обробка відповіді
submitButton.addEventListener('click', () => {
    const userAnswer = parseInt(currentAnswer);  // читаємо як ціле число
    const correctAnswer = currentProblem.correctAnswer;

    // Перевірка правильної відповіді — точне порівняння цілих
    const isCorrect = Number.isInteger(userAnswer) && userAnswer === correctAnswer;

    if (isCorrect) {
        score += 5;
        scoreDisplay.textContent = `${score}`;
        resultDisplay.textContent = '';
        const currentWidth = parseInt(progressBar.style.width) || 0;
        const newWidth = Math.min(currentWidth + 5, 100);
        progressBar.style.width = `${newWidth}%`;
        generateNewProblem();
        currentAnswer = '';
        userAnswerDisplay.textContent = '0';

        if (score % 30 === 0) {
            level++;
            levelDisplay.textContent = `${level}`;
        }
    } else {
        resultDisplay.textContent = `Неправильно. Правильна відповідь: ${correctAnswer}`;
        resultDisplay.style.color = 'red';
        pauseTimer(5000);
        setTimeout(() => {
            resultDisplay.textContent = '';
            generateNewProblem();
            currentAnswer = '';
            userAnswerDisplay.textContent = '0';
        }, 5000);
    }

    if (!isCorrect) {
        currentAnswer = '';
        userAnswerDisplay.textContent = '0';
    }
});

// Запуск при завантаженні
window.onload = () => {
    startTimer();
    generateNewProblem();
    scoreDisplay.textContent = `${score}`;
    progressBar.style.width = '0%';
};
```

```
body {
    font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    min-height: 100vh;
    text-align: center;
    background-size: cover;
    background-position: center;
    background-repeat: no-repeat;
    background-attachment: fixed;
    background-image: url(media/background.png);
}
header {
    width: 100%;
    background-color: #0052cc;
    color: white;
    font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
    display: flex;
    align-items: center;
    justify-content: flex-start;
    padding: 10px 30px;
    box-sizing: border-box;
    font-size: 1.7rem;
    top: 0;
    left: 0;
    right: 0;
    z-index: 1000;
    position: fixed;
}
.logo a{
    display: flex;
    align-items: center;
    padding-left: 20%;
    text-decoration: none;
    gap: 10px;
    font-size: 1.7rem;
    font-weight: lighter;
    color: white;
}
.logo img{
    height: 40px;
```

```
    width: auto;
    max-width: 100%;
    object-fit: contain;
}
@media (max-width: 480px) {
    .logo img {
      height: 30px;
    }

    .logo span {
      font-size: 1.2rem;
    }
  }
nav a{
    margin-left: 40%;
    color: white;
    text-decoration: none;
    font-size: 1.7rem;
    font-weight: lighter;
}
@media (max-width: 480px){
    nav a{
        font-size: 1.2rem;
    }
}
nav a:hover{
    font-weight: 400;
}
@media (max-width: 768px){
    .train_card{
        flex-direction: column;
        align-items: center;
    }
}
.title{
    background-color: #043784;
    color: white;
    padding: 10px 20px;
    border-radius: 20px;
    display: inline-block;
    margin-bottom: 20px;
    font-size: 25px;
    cursor: default;
    text-align: center;
    max-width: 100%;
    word-wrap: break-word;
    border: none;
    box-sizing: border-box;
}
.train_card{
```

```
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
    gap: 20px;
    padding: 20px;
    max-width: 1200px;
    width: 100%;
    box-sizing: border-box;
}
.content{
    margin-top: 70px;
    display: flex;
    flex-direction: column;
    align-items: center;
    padding: 10px;
    box-sizing: border-box;
}
.card {
    width: 220px;
    height: 160px;
    background-color: #0052cc;
    border-radius: 15px;
    padding: 20px;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
    text-align: center;
    color: white;
    flex-direction: column;
    justify-content: center;
}
.train_card a{
    text-decoration: none;
}
.train_card h3{
    text-transform: uppercase;
    font-size: 1.1rem;
}
.card:hover {
    transform: translateY(-5px);
}

.title {
    font-size: 28px;
    font-weight: bold;
}
.topics {
    margin-top: 20px;
    font-size: 18px;
    text-align: left;
    width: max-content;
    margin-left: auto;
```

```
    margin-right: auto;
}

.topics ul {
    list-style: none;
    padding: 0;
}

.topics li {
    padding: 10px;
    margin: 5px auto;
    border-radius: 10px;

}
.topics a{
    color: aliceblue;
    text-decoration: none;

}

body.calculator-page {
    font-family: Arial, sans-serif;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    background-color: #4A90E2;
    margin: 0;
}

.container {
    background: white;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
    width: 100%;
    max-width: 400px;
    text-align: center;
    margin-top: 0 auto;

}
.progress-container {
    margin: 15px 0;
}

.progress-bar {
    background: #ddd;
    width: 100%;
    height: 10px;
    border-radius: 5px;
```

```
}
.top-bar {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 10px;
}
.level {
    margin-right: auto;
    font-weight: bold;
}
.score{
    font-weight: bold;
}
.progress {
    background: #4CAF50;
    height: 100%;
    width: 0%;
    border-radius: 5px;
}
.problem-container {
    font-size: 35px;
    font-weight: bold;
    margin: 20px 0;
    color: #030208;
}
.answer-container {
    background: #c0d2f1;
    padding: 10px;
    height: 25px;
    border-radius: 5px;
    font-weight: bold;
    margin-bottom: 10px;
    font-size: 25px;
    color: #13097b;
}
.timer{
    color: #030208;
}
.progress-container{
    color: #030208;
    text-align: right;
}
.calculator {
    display: grid;
    grid-template-columns: repeat(4, 1fr);
    gap: 10px;
    width: 100%;
    max-width: 400px;
    margin: 0 auto;
```

```
    padding: 10px;
    box-sizing: border-box;
}
button {
    padding: 15px;
    font-size: 1.2rem;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    width: 100%;
    box-sizing: border-box;
}

.number-btn {
    background: #0052cc;
    color: white;
    font-weight: bold;
}

.number-btn:hover {
    background: #13097b;
}

.clear-btn {
    background: red;
    color: white;
    font-weight: bold;
}

.clear-btn:hover {
    background: darkred;
}

.zero {
    grid-column: span 3;
}
#submit-btn {
    background: #4CAF50;
    color: white;
    font-size: 18px;
    font-weight: bold;
    border-radius: 5px;
    grid-row: span 2;
}

#submit-btn:hover {
    background: rgb(9, 117, 46);
}

#result {
```

```
    position: fixed;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    background: white;
    padding: 30px;
    border-radius: 12px;
    box-shadow: 0 4px 10px rgba(0,0,0,0.2);
    text-align: center;
    z-index: 1000;
}
.back{
    background-color: #0052cc;
    color: white;
    padding: 10px 20px;
    border-radius: 20px;
}
.back a{
    color: white;
    text-decoration: none;
}
.hidden {
    display: none;
}
```