Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет центр післядипломної освіти
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів гейміфікації процесу онлайн навчання на прикладі розробки веб-застосунку для опанування навичок сліпого набору
(тема)

Виконав:
студент (ка) 2 курсу, групи ІПЗздм-22-1

Букатич А.В.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)

Тип програми освітньо-наукова

Керівник доцент Колесников Д.О.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет центр післядипломної освіти
Кафедра програмної інженерії
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми освітньо-наукова програма
Освітня програма Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Букатич Альоні Володимирівні
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів гейміфікації процесу онлайн навчання на прикладі розробки веб-застосунку для опанування навичок сліпого набору»
Затверджена наказом по університету від 22.04. 2024р. № 60 Стз
2. Термін подання студентом роботи до екзаменаційної комісії 17.06.2024
3. Вихідні дані до роботи приклади відкритих даних у різних форматах,реляційна база даних в форматі sql, приклади роботи з реляційними базами даних
4. Перелік питань, що потрібно опрацювати в роботі
аналіз предметної області, розробка FRONT-END частини для веб-системи автоматизації введення, розробка BACK-END частини для веб-системи автоматизації введення веб-застосунку.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання елементів роботи	Примітка
1	Аналіз предметної галузі	26.04.2024	Виконано
2	Моделі представлення системи та їх опис	06.05.2024	Виконано
3	Розробка FRONT-END частини	19.05.2024	Виконано
4	Проектування бази даних	19.05.2024	Виконано
5	Розробка BACK-END частини	02.06.2024	Виконано
6	Оформлення пояснювальної записки, додатків, графічного матеріалу	13.06.2024	Виконано
7	Нормоконтроль	14.06.2023	Виконано
8	Рецензування	15.06.2023	Виконано
9	Занесення диплому електронний архів	15.06.2023	Виконано
10	Попередній захист	15.06.2023	Виконано
11	Захист	17.06.2023	Виконано

Дата видачі завдання 26 квітня 2024 р.

Студент _____ Букатич А.В.

Керівник роботи _____ доц.кафедри ПІ Колесников Д.О.

РЕФЕРАТ / ABSTRACT

Кваліфікаційна робота магістра містить: 84с., 18 рисунок, 2 таблиці, 51 джер.

ГЕЙМІФІКАЦІЯ, ДЕСКТОП ДОДАТОК, МОТИВАЦІЯ, ОНЛАЙН ПЛАТФОРМА ДЛЯ НАВЧАННЯ, ОСВІТА, WEB-ДОДАТОК, JAVASCRIPT, REACT.

Метою роботи є створення та детальний опис розробки веб-додатка для онлайн навчання сліпої печаті.

Методи розробки базуються на таких технологіях, як JavaScript, React, NodeJS, Git.

У результаті роботи створено план та детальний опис розробки додатка, які можна використовувати як основу для створення програмної реалізації та стартап-проєкту у сфері освіти у вигляді веб-додатка для навчання.

GAMIFICATION, DESKTOP APPLICATION, ONLINE LEARNING PLATFORM, EDUCATION, WEB APPLICATION, MOTIVATION, JAVASCRIPT, REACT.

The purpose of the work is the creation and detailed description of the development of a web application for online teaching of blind printing.

Development methods are based on such technologies as JavaScript, React, NodeJS, Git.

As a result of the work, a plan and a detailed description of the development of the application were created, which can be used as a basis for creating a software implementation and a start-up project in the field of education in the form of a web application for training.

Я, Букатич Альона Володимирівна, студентка групи ІПЗздм-22-1, здобувачка вищої освіти на другому (магістерському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Гейміфікація процесу онлайн навчання на прикладі розробки веб-застосунку для опанування навичок сліпого набору», що буде представлена до ЕК для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлена з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

Вступ 8
1 Аналіз предметної області 11
1.1 Опис предметної області 11
2 Технології для створення програмного продукту 21
2.1 Фронтенд: React 21
2.2 Бекенд: Node.js та Fastify 23
2.3 База даних: MongoDB 26
2.4 Загальні рекомендації та висновки 28
3 Моделі представлення системи та їх опис 30
3.1 Модель фронтенду 30
3.2 Модель бекенду 33
3.3 Модель бази даних 34
3.4 Взаємодія між компонентами 35
3.5 Загальні рекомендації та висновки 36
4 Розробка FRONT-END частини для веб-системи 39
4.1 Дизайн і структура компонентів 39
4.2 Інтерактивність та валідація введених даних 42
4.3 Інтеграція з бекендом 42
4.4 Гейміфікація та користувацький досвід 43
4.5 Тестування та оптимізація 49
4.6 Загальні рекомендації та висновки 50
5 Проектування бази даних веб-додатку для тренування сліпого набору 58
5.1 Обґрунтування вибору бази даних для тренажера сліпої печаті 58
5.2 Альтернативні варіанти та переваги MongoDB 59
6 Розробка BackEnd частини 61
6.1 Опис основного функціоналу Back End частини веб-додатку 61
6.2 Аутентифікація та авторизація 62
6.3 Збереження даних прогресу до бази даних 63

Висновки 65
Перелік джерел посилання 66
Перелік джерел посилання науковими напрямами керівника та науковців кафедри Програмної інженерії 71
Додаток А 72
Додаток Б 73
Додаток В 81
Додаток Г 83

ВСТУП

У світі, де технології стають невід'ємною частиною нашого повсякденного життя, важливо визнати та підтримати навички, які допомагають нам легше та ефективніше взаємодіяти з цифровим середовищем. Однією з таких цінних навичок є вміння сліпої печаті.

Актуальність розвитку веб-додатків для навчання сліпої печаті очевидна. Забезпечення доступу до якісного навчання інструментами, які враховують потреби людей із різними здатностями, сприяє інклюзивності та розширює можливості самореалізації.

Проект має на меті не тільки розробку гейміфікованого веб-додатка, який вчить сліпому набору, надання користувачам інтерактивне та зручне середовище для вдосконалення цієї важливої навички, і також продумати як забезпечити доступність та інтерактивність. Сподіваюсь внести свій внесок у створення більш доступного та інклюзивного цифрового світу для всіх.

Навичка сліпого набору приносить ряд позитивних моментів, які можуть значно полегшити повсякденне життя та принести користь у різних аспектах. Декілька з них включають:

Швидкість та ефективність набору:

Люди, які володіють навичкою сліпого набору, можуть писати текст швидше та ефективніше, оскільки вони не потребують візуального спостереження за клавіатурою.

Незалежність від зору.

Сліпий набір дозволяє людям з візуальними обмеженнями або втратою зору повноцінно користуватися комп'ютерами та іншими пристроями, які вимагають введення тексту.

Робочі можливості.

Вміння сліпого набору може розширити можливості працевлаштування. На сьогодні майже всі галузі комп’ютеризовані. У багатьох вакансіях можна знайти вимоги, які пов'язані з роботою на комп'ютері та набором тексту.

Розвиток пам'яті та координації.

Вправи зі сліпим набором можуть підвищити рівень пам'яті та координації рук, оскільки вони вимагають точних рухів пальців та запам'ятовування розташування клавіш.

Ці позитивні аспекти роблять навичку сліпого набору корисною та цінною для різних груп людей, і розвиток проектів для її навчання може внести важливий внесок у підтримку різноманітності та інклюзивності.[1]

Метою цієї роботи є створення та детальний опис веб-додатку для навчанню сліпої печаті.

Для реалізації мети, у мене наступні завдання:

- провести аналітичну роботу щодо предметної галузі та аналогів бізнес ідеї;
- дослідити оптимальний стек технологій для FRONT-END та BACK-END частини
- визначити необхідні критерії для вибору бази даних для проєкта.

У результаті роботи я очікую створення повноцінного веб-додатку, який в подальшому можна буде використовувати як основу для створення програмної реалізації, а в подальшому і стартап-проєкту у сфері освіти у вигляді онлайн платформи для навчання.

Для залучення користувачів (учнів сліпого друку) я планую використати методи гейміфікації.

Методи, які я визначила як оптимальні, провівши дослідження в рамках науково-дослідній практиці: це комбінація методів: бейджи та досягнення, рівні та прогрес і елементи гри.

З часом можна бути додати стимулювання до співпраці у вигляді змагань по швидкості набору текста.

Для найкращого засвоєння інформації можна використовувати метод «інтервальних повторень» [2].

Завдання будуть створюватись таким чином, щоб у кожному новому використовувати знання, здобуті у попередньо пройдених завданнях.

Подробиці і результати дослідження будуть показані у даній роботі в окремому розділі.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Опис предметної області

Сліпий набір, або техніка набору тексту, коли людина може вводити текст, не дивлячись на клавіатуру, є важливою для підвищення ефективності та швидкості роботи з комп'ютером.[3]

Процес навчання сліпому набору може бути виснажливим та потребує великої кількості тренувань. У цьому контексті гейміфікація, тобто використання елементів гри в неігрових сценаріях, може стати ефективним методом для зацікавлення студентів та стимулювання їхнього прогресу в опануванні навичок сліпого набору.[4]

Розробка веб-застосунку для гейміфікованого онлайн навчання може об'єднати інноваційні методи навчання з цифровими технологіями та гейміфікацією. Персоналізовані завдання, візуалізація прогресу, використання бейджів та лідербордів можуть надати користувачам додатковий стимул для систематичного навчання та вдосконалення навичок сліпого набору.[5]

Такий веб-застосунок може бути корисним для широкого кола користувачів, оскільки навички сліпого набору стають універсальними та корисними в навчанні та професійному житті. Гейміфікація дозволяє зробити процес навчання цікавим та залучливим, стимулюючи студентів до досягнення високих результатів у опануванні цих навичок.

У сучасному суспільстві високотехнологічної інформаційної епохи, володіння навичками сліпого набору стає ключовим елементом успіху для багатьох людей. Незалежно від сфери діяльності, вміння швидко та ефективно вводити текст може значно підвищити продуктивність та комфорт користувача.

Розглянемо декілька прикладів існуючих веб-застосунків для навчання сліпому друку. Виявимо позитивні і негативні моменти, а також визначимо, які моменти можна запозичити, а яких краще уникати для кращого досвіду і залученості користувачів при створенні нашого гейміфікованого додатку.

Спочатку розглянемо додатки у функціоналі яких присутня можливість навчання сліпому друку українською:

Ratatype (див.рис.1.1 та рис.1.2).

Сервіс Ratatype позиціонується як традиційний тренажер з простим і зрозумілим інтерфейсом. Проте, щоденне набирання символів може стати монотонним, тому платформа рекомендує час від часу пройти тест на швидкість друку (з успішними результатами надсилається сертифікат). Крім того, ви можете випробувати свої сили в змаганнях: на сайті присутня таблиця досягнень інших учасників, а також можливість брати участь у групових змаганнях на швидкість набору тексту.

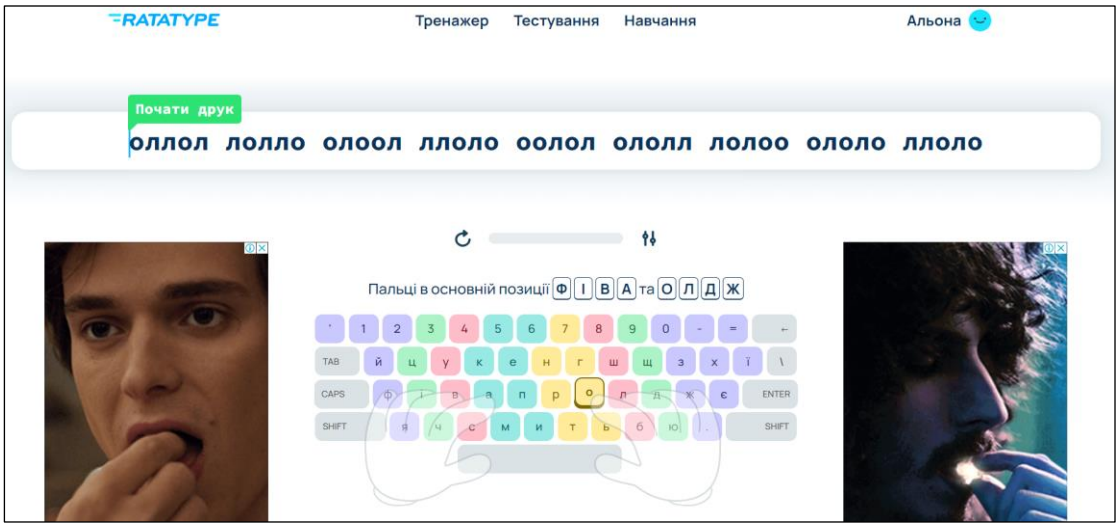

Рисунок 1.1 – Сервіс Ratatype виглядає непогано, але забагато реклами

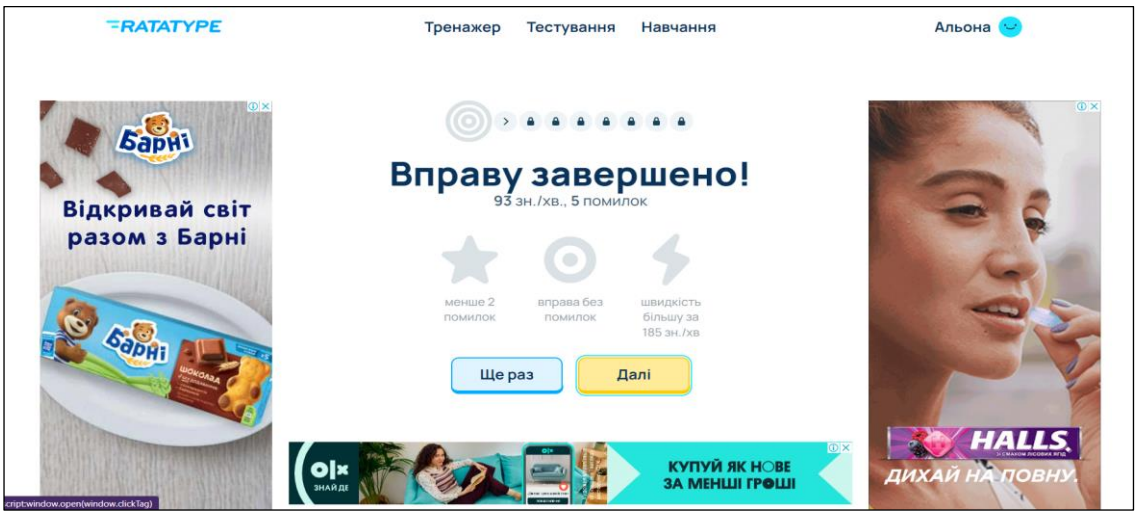

Рисунок 1.2 – Сервіс Ratatype мотивація швидкого і якісного виконання

Позитивні моменти:

- зрозумілий інтерфейс;
- можливість створювати групи;
- тестування поточного рівня друку наосліп (вимірюється кількістю надрукованих слів у хвилину);
- можливість отримати сертифікат;
- мотивація проходити завдання без помилок і якомога швидше.

Негативні моменти:

- забагато реклами у безкоштовній версії;
- немає елементів гри;
- незрозуміла програма. Заявлено про те, що для того, щоб навчитись потрібно 19 занять. Але перехід відбувається автоматично і невідомо що буде у наступному уроці;
- у початкових завданнях задіяна тільки 1 рука.

Touch Typing Study (див.рис. 1.3, 1.4).

Навчитися сліпому набору тексту пропонується за 15 уроків, кожен з яких розкладений на вправи. Кожен урок створений так, що спочатку вивчаються нові клавіші, потім з ними набираються слова, а в кінці перехід до друкування тексту. Крім того, в уроках містяться короткі підказки.

Паралельно з виконанням уроків можна складати тести на швидкість, користуватися набором випадкового тексту та грати у ігри, пов'язані з друкуванням. Також є можливість додатково вивчити набір на цифровій клавіатурі (блок з цифрами справа).

Позитивні моменти:

- тестування на швидкість;
- можливість грати в ігри;
- статистика по кожній букві.

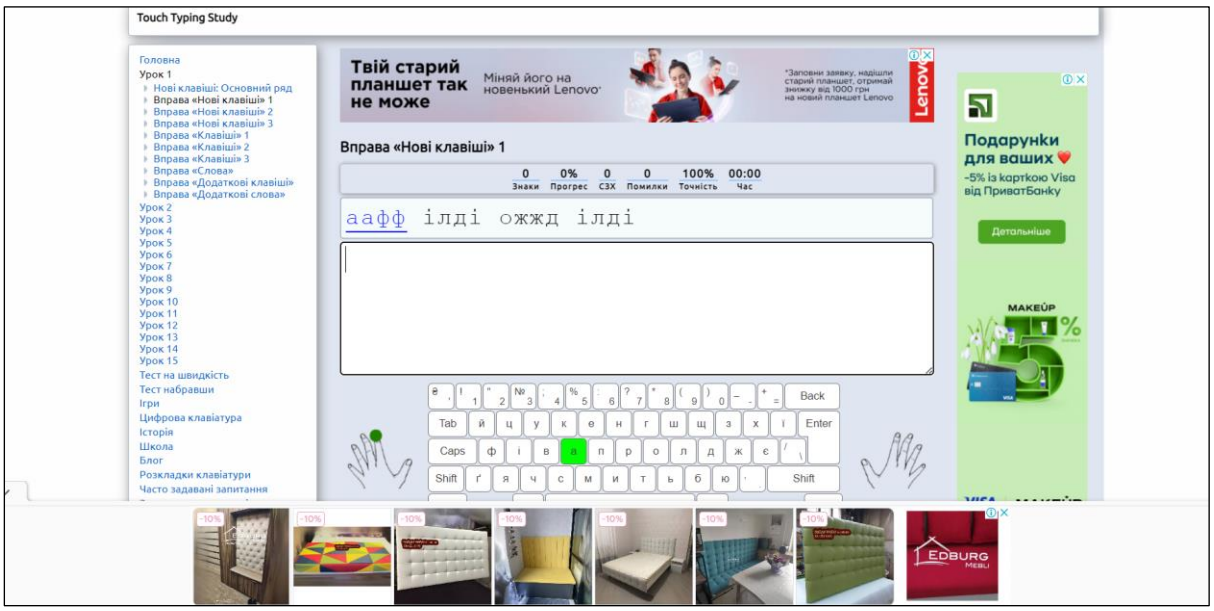

Рисунок 1.3 – Touch Typing Study непривабливий інтерфейс

Рисунок 1.4 – Touch Typing Study статистика по кожній літері

Негативні моменти:

- непривабливий інтерфейс;
- багато реклами;
- ігри зроблені окремими розділами;
- об’ємні уроки;
- новий текст з’являється і немає розуміння, коли урок закінчиться;
- багато літер вже у першому уроці (8 літер).

Прикладів додатків в англомовному сегменті (немає української) набагато більше. Розглянемо декілька з них.

TypingClub – це безкоштовний веб-сервіс для вивчення швидкого набору тексту. Він пропонує мобільні програми та інструменти, які допомагають вчителям планувати уроки (див.рис.1.5, 1.6).

Пропонує різні рівні, щоб підтримувати мотивацію та розвивати м’язову пам’ять.

Рисунок 1.5 – Typing club. Вигляд карток завдань у додатку

Рисунок 1.6 – Typing club. Інтерфейс гри

Позитивні моменти:

- привабливий і зрозумілий інтерфейс;
- є розподіл на блоки для вивчення розташування літер;
- виділяється декілька завдань на вивчення одних і тих же літер. Є час вивчити розташування клавиш;
- ігри після проходження кожного блоку завдань;
- система навчання починаючи з основних пальців (F та J). У кожному завданні задіюються пальці обох рук.;
- є можливість увімкнути і вимкнути показ клавіатури. Можна обрати стиль клавіатури;
- мотивує проходити без помилок, щоб отримати більше зірок, які вказуються на прев’ю кожного завдання;
- є бейджи.

Негативні моменти:

- немає української;
- платний сервіс;
- є повторення, але через занадто маленький проміжок часу.

Typing Cat – це тренажер для поглиблення навички набору тексту. Пропонує різноманітні методики та типи розкладок клавіатури, щоб допомогти покращити швидкість друку та зручність роботи з клавішами комп'ютера (див.рис.1.7, 1.8).

Позитивні моменти:

- структура занять по якій зрозуміло чого чекати в занятті;
- зрозумілий інтерфейс;
- ігри;
- початковий тест;

- система навчання починаючи з основних пальців (F та J). У кожному завданні задіюються пальці обох рук;
- додаткові рівні;
- статистика всередині самого завдання;
- немає реклами;
- хоча курс платний, коштує він дешево.

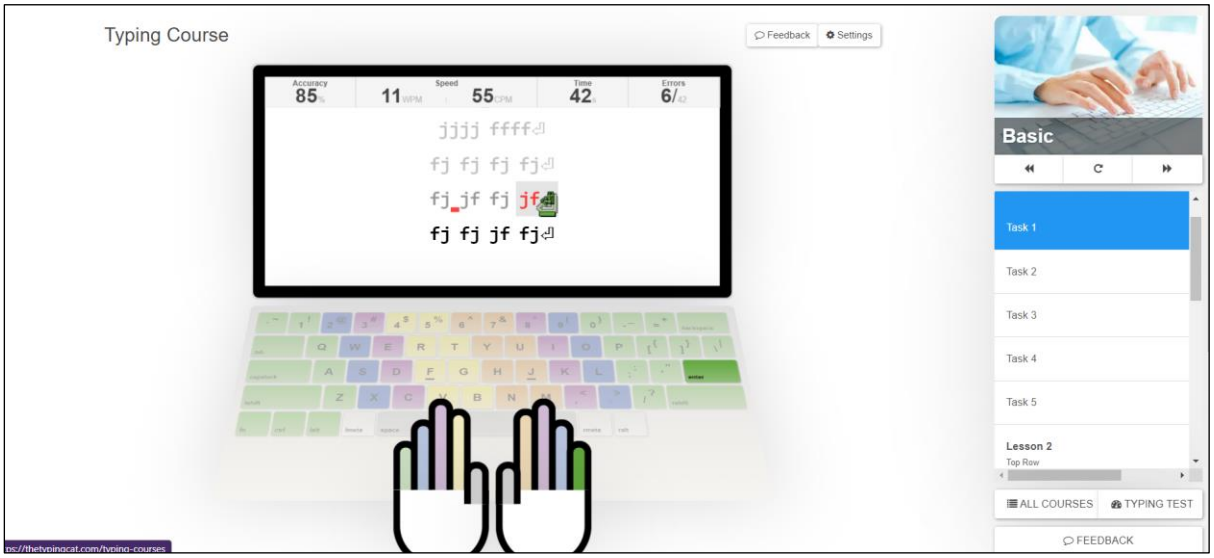

Рисунок 1.7 – Thetypingcat інтерфейс тренажера

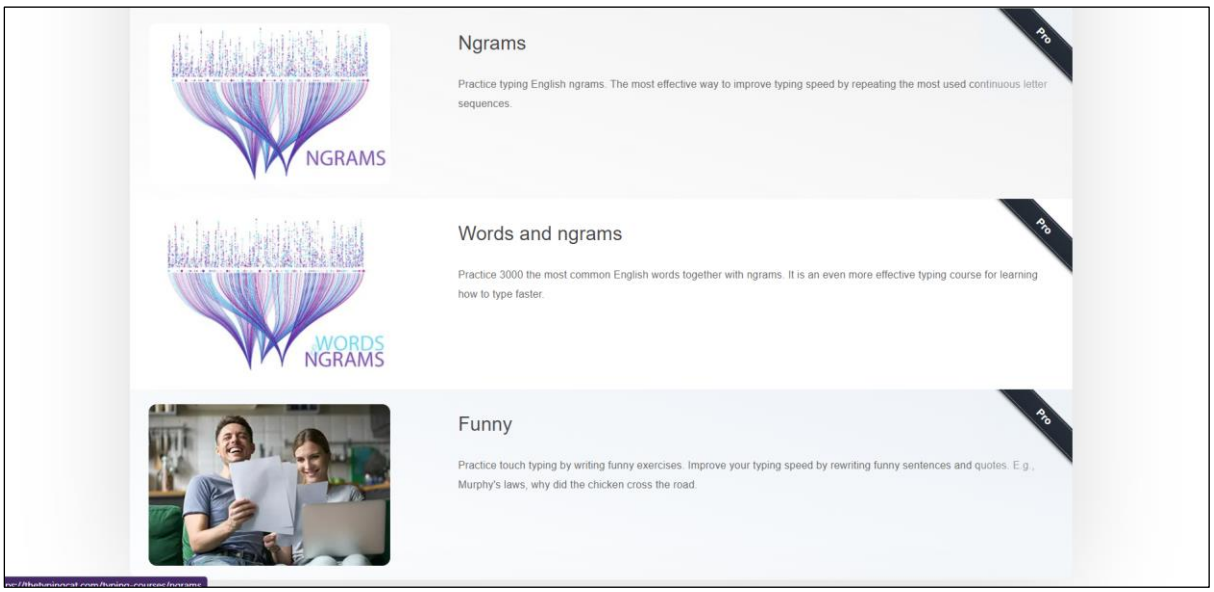

Рисунок 1.8 – Thetypingcat додаткові рівні тренажера

Негативні моменти:

- немає розуміння наскільки затягнеться урок, текст видно одразу не весь. Він поступово додається;
- немає ігрових елементів у базовому курсі. Ігри тільки для тих, хто пройшов базовий курс;
- занадто швидкі переходи до наступних літер;
- курс платний.

Vehicles Typing – це інструмент для набору тексту. Потрібно просто вводити літери у повітряних кульках. Тоді транспорт продовжуватиме працювати (див.рис.1.9).

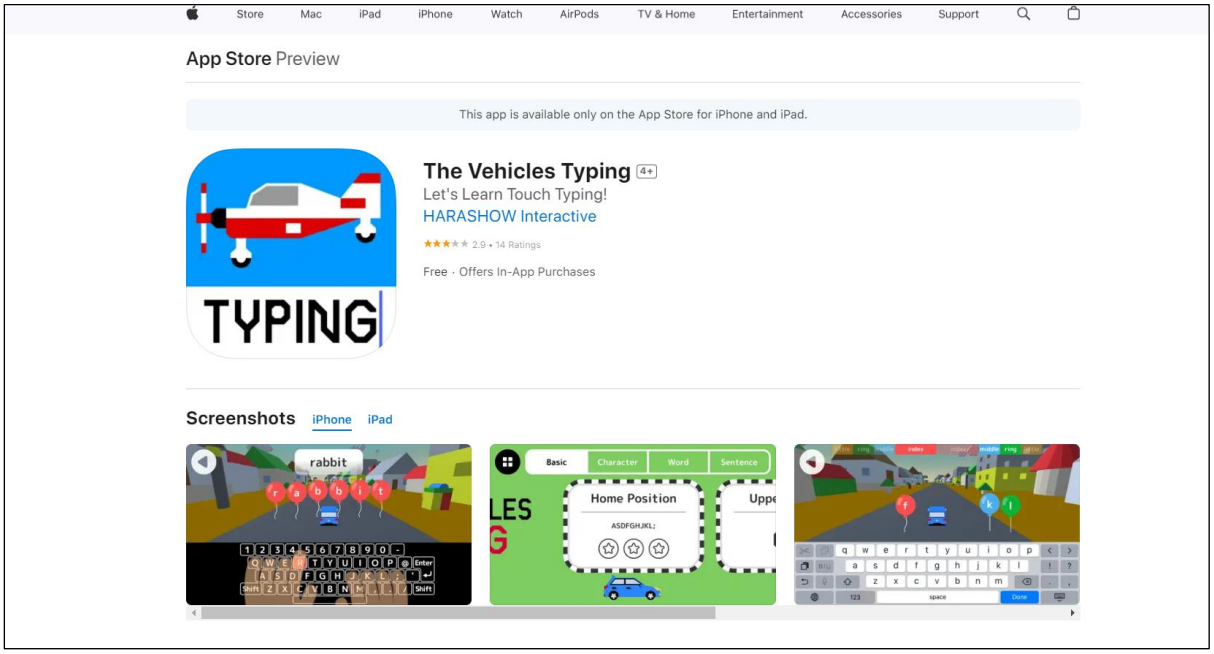

Рисунок 1.9 – The Vehicles Typing додаткові рівні тренажера

Позитивні моменти:

- пропонується більше тридцяти уроків;
- представлений у вигляді гри;
- анімація різних транспортних засобів, таких як автомобілі, поїзди, літаки, кораблі тощо;
- немає сторонньої реклами.

Негативні моменти:

- підтримувана платформа тільки iOS;
- завдання занадто прості, дорослим швидко може набриднути;
- не дуже високий рейтинг в App Store.

Порівняльний аналіз додатків виявив наступні моменти.

Додатків для навчання сліпому друку українською зовсім мало. В тих що наявні немає елементів гри у процесі навчання. Немає також підходу в якому з першого завдання залучаються обидві руки. На клавіатурі спеціально виділені дві клавіші (А та О) для того, щоб можна було знайти їх, навіть не дивлячись на клавіатуру. Це база, з якої потрібно починати навчання сліпому друку.

В додатку, який я розробляю все навчання побудовано саме на цьому. Спочатку потрібно міцно запам’ятати початкову позицію, щоб мати можливість рухатись далі.

При сліпому друку кожен палець відповідає за конкретну область. Рухаючись пальцями по відповідним областям ми завжди повертаємось у початкову точку (див.рис.1.10).

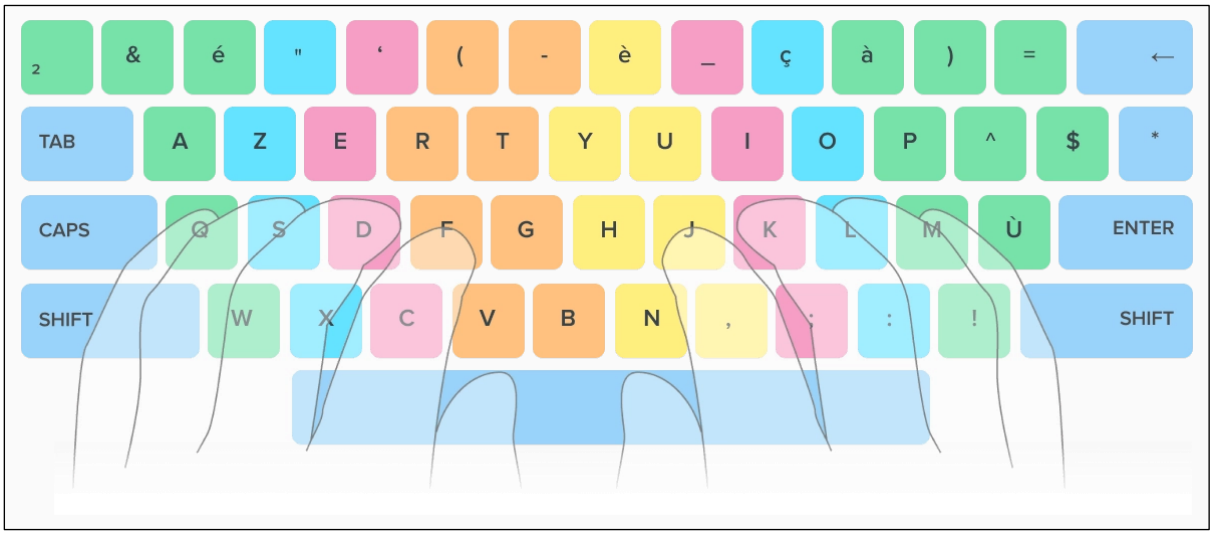

Рисунок 1.10 – Початкова позиція пальців. Відповідні області

Додаток Ratatype виглядає дуже добре. Однак, є моменти які можливо покращити у майбутньому додатку:

- зробити прев'ю карток завдань і відображати на них якість проходження (як у Typing Club);
- зробити декілька завдань для опанування літер (окремий блок для кожної пари);
- додати елемент гри – після проходження блоку влаштовувати ігру на швидкість набору вивчених літер;
- не додавати безліч реклами. Замість цього зробити привабливий платний функціонал;
- підсвічувати кольорову область на клавіатурі (підказці) під час виконання завдання;
- продумати систему повторень, щоб відтворити метод інтервальних повторень;
- стежити за статистикою неправильного набору літер. І, при збільшенні помилок на окремих літерах, пропонувати повертатись до занять з цими літерами, щоб відновити навичку;
- створити елемент суперництва у вигляді гри на швидкість набору тексту онлайн з іншими користувачами.

Таким чином, основною метою навчальної системи з гейміфікацією є покращення якості роботи програмістів. Це досягається шляхом створення цікавих та захоплюючих завдань з програмування, оформлених у вигляді ігор. Цільова аудиторія додатка з навчання сліпому друку – працівники технічних компаній, які прагнуть удосконалити свої знання та навички в програмуванні. Для досягнення цієї мети розглядається використання різних гейміфікаційних підходів і методів у навчальній системі, включаючи вибір та інтеграцію відповідних гейміфікаційних елементів [6].

Використання балів і досягнень за виконання конкретних цілей та завдань. Наприклад, учні можуть отримувати зірочки за успішне завершення завдань певної складності або за досягнення визначеного рівня знань у певній темі. Це сприяє підвищенню мотивації та підтримці інтересу учнів до навчання [7].

У перспективі використання історій та квестів дозволить користувачам зануритися у світ набору тексту методом сліпого друку. Користувачі можуть проходити квести та виконувати завдання, які допомагатимуть їм дізнатися більше про те, як ефективніше і вправніше набирати текст [8].

2 ТЕХНОЛОГІЇ ДЛЯ СТВОРЕННЯ ПРОГРАМНОГО ПРОДУКТА

Розділ "Технології для створення програмного продукту" відіграє ключову роль у розробці будь-якого програмного застосунку.

У цьому розділі розглянемо вибір технологій для фронтенду та бекенду для проєкту з гейміфікації процесу онлайн навчання.

2.1. Фронтенд: React

Однією з ключових технологій для створення користувацького інтерфейсу веб-застосунку є бібліотека React.

React дозволяє ефективно створювати інтерактивні та швидкі користувацькі інтерфейси, спрощуючи розробку компонентів та їх взаємодію. Використання React забезпечить ефективне управління станом, а також зручну організацію коду.

Альтернативами React можуть бути Angular та Vue.js. Однак враховуючи простоту вивчення, широкий спектр готових компонентів та велику активність спільноти, я вибрала React як оптимальний інструмент для фронтенд розробки.

React – це потужний інструмент для створення сучасних веб-додатків, і він має багато переваг, які роблять його ідеальним вибором для розробки додатків для навчання. Ось основні позитивні сторони React у цьому контексті.

Компонентна архітектура. Однією з головних переваг React є його компонентна архітектура. Вона дозволяє розбивати додаток на окремі, незалежні компоненти, які можна повторно використовувати та комбінувати. Для навчального додатку це означає можливість створення модульних блоків, таких як тести, відеоуроки, інтерактивні завдання тощо. Це спрощує розробку, тестування та підтримку додатку.

Висока продуктивність. React використовує віртуальний DOM, який мінімізує кількість маніпуляцій з реальним DOM і підвищує продуктивність додатку. Це особливо важливо для навчальних додатків, де швидкість відгуку та плавність взаємодії є критичними для забезпечення гарного користувацького досвіду.

Багата екосистема та підтримка. React має розвинену екосистему, яка включає в себе велику кількість бібліотек та інструментів, що можуть бути корисними для створення навчальних додатків. Наприклад, бібліотеки для керування станом (Redux, MobX), роутінгу (React Router), а також безліч UI-компонентів (Material-UI, Ant Design). Це дозволяє швидко інтегрувати потрібний функціонал і скоротити час розробки.

Відмінна документація та велика спільнота. React має детальну та добре структуровану документацію, яка допомагає швидко освоїтися з інструментом навіть новачкам. Крім того, велика та активна спільнота розробників постійно ділиться своїм досвідом, створює навчальні матеріали та підтримує новачків. Це особливо корисно при розробці навчальних додатків, де часто виникає потреба в інтеграції нових методик і технологій.

Кросплатформеність. Завдяки React Native, додаток, створений на основі React, можна легко адаптувати для мобільних платформ (iOS та Android). Це дозволяє охопити ширшу аудиторію учнів, які можуть використовувати додаток на різних пристроях. Навчальні додатки стають доступними скрізь – як на комп'ютерах, так і на смартфонах та планшетах.

Легке оновлення та підтримка. Завдяки модульній структурі та односпрямованому потоку даних, React-додатки легко оновлювати і підтримувати. Це дозволяє швидко вносити зміни в навчальний контент, додавати нові функції або виправляти помилки, не порушуючи роботи інших частин додатку.

Інтерактивність та динамічність. React дозволяє створювати високодинамічні та інтерактивні інтерфейси, що є важливим для навчальних додатків. Учні можуть взаємодіяти з контентом у реальному часі, виконувати інтерактивні завдання, проходити тести та отримувати миттєвий зворотний зв'язок. Це підвищує залученість і мотивацію до навчання.

Легкість інтеграції з іншими технологіями. React легко інтегрується з іншими технологіями та інструментами, такими як GraphQL для управління даними, Firebase для бекенду, або навіть з існуючими системами управління контентом

(CMS). Це дозволяє створювати комплексні навчальні рішення з мінімальними витратами часу і ресурсів.

Адаптивний дизайн. Завдяки таким бібліотекам, як Styled Components або CSS-in-JS, React дозволяє легко створювати адаптивний дизайн, який забезпечує гарний вигляд додатку на різних пристроях та розмірах екрану. Це критично важливо для навчальних додатків, де користувачі можуть працювати з різних пристроїв.

Постійний розвиток та інновації. React активно розвивається та підтримується Facebook і спільнотою. Постійні оновлення та впровадження нових можливостей дозволяють розробникам використовувати найсучасніші технології та підходи, що забезпечує високу конкурентоспроможність додатку на ринку.

Усі ці переваги роблять React чудовим вибором для створення ефективних, інтерактивних та зручних навчальних додатків, що сприяють кращому засвоєнню матеріалу та покращують досвід користувачів.

2.2 Бекенд: Node.js та Fastify

Для реалізації серверної частини веб-застосунку вибір пав на Node.js та Fastify.[9]

Node.js є високопродуктивним середовищем виконання JavaScript, що дозволяє розробникам використовувати ту ж мову програмування на фронтенді та бекенді, що полегшує взаємодію між цими частинами проєкту.

Fastify, який є швидким та легким веб-фреймворком для Node.js, обрано через його високу продуктивність та низький рівень обміну ресурсами.

Альтернативами можуть бути NestJS або Koa, але у зв'язку з особливостями нашого проекту, швидкість Fastify виявляється більш оптимальною.

Node.js та Fastify – це потужна комбінація для реалізації серверної частини додатків, зокрема й для навчальних додатків. Цей тандем забезпечує високу продуктивність, простоту в розробці та багатофункціональність. Розглянемо основні переваги вибору Node.js та Fastify для серверної частини навчального додатку.

Висока продуктивність. Node.js побудований на основі движка V8 від Google, який забезпечує високу швидкість виконання JavaScript-коду. Його асинхронна, неблокуюча архітектура дозволяє обробляти велику кількість одночасних запитів, що є критичним для додатків з високим навантаженням. Fastify, у свою чергу, спеціально оптимізований для максимальної продуктивності та низької латентності, що робить його ідеальним вибором для швидкодіючих веб-серверів.

Простота в розробці та підтримці. Node.js використовує JavaScript – мову програмування, яка популярна серед фронтенд-розробників. Це дозволяє використовувати єдину мову на всьому стеку розробки, що значно спрощує процес розробки та підтримки додатку. Fastify має простий та зрозумілий API, що дозволяє швидко налаштовувати та розширювати серверну частину, зменшуючи витрати часу на навчання та розробку.

Розширюваність та модульність. Fastify підтримує модульну архітектуру, що дозволяє легко додавати та видаляти плагіни відповідно до потреб додатку. Це робить його дуже гнучким та адаптивним до змін у функціональності. Node.js має велику кількість пакетів і модулів у своєму екосистемі (npm), що дає можливість швидко інтегрувати додаткові функції та сервіси, необхідні для навчального додатку.

Висока масштабованість. Завдяки неблокуючій моделі вводу/виводу Node.js, серверна частина додатку може ефективно обробляти велику кількість одночасних запитів, що є ключовим для масштабування. Fastify додатково забезпечує високу продуктивність та низьку латентність, що дозволяє легко масштабувати додаток для підтримки збільшення кількості користувачів та запитів.

Безпека. Безпека є критичним аспектом для будь-якого додатку, особливо для навчальних додатків, де обробляються особисті дані користувачів. Fastify забезпечує вбудовані механізми для захисту від відомих вразливостей, таких як атаки типу XSS та CSRF. Node.js також підтримує численні бібліотеки та плагіни для забезпечення безпеки додатків, що дозволяє створювати надійні та безпечні серверні рішення.

Підтримка REST та GraphQL. Node.js та Fastify дозволяють легко реалізувати RESTful API або GraphQL API, що забезпечує гнучкість у побудові архітектури додатку. Це дозволяє створювати масштабовані та легко розширювані серверні частини, які можуть інтегруватися з різноманітними фронтенд-технологіями та мобільними додатками.

Інтеграція з базами даних. Node.js забезпечує підтримку для багатьох баз даних, таких як MongoDB, MySQL, PostgreSQL, що дозволяє вибрати оптимальне рішення для зберігання даних навчального додатку. Fastify має плагіни для легкої інтеграції з різними базами даних, що спрощує роботу з даними та забезпечує високий рівень продуктивності.

Підтримка реального часу. Завдяки WebSocket та бібліотекам, таким як Socket.io, Node.js дозволяє реалізувати функціонал в реальному часі, що може бути корисним для інтерактивних навчальних додатків, де потрібна миттєва комунікація між користувачами або миттєві оновлення контенту.

Велика спільнота та підтримка. Node.js має велику та активну спільноту розробників, які постійно розвивають та вдосконалюють інструменти та бібліотеки. Fastify також набуває популярності завдяки своїй продуктивності та простоті. Це забезпечує доступ до великої кількості ресурсів, навчальних матеріалів та підтримки, що спрощує процес розробки та вирішення можливих проблем.

Економія ресурсів. Node.js дозволяє використовувати JavaScript на всьому стеку технологій, що зменшує потребу у вивченні та використанні різних мов програмування. Це економить ресурси на навчання нових розробників та спрощує процес розробки. Fastify, завдяки своїй продуктивності, зменшує потребу у використанні потужного апаратного забезпечення, що також сприяє економії ресурсів.

Усі ці переваги роблять Node.js та Fastify відмінним вибором для реалізації серверної частини навчального додатку, забезпечуючи високу продуктивність, гнучкість, безпеку та масштабованість.

2.3. База даних: MongoDB

Для зберігання даних обрали MongoDB. Це база даних, яка добре підходить для проектів з розвитком схеми та неструктурованими даними. MongoDB пропонує гнучкість та високу масштабованість, що важливо для динамічних систем гейміфікації.[10]

Альтернативами можуть бути PostgreSQL або MySQL, але MongoDB дозволяє ефективно працювати з різними форматами даних, що є ключовим для нашого проекту.

MongoDB – це потужна та гнучка NoSQL база даних, яка ідеально підходить для зберігання даних у навчальних додатках. Завдяки своїй продуктивності, масштабованості та гнучкості, MongoDB забезпечує ефективне управління та обробку великих обсягів даних. Ось основні переваги вибору MongoDB для зберігання даних у додатку для навчання.

Гнучка модель даних. MongoDB використовує документно-орієнтовану модель даних, що дозволяє зберігати дані у форматі BSON (Binary JSON). Ця модель забезпечує високу гнучкість у структурі даних, дозволяючи зберігати складні ієрархічні дані без необхідності у попередньому визначенні схеми. У навчальних додатках це означає можливість зберігати різнорідні дані, такі як інформація про курси, матеріали, оцінки та профілі користувачів, у зручному форматі.

Масштабованість. MongoDB підтримує горизонтальне масштабування за допомогою шардінгу, що дозволяє розподіляти дані на кілька серверів для підвищення продуктивності та надійності. Це особливо важливо для навчальних додатків, де кількість користувачів та обсяг даних можуть швидко зростати. Масштабованість MongoDB забезпечує стабільну роботу додатку навіть при значному навантаженні.

Висока продуктивність. Завдяки індексації, агрегації та оптимізації запитів, MongoDB забезпечує високу продуктивність при роботі з великими обсягами даних. Це дозволяє швидко отримувати доступ до необхідної інформації,

обробляти великі масиви даних та виконувати складні запити. У навчальних додатках це забезпечує швидку взаємодію користувачів з системою та миттєву реакцію на їх дії.

Легка інтеграція з JavaScript та Node.js. MongoDB чудово інтегрується з JavaScript та Node.js, що дозволяє використовувати єдину мову програмування на всьому стеку технологій. Це спрощує розробку та підтримку додатку, знижуючи витрати на навчання нових розробників. MongoDB також має офіційний драйвер для Node.js, який забезпечує ефективну та зручну роботу з базою даних.

Підтримка агрегаційних фреймворків. Агрегаційний фреймворк MongoDB дозволяє виконувати складні обчислення та аналіз даних безпосередньо у базі даних. Це особливо корисно для навчальних додатків, де часто потрібно генерувати звіти, аналізувати успішність учнів, проводити статистичні дослідження та інше. Використання агрегаційного фреймворку знижує навантаження на сервер додатку та покращує продуктивність.

Вбудована підтримка геопросторових даних. MongoDB підтримує зберігання та обробку геопросторових даних, що дозволяє легко реалізовувати функціонал, пов'язаний з геолокацією. У навчальних додатках це може бути використано для аналізу географічного розподілу учнів, організації локальних навчальних заходів або відслідковування відвідуваності.

Надійність та відмовостійкість. MongoDB забезпечує високий рівень надійності та відмовостійкості завдяки реплікації даних. Реплікаційні набори MongoDB автоматично створюють копії даних на кількох серверах, що дозволяє забезпечити безперервну роботу додатку навіть у разі відмови одного з серверів. Це критично важливо для навчальних додатків, де доступ до даних повинен бути постійно доступним.

Безпека. MongoDB підтримує численні механізми безпеки, включаючи аутентифікацію, авторизацію, шифрування даних на диску та в мережі. Це дозволяє забезпечити захист конфіденційних даних користувачів та відповідність вимогам різних стандартів безпеки. У навчальних додатках це особливо важливо для захисту особистої інформації учнів та викладачів.

Підтримка транзакцій. З версії 4.0 MongoDB підтримує багатодокументні транзакції, що дозволяє виконувати кілька операцій як одну атомарну операцію. Це забезпечує консистентність даних та підвищує надійність додатку. У навчальних додатках це може бути корисно для забезпечення цілісності даних при проведенні одночасних операцій, таких як реєстрація на курси або оплата навчання.

Велика спільнота та активна розробка. MongoDB має велику та активну спільноту розробників, яка постійно розширює можливості бази даних та ділиться своїм досвідом. Це забезпечує доступ до численних ресурсів, навчальних матеріалів та інструментів, що спрощує процес розробки та підтримки додатку. Активна розробка MongoDB гарантує регулярні оновлення та вдосконалення бази даних.

Завдяки цим перевагам, MongoDB є відмінним вибором для зберігання даних у навчальних додатках, забезпечуючи високу продуктивність, гнучкість, надійність та безпеку.

2.4. Загальні рекомендації та висновки

Хоча існують альтернативні технології для фронтенду, бекенду та бази даних, обрані React, Node.js (Fastify) та MongoDB оптимально взаємодіють між собою, забезпечуючи високу продуктивність та ефективність в розробці.

При виборі технологій важливо враховувати особливості проекту, його масштаб та вимоги.

Вибір технологій, які краще відповідають конкретним завданням проєкту, є ключовим етапом розробки, що може визначити успіх у подальшому.

Також, я маю великий досвід роботи з вибраними технологіями, що теж сприяло їх вибору.

Вибір технологій React, Node.js (разом з Fastify) та MongoDB є оптимальним для розробки навчальних додатків завдяки їх продуктивності, масштабованості, гнучкості та зручності розробки. Ці технології забезпечують ефективну взаємодію між фронтендом та бекендом, дозволяючи створювати інтерактивні, динамічні та надійні навчальні платформи.

Комбінація цих інструментів створює потужний та злагоджений стек, який підтримує сучасні вимоги до навчальних додатків, забезпечуючи відмінний користувацький досвід та ефективне управління даними.

Разом ці технології створюють потужний та злагоджений стек, який забезпечує безшовну інтеграцію між фронтендом і бекендом. React дозволяє створювати динамічні та інтерактивні інтерфейси користувача, які взаємодіють з серверною частиною, реалізованою на Node.js з використанням Fastify. Серверна частина забезпечує обробку запитів та управління даними, що зберігаються в MongoDB. Використання єдиної мови програмування (JavaScript) на всьому стеку технологій спрощує процес розробки та зменшує ризик помилок при передачі даних між різними частинами додатку.

Вибір технологій React, Node.js (разом з Fastify) та MongoDB для створення навчального додатку є оптимальним завдяки їх взаємодії та синергії, що забезпечує високу продуктивність, масштабованість та зручність розробки. Цей технологічний стек надає потужний інструментарій для розробки, підтримки та розширення функціональності додатку, забезпечуючи відмінний користувацький досвід.

3 МОДЕЛІ ПРЕДСТАВЛЕННЯ СИСТЕМИ ТА ЇХ ОПИС

Розділ "Моделі представлення системи та їх опис" визначає ключові аспекти архітектури та взаємодії всіх компонентів в системі гейміфікації онлайн навчання. У цьому розділі ми розглянемо основні моделі представлення системи та надамо їх детальний опис.

При розробці програмного продукту, включаючи гейміфікованого додатку,, вибір відповідних інструментів програмної реалізації є одним із ключових аспектів. Обрані інструменти впливають не лише на ефективність і продуктивність розробки, але й на якість, зручність та функціональність кінцевого продукту.

При виборі інструментів програмної реалізації слід враховувати такі фактори:

- функціональність: Інструменти повинні відповідати потребам проекту та забезпечувати необхідні можливості. Наприклад, якщо проект вимагає використання фронтенд-фреймворка, важливо обрати той, який дозволяє створювати інтерактивні інтерфейси [11];
- відкритість та підтримка: Обрані інструменти повинні мати активну спільноту розробників та бути підтримуваними. Це забезпечить можливість отримати допомогу при необхідності, а також доступ до актуальних версій, виправлень і нових функцій;
- досвід команди: Необхідно враховувати досвід та навички команди розробників. Якщо команда вже має експертів з певними інструментами, це може стати перевагою при їх виборі [12].

3.1 Модель фронтенду

Модель фронтенду включає в себе весь користувацький інтерфейс, який забезпечує спілкування користувача з системою. Застосунок розроблений з використанням React, що дозволяє створювати компоненти для кожної функціональної частини.

Компоненти, такі як вправи для сліпого друку, панель досягнень та статистика, впроваджені для забезпечення інтерактивного та зручного досвіду для користувачів.

Модель фронтенду для навчальних додатків, зокрема для додатків для навчання сліпому друку, зосереджується на створенні інтуїтивно зрозумілого, інтерактивного та адаптивного інтерфейсу.

Важливими аспектами є гнучкість у відображенні навчальних матеріалів, миттєвий зворотний зв'язок та ефективна взаємодія з бекендом для збереження та отримання даних. Вибір сучасних технологій та підходів до розробки забезпечує високу продуктивність та зручність користувацького досвіду, що є ключовими факторами для успіху навчальних додатків.

Модель фронтенду охоплює архітектурні та дизайнерські аспекти, які визначають, як користувач взаємодіє з додатком.

Фронтенд – це частина додатку, яку бачить і з якою взаємодіє користувач. Основні компоненти моделі фронтенду для таких додатків включають наступні частини.

Інтерфейс користувача (UI). Це візуальна частина додатку, яка включає всі елементи, з якими взаємодіє користувач:

- головний екран: Містить меню, вибір курсів;
- навчальні екрани: Екрани, на яких відбувається безпосередньо навчання сліпому друку. Вони включають текст для введення, клавіатурні підказки та індикатори швидкості;
- екран статистики: Відображає результати навчання, включаючи швидкість друку, кількість помилок, прогрес користувача.

Користувацький досвід (UX). Це аспект фронтенду, який визначає, наскільки зручно та ефективно користувач може взаємодіяти з додатком:

- навігація: Легка та інтуїтивно зрозуміла навігація між різними екранами та функціями додатку;

- відповідь додатку: Швидкість реакції на дії користувача, забезпечення плавного та безперебійного досвіду;
- інтерактивність: Використання анімацій, підказок та інших інтерактивних елементів, які допомагають користувачеві освоїти матеріал.

Функціональність та компоненти. Різні компоненти та функціональність, що реалізуються на фронтенді:

- текстові завдання: Тексти для введення, які адаптуються під рівень користувача;
- клавіатурні підказки: Візуалізація клавіатури з підсвіткою поточної клавіші для натискання;
- системи підказок: Включають підказки для розміщення рук, правильного натискання клавіш тощо;
- зворотний зв'язок: Миттєвий зворотний зв'язок про помилки та успіхи користувача, що допомагає коригувати навчальний процес.

Технології. Фронтенд додатку для навчання сліпому друку зазвичай створюється з використанням сучасних веб-технологій:

- JavaScript/TypeScript: Для інтерактивності та динамічних змін в інтерфейсі;
- фреймворки та бібліотеки: React для побудови модульних та масштабованих інтерфейсів.

Взаємодія з бекендом. Фронтенд додатку для навчання взаємодіє з серверною частиною для отримання та відправки даних:

- запити до сервера: Для отримання навчальних матеріалів, збереження прогресу користувача, отримання статистики;
- обробка даних: Фронтенд обробляє дані, отримані з сервера, і відображає їх у зрозумілій формі.

Адаптивність. Забезпечення адаптивного дизайну, який добре працює на різних пристроях (десктопи, планшети, мобільні телефони), дозволяє користувачам навчатися у зручний для них спосіб.

3.2. Модель бекенду

Модель бекенду використовує Node.js та Fastify для забезпечення серверної логіки. Сервер відповідає за обробку запитів фронтенду, здійснення взаємодії з базою даних MongoDB та надання необхідних ресурсів. Модель бекенду також включає в себе алгоритми гейміфікації, які визначають правила гри, нагороди та рівні складності для ефективного стимулювання навчання.

Вибір технологій Node.js, Fastify та MongoDB для реалізації бекенду є оптимальним рішенням для створення сучасних, масштабованих та ефективних додатків. Цей технологічний стек надає численні переваги, які сприяють зручній розробці, високій продуктивності та гнучкості системи. Розглянемо детальніше позитивні моменти вибору цих технологій для моделі бекенду.

Node.js є платформою на основі JavaScript, яка дозволяє виконувати серверний код швидко та ефективно.

Fastify – це високопродуктивний веб-фреймворк для Node.js, який надає можливості для швидкої та ефективної розробки бекенд-частини додатку.

MongoDB є документно-орієнтованою NoSQL базою даних, яка ідеально підходить для зберігання та управління даними у динамічних та масштабованих додатках.

Комбінація Node.js, Fastify та MongoDB створює потужний та злагоджений бекенд-стек, який забезпечує ефективну взаємодію між компонентами системи.

Безпека та надійність. Node.js та Fastify забезпечують надійні механізми для захисту додатку, включаючи аутентифікацію, авторизацію та захист від поширених веб-загроз. MongoDB надає засоби для шифрування даних, контролю доступу та реплікації, що забезпечує захист та цілісність даних.

Висновок. Вибір Node.js, Fastify та MongoDB для реалізації бекенду є оптимальним завдяки їх високій продуктивності, гнучкості та масштабованості.

Цей стек технологій забезпечує ефективну розробку та підтримку додатку, дозволяючи швидко реагувати на зміни та забезпечувати відмінний користувацький досвід.

Разом ці технології створюють потужну платформу для створення сучасних, масштабованих та надійних додатків, які можуть ефективно обробляти великі обсяги даних та підтримувати високу кількість одночасних користувачів.

3.3. Модель бази даних

Модель бази даних – це структура, яка визначає, як дані організовані, зберігаються та керуються у базі даних. Вона описує логічне розташування інформації, типи даних, взаємозв'язки між ними та правила, що регулюють їх взаємодію. Моделі баз даних включають кілька типів, таких як реляційні, документно-орієнтовані, графові та інші. Вибір моделі бази даних визначає, наскільки ефективно система може обробляти дані, масштабуватися та відповідати на запити.

Модель бази даних MongoDB. MongoDB є прикладом документно-орієнтованої NoSQL бази даних, яка відрізняється своєю гнучкістю та масштабованістю. Основні аспекти моделі бази даних MongoDB включають:

- документно-орієнтована структура: Дані зберігаються у форматі BSON (Binary JSON), що дозволяє зберігати складні, вкладені структури, такі як масиви та об'єкти. Це забезпечує природне відображення об'єктно-орієнтованих мов програмування та спрощує роботу з даними;
- гнучкі схеми: На відміну від реляційних баз даних, MongoDB не вимагає жорсткого визначення схем. Це дозволяє легко змінювати структуру документів без необхідності перебудови бази даних, що підвищує гнучкість та швидкість розробки;

- колекції та документи: Дані в MongoDB організовані в колекції, які містять документи. Колекції аналогічні таблицям у реляційних базах даних, а документи – це записи, представлені у форматі JSON;
- індексація: MongoDB підтримує створення індексів для прискорення запитів. Індекси можуть бути створені на одному або декількох полях документа, що забезпечує швидкий доступ до даних.

База даних MongoDB використовується для зберігання інформації про користувачів, їх досягнення, статистику та інші дані, необхідні для функціонування системи.

Схема даних організована таким чином, щоб забезпечити ефективний доступ до неї з боку бекенду та швидкий вилучення даних для відображення на фронтенді.

3.4. Взаємодія між компонентами

Взаємодія між фронтендом та бекендом відбувається через API, яке передає дані у форматі JSON.

Фронтенд відправляє запити, наприклад, для отримання завдань чи збереження прогресу, а бекенд обробляє ці запити та повертає необхідну інформацію. Комунікація із базою даних відбувається за допомогою мови запитів MongoDB.

Взаємодія між фронтендом та бекендом через API, яке передає дані у форматі JSON, є ефективним і сучасним підходом до розробки веб-додатків. Це забезпечує низку переваг, особливо для навчальних додатків, де важливо швидко та надійно передавати дані між користувачем та сервером. У такій архітектурі фронтенд відправляє запити до бекенду для отримання завдань чи збереження прогресу, а бекенд обробляє ці запити, взаємодіє з базою даних MongoDB та повертає необхідну інформацію. Давайте детальніше розглянемо позитивні моменти такого підходу:

- реальний час: Швидка та ефективна взаємодія між фронтендом та бекендом через API дозволяє надавати миттєвий зворотний зв'язок

користувачам, що є критично важливим для навчальних додатків. Користувачі можуть бачити свої результати та прогрес в режимі реального часу, що підвищує мотивацію та ефективність навчання;

- персоналізація: Взаємодія через API дозволяє легко зберігати та витягувати персоналізовані дані користувачів, такі як прогрес навчання, улюблені курси чи налаштування. Це дозволяє створювати індивідуальні навчальні шляхи та підвищує залученість користувачів;
- масштабованість для великої кількості користувачів: Навчальні додатки часто мають велику кількість користувачів, і використання ефективного API дозволяє обробляти велику кількість одночасних запитів без зниження продуктивності. Це забезпечує стабільну роботу додатку навіть при пікових навантаженнях.

Висновок. Взаємодія між фронтендом та бекендом через API з використанням формату JSON та бази даних MongoDB надає численні переваги для розробки сучасних навчальних додатків. Цей підхід забезпечує модульність, гнучкість, масштабованість та високу продуктивність системи. Простота використання JSON, ефективна обробка запитів через API та потужні можливості MongoDB роблять цей технологічний стек ідеальним вибором для створення ефективних, надійних та масштабованих навчальних платформ.

3.5. Загальні рекомендації та висновки

Загальні рекомендації:

1) чітка архітектура:
    1) модульність: Розділення додатка на окремі модулі, такі як користувацький інтерфейс, логіка гейміфікації, управління даними та аутентифікація дозволить спрощувати підтримку та оновлення окремих частин системи;

2) чітке визначення API: чіткі інтерфейси для взаємодії між фронтендом і бекендом полегшить інтеграцію та забезпечить надійну комунікацію між компонентами;

2) ефективна взаємодія між фронтендом та бекендом:
    1) використання JSON: JSON є легким, зручним для читання та легко парсується. Це спрощує обробку даних на фронтенді та бекенді;
    2) оптимізація запитів: Ефективне використання асинхронних запитів та обробка даних на бекенді знизить час відповіді та забезпечить високу продуктивність системи;
3) використання сучасних інструментів та бібліотек:
    1) React: застосування компонентів та хуків для створення динамічного та інтерактивного інтерфейсу користувача забезпечить високу продуктивність та зручність використання;
    2) Fastify: за допомогою Fastify можливе створення швидкого та легкого бекенду. Його висока продуктивність та підтримка плагінів дозволяють швидко розширювати функціональність;
    3) MongoDB: зручний для зберігання даних у гнучкому та масштабованому форматі. Агрегаційні фреймворки MongoDB ефективно обробляють дані та виконують складні запити;
4) безпека та надійність:
    1) аутентифікація та авторизація: необхідно подбати про надійні методи аутентифікації та авторизації користувачів, щоб захистити дані та обмежити доступ до чутливих інформації;
    2) шифрування даних: HTTPS для шифрування даних під час передачі між фронтендом та бекендом. Необхідний захист даних у базі за допомогою шифрування;
5) масштабованість та продуктивність:
    1) горизонтальне масштабування: потрібно використовувати можливості горизонтального масштабування MongoDB для обробки великих обсягів даних та підтримки високого навантаження;

2) кешування: Щоб зменшити навантаження на сервер та прискорити доступ до даних рекомендовано використовувати механізми кешування на різних рівнях системи;

6) Моніторинг та тестування
   1) автоматизоване тестування: Швидко виявляти та виправляти помилки допоможуть автоматизовані тести для перевірки функціональності та стабільності системи;
   2) моніторинг: Впровадження системи моніторингу для відстеження продуктивності та стану додатку допоможе швидко реагувати на проблеми та оптимізувати роботу системи.

Висновки. Використання React, Node.js з Fastify та MongoDB створює потужну платформу для реалізації системи гейміфікації онлайн навчання. Такий підхід дозволяє досягти високої продуктивності, гнучкості та масштабованості додатку. Чітка архітектура, ефективна взаємодія між фронтендом та бекендом, а також використання сучасних інструментів сприяють зручності користувачів та швидкій розробці нових функцій.

Забезпечення безпеки даних та використання механізмів кешування допомагають підтримувати стабільну роботу системи навіть при високому навантаженні. Автоматизоване тестування та моніторинг дозволяють підтримувати високу якість та надійність додатку.

Вибір таких технологій, як React, Node.js (Fastify) та MongoDB, сприяє ефективній розробці та інтеграції різних компонентів системи гейміфікації онлайн навчання.

Важливо підтримувати чітку архітектуру та забезпечити ефективну взаємодію між фронтендом та бекендом для досягнення високої продуктивності та зручності користувачів.

4 РОЗРОБКА FRONT-END ЧАСТИНИ ДЛЯ ВЕБ-СИСТЕМИ

У розділі "Розробка FRONT-END частини для веб-системи автоматизації введення" ми детально розглянемо процес розробки фронтенду нашого веб-застосунку, який спрямований на автоматизацію введення інформації для навчання сліпого набору. Використана технологія React взаємодіє з користувачем, забезпечуючи ефективність та зручність введення даних.[13]

4.1. Дизайн і структура компонентів

Першим етапом розробки було визначення дизайну та структури компонентів фронтенду.

Застосовано підходи Material UI для створення сучасного та інтуїтивно зрозумілого інтерфейсу (див.рис.4.1). Дизайн передбачає мінімум зайвих елементів і кольорів для того, щоб користувач міг повністю зосередитися на виконанні завдання.

Рисунок 4.1 – Головна сторінка застосунку

Компоненти були розподілені за функціональними блоками: розділи додатку (базовий курс, змагання, тематичні тексти), карточка завдання, меню додатка, вправи для тайпінгу, вхід у профіль, створення нового профілю тощо (див.рис.4.2).

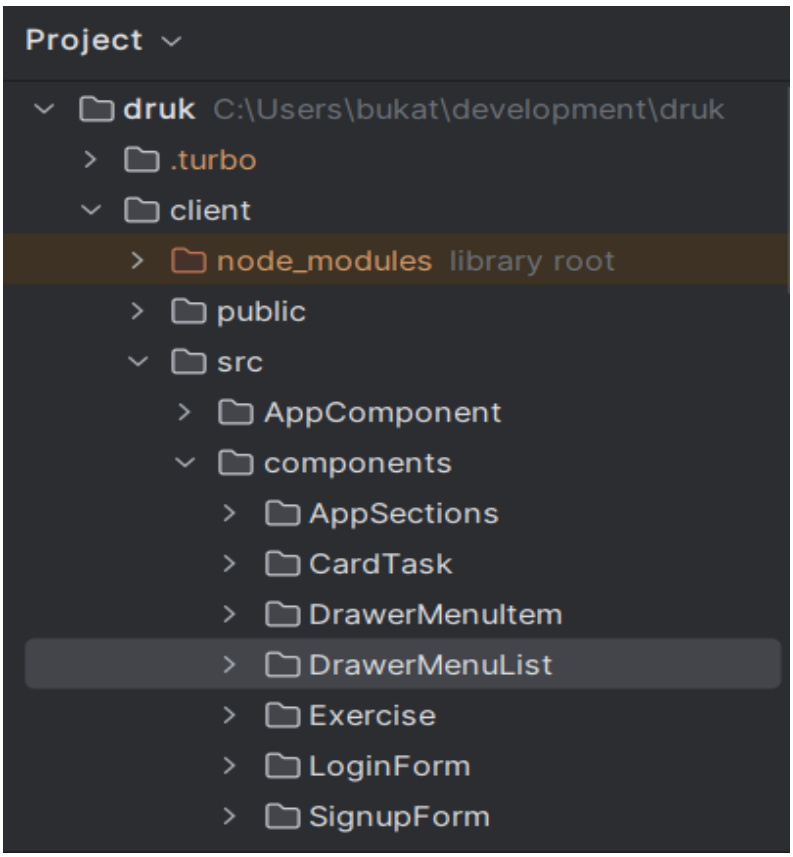

Рисунок 4.2 – Структура проєкта

Карточка завдання у застосунку буде мати інформацію про літери, які вивчатимуться при виконанні. Внизу зліва буде розміщений порядковий номер. Також, користувач буде бачити якість виконання виконаного завдання, не заходячи у саму картку. Якість буде показана у вигляді зірочок. Також, при дуже якісному виконанні на карточці завдання буде додаватись іконка з вогником (див.рис.4.3).

Рисунок 4.3 – Зовнішній вигляд карточки завдання

Клікнувши по карточці завдавдання, ми переходимо до виконання (див.рис.4.4).

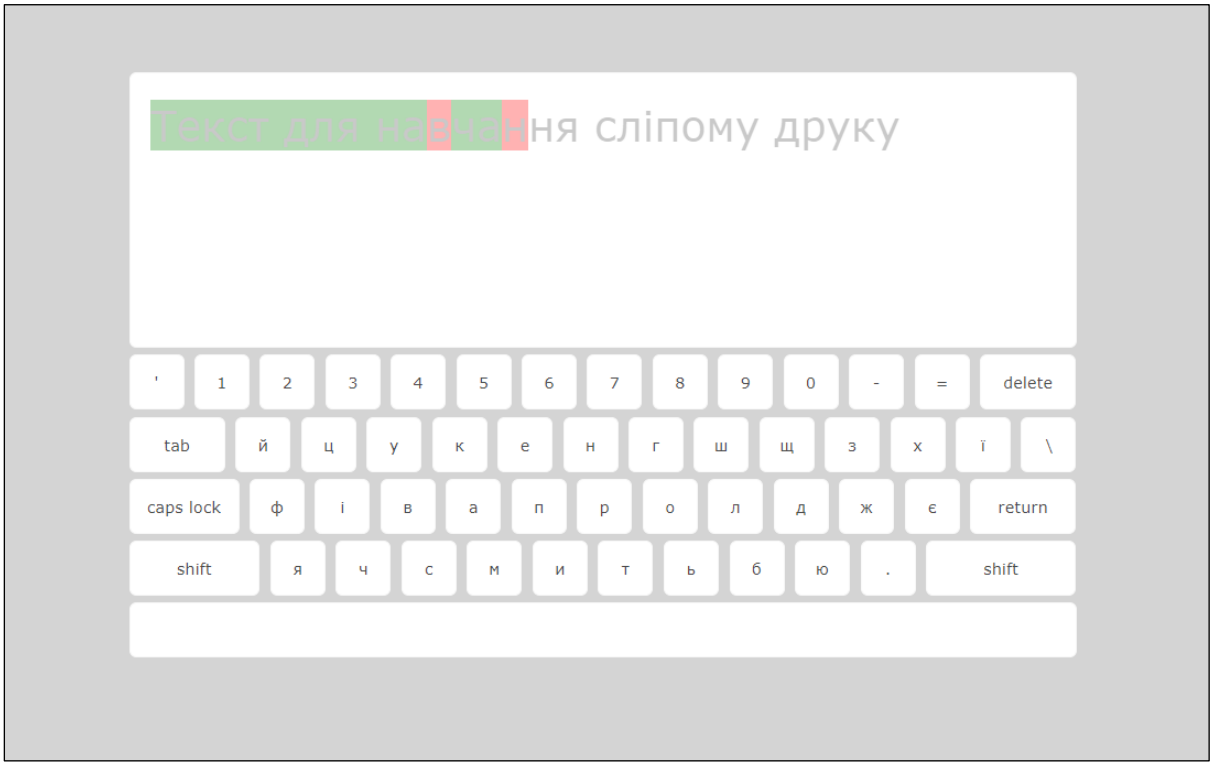

Рисунок 4.4 – Вигляд завдання всередині картки

При наборі тексту перевіряється правильність введення літер. Правльні підсвічені зеленим, неправильні – червоним. У подальшому статистика правильного вводу текста буде враховуватись для розрахунку якості.

Також, у застосунку реалізован функціонал для створення аккаунта користувача (див.рис.4.5).

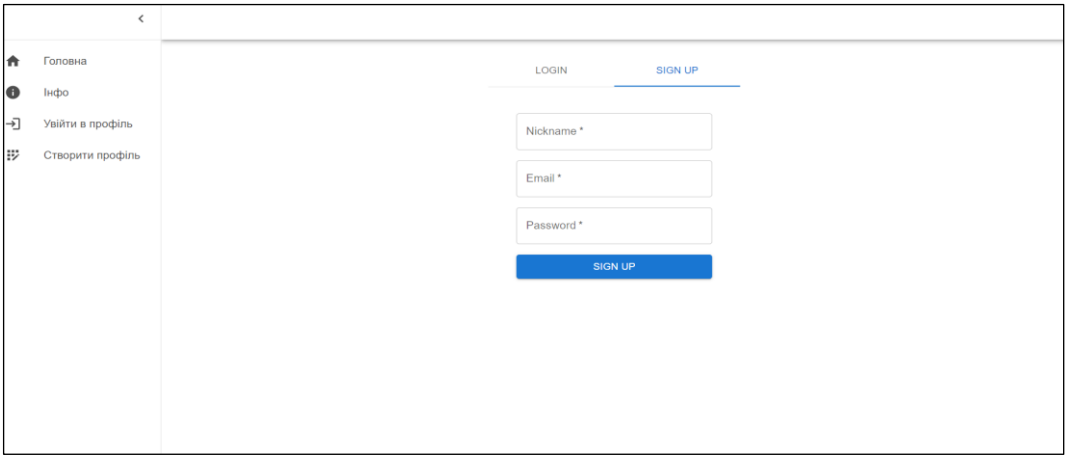

Рисунок 4.5 – Функціонал для створення профілю юзера

Для користувачів з існуючим профілем налаштований розділ меню для входу в аккаунт (див.рис.4.6).

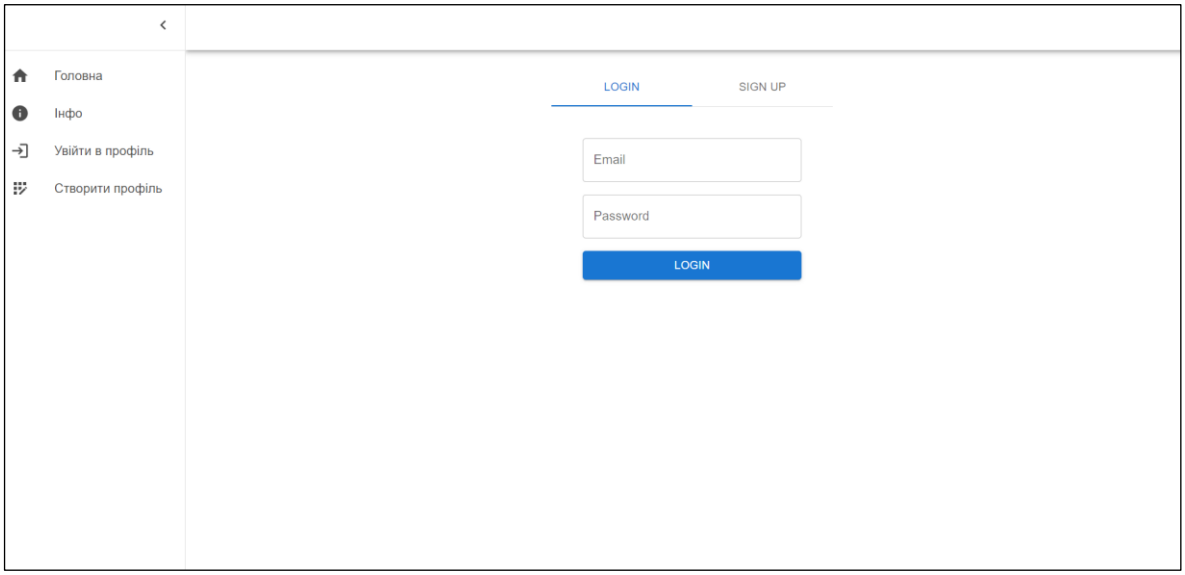

Рисунок 4.6 – Функціонал для входу в існуючий профіль

4.2. Інтерактивність та валідація введених даних

Для забезпечення інтерактивності та зручності користування вправами зі сліпим набором використано асинхронні запити для отримання та відправлення даних на бекенд.

Реалізована валідація введених даних для попередження помилок при введенні тексту у застосунку для навчання сліпому друку та забезпечення коректності результатів.

4.3. Інтеграція з бекендом

Інтеграція з бекендом у застосунку для навчання сліпому друку відбувається за допомогою API.

Бекенд веб-додатку представляє собою набір ендпоінтів відкритих для запитів через захищенний протокол https.

Частина ендпоінтів потребує аутентифікації юзера у застосунку для навчання сліпому друку.

Процес аутентифікації буде більш детально розглянуто у розділі 6.2.

Бекенд та фронтенд будуть обслуговуватись на одному хостингу та мати однаковий основний домен.

Бекенд буде доступний під піддоменом “api”.

Також для підвищення рівня безпеки запити на бекенд регулюються за допомогою політики CORS (Cross-Origin Resource Sharing) [14].

Ця політика не обробляє жодні запити надіслані не від основного домену, а також блокує можливість надсилати будь які дані за межі основного домену з клієнта.

Фронтенд надсилає запити на бекенд, який у свою чергу надсилає запити до бази даних у застосунку для навчання сліпому друку, після чого формує відповідь і надсилає її зворотньо на фронтенд.

Отриманні дані використовуються для рендерінгу відповідного стану додатку для навчання сліпому друку.

4.4. Гейміфікація та користувацький досвід

Гейміфікація полягає у впровадженні ігрових елементів та механізмів у неігрові ситуації, такі як освіта, щоб підвищити мотивацію, зацікавленість та ефективність навчального процесу [15].

У сучасному світі програмування є однією з найбільш динамічних сфер, тому важливо створювати ефективні методики навчання програмуванню, які відповідали б вимогам сучасного ринку праці. Використання гейміфікації в освітніх системах може сприяти зростанню зацікавленості та мотивації студентів, зменшенню втрати уваги під час навчання та покращенню засвоєння матеріалу [16].

У 1970-х роках інтерактивні комп'ютерні ігри набули популярності серед широкої аудиторії. Вони стали не тільки засобом розваги, але й ефективними інструментами для навчання та тренування [17].

Один із ранніх прикладів гейміфікації можна знайти ще в 1890-х роках, коли психолог Едвард Торндайк проводив дослідження поведінки тварин з використанням коробки-головоломки. Він встановив, що тварини навчалися швидше та ефективніше, коли отримували винагороду, таку як їжа, за вихід з

коробки. Це дослідження стало основою концепції оперантного кондиціонування, яке є важливим елементом гейміфікації.

Ще одна значна подія в історії гейміфікації – створення першої програми лояльності в 1896 році компанією Sperry & Hutchinson, яка пропонувала «Зелені марки» як винагороду за придбання певних товарів. Успіх цієї програми надихнув інші компанії на впровадження схожих систем винагород, таких як милі для часто літаючих пасажирів і бали за використання кредитних карток [18].

На початку 2000-х років, завдяки розвитку інтернет-технологій та зростанню популярності соціальних мереж, гейміфікація стала більш поширеним способом залучення та утримання уваги користувачів. Вона привернула більшу увагу дослідників та професіоналів, які почали вивчати її потенціал для підвищення залученості, мотивації та навчання у різних контекстах.

Термін «гейміфікація» був введений у 2002 році британським програмістом і винахідником Ніком Пеллінгом, який використовував його для опису процесу застосування ігрових механік до неігрових ситуацій [19].

З того часу гейміфікацію стали використовувати у багатьох сферах, включаючи освіту, охорону здоров'я, маркетинг і навчання персоналу.

З поширенням комп'ютерів і розвитком інформаційних технологій з'явилася можливість впроваджувати гейміфікацію в електронному вигляді. Перші гейміфіковані навчальні ігри здебільшого орієнтувалися на молодших учнів; вони були розважальними, легкими для розуміння і мали привабливий графічний дизайн, який дуже подобався школярам молодшого віку [20].

Основою стратегії гейміфікації є винагорода за виконані завдання. Гейміфікація передбачає використання ігрових механік та елементів гри для покращення процесу, продукту або послуги. Це можуть бути, наприклад, бейджі, рівні, участь у кланах, лідерборди, винагороди за досягнення певних цілей тощо. Методи гейміфікації використовуються для залучення користувачів до виконання потрібних дій, підвищення мотивації, та зменшення відчуття монотонності та нудьги [21].

Згідно з дослідженням, проведеним компанією IBM, 90% працівників, які брали участь у навчальній програмі з елементами гейміфікації, зазначили, що цей підхід допоміг їм краще зрозуміти матеріал і підвищити мотивацію [22].

За даними звіту компанії TalentLMS, впровадження гейміфікації в навчальні програми збільшує залученість учасників на 40%, покращує розуміння матеріалу на 55% і підвищує ймовірність успішного завершення курсу на 50% [23].

Розглянемо найбільш поширені методи гейміфікації.

Метод - д одавання елементів гри. Зазвичай елементи гри не займають абсолютно весь процес навчання. Як правило, по закінченню вивчення окремої теми або розділу, додається можливість зіграти у гру яка по суті є перевіркою знань.

Саме такий спосіб буде використаний при розробці застосунку для вивчення сліпого друку на клавіатурі.

Складові концепти методи гри являють собою (див.рис.4.7):

- дінаміку (це стосується реакцій і поведінок, що виникають в результаті взаємодії гравців з механіками. Вона включає емоційні та поведінкові аспекти, такі як мотивація, залучення, зворотній зв'язок і соціальна взаємодія. Динаміка визначає, як гравці відчувають і реагують на механіки гри протягом часу.);
- механіку (включає основні правила і структури, які визначають, як гравці взаємодіють з системою. Це фундаментальні елементи гри, що забезпечують дії та результати. Приклади механік включають завдання, рівні, виклики, нагороди, правила і обмеження, що визначають, як гра функціонує.);
- компоненти (це конкретні елементи або атрибути, які складають механіки гри. Вони можуть включати бейджі, бали, таблиці лідерів, рівні, аватари, нагороди, відзнаки і т.д. Ці компоненти використовуються для реалізації механік і створення динаміки, що залучає та мотивує гравців).

Рисунок 4.7 – Складові метода гри [24]

Разом ці три складові – механіка, динаміка та компоненти – формують повну систему гейміфікації, яка сприяє залученню, мотивації та ефективному навчанню або виконанню завдань (див.рис.4.8).[25]

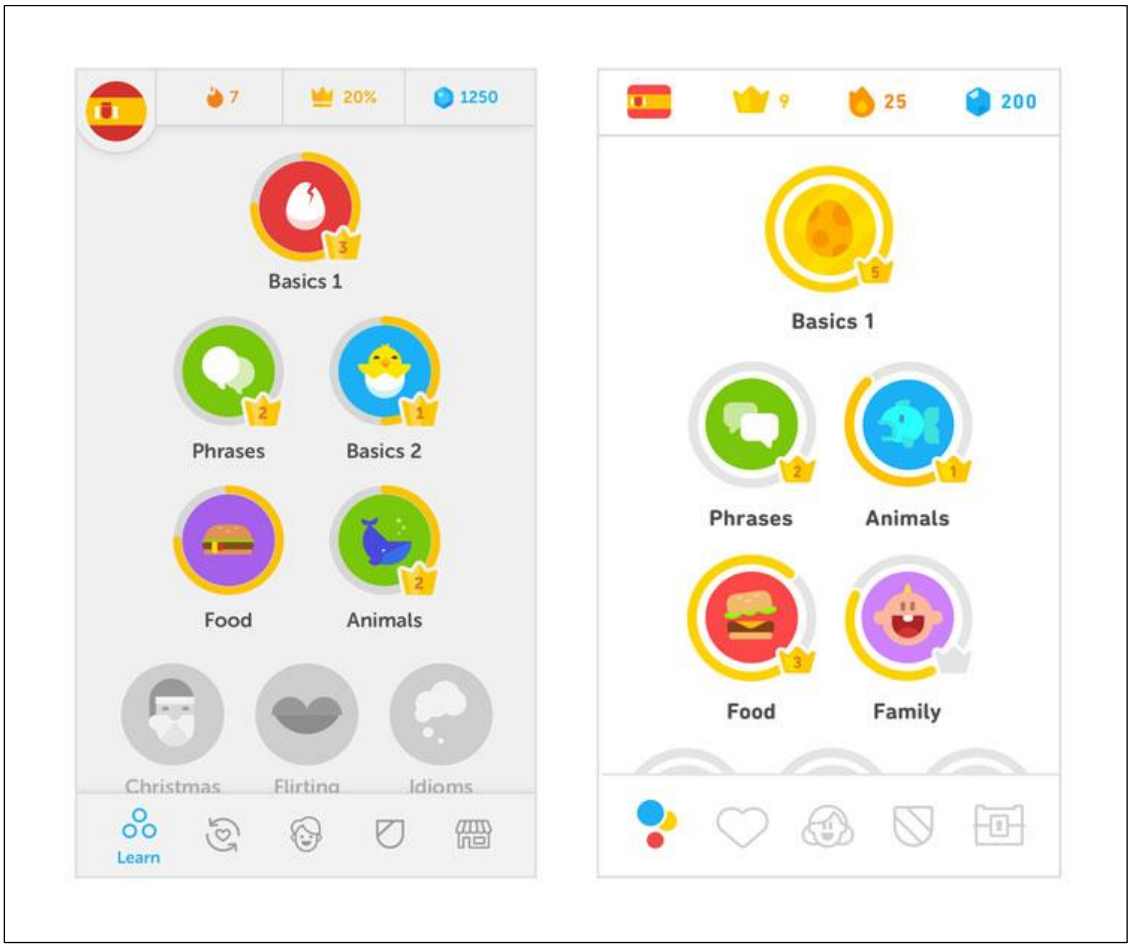

Рисунок 4.8 – Приклад методу “Бали та досягнення”

Метод "Бали та досягнення". Метод "Бали та досягнення" в гейміфікації використовується для підвищення мотивації та залученості користувачів. Бали нараховуються за виконання певних дій або завдань, дозволяючи користувачам бачити свій прогрес і порівнювати його з іншими. Досягнення, у свою чергу, нагороджують користувачів за досягнення конкретних цілей або виконання складних завдань. [26]

Це може бути представлено у вигляді бейджів, сертифікатів або інших візуальних нагород. Разом, бали та досягнення стимулюють користувачів до активної участі, допомагають підтримувати інтерес і створюють відчуття успіху та задоволення від виконаних завдань.[27]

Отже, застосування методу гейміфікації "Бали та досягнення" може зробити процес навчання програмуванню більш цікавим та продуктивним для користувачів. Використання системи нагород і досягнень може заохотити користувачів до активнішого та більш цілеспрямованого навчання [28].

Метод "Ранги та рейтинги". Метод гейміфікації "Ранги та рейтинги" використовує систему ранжування та рейтингів для стимулювання користувачів. Користувачі отримують певні ранги на основі їх досягнень та прогресу, а також можуть бачити свої рейтинги порівняно з іншими користувачами. Це створює змагальну атмосферу, підвищує мотивацію та заохочує користувачів до досягнення вищих результатів, покращуючи їхню залученість і продуктивність. [29]

Місії та завдання. Метод гейміфікації "Місії та завдання" передбачає надання користувачам конкретних цілей і завдань для виконання. Ці місії та завдання розробляються для поступового ускладнення, забезпечуючи чітку структуру і напрямок навчання або виконання роботи. Користувачі отримують винагороди або досягнення за успішне виконання місій, що стимулює їх до подальшої активності та прогресу. Цей метод підвищує мотивацію, робить процес більш захопливим і допомагає підтримувати інтерес, розбиваючи складні завдання на більш керовані етапи. [30]

Персонажі та історії. Метод гейміфікації "Персонажі та історії" використовує вигаданих персонажів і сюжетні лінії для створення захопливого контексту

навчання або роботи. Користувачі взаємодіють з персонажами, проходять через сюжетні повороти та виконують завдання в рамках історії. Це підвищує емоційну залученість, робить процес більш цікавим і допомагає користувачам легше запам'ятовувати інформацію через контекстуальне навчання. [31]

Соціальна інтеракція. Метод гейміфікації "Соціальна інтеракція" залучає користувачів через взаємодію з іншими учасниками. Це може включати спільні завдання, змагання, обмін досягненнями, коментування та підтримку один одного. Соціальна взаємодія сприяє створенню відчуття спільноти, підвищує мотивацію і залученість, оскільки користувачі надихаються успіхами інших і отримують соціальну підтримку та визнання. [32]

Аналіз проблем в гейміфікації та їх вирішення. Ефективність гейміфікованих навчальних систем залежить від ряду факторів, включаючи специфіку завдань і навчальних цілей, мотивацію та індивідуальні характеристики студентів. Важливо також враховувати можливі недоліки гейміфікації, такі як ризик штучного завищення результатів і ймовірність втрати інтересу студентів через перенасичення елементами гейміфікації [33].

Існує питання щодо того, наскільки гейміфіковані системи є ефективними і результативними порівняно з традиційними методами навчання. Наприклад, вибір невідповідних метрик може призвести до того, що користувачі будуть зосереджуватися на досягненні певних цілей, а не на розвитку важливих для роботи навичок. Необхідно ретельно підбирати метрики, щоб вони відображали реальну продуктивність користувачів [34].

Однією з головних проблем може бути залежність користувачів від гейміфікованої системи. Надмірне використання гейміфікованих елементів, як і у випадку з іншими іграми, може призвести до залежності, що негативно впливає на здоров'я та продуктивність. Відчуття залежності може знизити мотивацію користувачів до самостійного навчання, оскільки вони будуть очікувати нових гейміфікованих стимулів для отримання задоволення від навчального процесу. Спочатку бейджі або нагороди можуть бути ефективними, але з часом вони можуть не сприяти внутрішній мотивації та саморозвитку [35].

Навіть за умови ефективності гейміфікованої системи, вона може виявитися не затребуваною, якщо не забезпечує користувачам змістовний контекст або не відповідає їхнім потребам. Ще одна можлива проблема – відсутність індивідуалізації. Якщо система не враховує особисті потреби та рівень підготовки користувачів, це може призвести до нерівності між ними та зниження мотивації до навчання [36].

Варто пам'ятати, що гейміфікація не є універсальним рішенням для всіх ситуацій. У деяких випадках, особливо в серйозних навчальних або професійних контекстах, важливість серйозного підходу та глибокого розуміння предметної області може переважати над гейміфікаційними елементами [37].

Усі ці проблеми можна вирішити шляхом ретельного дослідження, тестування та вдосконалення методів гейміфікації. Важливо збалансовано оцінювати переваги й недоліки, а також враховувати зворотний зв'язок від учнів, користувачів та інших зацікавлених сторін.

Застосування методів гейміфікації може бути потужним засобом для покращення навчання та стимулювання учнів або користувачів. Проте, необхідно розглядати гейміфікацію як частину ширшого підходу до навчання та мотивації, враховуючи контекст, потреби та особливості цільової аудиторії [38].

У застосунку TypeRacing реалізовано елементи гейміфікації, такі як елемент гри, бали та досягнення, соціальна інтеракція для стимулювання користувачів. Крім того, забезпечено відображення статистики, досягнень для підвищення мотивації та покращення користувацького досвіду

4.5. Тестування та оптимізація

Тестування та оптимізація є критично важливими етапами розробки застосунку для навчання сліпому друку.

Впровадження ефективних методів тестування та оптимізації забезпечить високу якість, продуктивність та зручність користування додатком.

Впровадження комплексних заходів для тестування та оптимізації є ключовим фактором для успішного розробки та підтримки застосунку для навчання сліпому друку.

Автоматизоване тестування, тестування продуктивності, юзабіліті та безпекове тестування забезпечують високу якість та надійність додатку.

Оптимізація продуктивності фронтенду, бекенду та бази даних, а також моніторинг та аналіз продуктивності допомагають досягти високої швидкості роботи та зручності користування у застосунку для навчання сліпому друку.

Впровадження системи моніторингу для відстеження продуктивності додатку у реальному часі дозволить оперативно виявляти та реагувати на проблеми, що виникають.

Разом ці заходи створюють міцну основу для створення ефективного, надійного та зручного навчального додатку.

Розроблені компоненти та функціональність піддавалися системному тестуванню, включаючи тестування взаємодії з різними пристроями та браузерами.

Оптимізовано завантаження та відображення сторінок для забезпечення швидкої реакції та ефективності в роботі з великою кількістю даних.

4.6. Загальні рекомендації та висновки

Розробка FRONT-END частини для веб-системи планується реалізувати завдяки використанню технології React та підходів Material UI. Ефективна взаємодія з бекендом, елементи гейміфікації та оптимізація роботи гарантують зручний та стимулюючий досвід для користувачів, що вивчають сліпий набір.

Мною було проведено дослідження вибору основного фреймворка для реалізації веб-застосунка для навчанню сліпої печаті, в рамках ДКР по предмету «Теорія ігор та прийняття рішень».

Нижче я наводжу дане дослідження.

Дослідження вибору основного фреймворка для реалізації веб-застосунка для навчанню сліпої печаті.

Ідентифікація проблеми та постановка мети. Гейміфікація у навчанні – це захопливий та інноваційний підхід, який використовує елементи гри для покращення мотивації та залучення учнів до навчального процесу. Існує низка позитивних аспектів, які роблять гейміфікацію привабливою для освітнього середовища.

Гейміфікація стимулює конкуренцію та співпрацю, сприяючи формуванню командного духу серед учнів. Ігрові елементи, такі як лідерборди та виклики, можуть підштовхнути студентів до активності та взаємодії в рамках навчального процесу.[39]

Гейміфікація може зробити навчання більш доступним і цікавим для широкого кола учнів. Вона може враховувати індивідуальні потреби та рівні навичок, дозволяючи кожному учневі розвиватися власним темпом та отримувати особисті досягнення.[40]

Гейміфікація сприяє збільшенню мотивації до вивчення. Використання нагород, визнання досягнень та можливість просування на різні рівні може стимулювати учнів брати участь у навчальних активностях з більшим ентузіазмом та зацікавленістю.[41]

Гейміфікація може підвищити рівень взаємодії між вчителем та учнями. Вона надає можливість для персоналізації навчання, де вчителі можуть адаптувати завдання та відстежувати прогрес кожного студента, сприяючи більш ефективному навчанню.[42]

Загалом, гейміфікація може виявитися потужним інструментом для покращення навчання, забезпечуючи велику міру зацікавленості та підтримки з боку учнів. При правильному підході та врахуванні потреб учнів, вона може стати ефективним засобом для досягнення навчальних цілей.[43]

В рамках написання дипломного проекту я поставила за мету реалізувати популярні методи гейміфікації при розробці веб-застосунка, який допоможе опанувати навички сліпого набору для різних мов.

Щоб розробити подібний застосунок необхідно визначитись зі стеком технологій які будуть використовуватись у клієнтський та серверній частинах. В

межах поточної контрольної робити я хочу зробити вибір основного фреймфорка, який буде використаний для розробки клієнтської частини.

Інформаційна підготовка прийняття рішення. За останні 10 років розробка клієнтської частини веб-застосунків набула значного розвитку. Багато функціональної логіки перейшло з серверної частини на клієнтську з метою розвантажити потужності серверів та робити частину розрахунків за допомогою CPU та GPU клієнтських пристроїв. Основною мовою програмування для переважної більшості клієнтських фреймфорків є JavaScript.

Для роботи було обрано 5 фреймворків, які є найбільш популярними за даними сайту technostack.com:

- React;
- Angular;
- Vue;
- Svelte;
- Ember.

Кожен з цих фреймворків є вже багаторічним гравцем на ринку та має як свої переваги так і свої недоліки.

React, розроблений Facebook, є одним із найпопулярніших Front-End фреймворків. Він визначається декларативним підходом до створення інтерфейсів та широко використовується у великих веб-додатках.[44]

Angular, створений Google, відзначається своєю повністю інтегрованою системою, що дозволяє ефективно керувати всім життєвим циклом додатка. Він використовує TypeScript і пропонує розширені можливості для розробки масштабованих проектів.

Vue – це легкий фреймворк, який швидко набуває популярності завдяки своїй простоті та гнучкості. Він ідеально підходить для невеликих і середніх проектів, надаючи інтуїтивний API та зручні інструменти.

Svelte – це інноваційний фреймворк, який пропонує компіляцію в часі збирання, що дозволяє генерувати легкий та швидкий код. Svelte вирізняється тим, що весь фреймворк включений у кінцевий бандл, забезпечуючи ефективну роботу додатків.

Ember.js – це високорівневий Front-End фреймворк, який зорієнтований на конвенцію перед конфігурацією, надаючи структурованість та організацію для великих веб-додатків. Однією з ключових особливостей Ember є концепція "Конвенція над конфігурацією", що означає, що багато рішень і виборів вже зроблені за вас, спрощуючи процес розробки.

Також розглянемо вибір основних критеріїв, які будуть враховані при прийнятті рішення:

- кількість завантажень на тиждень згідно зі статистикою npm[45]. Цей показник є відображенням вживанності фреймворку у розробці, що в свою чергу на пряму корелює з розвиненістю допоміжних бібліотек для вирішення специфічних задач;
- кількість контріб’юторів на гітхабі[46] Чим більше у репозиторія з фреймворком контрибуторів, тим вище шанси, що баг існуючий у самому фреймворку буде пофікшено;
- кількість днів, що пройшли з дати релізу першої версії фреймворку. Чим старший фреймворк, тим краще, бо це в свою чергу зводить ймовірність виявлення раніше невідомого багу у самому фреймворку нижче;
- кількість днів, що пройшли з останнього релізу. Чим нижче цей показник, тим краще, бо він відповідає на питання на скільки фреймворк є актуальним і служить доповненням до минулого критерію;
- кількість розробників, що згадують фреймворк у своєму резюме на Linkedin[47]. Цей показник впливає на те, на скільки просто буде знайти розробників для підтримки програмного забезпечення, розробленого за допомогою фреймворку.

Вибір найкращої альтернативи. Згідно з обраними критеріями побудуємо таблицю, в якій вкажемо чисельні показники для кожного з фреймворків. Показники беруться з відкритих джерел. Для прийняття рішення скористаємось методом лінійної згортки з ваговими коефіцієнтами, тому що цей метод є найрозповсюдженішим, а також виявити вагові коефіцієнти згідно з пріоритетами для нашого випадку доволі проста задача. Додаємо вагові коефіцієнти від 1 до 5 для кожного з критеріїв, де 5 є найбільш вагомим, а 1 найменш вагомим.

Кількість контріб'юторів – є найбільш важливим критерієм з усіх, що ми обрали, бо він ілюструє скільки саме людей підтримують розробку даного фреймворку і корелює зі швидкістю виправлення багів у фреймворку, з популярністю серед опенсорс розробників, а також корелює з вірогідністю зупинки розвитку самого фреймворку та його екосистеми.

На наступному місці по вагомості є критерій з кількістю розробників, що вказало володіння навичками фреймворку у своєму резюме. Ці люди також є частиною ком'юніті, на яке спирається фреймворк і також, як було згадано раніше – цей показник відповідає на питання наскільки легко знайти розробників на підтримку чи подальшу розробку проекту, що базується на обраному фреймворку.

Дата першого релізу та дата останнього релізу мають для нас однакову вагу, ці показники доповнюють один-одного і вказують на стабільність та актуальність фреймворку.

Найменш вагомим фактором є кількість завантажень на тиждень. Опосередковано цей показник вказує на загальну вживанність фреймворку та його популярність, проте він також залежить від конкретних середовищ розробки та звичок окремих програмістів до переналаштувань середовищ, тому ми не можемо спиратись на цей показник, як на основний критерій для обрання фреймворку (див.табл.1).

Деякі з варіантів фреймворків значно поступаються іншим в окремих критеріях, проте враховуючи що критерії також мають різну вагу ми не можемо згідно до множини Парето відразу відкинути якісь варіанти, як очевидно програшні, тому вирішено провести нормалізацію даних для всіх п'яти варіантів.

Не всі критерії відповідають логіці “чим більше, тим краще”. Так для критерія “Кількість днів з останнього релізу” навпаки менший показник є кращим, тому проведемо нормалізацію даних вибравши за 1 найбільший показник для кожного з критеріїв, крім вищезгаданого, до нього застосуємо зворотню логіку.

Таблиця 1 – Оцінка фреймворків за обраними критеріями

	Завантаження на тиждень	Кількість контріб'юторів	Кількість днів з релізу першої версії	Кількість днів з останнього релізу	Кількість розробників
React	22907614	1643	3807	538	3040000
Angular	3341522	1825	2637	4	1040000
Vue	4422164	427	3593	5	528000
Svelte	809910	649	2561	3	2800
Ember	198498	820	4379	35	8800
Вар. Коеф.	2	5	3	3	4

Для нормалізації даних використаємо формулу співвідношення показника до еталонного значення (формула 1).

$$f = \frac{f_{вим}}{f_{ет}} \quad (1)$$

Для критерію з останньою датою релізу формула має зворотній вигляд (див.табл.2).

Таблиця 2. Оцінка фреймворків за обраними критеріями (нормалізовані дані)

	Завантаження на тиждень	Кількість контріб'юторів	Кількість днів з релізу першої версії	Кількість днів з останнього релізу	Кількість розробників
React	1,00	0,90	0,87	0,01	1,00
Angular	0,15	1,00	0,60	0,75	0,34

Vue	0,19	0,23	0,82	0,60	0,17
Svelte	0,18	0,36	0,58	1,00	0,00
Ember	0,01	0,45	1,00	0,09	0,00
Вар. Коеф.	0,12	0,29	0,18	0,18	0,24

Для розрахунків використаємо формулу лінійної адитивної згортки з ваговими коефіцієнтами (формула 2):

$$Z = max \sum_{j=1}^{n} \alpha_j \beta_j a_{ij} \quad (2)$$

Нижче вказані розрахунки для кожного фреймворка відповідно до формули:

React 1*0.12+0.9*0.29+0.87*0.18+0.01*0.18+1*0.24 = 0.78

Angular 0.15*0.12+1*0.29+0.6*0.18+0.75*0.18+0.34*0.24=0.63

Vue 0.19*0.12+0.23*0.29+0.82*0.18+0.6*0.18+0.17*0.24=0.39

Svelte 0.18*0.12+0.36*0.29+0.58*0.18+1*0.18+0*0.24=0.41

Ember 0.01*0.12+0.45*0.29+1*0.18+0.09*0.18+0*0.24=0.33

Таким чином відповідно до розрахунків найбільш вдалим вибором буде React.

Висновки. В результаті виконання домашньої контрольної роботи була сформована задача багатокритеріального вибору та обрані критерії для оцінки. Всі показники були нормалізовані від нуля до одиниці та до них була застосована формула лінійної адитивної згортки з ваговими коефіцієнтами.

За результатами розрахунків було визначено, що найкращим фреймворком для розробки клієнтської частини веб-застосунку у рамках написання дипломного проекту стане React.

Крім перерахованих переваг, вибір React також зумовлений його здатністю працювати з комплексними додатками із складним станом та взаємодією.

Використовуючи React, ми можемо ефективно керувати станом додатку і забезпечувати швидке оновлення інтерфейсу користувача при зміні даних або

взаємодії з ним. React також дозволяє розробляти компоненти, які можна повторно використовувати в різних частинах додатку. Це сприяє створенню модульних та масштабованих компонентів, що прискорює процес розробки та спрощує підтримку [48].

Під час розробки було застосовано різноманітні інструменти та бібліотеки для підвищення ефективності та якості роботи.

Наприклад, для управління залежностями проекту використовувався npm (Node Package Manager). Npm використовувався з кількох причин. Управління залежностями: npm дозволяє легко контролювати залежності проекту. Завдяки npm можна швидко встановлювати, оновлювати та видаляти пакети, необхідні для розробки та функціонування додатку. Це допомагає забезпечити узгодженість середовища розробки та мінімізувати можливі проблеми, пов'язані з сумісністю версій пакетів [49].

5 ПРОЕКТУВАННЯ БАЗИ ДАНИХ ВЕБ-ДОДАТКУ ДЛЯ СЛІПОГО НАБОРУ

5.1 Обґрунтування вибору бази даних для тренажера сліпої печаті

Розробка додатка для сліпої печаті – завдання, що вимагає вдумливого вибору технологій для забезпечення ефективного та стабільного функціонування. Одним із ключових етапів в цьому процесі є вибір системи управління базами даних (СУБД), яка відповідає конкретним потребам проекту. У данному розділі ми обгрунтуємо вибір MongoDB для розробки додатка для сліпої печаті.

MongoDB – це NoSQL база даних, яка відрізняється гнучкістю схеми даних. У звичайних реляційних базах даних, схема даних зазвичай задає структуру та типи даних. У MongoDB, ви можете легко змінювати схему без необхідності перекомпіляції бази даних або призупинення роботи системи. Це дозволяє зручно впроваджувати зміни в додаток для сліпої печаті під час його розвитку, не переймаючись про ризик втрати даних або втрату часу на міграцію даних.

Додаток для сліпої печаті може включати велику кількість користувачів та генерувати значну кількість даних. MongoDB підтримує розподілені запити, що дозволяє розподіляти базу даних на кілька серверів для забезпечення високої продуктивності та доступності. Це особливо важливо для додатка, який має обробляти великі об'єми інформації та запитів.

MongoDB використовує JSON-подібні документи для зберігання даних. Це дозволяє представляти дані у форматі, близькому до об'єктно-орієнтованого програмування, що полегшує взаємодію з базою даних та обробку даних в додатку для сліпої печаті. Документи можуть включати будь-яку кількість вкладених полів, що робить дані більш структурованими та легкими для розуміння та обробки.

Додаток для сліпої печаті може вимагати відображення географічних даних, таких як локація користувачів або інші об'єкти в просторі. MongoDB має вбудовану підтримку для геоданих, що дозволяє легко виконувати операції з просторовими запитами та інтегрувати геодані у додаток.

MongoDB є популярною технологією з активною спільнотою розробників. Це забезпечує швидку підтримку, постійні оновлення та доступ до багатьох корисних ресурсів. Розробка додатка для сліпої печаті з використанням MongoDB дозволяє скористатися передовими технологіями та найновішими розробками.

Вибір MongoDB для додатка для сліпої печаті обгрунтований його гнучкістю, високою масштабованістю, зручним обробником документів, підтримкою геоданих та активною спільнотою. Використання цієї NoSQL бази даних дозволить оптимізувати розробку та забезпечити ефективну роботу додатка в умовах постійного росту та розвитку.

5.2 Альтернативні варіанти та переваги MongoDB

MySQL та PostgreSQL є популярними реляційними базами даних, які використовують SQL для взаємодії з даними. Однак у порівнянні з MongoDB вони менш гнучкі щодо схеми даних. MongoDB дозволяє працювати з гнучкою схемою, тоді як реляційні бази мають строгу структуру. Для додатка для сліпої печаті, де може бути необхідно швидко змінювати та доповнювати дані, гнучкість MongoDB виявляється важливою перевагою.

Cassandra є NoSQL базою даних, спеціалізованою на високій доступності та розподіленому зберіганні. Однак MongoDB виграє у простоті використання та розробки, а також має потужні засоби для запитів та агрегацій. Для розробки додатка для сліпої печаті, де простота та ефективність грають важливу роль, MongoDB може бути більш практичним вибором.

Neo4j – графова база даних, спеціалізована на роботі з взаємозв'язками між об'єктами. У випадку, коли аспекти графової моделі сильно підходять для додатка, Neo4j може бути альтернативою. Проте, MongoDB, за своєю природою документо-орієнтованої бази даних, також може зручно вирішувати питання, пов'язані зі зберіганням та опрацюванням зв'язків між даними.

Переваги MongoDB:

- гнучкість Схеми: MongoDB дозволяє легко змінювати схему даних без переймання на себе великої кількості трудовитрат та ризику втрати даних;

- висока Масштабованість: Здатність розподілених запитів та горизонтальне масштабування роблять MongoDB ефективним для великих та розширених додатків;
- обробник Документів: Використання JSON-подібних документів полегшує роботу з даними та інтеграцію з програмним кодом;
- підтримка Геоданих: Інтеграція з геоданими спрощує обробку локаційної інформації у додатку для сліпої печаті;
- активна Спільнота та Розвиток: MongoDB має велику та активну спільноту розробників, що гарантує швидку підтримку та доступ до нових функцій.

Усі ці переваги роблять MongoDB потужним вибором для ефективного розвитку та експлуатації додатка для сліпої печаті.

6 РОЗРОБКА BACKEND ЧАСТИНИ

Розробка бекенду для веб-додатка тренажера для методу сліпого набору є ключовою частиною створення повноцінного та ефективного застосунку. У цьому розділі ми розглянемо основний функціонал, аутентифікацію та авторизацію, а також збереження даних прогресу до бази даних

6.1 Опис основного функціоналу Back End частини веб-додатку

Для написання бекенду веб-додатку було обрано стек технологій, який базується на мові програмування JavaScript. Основа бекенда написана за допомогою Node.js. У якості базового фреймворку було обрано популярний та швидкий Fastify. Основна база даних у проекта це MongoDB.

Основна причина вибору Node.js у якости основи бекенду є те що це частина екосистеми побудованої навколо мови програмування JavaScript. Ця мова є основою фронт-енд частини, тому використання однієї мови як для фронтенду, так і для бекенду є логічним кроком, який дозволяє легко використовувати одну кодову базу для всього проекту. Цей крок також дозволяє в подальшому підтримувати та розвивати кодову базу за допомогою меншої кількості програмістів.

Бекенд для даного веб-додатку виявляє собою набір REST API ендпоінтів, у відповідь на запит до яких надається інформація, яка потрібна для відображення у фронтенд частині. Доступ до більшості ендпоінтів вимагає авторизації юзера і знаходиться під захистом.

Для полегшення написання коду для бекенду використовується сучасний фреймворк Fastify. Цей фреймворк надає можливість уникнути низькорівневого програмування базових функцій для запуска та роботи звичайного http сервера. Цей фреймворк є альтернативою популярному фреймворку Express.js, який в 2024 році перестає отримувати підтримку. Саме цей факт і вплинув на пошук альтернативного фреймворку. Fastify - це сучасне та надійне рішення, яке значно прискорює розробку і отриманий результат є більш надійним за рахунок високого покриття юніт тестами коду самого фреймворку.

Основою базою даних для проекта було обрано MongoDB. Це не релятивна база даних, яка легко інтегрується з обраними раніше фреймворками. Не релятивна база даних дає можливість зберігати дані у більш гнучкому форматі. Це стає можливим завдяки відсутності чітко визначеної схеми даних, в порівнянні з релятивними базами даних. Ця перевага надасть можливість у подальшому по мірі розвитку проекта додавати якісь поля у збережені записи та не проводити великих міграцій даних.

6.2 Аутентифікація та авторизація

Викликом для веб-додатків є правильно побудована система аутентифікації та авторизації. Основна задача аутентифікації є визначення, що юзер має право на вхід у систему. Авторизація – це розділення доступу внутри системи в залежності від того, які можливості надаються кожному типу юзерів.

В нашому випадку аутентифікація побудована на основі JWT токену (JSON Web Token). Це токен побудований за стандартом RFC-7519 [50]. Процес генерації токену відбувається за умови успішної аутентифікації юзера.

Для аутентифікації у веб-додатку використовується пароль та логін. У якості логіна юзер має використовувати існуючу email адресу. Для реєстрації акаунту юзер має заповнити спеціальну форму на фронтенді додатку. У формі юзер має надати email, нікнейм та пароль. Ця інформація має передаватися у вигляді POST запиту на відповідний ендпоінт бекенду у зашифрованому вигляді через https. Первинна валідація даних відбувається у кожному окремому полі на фронтенді. Бекенд перевіряє ще раз валідність даних та повідомляє у разі виникнення невідповідностей. Якщо всі дані валідні, то відбувається збереження наданої інформації у колекцію юзерів у MongoDB.

Для захисту акаунтів від можливого витоку даних, або несанкціонованого доступу паролі зберігаються у вигляді хешів згенерованих за допомогою алгоритму хешування MD5 у відповідності до стандарту RFC-1321 [51]. Хешування відбувається кожного разу коли юзер буде виконувати спробу аутентифікації і для успішної аутентифікації хеш переданого паролю та збереженого паролю у базі

даних мають співпадати. Хешування є незворотньою процедурою, відповідно отримати вихідні дані за наявними хешем вважається умовно неможливим.

У разі успішної аутентифікації у відповідь сервер надсилає сформований JWT токен, який кодується у форматі base64. В подальшому цей токен має бути доданий у cookies на стороні клієнта та додаватись до кожного наступного запиту до сервера у відповідному заголовці http запиту. Таким чином сервер може ідентифікувати, що запит надійшов від авторизованного юзера. Також цей токен зберігається у іншій колекції з назвою sessions у базі даних. Кожен токен має всередині інформацію о даті його створення, даті його дезактивації, інформацію о клієнті, для якого було створено токен, а також рандомний набір символів. У разі, якщо токен збережений у базі даних не відповідає поточному токену надісланому у заголовці запиту – юзер переадресовується на форму логіна де він має пройти повторну процедуру аутентифікації.

Інформація про тип акаунта також вноситься до JWT токена і таким чином вирішується проблема авторизації. Юзеру надається доступ до функціоналу певного рівня тільки у разі якщо JWT токен юзера є валідним та містить поле type всередині з необхідним значенням.

6.3 Збереження даних прогресу до бази даних

Збереження даних прогресу до бази даних є критично важливою складовою для ефективної реалізації тренажера для методу сліпого набору. Цей аспект бекенду дозволяє ефективно відстежувати та управляти інформацією щодо виконаних уроків, досягнень та прогресу кожного користувача. Нижче розглянемо деталі цього процесу.

Для того, щоб зберігати дані про поточний прогрес юзера створюється колекція даних у базі MongoDB з назвою progress-data. Всередині цієї колекції створюється документ індивідуальний для кожного юзера. В цьому документі зберігається унікальний ідентифікатор юзера і дані про його поточний рівень прогресу.

Для кожної окремої вправи також створюється своя колекція. В неї додаються унікальні документи для кожного юзера в яких зберігається історія спроб юзера виконати цю вправу. В історію записуються дані про успішні та неуспішні спроби, відсоток правильно набраних символів, швидкість друку у словах на хвилину та конкретний час кожної спроби.

Такий підхід дозволяє доволі гнучко розширяти кількість збережених даних і не потребує складних маніпуляцій для збереження даних. Всі запити до бази даних відбуваються виключно через бекенд після валідації та санітизації отриманих даних. Такий підхід дозволяє мінімізувати ризики злому бази даних через атаки типу ін’єкцій та XSS.

Загальна реалізація збереження даних прогресу до бази даних включає в себе не лише технічні аспекти, але й стратегічне планування та організацію, щоб гарантувати високу ефективність та безпеку використання цих даних в тренажері для сліпого набору.

ВИСНОВКИ

Розробка веб-системи гейміфікованого онлайн навчання сліпого набору запланована з використанням технологій React, Node.js, Fastify та MongoDB. Нижче представлені основні висновки, отримані під час розробки та аналізу проєкту.

Вибір React для фронтенду: Використання бібліотеки React дозволить створити інтерактивний та зручний інтерфейс, що забезпечить користувачам зручність при вивченні сліпого набору. Компонентний підхід сприяє легкості розширення та модифікації.

Вибір Node.js та Fastify для бекенду: Використання Node.js та Fastify гарантує високу швидкість відповідей сервера та ефективне використання ресурсів. Це особливо важливо для системи гейміфікації, де час реакції впливає на користувацький досвід.

Гейміфікація для стимулювання навчання: Введення елементів гейміфікації, таких як бейджі, відзнаки та віртуальні нагороди, покращить мотивацію користувачів до активного вивчення сліпого набору. Це додасть процесу навчання додатковий інтерес.

Використання MongoDB: Використання бази даних MongoDB дозволить зберігати та витягувати дані швидко та ефективно. Систему можна буде легко масштабувати у майбутньому та пристосовувати її до змінних вимог.

Тестування та оптимізація: Системний підхід до тестування, включаючи взаємодію з різними пристроями та браузерами, дозволить гарантувати стабільність та коректну роботу системи. Оптимізація завантаження та відображення сторінок посприяє швидкій реакції та задоволенню користувачів.

Усі ці аспекти разом з ефективною взаємодією між компонентами системи створять навчальне середовище, яке не лише посприяє опануванню сліпого набору, а й забезпечить користувачів позитивний та захоплюючий досвід. Загальна архітектура системи буде мати гнучкість та готовність до майбутніх розширень та вдосконалень.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Hainey, T., Connolly, T. M., Stansfield, M., & Boyle, E. A. (2011). Evaluation of a game to teach requirements collection and analysis in software engineering at tertiary education level. Computers & Education, 56(4), 1219-1230.

2. Баланова Т. В. Використання методу інтервального повторення слів в процесі вивчення англійської мови. 2017. URL: http://eprints.zu.edu.ua/26807/ (дата звернення 15.05.24)

3. Plass, J. L., Homer, B. D., & Kinzer, C. K. (2015). Foundations of game based learning. Educational Psychologist, 50(4), 258-283.

4. Barata, G., Gama, S., Jorge, J., & Gonçalves, D. (2013). Improving participation and learning with gamification. In Proceedings of the First International Conference on Gameful Design, Research, and Applications (pp. 10-17).

5. Ren, Y., & Lu, Y. (2019). Gamification of education: A review of literature. Eurasia Journal of Mathematics, Science and Technology Education, 15(6), em1716.

6. Csikszentmihalyi, M. (1991). Flow: The psychology of optimal experience. HarperCollins.

7. Zichermann, G., & Cunningham, C. (2011). Gamification by design: Implementing game mechanics in web and mobile apps. O'Reilly Media, Inc.

8. Hunicke, R., LeBlanc, M., & Zubek, R. (2004). MDA: A formal approach to game design and game research. In Proceedings of the AAAI Workshop on Challenges in Game AI (pp. 1-5).

9. Документація Fastify. URL: https://fastify.dev/ (дата звернення 15.05.24)

10. Документація MongoDB. URL: https://www.mongodb.com/ (дата звернення 15.05.24)

11. Hoffman, J. A., & Duarte, P. (2014). Learning through game design: An investigation of age and gender differences in game design activities. Computers & Education, 73, 30-45.

12. Kohn, A. (1993). Punished by rewards: The trouble with gold stars, incentive plans, A's, praise, and other bribes. Houghton Mifflin Harcourt.

13. Top Frontend Frameworks of 2023 for Web Development. URL: https://technostacks.com/blog/best-frontend-frameworks/ (дата звернення 15.05.24)

14. Cross-Origin Resource Sharing (CORS) . URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS/ (дата звернення 15.05.24)

15. Burke, B. (2014). Gamify: How gamification motivates people to do extraordinary things. Bibliomotion.

16. Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: Defining "gamification". In Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments (pp. 9-15). ACM.

17. Hamari, J., & Sjöklint, M. (2015). Does gamification work?–A literature review of empirical studies on gamification. In Proceedings of the 2015 48th Hawaii International Conference on System Sciences (pp. 3025-3034). IEEE.

18. Landers, R. N., & Landers, A. K. (2014). An empirical test of the theory of gamified learning: The effect of leaderboards on time-on-task and academic performance. Simulation & Gaming, 45(6), 769-785.

19. Lee, J. J., & Hammer, J. (2011). Gamification in education: What, how, why bother?. Academic Exchange Quarterly, 15(2), 1-5.

20. Mekler, E. D., Brühlmann, F., Tuch, A. N., & Opwis, K. (2017). Towards understanding the effects of individual gamification elements on intrinsic motivation and performance. Computers in Human Behavior, 71, 525-534.

21. Kapp, K. M. (2012). The gamification of learning and instruction fieldbook: Ideas into practice. John Wiley & Sons.

22. Nacke, L. E., Bateman, C., & Mandryk, R. L. (2014). BrainHex: A neurobiological gamer typology survey. Entertainment Computing, 5(1), 55-62.

23. Pivec, M., & Dziabenko, O. (2015). Game-based learning and gamification in higher education: Impact study on lecturers. Journal of Interactive Learning Research, 26(1), 27-48.

24. Sailer, M., Hense, J. U., Mayr, S. K., & Mandl, H. (2017). How gamification motivates: An experimental study of the effects of specific game design elements on psychological need satisfaction. Computers in Human Behavior, 69, 371- 380.

25. Werbach, K., & Hunter, D. (2012). For the win: How game thinking can revolutionize your business. Wharton Digital Press.

26. Zichermann, G., & Linder, J. (2013). The gamification revolution: How leaders leverage game mechanics to crush the competition. McGraw Hill Professional.

27. Werbach, K., & Hunter, D. (2012). For the win: How game thinking can revolutionize your business. Wharton Digital Press.

28. Reeves, B., & Read, J. L. (2009). Total engagement: Using games and virtual worlds to change the way people work and businesses compete. Harvard Business Press.

29. Barata, G., Gama, S., Jorge, J., & Gonçalves, D. (2013). Improving participation and learning with gamification. In Proceedings of the First International Conference on Gameful Design, Research, and Applications (pp. 10-17).

30. Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: Defining "gamification". In Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments (pp. 9-15). ACM.

31. Hamari, J., Koivisto, J., & Sarsa, H. (2014). Does gamification work?–A literature review of empirical studies on gamification. In Proceedings of the 47th Hawaii International Conference on System Sciences (pp. 3025-3034). IEEE.

32. Nicholson, S. (2012). A user-centered theoretical framework for meaningful gamification. In Proceedings of Games+ Learning+ Society (Vol. 8)

33. Reeves, B., Malone, T. W., & O'Driscoll, T. (2008). Leadership's online labs. Harvard Business Review, 86(12), 62-70.

34. Landers, R. N., Bauer, K. N., Callan, R. C., & Armstrong, M. B. (2017). Psychological theory and gamification: The study of motivation and learning. In Gamification in Education and Business (pp. 25-49). Springe

35. Morschheuser, B., Werder, K. P., & Hamari, J. (2019). How to design gamification? A method for engineering gamified software. Information and Software Technology, 109, 13-31.

36. Kapp, K. M. (2012). The gamification of learning and instruction: Gamebased methods and strategies for training and education. John Wiley & Sons

37. Johnson, L., Adams Becker, S., Estrada, V., Freeman, A., & Hall, C. (2015). NMC horizon report: 2015 higher education edition. The New Media Consortium.

38. Kiili, K. (2005). Digital game-based learning: Towards an experiential gaming model. The Internet and Higher Education, 8(1), 13-24.

39. Dicheva, D., Dichev, C., Agre, G., & Angelova, G. (2015). Gamification in education: A systematic mapping study. Journal of Educational Technology & Society, 18(3), 75-88.

40. Lister, M., West, J. H., Cannon, B., Sax, T., Brodegard, D., & Just, E. (2014). Gamification of learning in statistics and analytics. International Journal of Teaching and Learning in Higher Education, 26(3), 349-359.

41. Nicholson, S. (2015). A user-centered theoretical framework for meaningful gamification. In Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play (pp. 201-214).

42. Lee, J. J., & Hammer, J. (2011). Gamification in education: What, how, why bother? Academic Exchange Quarterly, 15(2), 1-5.

43. Huang, Y. M., & Backman, S. J. (2011). Gamification and user engagement. In Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining (pp. 273-276).

44. Документація React. URL: https://uk.legacy.reactjs.org/docs/getting-started.html/ (дата звернення 15.05.24)

45. Node Package Manager. URL: https://www.npmjs.com/ (дата звернення 15.05.24)

46. Code hosting platform for version control and collaboration. URL: https://github.com/(дата звернення 15.05.24).

47. World's largest professional network on the internet. URL: https://www.linkedin.com/ (дата звернення 15.05.24).

48. Hamari, J., Koivisto, J., & Sarsa, H. (2014). Does gamification work?-A literature review of empirical studies on gamification. In Proceedings of the 47th 55 Hawaii International Conference on System Sciences (pp. 3025-3034).

49. Landers, R. N., Bauer, K. N., & Callan, R. C. (2017). Gamification of task performance with leaderboards: A goal setting experiment. Computers in Human Behavior, 71, 508-515.

50. JSON Web Token (JWT). URL: https://datatracker.ietf.org/doc/html/rfc7519/ (дата звернення 15.05.24).

51. The MD5 Message-Digest Algorithm. URL: https://datatracker.ietf.org/doc/html/rfc1321/ (дата звернення 15.05.24).

52. Study of Prediction and Classification Models in the Problems of Diabetes Among Patients with a Stroke in Different Living Conditions / Huliiev, N., Peretiaha, M., Khovrat, A., Teslenko, D., Nazarov, A. // Innovative Technologies and Scientific Solutions for Industries, (2 (24)), 54-61. URL: https://journals.uran.ua/itssi/article/view/285501/279567 (дата звернення 15.05.24).

53. Selection of Artificial Neural Networks for Disease Prediction / Iryna Kyrychenko, Oleksii Nazarov, Nural Huliiev, Oleksii Avdieiev // COLINS-2023: 7th International Conference on Computational Linguistics and Intelligent Systems, April 20–21, 2023, Kharkiv, Ukraine (CEUR Workshop Proceedings (CEUR-WS.org)). Volume I: Machine Learning Workshop, pp. 236–248. URL: https://ceur-ws.org/Vol-3387/paper18.pdf (дата звернення 15.05.24).

54. Parallelization of the VAR Algorithm Family to Increase the Efficiency of Forecasting Market Indicators During Social Disaster / Artem Khovrat, Volodymyr Kobziev, Alexei Nazarov, Sergiy Yakovlev // Information Technology and Implementation (IT&I-2022), November 30 - December 02, 2022, Kyiv, Ukraine. – 12 pp. CEUR Workshop Proceedings, 2022, 3347, pp. 222–233. – URL: https://ceur-ws.org/Vol-3347/Paper_19.pdf (дата звернення 15.05.24).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ НАУКОВИМИ НАПРЯМАМИ КЕРІВНИКА ТА НАУКОВЦІВ КАФЕДРИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

52. Study of Prediction and Classification Models in the Problems of Diabetes Among Patients with a Stroke in Different Living Conditions / Huliiev, N., Peretiaha, M., Khovrat, A., Teslenko, D., Nazarov, A. // Innovative Technologies and Scientific Solutions for Industries, (2 (24)), 54-61. URL: https://journals.uran.ua/itssi/article/view/285501/279567 (дата звернення 15.05.24).

53. Selection of Artificial Neural Networks for Disease Prediction / Iryna Kyrychenko, Oleksii Nazarov, Nural Huliiev, Oleksii Avdieiev // COLINS-2023: 7th International Conference on Computational Linguistics and Intelligent Systems, April 20–21, 2023, Kharkiv, Ukraine (CEUR Workshop Proceedings (CEUR-WS.org)). Volume I: Machine Learning Workshop, pp. 236–248. URL: https://ceur-ws.org/Vol-3387/paper18.pdf (дата звернення 15.05.24).

54. Parallelization of the VAR Algorithm Family to Increase the Efficiency of Forecasting Market Indicators During Social Disaster / Artem Khovrat, Volodymyr Kobziev, Alexei Nazarov, Sergiy Yakovlev // Information Technology and Implementation (IT&I-2022), November 30 - December 02, 2022, Kyiv, Ukraine. – 12 pp. CEUR Workshop Proceedings, 2022, 3347, pp. 222–233. – URL: https://ceur-ws.org/Vol-3347/Paper_19.pdf (дата звернення 15.05.24).