Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

(повна назва факультету)

(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

(назва освітнього ступеня)

на тему: _____

Виконав(ла): студент(ка) _____ курсу, групи _____

спеціальності _____

(шифр і назва спеціальності)

_____ _____
(підпис) (прізвище та ініціали)

Керівник _____ _____
(підпис) (прізвище та ініціали)

Нормоконтроль _____ _____
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ _____
(підпис) (прізвище та ініціали)

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль
20___

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет _____
(повна назва факультету)

Кафедра _____
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____ _____
(підпис) (прізвище та ініціали)
« » 20___ р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня _____
(назва освітнього ступеня)

за спеціальністю _____
(шифр і назва спеціальності)

студенту _____
(прізвище, ім'я, по батькові)

1. Тема роботи _____

Керівник роботи _____
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «___» _____ 20___ року № _____

2. Термін подання студентом завершеної роботи _____

3. Вихідні дані до роботи _____

4. Зміст роботи (перелік питань, які потрібно розробити)

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

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка

Студент _____ _____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Система тестування для визначення рівня володіння іноземною мовою // Кваліфікаційна робота освітнього рівня «Бакалавр» // Бурса Валерія Віталіївна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра інженерії програмного забезпечення, група СП-41 // Тернопіль, 2024 // С. - 98, рис. – 22, табл. – 3, кресл. – 0, додат. – 3, бібліогр. – 31.

Ключові слова: система тестування, англійська мова, веб-сервіс, рівень володіння, архітектура системи, інтерфейс користувача, тестування системи.

Кваліфікаційна робота присвячена розробці системи тестування, яка допоможе користувачам точно дізнатись свій рівень володіння англійською мовою та надасть рекомендації для покращення знань.

У першому розділі кваліфікаційної роботи було проведено детальний аналіз вимог до продукту та можливих ризиків. Проведено порівняння з існуючими рішеннями, вивчено схожі системи тестування, їхні функціональні можливості та методології розробки.

У другому розділі роботи було спроектовано та побудовано архітектуру та інтерфейс системи тестування. Було створено всі необхідні компоненти системи, забезпечено їх взаємодію та інтеграцію для забезпечення стабільної роботи продукту.

У третьому розділі кваліфікаційної роботи було проведено тестування розробленої системи. Тестування включало перевірку функціональності, виявлення недоліків та їхнє виправлення.

ANNOTATION

Testing System for Determining Foreign Language Proficiency // Bachelor's Qualification Work // Bursa Valeriia Vitaliivna // Ternopil National Technical University named after Ivan Puluj, Faculty of Computer Information Systems and Software Engineering, Department of Software Engineering, Group SP-41 // Ternopil, 2024 // Pages - 98, Figures - 22, Tables - 3, Drawings - 0, Appendices - 3, References - 31.

Keywords: testing system, English language, web service, proficiency level, system architecture, user interface, system testing

The qualification work is dedicated to the development of a testing system that helps users accurately determine their level of English proficiency and provides recommendations for improving their knowledge.

In the first section of the qualification work, a detailed analysis of product requirements and potential risks was conducted. Comparisons with existing solutions were made, and similar testing systems, their functional capabilities, and development methodologies were studied.

In the second section of the work, the architecture and interface of the testing system were designed and constructed. All necessary components of the system were created, ensuring their interaction and integration to provide stable product operation.

In the third section of the qualification work, testing of the developed system was carried out. The testing included checking functionality, identifying shortcomings, and rectifying them.

ЗМІСТ

ВСТУП

У сучасному світі, де глобалізація та міжнародна співпраця набувають все більшого значення, знання іноземної мови стає не просто бажаним, а й необхідним фактором для успішного життя.

Згідно з звітом EF Education First у статті "English Language Proficiency 2021", рівень володіння англійською мовою в Україні за останні 5 років зріс на 10 пунктів, що свідчить про зростаючу потребу у цій мові.

Англійська мова відкриває двері до кращих освітніх можливостей, кар'єрного зростання, подорожей та культурного обміну. Вона дозволяє спілкуватися з людьми з усього світу, отримувати доступ до інформації та розширювати свій кругозір.

З появою нових технологій, таких як штучний інтелект та машинний переклад, англійська мова стає ще важливішою. Ці технології потребують великих обсягів даних, які зазвичай доступні англійською мовою. Тому, щоб успішно працювати в цій сфері, необхідно мати високий рівень володіння англійською мовою.

Правильне та точне визначення рівня володіння іноземною мовою є однією з перших проблем, яка постає перед учнем. Цей етап є майже найважливішим, адже якщо людина не знатиме точно свого рівня, вона може почати вчити не з тієї теми і витратити багато часу на непотрібні речі.

Система тестування може допомогти учням оцінити свої знання та визначити, на якому рівні вони володіють мовою. Це дозволить їм обрати правильні навчальні матеріали та скласти план навчання, який буде відповідати їхнім потребам. Система тестування також може бути корисною для вчителів. Вона може допомогти їм визначити рівень своїх учнів, планувати уроки та оцінювати прогрес учнів.

Метою цього дослідження є розробка веб-сайту для визначення рівня володіння англійською мовою та її вивчення. Очікується, що в результаті цього

дослідження буде розроблено веб-сайт, який буде корисною та ефективною онлайн-платформою для визначення рівня володіння англійською мовою та її вивчення. Цей сайт буде доступний для користувачів з усього світу і допоможе їм покращити свої знання англійської мови та досягти своїх цілей.

Теоретична значимість цього дослідження полягає в тому, що воно вносить новий вклад у розробку методів визначення рівня володіння англійською мовою. Практична значимість цього дослідження полягає в тому, що воно має потенціал для покращення процесу вивчення англійської мови.

Розроблений веб-сайт буде доступний для людей з будь-яким рівнем володіння англійською мовою. Це дозволить їм точно визначити свій поточний рівень володіння мовою.

Після проходження тестування користувачі отримають детальні результати з порадами для кращого вивчення мови та підняття рівня володіння. Це значно збільшить шанси людей правильно і коректно сформулювати свій план навчання найбільш ефективним чином. Таким чином, це дослідження може допомогти людям вивчати англійську мову більш ефективно та досягати своїх цілей.

Крім того, веб-сайт може бути використаний для дослідницьких цілей. Він може допомогти дослідникам вивчати процес вивчення англійської мови та розробляти нові методи навчання. Очікується, що це дослідження матиме значний вплив на теорію та практику вивчення англійської мови.

Підсумовуючи, можна сказати, що розробка системи тестування для визначення рівня володіння іноземною мовою є актуальною та значущою проблемою. Така система може допомогти учням та вчителям ефективніше вивчати та викладати іноземну мову.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ ТА АНАЛІЗ ВИМОГ

1.1 Визначення мети та цілей проєкту

Постановка мети та завдань є найпершим кроком у будь-якому проекті. Це фундамент, на якому буде будуватися вся робота. Чітко визначена мета дає можливість всім залученим до проекту людям чітко зрозуміти, що саме розробляється, чому це робиться та яких результатів очікується. Правильно поставлена мета також допомагає зосередитися на головному та не витрачати час на незначні деталі. Крім того, чітко визначені завдання дозволяють скалсти ефективний план роботи, відстежувати прогрес та вносити необхідні корективи.

Метою проекту є розробка інтерактивної системи тестування для визначення рівня володіння іноземною мовою, зокрема англійською. Проект спрямований на вирішення проблеми неефективного навчання через неправильно складені навчальні плани, що не відповідають реальному рівню знань користувача. Система забезпечить точну оцінку мовних навичок, що дозволить користувачам починати навчання з відповідного рівня, оптимізуючи тим самим процес навчання та зменшуючи витрати часу на вивчення матеріалу, який вже засвоєний або є надто складним. Крім того, проект задовольнить потребу в доступному та зручному інструменті для самостійного оцінювання рівня володіння мовою, що допоможе ефективніше планувати навчальний процес.

Цілі проекту були сформульовані з використанням методології SMART, яка є ефективним інструментом для визначення та досягнення цілей. SMART - це акронім, що розшифровується як Specific (конкретний), Measurable (вимірюваний), Achievable (досяжний), Relevant (релевантний) та Time-bound (обмежений у часі). Використання цієї методології дозволяє чітко окреслити цілі, що сприяє їх більш ефективному досягненню [2].

Отже основними цілями проекту є:

- Розробка веб-сайту, що дозволяє користувачам точно визначити свій рівень володіння англійською мовою завдяки комплексному підходу до оцінки, що включає перевірку граматики, словникового запасу, аудіювання та читання.
- Розробка системи рекомендацій, яка буде допомагати користувачам покращити свої знання англійської мови.
- Інтегрування в систему можливість вивчення граматики, поділеної по темах, та практики за допомогою різних граматичних вправ, що відповідає потребам користувачів у побудові ефективного навчального плану.

Після визначення цілей проекту важливо окреслити конкретні завдання, необхідні для їх досягнення. Завдання проекту мають бути чіткими та конкретними, щоб забезпечити послідовне виконання робіт і досягнення поставлених цілей. Це дозволяє чітко визначити кроки, необхідні для успішного завершення проекту, і забезпечує відповідальність та ефективне управління ресурсами.

Отже, завданнями для даного проекту будуть наступні:

- Створення гарного юзер-френдлі інтерфейсу: Розробити інтерфейс веб-сайту, який буде зручним та інтуїтивно зрозумілим для користувачів, забезпечуючи легкий доступ до всіх функцій та модулів системи.
- Підключення бази даних: Створити та інтегрувати базу даних для зберігання результатів тестування, профілів користувачів та іншої необхідної інформації, забезпечуючи її надійність та безпеку.
- Розробка системи рекомендацій: Створити систему рекомендацій, яка надаватиме користувачам індивідуальні поради щодо покращення їх знань англійської мови на основі результатів тестування.
- Інтегрування навчальних матеріалів: Впровадити в систему можливість вивчення граматики, поділеної по темах, та практики за допомогою різних граматичних вправ, що дозволить користувачам будувати ефективний навчальний план.

Виконання цих завдань допоможе створити веб-сайт, який буде цінним ресурсом для людей, які хочуть вивчати англійську мову.

1.2 Аналіз конкурентів

Завершивши визначення цілей і завдань проекту, важливо провести аналіз конкурентів. Це дозволить визначити сильні та слабкі сторони існуючих рішень на ринку, знайти можливості для вдосконалення та забезпечити конкурентоспроможність нашого продукту.

Проведення аналізу конкурентів перед початком розробки проекту має кілька ключових переваг. Виявлення сильних сторін конкурентів допомагає зрозуміти, які аспекти варто запозичити або вдосконалити, а слабкі сторони вказують на можливості для покращення нашого продукту. Знання сильних та слабких сторін конкурентів допомагає краще організувати процес розробки, визначити пріоритети та уникнути поширених помилок.

Для проведення аналізу конкурентів ми будемо оцінювати їх за наступними критеріями:

- Інтерфейс: Зручність та привабливість користувацького інтерфейсу.
- Доступність: Наскільки легко користувачі можуть отримати доступ до всіх функцій та матеріалів.
- Наявність розділів для вивчення граматики та практики: Чи є можливість вивчення граматики, поділеної по темах, та практики за допомогою різних граматичних вправ.
- Чіткість результату: Точність та зрозумілість результатів тестування.

Серед великої кількості конкурентів, які наразі є присутніми на ринку, ми розглянемо більш детально наступні:

1. Duolingo

Duolingo - це безкоштовний веб-сайт та мобільний додаток, який допомагає людям вивчати іноземні мови. Він пропонує інтерактивні уроки, ігри та тести, які роблять процес навчання цікавим та ефективним. Duolingo доступний більше ніж 300 мільйонам користувачів у всьому світі [3].

Серед переваг даного сайту слід виділити наступні:

- Duolingo безкоштовний для використання, що робить його доступним для людей з будь-яким бюджетом.
- Користувачі можуть завантажувати уроки для офлайн-доступу, що зручно для використання в подорожах.
- Duolingo пропонує інтерактивні уроки та ігри, які роблять процес навчання цікавим та ефективним.

Проте веб-сайт також має низку недоліків, серед них:

- Duolingo використовує алгоритми машинного навчання, які не завжди є точними.
- Duolingo не пропонує вивчення всіх аспектів мови, таких як усне мовлення та листове спілкування.
- Duolingo містить рекламу, яка може дратувати деяких користувачів.

Також дуже важливо знати і розуміти чим розроблювана система переважатиме над конкурентом, у даному випадку це:

- Більш точна оцінка рівня володіння мовою, завдяки використання комплексного підходу до оцінки, який включає різні типи завдань.
- Більш глибоке вивчення всіх аспектів мови, включаючи усне мовлення та листове спілкування.
- Відсутність реклами, що зробить його більш приємним для користувачів.

2. Langeek

Langeek - це платформа для вивчення мов, яка пропонує комплексні тести для визначення рівня володіння мовою, а також курси та ресурси для вивчення мови.

Langeek використовує штучний інтелект, щоб персоналізувати навчання для кожного користувача [4].

До переваг цього сайту можна віднести наступні:

- Langeek використовує штучний інтелект, щоб оцінити рівень володіння мовою користувачів з високою точністю.
- Langeek використовує штучний інтелект, щоб персоналізувати навчання для кожного користувача.
- Langeek пропонує широкий спектр курсів та ресурсів для вивчення мови.

Однак, веб-сайт також має кілька недоліків, серед яких:

- Langeek є платним сервісом, що може зробити його недоступним для деяких людей.
- Користувачі не можуть завантажувати уроки для офлайн-доступу, що незручно для використання в подорожах.
- Langeek не пропонує можливості спілкуватися з носіями мови, що може бути важливо для практики усного мовлення.

Також важливо усвідомлювати переваги розроблюваної системи над конкурентами, зокрема це:

- Більш глибоке вивчення всіх аспектів мови, включаючи усне мовлення та листове спілкування.
- Безкоштовність, що зробить його доступним для більшої кількості людей.

3. Test-english

Test-English - це безкоштовний веб-сайт, який пропонує тести з граматики, словникового запасу, читання та усного мовлення, а також рекомендації щодо навчальних матеріалів. Test-English використовується більше ніж 1 мільйоном користувачів у всьому світі [5].

Наступні аспекти є перевагами цього сайту:

- Test-English безкоштовний для використання, що робить його доступним для людей з будь-яким бюджетом.
- Чіткі та зрозумілі результати: Test-English надає користувачам чіткі та зрозумілі результати тестування.

- Test-English пропонує тести з граматики, словникового запасу, читання та усного мовлення.

Тим не менш, цей веб-сайт має деякі недоліки, зокрема:

- Test-English використовує простий дизайн, який не є дуже привабливим.
- Test-English використовує прості тести, які не завжди є точними для оцінки рівня володіння мовою.
- Test-English доступний лише на веб-сайті, що може бути незручно для людей, які хочуть використовувати його на ходу.

Крім того, необхідно знати і розуміти, чим розроблювана система буде кращою за конкурентів, а саме:

- Використання більш комплексного підходу до оцінки рівня володіння мовою, що зробить результати більш точними.
- Використання сучасного та інтуїтивно-зрозумілого інтерсейсу.
- Більш глибоке вивчення всіх аспектів мови, завдяки використанню комплексного підходу до побудови тестів.

Проведення детального аналізу цих конкурентів допоможе нам виявити кращі практики та можливості для вдосконалення нашого проекту, що дозволить створити високоякісну та конкурентоспроможну систему тестування для визначення рівня володіння іноземною мовою.

1.3 Аналіз ризиків проєкту

Ризики проекту – це потенційні проблеми чи непередбачувані події, які можуть негативно вплинути на успіх проекту. Визначення ризиків на ранніх етапах розробки є ключовим для успішного управління проектом, оскільки дозволяє вчасно розробити стратегії для їх мінімізації або уникнення.

Визначення ризиків на початку проекту дозволяє проактивно підготуватися до можливих проблем, що можуть виникнути під час реалізації. Це забезпечує

можливість розробки відповідних планів дій, що допомагає уникнути чи зменшити негативні наслідки ризиків.

Ризики проекту можна класифікувати на кілька категорій:

- Технічні ризики: Включають проблеми з технологіями, програмним забезпеченням або обладнанням, що використовуються у проекті.
- Фінансові ризики: Пов'язані з бюджетом проекту, включаючи перевищення витрат та недостатнє фінансування.
- Операційні ризики: Включають проблеми з управлінням проектом, комунікацією в команді або організаційними процесами.
- Ризики якості: Виникають, коли кінцевий продукт не відповідає вимогам чи стандартам якості.
- Ризики термінів: Пов'язані з затримками у виконанні завдань або невідповідністю плановим строкам.

Ризики можуть суттєво вплинути на успіх проекту. Наприклад, технічні проблеми можуть призвести до значних затримок у розробці, що, у свою чергу, може призвести до перевищення бюджету та втрати клієнтів. Фінансові ризики можуть змусити скоротити обсяг проекту або змінити його основні характеристики, що негативно позначиться на якості кінцевого продукту.

Не визначення ризиків на ранніх етапах та відсутність заходів для протидії їм може призвести до значних фінансових та часових втрат. Наприклад, якщо ризик перевищення бюджету не враховано на початку проекту, це може призвести до фінансових проблем у майбутньому. Затримки у виконанні завдань можуть зірвати графік проекту, що, у свою чергу, може викликати невдоволення клієнтів та втрату репутації.

Після визначення цілей, завдань і аналізу конкурентів важливо провести аналіз ризиків проекту. Для цього проекту ризики будуть представлені у формі таблиці для кращого розуміння.

Таблиця 1.1 – Ризики проекту

Назва ризику	Короткий опис	Класифікація	Вплив на проект	Імовірність виникнення
Технічні проблеми	Проблеми з інтеграцією системи або несправності обладнання	Технічні	Затримки у розробці, перевищення бюджету	3
Проблеми з якістю	Продукт не відповідає вимогам чи стандартам якості	Ризики якості	Зниження задоволеності користувачів, необхідність переробок	4
Неправильна оцінка результатів	Тести не точно визначають рівень володіння мовою	Ризики якості	Зниження задоволеності користувачів, втрати довіри до системи	2
Непорозуміння в команді	Проблеми з комунікацією та управлінням проектом	Операційні	Зниження ефективності роботи, конфлікти в команді	3
Зміна ринкових умов	Зміна попиту на системи тестування для визначення рівня володіння іноземною мовою	Фінансові, ризики якості	Зниження попиту на систему тестування, втрата інвестицій	2

Заходи для запобігання ризикам:

- Чітке визначення та документування вимог до системи тестування на початковому етапі проекту.
- Залучення до проекту кваліфікованої команди розробників з досвідом розробки систем тестування.
- Проведення ретельного тестування системи на всіх етапах розробки.

- Забезпечення достатнього фінансування, людських та комп'ютерних ресурсів для проекту.
- Моніторинг ринкових умов та внесення необхідних змін до системи тестування.
- Вжиття заходів для забезпечення безпеки системи тестування.

Аналіз ризиків є важливим етапом в управлінні проектом. Визначення потенційних ризиків та розробка стратегій для їх мінімізації дозволяє уникнути непередбачуваних проблем та забезпечити успішне завершення проекту. Ефективне управління ризиками допоможе досягти високої якості кінцевого продукту та задовольнити потреби користувачів, що є ключовими факторами для успішної реалізації проекту.

1.4 Виявлення та встановлення вимог

Виявлення вимог – це процес збору, аналізу, документування та управління потребами та очікуваннями зацікавлених сторін проекту. Встановлення вимог є важливим кроком, оскільки воно визначає основні параметри та функціональні можливості майбутньої системи, що дозволяє уникнути недоліків та невідповідностей у процесі розробки.

Виявлення та встановлення вимог є критично важливими для успіху будь-якого проекту. Виявлення вимог дозволяє ідентифікувати основні функції, характеристики та обмеження проекту, що допомагає уникнути помилок на етапі розробки та забезпечити ефективне управління проектом [6].

1.4.1 Функціональні вимоги

Функціональні вимоги визначають, як система повинна взаємодіяти з користувачами, іншими системами та зовнішніми елементами. Вони описують специфічні функції, які система повинна підтримувати для досягнення своїх цілей. Для даного проекту, спрямованого на розробку системи тестування для визначення рівня володіння іноземною мовою, функціональні вимоги можуть включати наступні:

1. Тестування рівня володіння мовою: Система повинна надавати тести для оцінки рівня володіння мовою, які охоплюють різні аспекти, такі як граматика, словниковий запас, аудіювання та читання. Тести повинні бути адаптивними та відповідати міжнародним стандартам оцінки рівня володіння мовою.
2. Відображення результатів: Після завершення тесту система повинна надавати користувачам детальні результати оцінки, включаючи поточний рівень володіння мовою, сильні та слабкі сторони, а також рекомендації щодо подальшого навчання.
3. Навчальні матеріали: Система повинна пропонувати користувачам навчальні матеріали, поділені по темах, для вивчення граматики, словникового запасу та інших аспектів мови. Матеріали повинні бути доступні в різних форматах (тексти, аудіо, відео) та містити інтерактивні вправи.
4. Практичні вправи: Система повинна надавати інтерактивні вправи для практики різних аспектів мови. Вправи повинні бути адаптовані до рівня володіння мовою кожного користувача та містити зворотній зв'язок для покращення навчання.
5. Система рекомендацій: Система повинна мати механізм рекомендацій, який допомагатиме користувачам покращувати свої знання на основі результатів тестування та історії навчання. Рекомендації повинні включати поради щодо вибору навчальних матеріалів та вправ.

1.4.2 Нефункціональні вимоги

Нефункціональні вимоги описують характеристики та властивості системи, які не стосуються конкретних функцій, але мають значний вплив на загальну якість і користувацьке сприйняття. Для даного проекту, спрямованого на розробку системи тестування для визначення рівня володіння іноземною мовою, основні нефункціональні вимоги можуть включати:

1. Продуктивність: Система повинна забезпечувати високу швидкість обробки даних і швидке реагування на дії користувачів. Тести та навчальні матеріали повинні завантажуватися та оброблятися без затримок, щоб забезпечити комфортне користування.
2. Надійність: Система повинна бути стабільною і надійною, без частих збоїв або помилок. Важливо забезпечити високу доступність сервісу та мінімальні простої, щоб користувачі могли постійно використовувати платформу.
3. Масштабованість: Система повинна мати здатність до масштабування для обробки зростаючої кількості користувачів та збільшення обсягу даних. Це забезпечить можливість розширення функціоналу та підтримки більшої кількості користувачів у майбутньому.
4. Безпека: Система повинна захищати дані користувачів та забезпечувати безпеку інформації під час передачі та зберігання. Важливо врахувати вимоги до конфіденційності, цілісності та доступності даних.
5. Юзабіліті: Інтерфейс системи повинен бути інтуїтивно зрозумілим та зручним для користувачів. Навігація повинна бути простою, а функціонал – легко доступним і зрозумілим навіть для користувачів з мінімальними технічними навичками.
6. Сумісність: Система повинна бути сумісною з різними пристроями та операційними системами. Важливо забезпечити коректну роботу як на десктопних, так і на мобільних платформах, з урахуванням різних браузерів.

1.4.3 Вимоги до захисту та доступу

Одним із ключових аспектів розробки системи тестування для визначення рівня володіння іноземною мовою є визначення вимог до захисту та доступу. Вимоги до захисту та доступу спрямовані на забезпечення конфіденційності, цілісності та доступності даних, а також надання відповідного рівня доступу до системи різним категоріям користувачів. Серед них слід виділити наступні:

1. Конфіденційність: Система повинна забезпечувати захист даних користувачів від несанкціонованого доступу. Це включає використання шифрування для захисту даних як при зберіганні, так і при передачі, а також реалізацію політики доступу, що обмежує доступ до конфіденційної інформації тільки авторизованим користувачам.
2. Цілісність: Важливо забезпечити, щоб дані в системі залишалися точними і не були змінені або знищені несанкціонованим способом. Це досягається за рахунок використання механізмів виявлення та запобігання втручання, таких як контроль версій та журналювання змін.
3. Доступність: Система повинна бути доступною для користувачів у будь-який час, що включає захист від атак типу Denial of Service (DoS) та забезпечення стійкості до відмов. Важливо передбачити резервне копіювання даних і механізми швидкого відновлення після збоїв.
4. Захист від зовнішніх загроз: Система повинна бути захищена від різних видів кіберзагроз, таких як шкідливе програмне забезпечення, фішинг та інші види атак. Це включає регулярне оновлення програмного забезпечення, проведення аудиту безпеки та впровадження системи виявлення вторгнень.
5. Відповідність законодавству: Система повинна відповідати вимогам законодавства та нормативних актів щодо захисту персональних даних, таких як GDPR (General Data Protection Regulation) у Європейському Союзі.

1.4.4 Вимоги до інтерфейсу системи та програмного забезпечення

Одним із важливих аспектів розробки системи тестування для визначення рівня володіння іноземною мовою є визначення вимог до інтерфейсу системи та програмного забезпечення. Ці вимоги включають у себе всі аспекти, що стосуються взаємодії користувачів із системою та функціональних можливостей програмного забезпечення.

Вимоги до інтерфейсу системи визначають, як користувачі взаємодіють із програмним забезпеченням. Серед них:

1. Зручність використання (Usability): Інтерфейс повинен бути інтуїтивно зрозумілим і легким у використанні, з мінімальною кількістю кроків для виконання основних завдань. Навігація повинна бути простою та логічною, а основні функції легко доступними.
2. Доступність (Accessibility): Система повинна бути доступною для всіх категорій користувачів, включаючи людей з обмеженими можливостями. Це включає підтримку стандартів доступності, таких як WCAG (Web Content Accessibility Guidelines), і забезпечення можливості використання системи за допомогою альтернативних пристроїв введення.
3. Естетичний вигляд (Aesthetic Design): Інтерфейс повинен бути привабливим і професійно виконаним, з урахуванням принципів сучасного дизайну. Важливо використовувати зручні для читання шрифти, контрастні кольори та добре структуровані елементи інтерфейсу.
4. Адаптивність (Responsiveness): Інтерфейс повинен коректно відображатися на різних пристроях, включаючи настільні комп'ютери, планшети та смартфони. Це забезпечує гнучкість у використанні системи в будь-яких умовах.

Вимоги до програмного забезпечення визначають функціональні та нефункціональні аспекти системи, такі як продуктивність, безпека, масштабованість та інтеграція з іншими системами.

1. Продуктивність (Performance): Система повинна працювати швидко і ефективно, з мінімальним часом завантаження сторінок і виконання операцій. Важливо забезпечити стабільну роботу навіть при великому навантаженні користувачів.
2. Безпека (Security): Програмне забезпечення повинно мати високий рівень захисту від несанкціонованого доступу та атак. Це включає шифрування даних, надійну аутентифікацію користувачів та регулярне оновлення системи для усунення вразливостей.
3. Масштабованість (Scalability): Система повинна бути здатна до масштабування для обслуговування збільшеної кількості користувачів без зниження продуктивності. Це забезпечується за рахунок використання відповідної архітектури та технологій.
4. Інтеграція (Integration): Система повинна мати можливість інтеграції з іншими програмними рішеннями та сервісами, такими як бази даних, системи керування контентом та зовнішні API. Це дозволяє розширювати функціональність системи та забезпечувати її гнучкість.

Виявлення та встановлення вимог є фундаментальним етапом у розробці системи тестування для визначення рівня володіння іноземною мовою. Використання різних методів для збору вимог дозволяє забезпечити точність та повноту вимог, що сприяє успішній реалізації проекту та задоволенню потреб користувачів.

1.5 Методи визначення рівня володіння іноземною мовою

Рівень володіння іноземною мовою – це міра знань та навичок, які дозволяють людині використовувати мову в різних комунікативних ситуаціях. Визначення рівня володіння мовою є критично важливим для забезпечення ефективного навчання та оцінювання прогресу учнів. Точне визначення рівня

володіння мовою дозволяє розробити індивідуальні навчальні плани, що відповідають конкретним потребам кожного учня, і таким чином підвищує ефективність навчального процесу.

Існують кілька основних підходів до визначення рівня володіння мовою, які використовуються для оцінювання різних аспектів мовних навичок [7]:

- Тестування розуміння мови: Це включає аудіювання та читання. Тести на аудіювання перевіряють здатність розуміти мову на слух, а тести на читання – здатність розуміти писаний текст. Ці методи дозволяють оцінити пасивні мовні навички.
- Письмове тестування: Цей підхід включає написання творів і перекладів. Письмове тестування перевіряє здатність висловлювати думки в письмовій формі, що є важливим аспектом активних мовних навичок.
- Усне тестування: Це включає інтерв'ю та розмову. Усне тестування дозволяє оцінити здатність використовувати мову в усному спілкуванні, що є ключовим показником рівня володіння мовою.

Для даного проекту було обрано комплексний підхід до перевірки рівня володіння мовою, який включає всі перераховані методи. Такий підхід забезпечує найбільш точне та всебічне оцінювання мовних навичок, оскільки враховує як пасивні, так і активні аспекти володіння мовою. Це дозволяє не лише точно визначити поточний рівень знань, але й виявити слабкі місця, що потребують додаткової уваги в процесі навчання. Використання комплексного підходу сприяє розробці ефективних навчальних програм, що допомагають учням досягати своїх мовних цілей.

1.6 Опис основних технологій проектування

1.6.1 Вибір методології проектування

Для успішної реалізації системи тестування рівня володіння іноземною мовою важливо вибрати відповідну методологію проєктування. Вона має відповідати кільком ключовим вимогам: гнучкість, можливість подальшого редагування та вдосконалення, швидкість запуску продукту, а також невибагливість щодо часу та фінансів. Розглянемо кілька популярних методологій проєктування та оберемо найкращу для нашого проєкту.

1. Каскадна методологія (Waterfall): Ця методологія передбачає послідовне виконання етапів проєкту, де кожен наступний етап починається лише після завершення попереднього. Вона є зрозумілою та структурованою, але має недоліки, пов'язані з негнучкістю та складністю внесення змін після завершення етапів [8].
2. Гнучка методологія (Agile): Agile підходить для проєктів, що потребують швидкого запуску продукту з можливістю поступового вдосконалення. Вона дозволяє ефективно реагувати на зміни вимог та збирати відгуки користувачів для подальшого покращення. Цей підхід є гнучким, швидким і економічно вигідним, але потребує активної участі користувачів та частих ітерацій.
3. Методологія швидкого прототипування (Rapid Prototyping): Ця методологія базується на швидкому створенні робочого прототипу, який потім вдосконалюється на основі зворотного зв'язку від користувачів. Вона дозволяє швидко випустити початкову версію продукту, збирати відгуки та вносити корективи. Це ідеальний варіант для нашого проєкту, оскільки забезпечує гнучкість, швидкість запуску та економію часу та коштів [9].

Отже, для проєкту системи тестування рівня володіння іноземною мовою найкраще підходить методологія швидкого прототипування. Вона дозволить швидко запустити початкову версію продукту, отримати відгуки від користувачів і

вдосконалювати систему на основі цих відгуків. Це забезпечить максимальну гнучкість та економічну ефективність, дозволяючи створити кінцевий продукт, який найкраще відповідатиме потребам користувачів.

1.6.2 Вибір технологій для розробки

Вибір мови програмування для розробки системи тестування рівня володіння іноземною мовою є критично важливим етапом, який впливає на ефективність, продуктивність та зручність використання кінцевого продукту. Мова програмування повинна відповідати таким вимогам: підтримка сучасних веб-технологій, легкість інтеграції з HTML та CSS, наявність багатого набору бібліотек для обробки даних, гнучкість для подальшого розширення та вдосконалення системи, а також підтримка різних браузерів. Розглянемо декілька популярних мов програмування, які можуть бути використані для розробки даного проекту:

1. Python: містить велику кількість бібліотек, зручна у використанні, має високу читабельність коду. Проте дана мова програмування потребує додаткових інструментів для інтеграції з веб-технологіями.
2. Ruby: має елегантний синтаксис, швидку розробку завдяки фреймворку Ruby on Rails. Проте може мати складність у масштабуванні великих проектів.
3. JavaScript: ця мова програмування має широке використання в веб-розробці, велику кількість бібліотек та фреймворків (наприклад, React, Angular, Vue.js), підтримку сучасних браузерів. Проте вона може бути складною для новачків через асинхронну природу [10].

Враховуючи всі вимоги до даного проекту, найбільш підходящою мовою програмування є JavaScript. Використання JavaScript забезпечить швидкий запуск продукту, можливість подальшого вдосконалення на основі відгуків користувачів, а також гнучкість та ефективність у розробці функціоналу системи тестування.

Важливим аспектом при розробці системи тестування для визначення рівня володіння іноземною мовою є вибір середовища розробки. Середовище розробки повинно відповідати кільком основним вимогам: бути зручним у використанні, забезпечувати підтримку необхідних мов програмування, мати можливості для розширення за допомогою додаткових інструментів і плагінів, а також підтримувати інтеграцію з іншими системами та інструментами, необхідними для розробки і тестування.

Одним з найпопулярніших середовищ розробки для веб-програмування є Visual Studio Code (VS Code). Він забезпечує зручний та інтуїтивно зрозумілий інтерфейс, підтримку великої кількості мов програмування та можливість додавання розширень для покращення функціональності. Особливо корисними є розширення для роботи з JavaScript, HTML та CSS, що робить VS Code ідеальним вибором для веб-розробників.

Ще одним популярним середовищем розробки є WebStorm від компанії JetBrains. WebStorm спеціально розроблений для веб-розробки і пропонує потужні інструменти для роботи з JavaScript, HTML, CSS та іншими веб-технологіями.

Ще одним варіантом є Sublime Text, який також підтримує роботу з багатьма мовами програмування і пропонує швидкий та зручний інтерфейс. Sublime Text дозволяє додавати розширення для покращення функціональності, проте в порівнянні з VS Code та WebStorm його можливості можуть бути менш розширеними без додаткових налаштувань.

Для даного проекту найкращим вибором буде Visual Studio Code. Це середовище забезпечує всі необхідні інструменти для розробки функціоналу на JavaScript, підтримує численні розширення для веб-розробки, а також є безкоштовним і має широкий спектр налаштувань та можливостей.

Розробка системи тестування для визначення рівня володіння іноземною мовою вимагає вибору відповідного сервісу для побудови графічного інтерфейсу. Цей сервіс повинен відповідати кільком основним вимогам: бути зручним у використанні, забезпечувати інтерактивність та функціональність, мати

можливість для співпраці в режимі реального часу, а також підтримувати експорт та інтеграцію з іншими інструментами розробки.

Одним з найпопулярніших сервісів для створення графічного інтерфейсу є Figma. Цей сервіс забезпечує зручний та інтуїтивно зрозумілий інтерфейс для дизайну, підтримує роботу в команді в режимі реального часу, а також дозволяє легко експортувати код стилів, що значно спрощує роботу веб-розробників.

Ще одним популярним сервісом є Adobe XD. Він також пропонує потужні інструменти для дизайну та прототипування, підтримує інтерактивність та анімацію, а також забезпечує інтеграцію з іншими продуктами Adobe, що може бути корисним для розробників, які вже використовують інші інструменти Adobe в своїй роботі.

Sketch є ще одним популярним інструментом для дизайну, який особливо популярний серед дизайнерів, які працюють на платформі macOS. Sketch пропонує потужні інструменти для створення інтерфейсів, а також підтримує численні плагіни та розширення для покращення функціональності. Однак, Sketch не підтримує роботу в режимі реального часу так добре, як Figma, і обмежений лише платформою macOS.

Для даного проекту найкращим вибором буде Figma. Цей сервіс забезпечує всі необхідні інструменти для створення графічного інтерфейсу, підтримує інтерактивність та роботу в команді в режимі реального часу, а також дозволяє легко експортувати код стилів для подальшої розробки.

1.6.3 Вибір технологій для зберігання даних

База даних для проекту з розробки системи тестування для визначення рівня володіння іноземною мовою повинна відповідати певним вимогам: бути надійною, масштабованою, забезпечувати високу продуктивність та легко інтегруватися з іншими технологіями, використовуваними в проекті.

Серед найпопулярніших баз даних варто виділити такі [11]:

1. MySQL: Відкритий код, висока продуктивність, підтримка масштабування, добре інтегрується з багатьма іншими технологіями, зручний у використанні.
2. PostgreSQL: Відкритий код, підтримка багатоваріантної обробки транзакцій, високий рівень відповідності стандартам SQL, потужні можливості для збереження та обробки складних даних.
3. MongoDB: База даних типу NoSQL, гнучка модель даних, хороша масштабованість, підходить для зберігання великих обсягів неструктурованих даних.

Відповідно до вимог проекту, MySQL є найкращим вибором. Це пояснюється її популярністю, легкістю у використанні та високою сумісністю з іншими технологіями, що забезпечує надійну основу для розробки та експлуатації системи.

1.6.4 Вибір технологій для розгортання та управління проектом

Хостинг для проекту з розробки системи тестування для визначення рівня володіння іноземною мовою повинен відповідати наступним вимогам: бути надійним, мати високу доступність, забезпечувати достатню продуктивність для обробки запитів користувачів, підтримувати роботу з базами даних та мати зручний інтерфейс для керування ресурсами.

Серед найпопулярніших хостингів варто виділити такі:

1. InfinityFree: Безкоштовний хостинг, зручний у використанні, має гарний інтерфейс, вбудований файловий менеджер, можливість працювати з базою даних.
2. Hostinger: Пропонує доступні тарифи, високу продуктивність, підтримку різних CMS, зручний інтерфейс та якісну технічну підтримку.
3. Bluehost: Надійний хостинг з відмінною продуктивністю, підтримкою баз даних, зручним інтерфейсом та високою доступністю.

Відповідно до вимог проекту, найкращим вибором є InfinityFree. Він забезпечує всі необхідні функції для запуску та підтримки веб-сайту, при цьому є безкоштовним, що значно знижує витрати на проект. Зручний інтерфейс та вбудований файловий менеджер полегшують управління ресурсами, що робить InfinityFree оптимальним вибором для даного проекту.

2 ПРОЕКТУВАННЯ ТА РОЗРОБКА СИСТЕМИ

2.1 Визначення основних акторів системи та побудова діаграми варіантів використання

В аналізі систем актор (actor) представляє користувача або іншу систему, яка взаємодіє з системою, що розробляється. Варіант використання (use case) описує конкретну функціональність, яку система надає акторам, та визначає послідовність дій для досягнення певної мети.

Визначення основних акторів та варіантів використання є ключовим етапом у процесі розробки системи тестування для визначення рівня володіння іноземною мовою. Це допомагає зрозуміти, які користувачі будуть взаємодіяти з системою і які функції вона повинна виконувати для задоволення їхніх потреб. Чітке визначення акторів та варіантів використання дозволяє виявити всі необхідні функції та забезпечити їх правильну реалізацію, що, в свою чергу, підвищує ефективність та зручність використання системи.

Основними акторами в даній системі можуть бути студенти, викладачі та інші люди, зацікавлені у перевірці рівня володіння іноземної мови. Кожен з цих акторів має різні потреби і буде використовувати систему для виконання різних завдань.

2.1.1 Визначення акторів системи

У розробці системи тестування для визначення рівня володіння іноземною мовою важливим етапом є визначення основних акторів системи та побудова діаграми варіантів використання. Актори системи — це люди, організації або інші системи, які взаємодіють із розроблюваною системою. У даному випадку ми будемо розглядати лише одного основного актора — користувача.

Користувач — це особа, яка заходить на сайт, натискає на необхідні кнопки, проходить тести, читає граматику та виконує практичні вправи. Він є єдиним актором у даній системі, оскільки вся взаємодія відбувається безпосередньо з інтерфейсом сайту та має наступні варіанти використання системи:

- Вхід у систему: користувач входить в систему через посилання.
- Перевірка рівня володіння іноземною мовою: Користувач натискає на кнопку перевірки рівня іноземною мовою і система перенаправляє його на сторінку з розділами тестування.
- Вибір розділу: після потрапляння на сторінку розділів тесту, користувач може обрати який з чотирьох розділів тесту (перевірка словникового запасу, граматики, читання та аудіювання) він хоче пройти.
- Виконання завдань зі словникового запасу: користувач читає речення з пропущеним словом або фразою, та повинен вибрати правильний варіант серед запропонованих. Система перевіряє правильність вибору та зберігає результат для подальшого виведення.
- Виконання завдань з граматики: користувач читає речення або фрагмент тексту з пропущеною частиною та вибирає правильний варіант граматичної конструкції серед запропонованих. Система перевіряє правильність виконання завдання та зберігає результат.
- Виконання завдань з читання: користувач читає наданий системою текст та відповідає на запитання по ньому. Система перевіряє правильність виконання завдання та зберігає результат.
- Виконання завдань з аудіювання: користувач виконує завдання відповідно до почутого на аудіодоріжці. Система зберігає отриманий результат.
- Увімкнення аудіодоріжки: користувач вмикає аудіодоріжку, натисканням на відповідну кнопку, під час проходження тесту з аудіювання, для виконання завдання.
- Отримання результатів тестування: після проходження всіх розділів тесту користувач отримує свій рівень володіння іноземною мовою,

ознайомлюється з балами за кожен розділ та читає поради щодо покращення знань з ресурсами для вивчення.

- Вивчення граматики: користувач може перейти на сторінку навчання, де у вигляді меню будуть написані різні граматичні теми, при натисканні на які відкриється відповідний текст з простим та зрозумілим поясненням вибраної теми.
- Проходження граматичних вправ: користувач може перейти на сторінку практики, де матиме можливість виконати певні практичні завдання розділені по розділах та перевірити отримані результати з позначенням в якому саме завданні була помилка.
- Вихід з системи: після виконання всього, що планувалось, користувач виходить з системи.

2.1.2 Діаграма варіантів використання

Діаграма варіантів використання (Use Case Diagram) є візуальним представленням взаємодії акторів із системою через різні функціональні сценарії (варіанти використання). Вона допомагає зрозуміти, які дії можуть виконувати користувачі в системі, та як ці дії пов'язані між собою. Діаграма варіантів використання є важливою для наочного розуміння функціоналу системи, що дозволяє більш ефективно планувати та реалізовувати її компоненти [12].

Існує багато інструментів для побудови діаграм варіантів використання. Для даного проекту діаграма варіантів використання була побудована за допомогою веб-сервісу Draw.io. Це безкоштовний та простий у використанні онлайн-інструмент, який дозволяє створювати UML-діаграми, в тому числі діаграми варіантів використання. Він має широкий спектр функцій, які дозволяють створювати професійні діаграми з мінімальними зусиллями.

Вигляд діаграми варіантів використання див. рисунок 2.1.1.

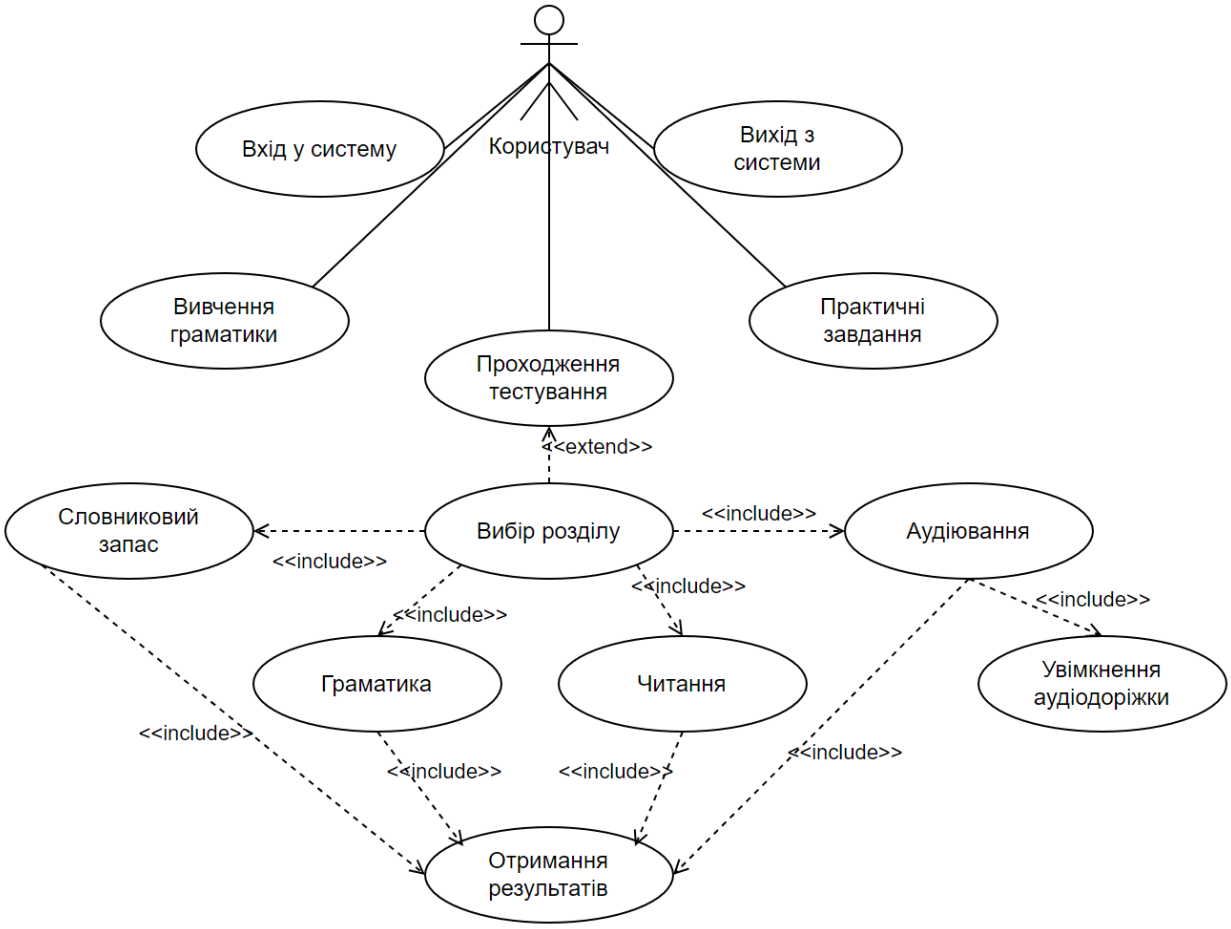

Рисунок 2.1 – Діаграма варіантів використання

Визначення акторів і варіантів використання є критично важливими для правильного розуміння функціональності системи. Це допомагає розробникам чітко визначити, які саме функції необхідно реалізувати, а також як ці функції будуть взаємодіяти між собою. Це забезпечує цілісність і логічну послідовність системи, що в кінцевому результаті призводить до створення ефективного і зручного для користувачів продукту.

2.2 Побудова діаграми класів

UML (Unified Modeling Language) діаграма класів — це візуальне подання структури системи, яке показує класи, їх атрибути, методи та зв'язки між класами.

Діаграми класів використовуються для моделювання об'єктно-орієнтованої архітектури програмного забезпечення, допомагаючи розробникам зрозуміти та оптимізувати взаємодію між різними компонентами системи. Використання UML діаграм дозволяє побудувати більш організовану та оптимізовану архітектуру, забезпечуючи кращу підтримку і розвиток програмного забезпечення [13].

Для даної системи було виділено шість основних класів, які правильно відображатимуть її роботу.

Клас User містить змінні id, name, та currentTest, що представляють ідентифікатор користувача, його ім'я та поточний тест відповідно. Методи startTest(testId), submitAnswer(questionId, answer) і getResult() дозволяють користувачеві розпочати тест, відправити відповідь та отримати результати.

Клас Test включає змінні id, title та questions, що представляють ідентифікатор тесту, його назву та масив запитань. Методи getQuestion(questionId) і submitAnswer(userId, questionId, answer) забезпечують отримання конкретного запитання та відправлення відповіді користувача.

Клас Question містить змінні id, text, options та correctAnswer, що представляють ідентифікатор запитання, текст запитання, масив варіантів відповідей та правильну відповідь відповідно. Метод checkAnswer(answer) перевіряє, чи є відповідь правильною.

Клас Option містить змінні id та text, що представляють ідентифікатор та текст варіанту відповіді. Цей клас використовується у класі Question для зберігання варіантів відповідей.

Клас Result включає змінні userId, testId, score та feedback, що представляють ідентифікатор користувача, ідентифікатор тесту, оцінку та зворотний зв'язок відповідно. Методи calculateScore(answers) і generateFeedback() забезпечують обчислення оцінки та генерацію зворотного зв'язку для користувача.

Клас TestSystem містить змінні users та tests, що представляють масиви користувачів і тестів. Методи addUser(user), addTest(test), getUser(userId) і getTest(testId) забезпечують додавання користувачів та тестів, а також отримання користувача або тесту за їх ідентифікаторами.

На основі описаних класів було створено діаграму класів у веб-сервісі Draw.io. Вона представлена на рисунку 2.2.

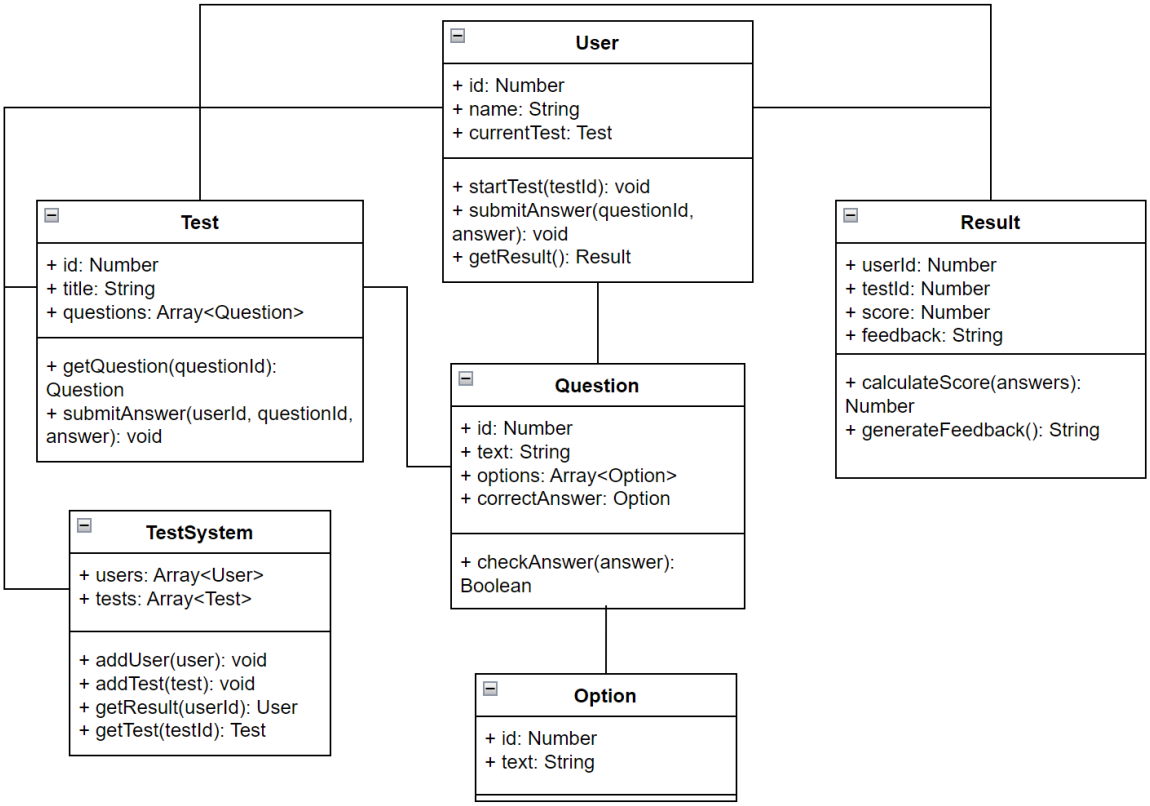

Рисунок 2.2 – Діаграма класів

Дана UML діаграма класів дозволяє розробникам краще зрозуміти структуру і взаємодію між компонентами системи тестування. Це сприяє створенню більш організованого та оптимізованого коду, полегшує підтримку та подальший розвиток системи. Використання таких діаграм допомагає уникнути помилок на етапі проектування та забезпечує ефективну реалізацію програмного забезпечення.

2.3 Проектування компонентів системи

Компонент системи - це окрема, самостійна частина системи, яка виконує певну функцію та взаємодіє з іншими компонентами для досягнення загальної мети системи. Простими словами, компонент системи можна порівняти з деталями конструктора, які при складанні утворюють єдиний механізм.

У системі тестування для визначення рівня володіння іноземною мовою компонентами можуть бути модулі, що відповідають за тестування граматики, читання, словникового запасу, аудіювання, а також компоненти для відображення результатів тесту, перегляду граматичних матеріалів та виконання практичних вправ.

В системі тестування для визначення рівня володіння іноземною мовою можна виділити наступні основні модулі:

1. Модуль користувача: Відповідає за інтерфейс користувача, через який користувач взаємодіє з системою. Включає функціонал для навігації по сайту, запуску тестів, перегляду результатів та навчальних матеріалів.
2. Модуль тестування: Відповідає за проведення тестів. Включає тест з граматики, читання, словникового запасу, аудіювання, а також забезпечує можливість програвання аудіотреків для завдань з аудіювання.
3. Модуль результатів: Обробляє та зберігає результати тестів, надає детальні рекомендації щодо покращення рівня володіння іноземною мовою.
4. Модуль граматичних матеріалів: Включає функціонал для відображення граматичних матеріалів, розділених по темах.
5. Модуль практичних вправ: Забезпечує можливість виконання практичних вправ з граматики, розділених по рівнях, та перевіряє правильність виконання завдань.
6. Модуль аудіо: Відповідає за зберігання та відтворення аудіотреків, необхідних для тестів з аудіювання.

7. Модуль бази даних: Забезпечує зберігання всієї інформації про тести, результати, граматичні матеріали та користувацькі дії.

Дані модулі системи взаємодіють між собою для забезпечення коректної роботи та інтеграції функціоналу. Модуль користувача взаємодіє з усіма іншими модулями для надання користувачеві можливості взаємодії з системою. Наприклад, він запускає модуль тестування при початку тесту, звертається до модуля результатів для отримання підсумків тесту та рекомендацій.

Модуль тестування отримує дані з модуля бази даних про тести та зберігає результати після завершення тестування. Він також взаємодіє з модулем аудіо для програвання аудіотреків під час тесту з аудіювання.

Модуль результатів зберігає оброблені результати в модулі бази даних і надає їх модулю користувача для відображення користувачеві.

Модуль граматичних матеріалів і модуль практичних вправ звертаються до модуля бази даних для отримання відповідного контенту та зберігають результати виконання вправ.

Для кращого розуміння залежностей між різними модулями буде розроблено діаграми послідовностей (див. рисунок 2.2.1 - 2.2.4). Діаграми послідовностей наочно демонструють порядок та взаємодію між модулями під час виконання різних сценаріїв використання системи. Це дозволить глибше зрозуміти, як саме функціонують модулі, які дані передаються між ними, і як забезпечується цілісність та коректність роботи системи в цілому [14].

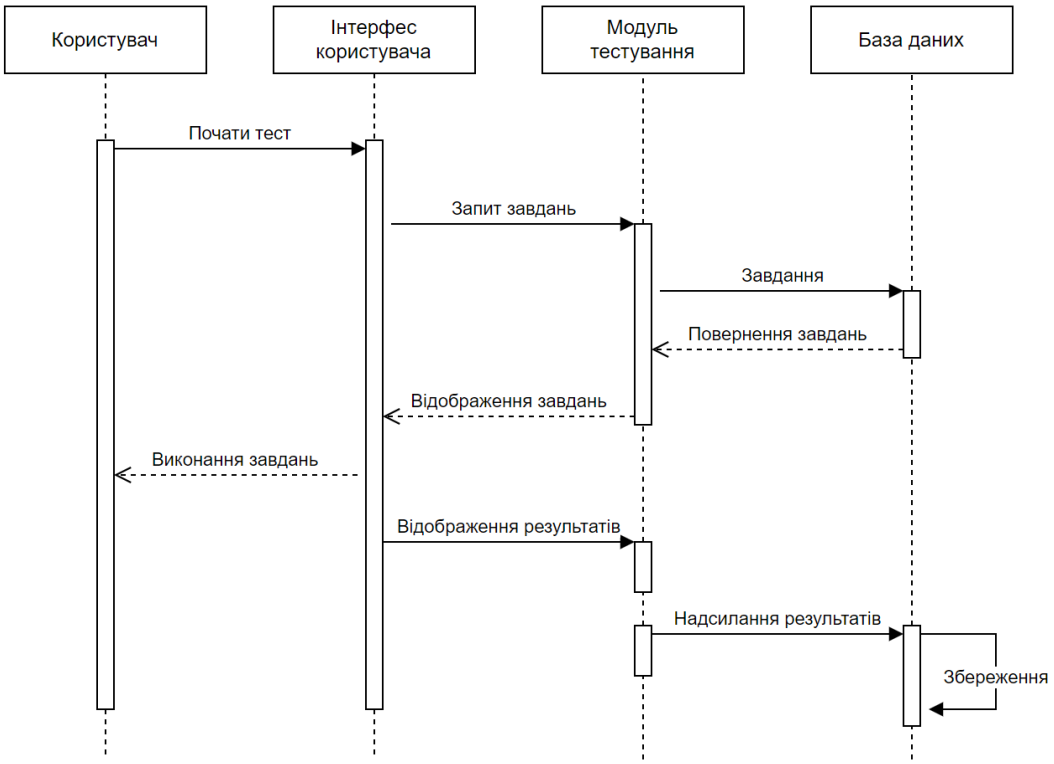

Рисунок 2.3 – Діаграма послідовностей проходження тесту

На діаграмі, що описує процес проходження тесту, користувач натискає кнопку "Почати тест" в інтерфейсі користувача, після чого інтерфейс надсилає запит до модуля тестування. Модуль тестування запитує завдання з бази даних і передає їх інтерфейсу користувача, де користувач виконує завдання. Для тесту з аудіювання інтерфейс також надсилає запит на відтворення аудіо до відповідного модуля, який відтворює аудіофайл. Після завершення тесту результати надсилаються модулю тестування і зберігаються у базі даних.

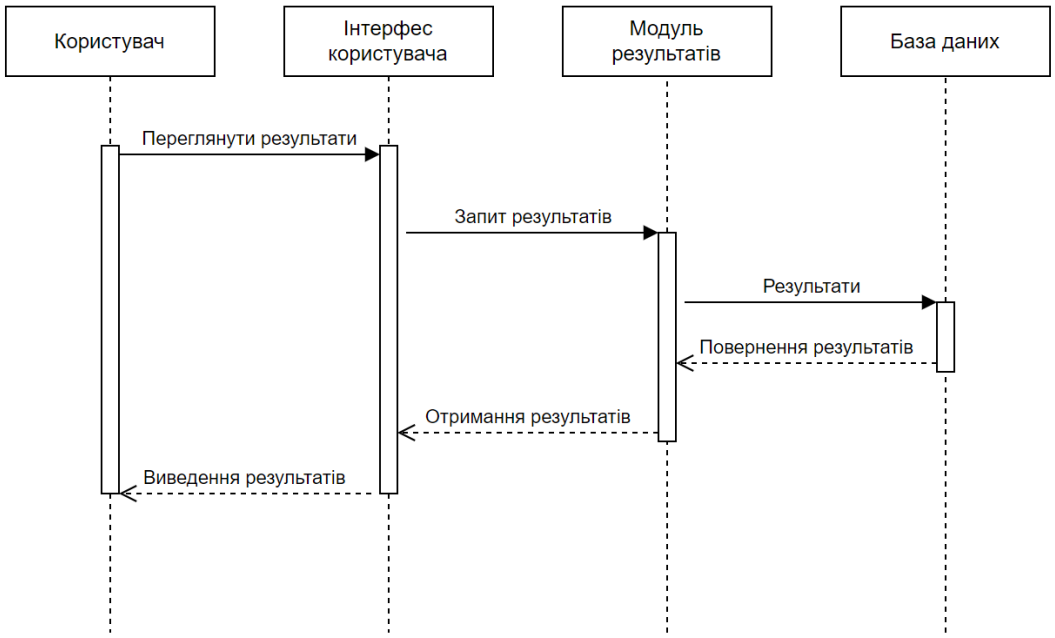

Рисунок 2.4 – Діаграма послідовностей перегляду результатів

Діаграма послідовностей для перегляду результатів демонструє, як користувач натискає кнопку "Переглянути результати" в інтерфейсі користувача, що ініціює запит до модуля результатів. Модуль результатів запитує необхідні дані з бази даних і повертає їх інтерфейсу користувача, який відображає результати користувачу.

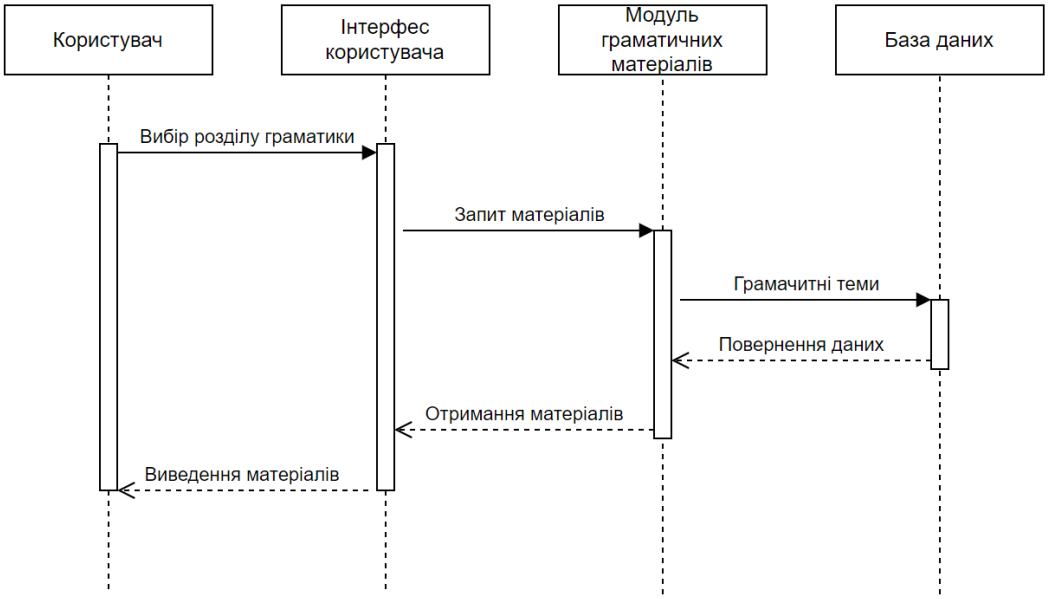

Рисунок 2.5 – Діаграма послідовностей читання граматики

Для процесу читання граматики користувач обирає розділ граматики в інтерфейсі користувача, який надсилає запит до модуля граматичних матеріалів. Модуль граматичних матеріалів отримує відповідні дані з бази даних і передає їх інтерфейсу користувача для відображення користувачу.

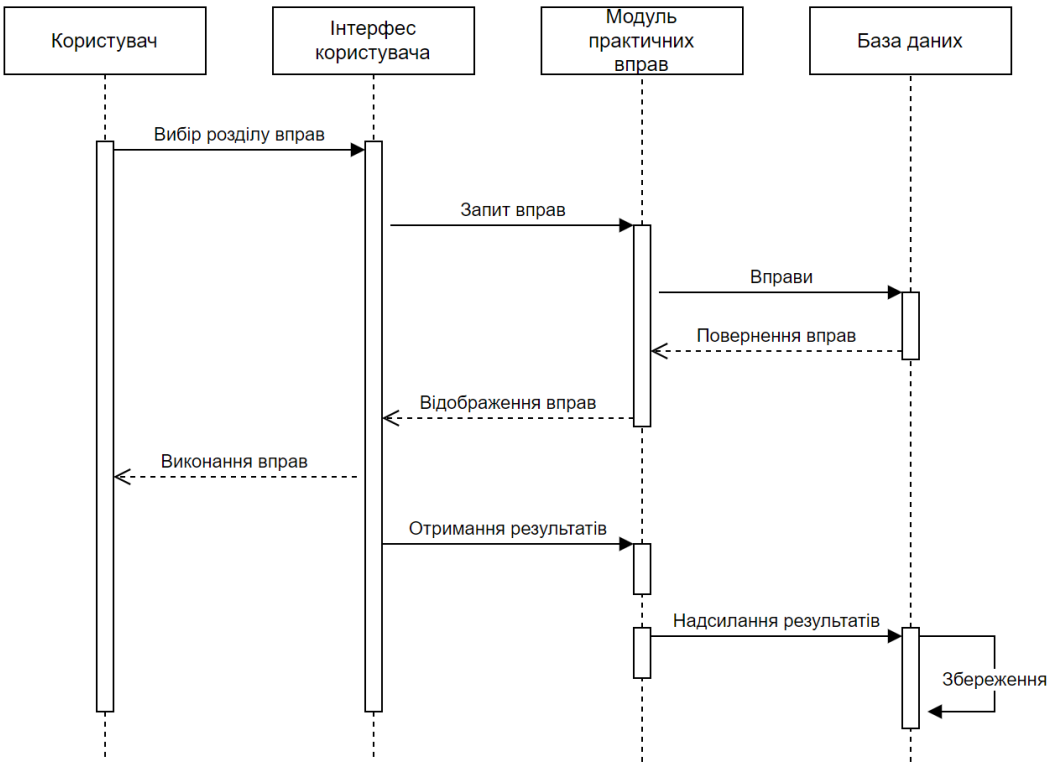

Рисунок 2.6 – Діаграма послідовностей виконання практичних вправ

У випадку виконання практичних вправ користувач обирає потрібну вправу, інтерфейс користувача надсилає запит до модуля практичних вправ, який отримує дані з бази даних і передає їх інтерфейсу користувача. Після виконання вправ користувач надсилає результати назад інтерфейсу користувача, який передає їх модулю практичних вправ для збереження у базі даних та відображення користувачу.

Детальне визначення основних модулів та їх взаємодій є важливим етапом у розробці системи тестування для визначення рівня володіння іноземною мовою. Використання діаграм послідовностей допомагає візуалізувати та зрозуміти

процеси, що відбуваються в системі, забезпечуючи прозорість та ефективність їх реалізації.

2.4 Проектування інтерфейсу системи

2.4.1 Структура та навігація сайту

Для забезпечення інтуїтивно зрозумілої навігації та зручності користування сайтом для тестування рівня володіння іноземною мовою, передбачено декілька основних розділів і категорій контенту. Головна сторінка сайту міститиме кнопки для переходу до трьох основних розділів: тестування, читання граматики та виконання практичних вправ.

Розділ тестування матиме окрему сторінку з підрозділами: словниковий запас, граматика, читання та аудіювання. Кожен підрозділ міститиме сторінки з одним запитанням на сторінку, що забезпечить фокусування користувача на кожному завданні. По завершенні тесту користувач зможе отримати загальні результати з рекомендаціями для подальшого покращення рівня володіння мовою.

Розділ граматики міститиме сторінку з меню для вибору теми, яку користувач бажає вивчати. Це забезпечить структуроване вивчення граматичних правил, розподілених за темами.

Розділ виконання практичних вправ передбачає вибір граматичної теми, за якою користувач виконуватиме завдання. Користувач зможе перевірити правильність виконання завдань у системі, що сприятиме ефективному закріпленню знань.

Такий вибір сторінок і їхня кількість забезпечать зручність у використанні сайту користувачами, дозволяючи їм легко знаходити необхідну інформацію та виконувати завдання. Це сприятиме досягненню високої функціональності сайту, відповідаючи всім вимогам для ефективного вивчення іноземної мови та оцінювання рівня її володіння.

Вибір стилю та візуальної мови для системи тестування рівня володіння іноземною мовою є важливим аспектом, що забезпечує комфортне та ефективне користування сайтом. Оскільки наш сайт орієнтований на широкий спектр користувачів, включаючи тих, хто має мінімальні навички роботи з ПК, ми обрали простий та інтуїтивно зрозумілий стиль. Це забезпечує легке сприйняття та навігацію по сайту.

Шрифти будуть простими, але елегантними, що дозволить легко читати текст та завдання. Такий вибір шрифтів сприяє зменшенню втоми очей та підвищенню зосередженості користувачів на тестах і навчальних матеріалах. Кольори, які будуть використані під час розробки сайту, переважно пастельні, м'які та приємні для очей. Це допомагає користувачам зосередитися на виконанні тестів і вивченні матеріалів, не відволікаючись на яскраві кольори або занадто насичені елементи дизайну.

Такий підхід до стилю та візуальної мови забезпечує кілька ключових переваг. По-перше, він зменшує когнітивне навантаження, дозволяючи користувачам краще концентруватися на завданнях, а не на елементах інтерфейсу. По-друге, пастельні кольори створюють приємну атмосферу, що сприяє тривалому та продуктивному використанню сайту. По-третє, відсутність нав'язливої реклами та складних графічних елементів робить сайт більш доступним і зрозумілим для всіх користувачів.

Важливість забезпечення чіткої та зрозумілої структури інтерфейсу для системи тестування рівня володіння іноземною мовою важко переоцінити. Правильно підібраний стиль є однією з основних складових успіху майбутнього продукту. Інтерфейс повинен бути інтуїтивно зрозумілим і легко доступним, що дозволить користувачам без зайвих зусиль орієнтуватися на сайті та виконувати необхідні дії. Це особливо важливо, оскільки основна аудиторія може мати різний рівень володіння комп'ютером.

Відповідність стилю темі та меті системи грає ключову роль у забезпеченні позитивного користувацького досвіду. Наприклад, спокійні пастельні кольори та

прості, елегантні шрифти створюють сприятливу атмосферу для навчання та тестування, не відволікаючи увагу від основних завдань.

2.4.2 Дизайн інтерфейсу

Опис елементів інтерфейсу є ключовим для створення системи тестування, яка визначає рівень володіння іноземною мовою. Правильно спроектований інтерфейс забезпечує користувачам зручність і інтуїтивність, що сприяє кращому використанню ресурсу. У цьому розділі ми розглянемо детальний опис основних сторінок сайту та їх функціональних елементів.

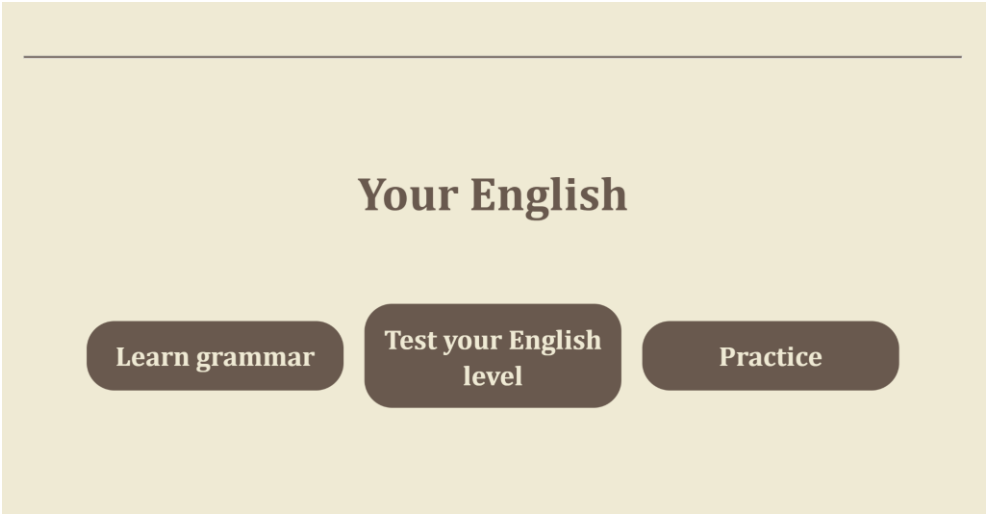

Рисунок 2.7 – Головна сторінка сайту

На головній сторінці сайту розташована назва сервісу по центру, що відразу привертає увагу користувача. Під назвою знаходяться три кнопки, кожна з яких перенаправляє користувача до відповідного розділу сайту: тестування, читання граматичних матеріалів та виконання практичних вправ. Такий підхід забезпечує просту та зрозумілу навігацію.

Рисунок 2.8 – Сторінка тестування

Ця сторінка містить чотири однакові кнопки для кожного розділу тесту: словниковий запас, граматика, читання та аудіювання. Користувач може обрати будь-який розділ, і його перенаправить на відповідну сторінку тесту. Також є кнопка для отримання загальних результатів рівня володіння іноземною мовою після завершення тестування.

Рисунок 2.9 – Сторінка розділів тесту

На кожній сторінці розділу тесту відображається по одному питанню з варіантами відповідей. Кожна сторінка містить номер питання, таймер зворотного відліку часу до завершення тесту та дві кнопки для переходу до наступного або попереднього питання. Для розділу читання додатково міститься текст для читання, а для аудіювання - аудіодоріжка для прослуховування.

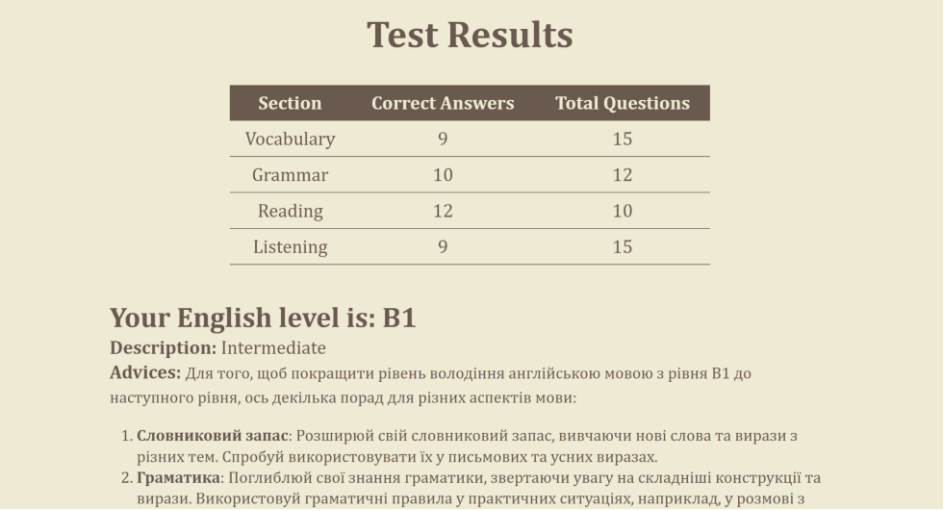

Рисунок 2.10 – Сторінка результатів

На цій сторінці відображається таблиця з кількістю набраних балів та максимально можливою кількістю. Під таблицею наведений поточний рівень володіння іноземною мовою користувача, а також короткі рекомендації щодо покращення знань та посилання на відповідні сервіси для навчання.

Рисунок 2.11 – Сторінка вивчення граматики

На цій сторінці розташоване бічне меню, яке можна згорнути. У меню представлені різні граматичні теми, при натисканні на які відображається текстовий матеріал відповідної теми. Це дозволяє користувачу зручно переходити між темами та вивчати граматику.

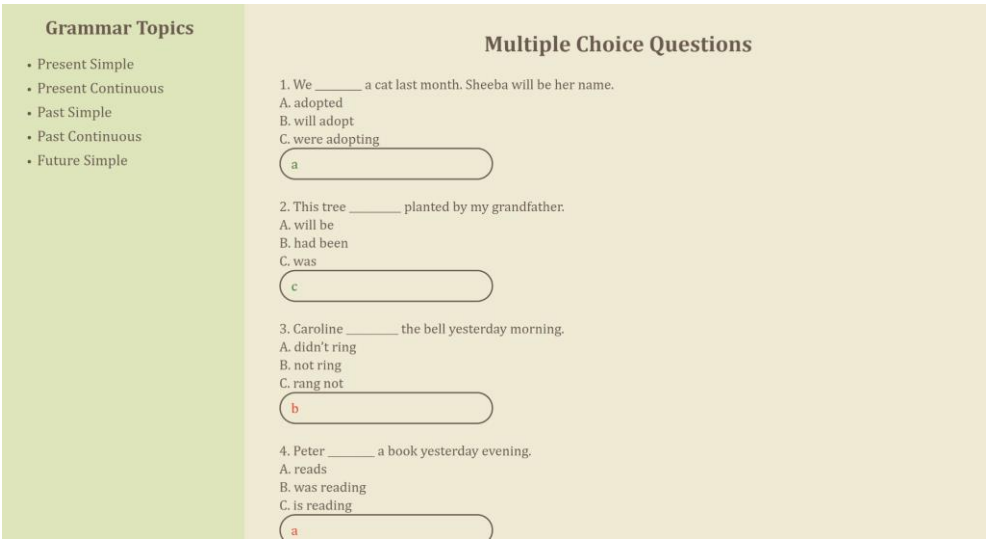

Рисунок 2.12 – Сторінка виконання практичних вправ

На цій сторінці також є бічне меню з різними граматичними темами, яке не згортається. При натисканні на тему відображаються завдання з спеціальним полем для введення відповідей. Після виконання завдань користувач може натиснути кнопку перевірки, яка відобразить кількість правильних та неправильних відповідей, виділяючи їх відповідними кольорами (зеленим для правильних та червоним для неправильних).

Такий підхід до структури та дизайну інтерфейсу забезпечує зручність користування сайтом, сприяє ефективному виконанню завдань і забезпечує високу функціональність системи тестування для визначення рівня володіння іноземною мовою.

2.5 Конструювання програмного забезпечення

2.5.1 Проектування та розробка бази даних

Проектування та розробка бази даних є невід'ємною частиною створення системи тестування для визначення рівня володіння іноземною мовою. У цьому

розділі розглянемо процес підключення та проектування бази даних, використовуючи phpMyAdmin [15].

phpMyAdmin — це потужний інструмент для адміністрування MySQL баз даних, який забезпечує зручний веб-інтерфейс для створення, управління та редагування баз даних. Основні функціональні можливості phpMyAdmin включають створення та видалення баз даних, таблиць, полів, виконання SQL-запитів, управління користувачами та правами доступу [16].

У даному проекті для створення бази даних використовувався phpMyAdmin, вбудований у хостинг InfinityFree.net. Було створено дві таблиці: одну для граматичних тем і одну для вправ. У таблиці для граматичних тем були створені поля з назвою та даними теми. Таблиця для вправ містить поля із завданнями та відповідями.

Використання бази даних значно зменшує розмір самого сайту, що дозволяє прискорити його завантаження та підвищити швидкість обробки запитів. Замість того, щоб зберігати великий обсяг інформації у вигляді статичних сторінок, база даних дозволяє динамічно отримувати необхідні дані та швидко їх обробляти. Це забезпечує ефективну роботу сайту та зручність для користувачів.

Підключення до бази даних відбувається через PHP-код, який виконує запити до бази даних і обробляє результати. Для цього необхідно налаштувати з'єднання з базою даних, використовуючи відповідні параметри, такі як ім'я сервера, ім'я користувача, пароль та назва бази даних. Після встановлення з'єднання можна виконувати SQL-запити для отримання, вставки, оновлення та видалення даних у базі.

Таким чином, проектування та розробка бази даних за допомогою phpMyAdmin та PHP забезпечують ефективне управління даними та підвищують продуктивність системи тестування для визначення рівня володіння іноземною мовою.

2.5.2 Реалізація основних модулів системи

Реалізація основних модулів системи є наступним кроком у створенні ефективної системи тестування для визначення рівня володіння іноземною мовою. На цьому етапі важливо зосередитися на реалізації функціональних можливостей, які забезпечать коректну роботу системи.

Одним із ключових модулів є модуль тестування, який відповідає за обробку відповідей користувачів і збереження результатів. Для цього необхідно реалізувати функцію, яка перевірятиме правильність вибраних відповідей і зберігатиме кількість правильних відповідей у локальному сховищі.

Розглянемо приклад такої функції на JavaScript на рисунку 2.13.

```
function selectAnswer(selected) {
  const current = questions[currentQuestion];

  // Перевірка, чи вибрана відповідь є правильною
  if (selected === current.answer) {
    correctAnswers1++;
  }

  // Увімкнення кнопки "Далі"
  document.getElementById("nextButton").disabled = false;

  localStorage.setItem("correctAnswers1", correctAnswers1);
}
```

Рисунок 2.13 – Функція перевірки правильних відповідей

Ця функція має на меті кілька важливих завдань. По-перше, вона отримує поточне питання з масиву питань та перевіряє, чи вибрана користувачем відповідь є правильною. Якщо відповідь правильна, лічильник правильних відповідей збільшується на одиницю. По-друге, функція активує кнопку "Далі", що дозволяє користувачеві перейти до наступного питання після вибору відповіді. Нарешті, функція зберігає кількість правильних відповідей у локальному сховищі браузера для подальшої обробки та відображення результатів.

Наступною не менш важливою функцією є підрахунок результатів тестування, що дозволяє оцінити рівень знань користувача.

Розглянемо код для підрахунку результатів тестування на рисунку 2.14.

```
      // Отримання кількості правильних відповідей з localStorage
      let correctAnswers1 =
parseInt(localStorage.getItem("correctAnswers1"));
      let correctAnswers2 =
parseInt(localStorage.getItem("correctAnswers2"));
      let correctAnswers3 =
parseInt(localStorage.getItem("correctAnswers3"));
      let correctAnswers4 =
parseInt(localStorage.getItem("correctAnswers4"));

      // Підрахунок загальної кількості правильних та неправильних
відповідей
      let total_correct = correctAnswers1 + correctAnswers2 +
correctAnswers3 + correctAnswers4;
      let total_questions = 15 + 12 + 10 + 15;
      let result_calc = total_questions - total_correct;

      console.log(total_correct);
      console.log(total_questions);
      console.log(result_calc);
```

Рисунок 2.14 – Код для підрахунку результатів

Ця частина коду виконує кілька важливих завдань. По-перше, вона отримує кількість правильних відповідей для кожного з чотирьох розділів тесту з локального сховища браузера (localStorage). Це дозволяє зберігати проміжні результати та забезпечує безперервність тестування, навіть якщо користувач оновлює сторінку або повертається до тесту пізніше.

Далі код підраховує загальну кількість правильних відповідей, додаючи результати всіх чотирьох розділів. Загальна кількість питань також розраховується шляхом додавання кількості питань у кожному розділі. Після цього обчислюється кількість неправильних відповідей шляхом віднімання кількості правильних відповідей від загальної кількості питань.

Наступною важливою частиною є модуль для читання граматичних матеріалів, який забезпечує динамічне завантаження контенту на сторінку, що

підвищує зручність користування сайтом і забезпечує швидкий доступ до необхідної інформації.

Розглянемо приклад коду для завантаження граматичних тем на рисунку 2.15.

```
function showText(num) {
  const xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function () {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("content-wrap").innerHTML =
this.responseText;
      currentText = num;
    }
  };
  xhttp.open("GET", "./php/get_topic.php?id=" + num, true);
  xhttp.send();
}
```

Рисунок 2.15 – Код для завантаження граматичних тем

Ця частина коду виконує кілька важливих завдань. По-перше, функція showText ініціалізує новий запит XMLHttpRequest для завантаження контенту з сервера. Функція приймає параметр num, який використовується для ідентифікації теми граматики, що повинна бути завантажена.

Далі, метод onreadystatechange перевіряє стан запиту. Коли запит досягає стану readyState == 4 і статусу status == 200, що вказує на успішне завершення запиту, відповідь сервера (текст граматичної теми) вставляється в елемент з ідентифікатором content-wrap на сторінці. Це дозволяє динамічно оновлювати зміст сторінки без необхідності її повного перезавантаження.

Метод open налаштовує параметри запиту, а метод send відправляє запит на сервер. Використання асинхронного запиту (true у методі open) дозволяє не блокувати основний потік виконання, забезпечуючи плавну та швидку роботу сайту.

Ця функція є важливою для системи, оскільки вона дозволяє ефективно управляти контентом, зменшуючи навантаження на сервер і підвищуючи швидкість завантаження сторінок. Реалізація цього модуля забезпечує зручний доступ до граматичних матеріалів, що сприяє більш ефективному вивченню мови користувачами.

Наступною важливою частиною для реалізації системи є модуль виконання практичних вправ. Цей модуль дозволяє користувачам закріплювати знання шляхом виконання різних завдань, таких як питання з множинним вибором та завдання на заповнення пропусків. Нижче наведено приклад коду на рисунку 2.16, який відповідає за відображення вправ.

```
function showExercise(topic) {
  const exercisesList = exercises[topic];
  const exercisesHTML = exercisesList
    .map((exercise, index) => {
      let html = "";
      if (exercise.type === "multiple_choice") {
        html += `<h3><strong>Multiple Choice
Questions</strong></h3>`;
        exercise.questions.forEach((question, i) => {
          html += `<p>${i + 1}. ${question.question}</p>`;
          question.answers.forEach((answer, answerIndex) => {
            html += `<label for="question${index}<span
class="math-inline">
                        ${answer}
                      </label><br>`;
          });
          html += `<input type="text" id="answer${index}${i}"
placeholder="Enter the correct letter">`;
        });
      } else if (exercise.type === "fill_in_the_blanks") {
        html += `<h3><strong>Fill in the Blanks</strong></h3>`;
        exercise.questions.forEach((question, i) => {
          html += `<p>${i + 1}. ${question.question}</p>`;
          html += `<input type="text" id="answer${index}${i}"><br>`;
        });
      }
      return html;
    })
    .join("");
  document.getElementById("exercises").innerHTML = exercisesHTML;
  saveData("currentTopic", topic);
}
```

Рисунок 2.16 – Функція для відображення вправ

Ця частина коду виконує кілька ключових функцій. Функція showExercise приймає параметр topic, який визначає тему вправ, що повинні бути відображені. Далі, список вправ (exercisesList) генерується на основі вибраної теми. Для кожної вправи створюється HTML-код, який містить різні типи завдань, такі як питання з множинним вибором та завдання на заповнення пропусків.

Для питань з множинним вибором створюються заголовки, запитання та варіанти відповідей, а також текстове поле для введення правильної відповіді. Для

завдань на заповнення пропусків створюються відповідні запитання та текстові поля для введення відповідей. Згенерований HTML-код вставляється в елемент з ідентифікатором exercises на сторінці, що дозволяє динамічно відображати зміст відповідно до вибраної теми. Функція saveData зберігає поточну тему в локальне сховище для подальшого використання.

Реалізація основних модулів системи є ключовим етапом у розробці ефективного інструменту для визначення рівня володіння іноземною мовою. Включення таких функціональних компонентів, як модулі тестування, читання граматики та виконання практичних вправ, забезпечує всебічний підхід до оцінювання та навчання користувачів. Кожен з модулів грає важливу роль у створенні інтерактивного та зручного інтерфейсу, який сприяє активному залученню користувачів і покращенню їхніх мовних навичок. Завдяки ретельному проектуванню та реалізації цих модулів, система стає ефективним інструментом для вивчення іноземних мов.

3 ВАЛІДАЦІЯ ТА ВЕРИФІКАЦІЯ ВИМОГ

Наступним критичним етапом розробки системи тестування для визначення рівня володіння іноземною мовою є валідація та верифікація вимог. Метою тестування програмного забезпечення є виявлення дефектів на ранніх стадіях розробки, що дозволяє їх своєчасно виправити перед випуском кінцевого продукту.

Тестування програмного забезпечення поділяється на два основні види: функціональне та нефункціональне тестування. Функціональне тестування перевіряє, чи виконує система свої функції відповідно до специфікацій. Воно включає такі підвиди, як модульне тестування, інтеграційне тестування, системне тестування та приймальне тестування. Метою нефункціонального тестування є оцінка таких аспектів, як продуктивність, безпека, зручність користування та сумісність системи.

3.1 Функціональне тестування

Функціональне тестування має на меті перевірку, чи відповідає створена система початковим функціональним вимогам. Це дозволяє переконатися, що всі необхідні функції реалізовані та працюють коректно. Завдання функціонального тестування включають виявлення невідповідностей, помилок та недоліків у функціональності системи [17].

3.1.1 Методи функціонального тестування

Методи функціонального тестування поділяються на три основні категорії: тестування чорного ящика, тестування білого ящика та тестування на основі моделей [18].

1. Тестування чорного ящика - це метод, при якому тестувальник не має доступу до внутрішньої структури або коду програми. Тестування проводиться на основі специфікацій та функціональних вимог. Тестувальник перевіряє, як система обробляє вхідні дані і які результати вона повертає.
2. Тестування білого ящика - це метод, при якому тестувальник має доступ до внутрішньої структури та коду програми. Тестування проводиться з урахуванням логіки роботи коду. Перевіряється кожна гілка, цикл та умова в програмі.
3. Тестування на основі моделей - це метод, при якому тестування проводиться на основі моделей, що описують очікувану поведінку системи. Використовуються формальні моделі для створення тестових сценаріїв. Моделі можуть бути у вигляді діаграм, математичних моделей або специфікацій.

Функціональне тестування забезпечує впевненість у тому, що система працює належним чином і виконує всі необхідні завдання. Вибір методу тестування залежить від конкретних вимог та характеристик проекту, а також від доступних ресурсів і досвіду команди тестувальників.

3.1.2 Техніки функціонального тестування

У процесі розробки системи тестування для визначення рівня володіння іноземною мовою необхідно застосовувати різноманітні техніки функціонального

тестування. Основні техніки функціонального тестування включають еквівалентне розбиття, тестування межових значень та тестування сценаріїв[19].

1. Еквівалентне розбиття: Ця техніка полягає в поділі вхідних даних на класи еквівалентності, в межах яких система повинна реагувати однаково. Це дозволяє зменшити кількість тестових випадків, оскільки достатньо перевірити лише один приклад з кожного класу.
2. Межові значення: Ця техніка включає тестування системи на межах допустимих значень вхідних даних. Це дозволяє виявити помилки, які можуть виникати при обробці граничних значень, що часто є слабкими місцями в програмному забезпеченні.
3. Тестові сценарії: Ця техніка передбачає створення набору сценаріїв, які імітують реальні ситуації використання системи. Це дозволяє перевірити поведінку системи в умовах, наближених до реальних.

Кожна з цих технік є ефективною для тестування програмного забезпечення і дає хороший результат. У даному випадку було створено декілька тестових сценаріїв для перевірки коректності роботи системи в різних випадках.

Для першого тестового сценарію було розглянуто приклад перевірки правильності відповідей у модулі граматичних вправ. Було взято три типи вхідних даних для відповідей: правильні, неправильні та порожні відповіді.

При введенні правильної відповіді система повинна виводити повідомлення про правильну відповідь. При введенні неправильної відповіді система повинна виводити повідомлення про неправильну відповідь. При введенні порожньої відповіді система повинна виводити повідомлення з проханням заповнити всі поля.

Було створено три тестові випадки: введення правильної відповіді, введення неправильної відповіді та залишення поля відповіді порожнім.

При введенні правильної відповіді система коректно вивела повідомлення про правильну відповідь. При введенні неправильної відповіді система правильно вказала на неправильність відповіді. При порожній відповіді система зарахувала це

як помилку, що не є критичною проблемою, але очікуваний результат повинен був бути іншим.

Система коректно реагує на правильні та неправильні відповіді, однак реагує некоректно на порожні відповіді. Ця незначна проблема не впливає на загальний функціонал системи, проте потребує уваги для забезпечення повної відповідності вимогам. Загалом, дана техніка функціонального тестування показала, що система працює належним чином.

У другому тестовому сценарії було перевірено правильність відповіді системи при отриманні результату тестування. У першому випадку користувач проходить всі розділи тесту і намагається отримати свій рівень володіння мовою. У другому - користувач намагається отримати свій рівень володіння мовою, не пройшовши всі розділи тесту.

При спробі дізнатись результат пройшовши всі розділи система повинна правильно надати результат, а при спробі зробити це без результатів з деяких розділів - система видає повідомлення про те, що не всі тести пройдені, і не надає результату.

Під час проведення тестування було виявлено, що система не надає результату стосовно рівня володіння іноземною мовою без проходження всіх розділів тесту і видає повідомлення про те, що потрібно пройти всі розділи.

Отже, система правильно реагує на проходження всіх тестів і надає результат, що підтверджує її коректну роботу в цьому сценарії.

Для останнього прикладу тестового сценарію у нас є наступні вхідні дані: Користувач намагається перейти до наступного питання відповівши на поточне питання та не відповівши на нього.

Система повинна дозволити перехід до наступного питання тільки після того, як користувач відповів на поточне питання. Це забезпечить концентрацію користувача на кожному окремому питанні, зменшить можливість пропуску

запитань і підвищить точність тестування. Кнопка переходу до наступного питання повинна бути неактивною, доки не обрано варіант відповіді.

Під час проведення перевірки система не дозволяє перехід до наступного питання, поки користувач не відповів на поточне питання. Кнопка переходу стає активною лише після вибору відповіді.

Отже, система працює коректно, забезпечуючи, що користувач відповідає на кожне питання перед переходом до наступного. Це сприяє точнішому визначенню рівня володіння мовою і запобігає пропускам питань, що є важливим для досягнення надійних результатів тестування.

Проведення тестування з використанням різних сценаріїв підтвердило, що система тестування для визначення рівня володіння іноземною мовою працює відповідно до вимог. Основні функціональні можливості реалізовані коректно, що дозволяє забезпечити якісний процес тестування. Однак деякі незначні аспекти, як-от обробка порожніх відповідей, потребують подальшого вдосконалення для покращення користувацького досвіду. В цілому, система відповідає заданим вимогам і забезпечує надійні результати тестування.

3.2 Нефункціональне тестування

Нефункціональне тестування спрямоване на оцінку таких аспектів системи, як продуктивність, надійність, масштабованість, зручність використання та безпека. Основною метою цього виду тестування є визначення, чи здатна система ефективно виконувати свої функції під різними умовами експлуатації. Завданням нефункціонального тестування є виявлення можливих проблем, які можуть виникнути під час реального використання системи, що дозволяє вчасно їх усунути та забезпечити високий рівень задоволеності користувачів.

3.2.1 Види нефункціонального тестування

Нефункціональне тестування включає кілька різновидів, кожен з яких має свою специфіку і мету. Серед основних видів нефункціонального тестування можна виділити тестування продуктивності, тестування безпеки, тестування юзабіліті, тестування надійності та тестування сумісності [20].

Тестування продуктивності. Це тестування спрямоване на оцінку швидкості реакції та стабільності системи під різними навантаженнями. Воно включає в себе такі підвиди, як тестування навантаження (load testing) та стрес-тестування (stress testing). Тестування продуктивності дозволяє визначити, як система поводиться при великій кількості одночасних користувачів або обробці великих обсягів даних, і чи здатна вона підтримувати необхідний рівень продуктивності у реальних умовах експлуатації [21].

Тестування безпеки. Метою цього тестування є виявлення та усунення вразливостей у системі, які можуть бути використані для несанкціонованого доступу або інших шкідливих дій. Тестування безпеки включає перевірку автентифікації та авторизації користувачів, захисту даних, безпеки мережевих з'єднань та інших аспектів.

Тестування юзабіліті. Це тестування оцінює зручність використання системи з точки зору кінцевого користувача. Воно включає аналіз інтерфейсу користувача, зрозумілість навігації, зручність введення даних та загальну взаємодію з системою. Метою тестування юзабіліті є забезпечення того, що система є інтуїтивно зрозумілою та зручною для користувачів, що підвищує їх задоволеність та ефективність роботи з системою.

Тестування надійності. Це тестування спрямоване на визначення здатності системи безперервно виконувати свої функції без відмов протягом певного часу. Воно включає виявлення можливих помилок, перевірку механізмів відновлення після збоїв та аналіз стабільності системи під час тривалого використання.

Тестування надійності дозволяє забезпечити стабільну роботу системи у реальних умовах експлуатації [22].

Тестування сумісності. Метою цього тестування є перевірка коректної роботи системи на різних платформах, операційних системах, браузерах та пристроях. Воно включає тестування сумісності з різними версіями програмного забезпечення та обладнання, що використовуються користувачами.

3.2.2 Навантажувальне тестування

Навантажувальне тестування є важливим етапом в розробці системи тестування для визначення рівня володіння іноземною мовою. Це тип нефункціонального тестування, який спрямований на оцінку здатності системи працювати при високому навантаженні. Його основна мета – визначити, як система поводиться, коли на неї одночасно звертається велика кількість користувачів. Навантажувальне тестування дозволяє виявити можливі проблеми з продуктивністю, передбачити реакцію сервісу на пікові навантаження та вжити необхідних заходів для оптимізації роботи системи [23].

Для проведення навантажувального тестування в нашому випадку було обрано веб-сервіс BlazeMeter. Основними перевагами BlazeMeter є простота у використанні, інтуїтивно зрозумілий інтерфейс, можливість створення детальних звітів про результати тестування та безкоштовність.

BlazeMeter дозволяє швидко і легко налаштувати тестування, вибрати необхідні параметри навантаження та отримати детальну інформацію про роботу системи під різними умовами. Це дозволяє виявити вузькі місця в системі та провести їх оптимізацію, що в свою чергу забезпечить надійну роботу системи навіть при великій кількості одночасних користувачів.

Перший тест проводився з навантаженням у 10 користувачів протягом 20 хвилин. Це дозволило оцінити роботу сайту в умовах невеликого навантаження,

наближеного до реальних умов використання. Результати цього тесту представлені на рисунку 3.1.

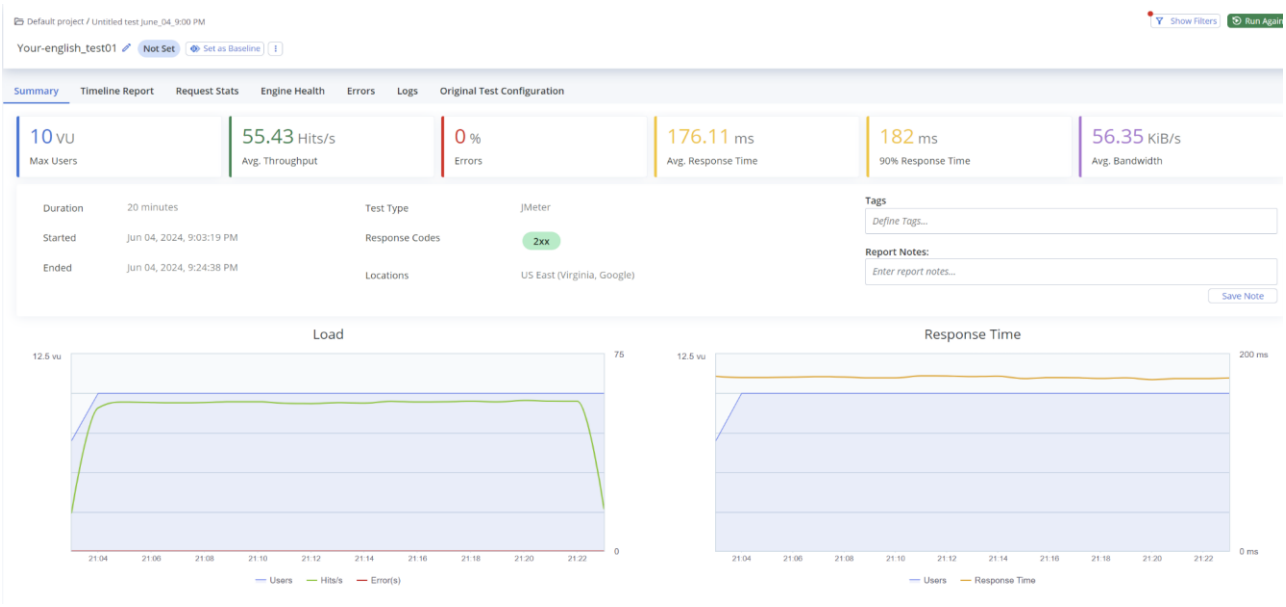

Рисунок 3.1 – Результати першого навантажувального тестування

При такому навантаженні сайт працював добре, майже не видавав помилок, а час очікування відповіді був швидким. Усі функції системи працювали коректно, що свідчить про стабільну роботу сайту при невеликій кількості користувачів.

Другий тест проводився з навантаженням у 20 користувачів протягом 20 хвилин. Результати цього тесту представлені на рисунку 3.2.

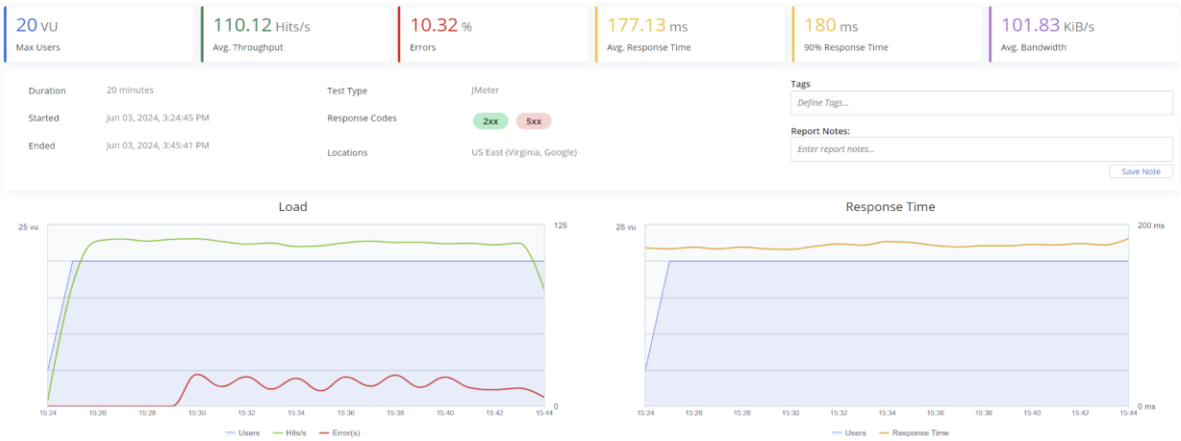

Рисунок 3.2 – Результати другого навантажувального тестування

Під час цього тесту стало помітно, що зі збільшенням кількості користувачів час відповіді системи трохи збільшується, а також починають з'являтись незначні помилки. Проте ці помилки майже не впливають на загальну роботу сайту, і він продовжує повноцінно функціонувати.

Третій тест проводився з навантаженням у 40 користувачів протягом 20 хвилин. Результати цього тесту представлені на рисунку 3.3.

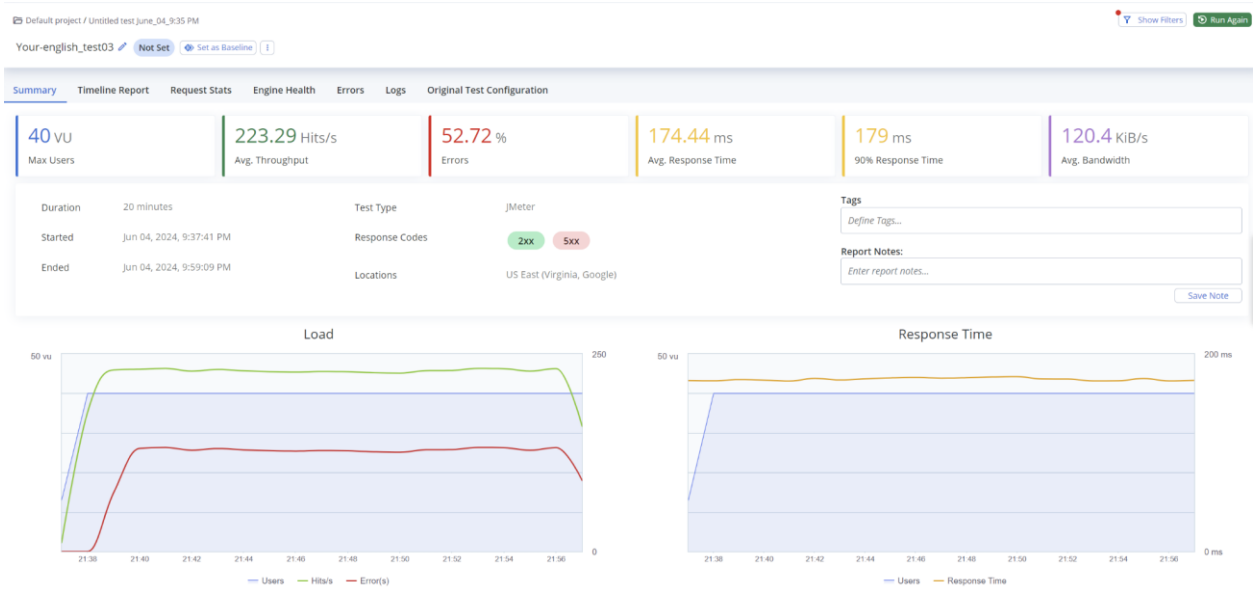

Рисунок 3.3 – Результати третього навантажувального тестування

Цей тест показав, що сайт починає зазнавати значного навантаження. Середній час відповіді системи значно зріс порівняно з попередніми тестами, з'явилися помилки, які, хоч і не повністю блокують роботу сайту, але роблять його повільнішим у відповіді на запити користувачів.

Після проведення навантажувального тестування було виявлено, що система чудово працює при невеликому навантаженні, проте з'являються певні, некритичні помилки при збільшенні кількості користувачів. У цілому сайт залишається функціональним навіть при значному навантаженні, хоча й з деяким зниженням швидкості відповіді. Ці результати вказують на необхідність оптимізації системи

для покращення продуктивності при великих навантаженнях, що забезпечить стабільну та ефективну роботу сайту за будь-яких умов.

3.2.3 Тестування на стійкість до збоїв

Тестування на стійкість до збоїв (Fault Tolerance Testing) є важливим етапом перевірки системи, що дозволяє визначити, як сайт реагує на виключні ситуації та збої, і виявити потенційні недоліки. Метою цього тестування є забезпечення стабільної роботи сайту навіть у випадку непередбачених подій, таких як відмови компонентів або різні технічні проблеми. Виявлення та виправлення цих недоліків на ранніх етапах дозволяє підвищити надійність системи та уникнути критичних проблем у майбутньому [24].

Для проведення тестування на стійкість до збоїв була обрана платформа UptimeRobot. Цей веб-сервіс відзначається простотою у використанні, зрозумілим інтерфейсом та можливістю моніторити доступність сайту в режимі реального часу. Окрім того, сервіс є безкоштовним, що робить його доступним для широкого кола користувачів.

За допомогою UptimeRobot було проведено моніторинг доступності сайту протягом 24 годин. За цей час сайт був доступний 99,9% часу. Тестування доступності веб-сервісу представлено на рисунку 3.4.

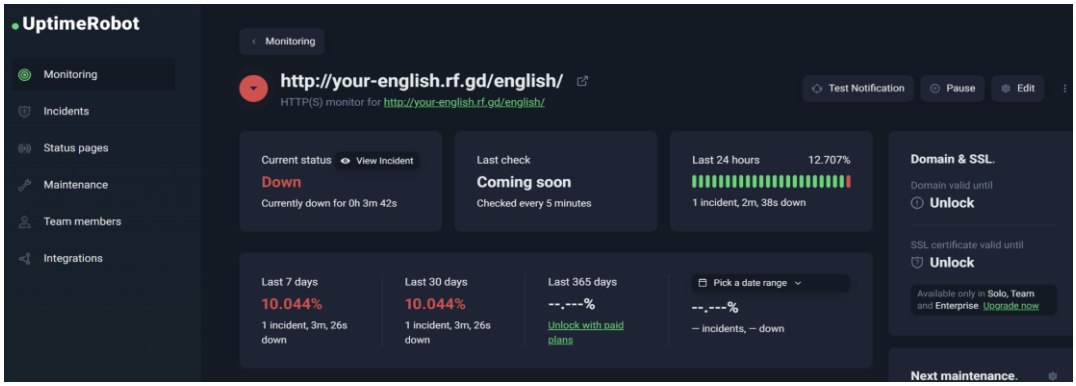

Рисунок 3.4 – Моніторинг доступності

Даний вид тестування системи дозволив виявити, як сайт реагує на різні типи збоїв і непередбачувані ситуації, та оцінити його здатність до швидкого відновлення. Проведене тестування підтвердило важливість цього етапу для забезпечення надійної та стабільної роботи сайту, що є критично важливим для користувачів.

3.2.4 Тестування на різних платформах та браузерах

Тестування на різних платформах та пристроях, або кросбраузерне тестування, є важливим етапом у розробці веб-системи, оскільки дозволяє перевірити, як сайт виглядає і функціонує на різних операційних системах, пристроях та веб-браузерах. Метою цього тестування є забезпечення коректного відображення і функціонування сайту на будь-якому пристрої, з якого користувач може зайти на сайт. Це допомагає налаштувати стилі та функціонал таким чином, щоб користувацький досвід був позитивним незалежно від платформи чи браузера [25].

Для проведення кросбраузерного тестування була обрана платформа LambdaTest. Цей веб-сервіс відзначається простотою у використанні, дозволяє одразу перевіряти сайт на адаптивність та кросбраузерність. LambdaTest надає зручний і лаконічний інтерфейс, показуючи результати тестування чітко і детально. Результати тестування веб-сервісу на різних пристроях та браузерах наведено на рисунках 3.5-3.6.

Рисунок 3.5 – Вигляд веб-сервісу на операційній системі Windows 8.1, браузері Firefox та розширенні екрану 1366x768

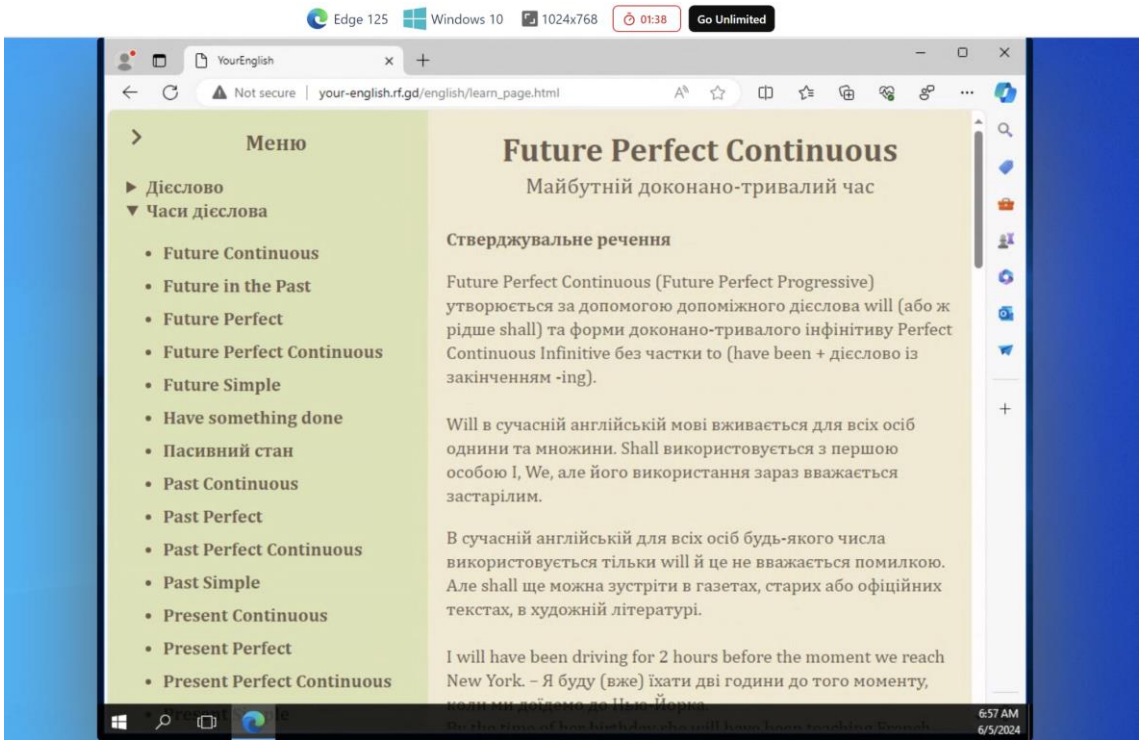

Рисунок 3.6 – Вигляд веб-сервісу на операційній системі Windows 10, браузері Edge та розширенні екрану 1024x768

Після проведення кросбраузерного тестування було виявлено, що сайт коректно відображається та функціонує на різних платформах і пристроях. Це

означає, що сайт готовий до використання широкою аудиторією і забезпечує належний користувацький досвід незалежно від обраного пристрою чи браузера.

Після проведення всіх етапів тестування, включаючи навантажувальне, тестування на стійкість до збоїв та кросбраузерне тестування, було виявлено деякі незначні недоліки, які варто виправити. Проте ці недоліки не є критичними і не заважають повноцінному функціонуванню сайту. В цілому, сайт готовий до використання і забезпечує стабільну та коректну роботу навіть за наявності незначних відхилень. Це підтверджує, що система тестування для визначення рівня володіння іноземною мовою є надійною та готовою до експлуатації.

4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Допомога при теплових і сонячних ударах

Робота за комп'ютером в жарку погоду, особливо в умовах недостатньої вентиляції та відсутності кондиціонування, може значно підвищити ризик теплового або сонячного удару. Особливо це актуально для працівників, які проводять багато часу за ПК в приміщеннях з великими вікнами, через які потрапляє сонячне світло.

Тепловий удар — це стан, який виникає внаслідок тривалого впливу високих температур і підвищеної вологості. Він характеризується значним підвищенням температури тіла, порушенням терморегуляції, та може призвести до серйозних наслідків для здоров'я, включаючи смерть, якщо не надати своєчасну допомогу. Для ефективного надання допомоги при теплових ударах важливо розуміти симптоми, основні заходи першої допомоги, а також профілактичні заходи.

Основні симптоми теплового удару включають:

- Висока температура тіла (понад 40°C).
- Відсутність потіння при високій температурі навколишнього середовища.
- Червона, гаряча і суха шкіра.
- Прискорене серцебиття.
- Слабкість, запаморочення, головний біль.
- Нудота або блювання.
- Сплутаність свідомості або втрата свідомості.

Згідно наказу МОЗ №441 від 09.03.2022р. “Про затвердження порядків надання домедичної допомоги особам при невідкладних станах” [26] надання першої допомоги при тепловому ударі включає наступні кроки:

- Перемістити потерпілого у прохолодне місце, де немає прямих сонячних променів.
- Зняти з нього зайвий одяг, щоб полегшити охолодження тіла.

- Охолодити потерпілого, використовуючи прохолодну воду. Це можна зробити за допомогою мокрих рушників або обтирання шкіри губкою.
- Забезпечити вживання прохолодної води невеликими порціями, якщо потерпілий при свідомості і може пити.
- Викликати швидку допомогу, навіть якщо стан потерпілого покращився.

Одним з найкращих методів профілактики теплових ударів є споживання достатньої кількості води. Для підтримання оптимального водного балансу в умовах високих температур рекомендується випивати 200-300 мл води кожні 20-30 хвилин [27]. Це запобігає дегідратації і допомагає організму ефективніше охолоджуватись. Рекомендований об'єм води залежно від маси тіла наведено в таблиці 4.1. Загальна кількість води, яку потрібно споживати протягом дня, розраховується за формулою:

$$V = 0.033 \times M \ ,$$

де V — об'єм води (літри), необхідний на день, M — маса тіла людини (кг).

Таблиця 4.1 – Рекомендований об'єм води залежно від маси тіла

Маса тіла (кг)	Рекомендований об’єм води (літри)
50	1.65
60	1.98
70	2.31
80	2.64
90	2.97

Сонячний удар є небезпечним станом, що виникає внаслідок тривалого перебування під прямими сонячними променями, коли тепловіддача організму знижується, і відбувається накопичення тепла. Основні симптоми сонячного удару включають головний біль, запаморочення, нудоту, блювання, почервоніння шкіри, підвищення температури тіла, втрату свідомості, та в важких випадках судоми.

Негайна допомога при сонячному ударі є вкрай важливою для збереження здоров’я та життя постраждалого.

Основні заходи першої допомоги при сонячному ударі [28]:

- Переміщення постраждалого в прохолодне місце. Важливо забезпечити доступ свіжого повітря та захист від прямих сонячних променів.
- Охолодження тіла. Потрібно зняти зайвий одяг, обернути постраждалого в мокру тканину або обтирати його холодною водою. Можна прикладати холодні компреси до голови, шиї, пахв та паху.
- Пиття великої кількості води. Постраждалого необхідно змусити пити прохолодну воду або розчини електролітів для відновлення рідини та солей в організмі.
- Виклик швидкої допомоги. У важких випадках необхідно негайно звернутися за медичною допомогою.

Для оцінки ризику сонячного удару можна використовувати формули для розрахунку індексу теплового комфорту та інших показників [29].

Індекс теплового комфорту (ІТК):

$$ITK = T - (\frac{0.55 - 0.0055 \times RH}{100}) \times (T - 14.5) ,$$

де T - температура повітря (°C), RH - відносна вологість повітря (%)

Відповідно до даної формули при ІТК менше 26 ризик сонячного удару є низьким, при 26-32 та 32-38 - помірним та високим відповідно. При індексі теплового комфорту більше 38 ризик сонячного удару є дуже високим, тому важливо уникати тривалого перебування на сонці та підтримувати водний баланс.

У підсумку, своєчасне надання першої допомоги при сонячних ударах та дотримання профілактичних заходів дозволяють знизити ризик серйозних наслідків для здоров'я. Особлива увага повинна приділятися умовам праці, особливо під час роботи за ПК в жарку погоду, щоб забезпечити комфортні умови та знизити ризик теплових уражень.

4.2 Організація служби охорони праці на підприємстві

Організація служби охорони праці на підприємстві є одним з найважливіших аспектів забезпечення безпеки працівників та ефективного функціонування підприємства. Служба охорони праці забезпечує контроль за виконанням вимог охорони праці, сприяє попередженню нещасних випадків і професійних захворювань, а також підвищує загальний рівень безпеки на робочих місцях. У контексті розробки веб-сервісу чи інших IT-продуктів, особливо важливо правильно організувати службу охорони праці для збереження здоров'я та життя працівників.

Основні завдання служби охорони праці [30]:

1. Розробка та впровадження політики охорони праці. Політика охорони праці має охоплювати всі аспекти діяльності підприємства та бути затвердженою керівництвом.
2. Проведення навчання та інструктажу працівників. Регулярне проведення інструктажів з охорони праці допомагає підтримувати високий рівень обізнаності серед працівників про ризики та заходи безпеки.
3. Моніторинг та контроль за виконанням вимог охорони праці. Регулярні перевірки робочих місць, обладнання та процесів дозволяють вчасно виявляти та усувати порушення.
4. Розробка та впровадження заходів для покращення умов праці. Постійне вдосконалення умов праці сприяє зниженню ризиків для здоров'я працівників.

Організація охорони праці на підприємстві починається з формування відповідного підрозділу або призначення відповідальної особи, залежно від кількості працівників та специфіки підприємства. Підрозділ з охорони праці створюється з метою забезпечення виконання вимог законодавства про охорону праці та запобігання виробничим травмам і професійним захворюванням.

На малих підприємствах, де кількість працівників не перевищує 50 осіб, зазвичай призначається одна відповідальна особа, яка буде виконувати функції з охорони праці. Ця особа повинна мати відповідну кваліфікацію та пройти спеціальне навчання. Вона відповідає за організацію та контроль за виконанням заходів з охорони праці, проведення інструктажів, облік і аналіз нещасних випадків та інших питань, пов'язаних з охороною праці.

На середніх підприємствах, де кількість працівників коливається від 50 до 500 осіб, створюється окремий підрозділ з охорони праці. Його очолює начальник відділу з охорони праці, який підпорядковується безпосередньо керівникові підприємства. У складі підрозділу можуть бути інженери з охорони праці, які відповідають за різні напрямки роботи, такі як контроль за дотриманням техніки безпеки, організація навчання працівників, проведення атестації робочих місць тощо.

На великих підприємствах, де кількість працівників перевищує 500 осіб, підрозділ з охорони праці є більш чисельним і структурованим. Очолює його начальник служби охорони праці, який може мати у своєму підпорядкуванні кілька відділів або секторів, кожен з яких відповідає за певний аспект охорони праці. Наприклад, може бути створений відділ з техніки безпеки, відділ з промислової безпеки, відділ з медичного обслуговування та інші. Кожен відділ очолює керівник, який координує роботу своїх підлеглих і підпорядковується начальнику служби охорони праці.

Відповідно до законодавства, підприємства зобов’язані забезпечувати належний рівень охорони праці, призначати відповідальних осіб або створювати спеціалізовані підрозділи, які будуть здійснювати контроль та організацію заходів з охорони праці. Важливо враховувати аспекти охорони праці при розробці IT-продуктів. Наприклад, під час роботи за ПК в жарку погоду, необхідно забезпечити відповідні умови праці, включаючи належну вентиляцію та кондиціонування повітря. У таблиці 4.3 наведено показники температури та рекомендований час перебування на робочому місці в умовах високих температур.

Таблиця 4.3 – Рекомендовані час роботи та перерви

Температура повітря (°C)	Максимальний час роботи (години)	Рекомендований час перерви (хвилини)
25 - 28	8	15
28 - 30	6	20
30 - 32	4	30
понад 32	2	45

Організація служби охорони праці на підприємстві є ключовим елементом забезпечення безпеки працівників та ефективного функціонування підприємства. Дотримання рекомендацій та стандартів з охорони праці дозволяє створити безпечне та комфортне робоче середовище, що позитивно впливає на продуктивність та загальний рівень безпеки [31].

Для ефективного управління охороною праці на підприємстві необхідно постійно вдосконалювати систему охорони праці, враховувати новітні досягнення науки і техніки, а також досвід передових підприємств. Таким чином, правильна організація служби охорони праці на підприємстві є запорукою збереження здоров'я та життя працівників, а також підвищення ефективності та продуктивності праці.

ВИСНОВКИ

У результаті розробки системи тестування для визначення рівня володіння іноземною мовою було досягнуто всіх поставлених на початку цілей. Основна мета проєкту полягала у створенні ефективної та інтуїтивно зрозумілої системи, яка дозволяє користувачам проходити тести для визначення їхнього рівня знань іноземної мови та отримувати детальні рекомендації для покращення. Досягнуті цілі включають розробку функціональної веб-системи, що забезпечує користувачам можливість проходити тести з граматики, читання, словникового запасу та аудіювання; забезпечення зручності використання через простий та інтуїтивно зрозумілий інтерфейс; впровадження системи рекомендацій, що надає користувачам детальні поради для покращення їхніх мовних навичок.

Досягнення цих цілей забезпечило створення надійної, зручної та ефективної системи тестування, яка відповідає вимогам сучасних користувачів та сприяє покращенню їхніх знань іноземної мови. У процесі розробки були враховані всі аспекти, від функціональності та користувацького досвіду до стійкості та надійності системи. Завдяки цьому, кінцевий продукт відповідає високим стандартам якості та забезпечує позитивний користувацький досвід.

Перед початком розробки, було проведено детальний аналіз вимог. Основні вимоги включали створення інтуїтивно зрозумілого інтерфейсу, здатність системи обробляти різні типи тестів, надання детальних рекомендацій користувачам після проходження тестів та забезпечення високої стійкості та надійності системи. Аналіз вимог дозволив чітко визначити ключові функціональні та нефункціональні вимоги, що стало основою для подальшої розробки.

Після аналізу вимог було побудовано UML-діаграми класів, які відображають структуру системи та взаємодію між основними компонентами. Діаграми класів є важливим інструментом в процесі проєктування, оскільки вони допомагають краще зрозуміти архітектуру системи, оптимізувати її структуру та забезпечити коректну взаємодію між компонентами.

Зокрема, створена веб-система демонструє високу продуктивність і стійкість до навантажень. Проведене навантажувальне тестування підтвердило, що система здатна обробляти значну кількість користувачів без суттєвих збоїв чи затримок. Це гарантує, що користувачі можуть отримувати стабільні та надійні результати тестування незалежно від кількості одночасних відвідувачів сайту. Виявлені незначні помилки при високих навантаженнях не впливають на загальну роботу системи і можуть бути легко виправлені в подальшому.

Крім того, тестування на різних платформах та пристроях забезпечило повну кросплатформність системи. Це означає, що користувачі можуть проходити тести та отримувати результати на будь-якому пристрої чи браузері, що робить систему зручною та доступною для широкого кола користувачів.

У результаті розробки було створено систему, яка не тільки відповідає всім поставленим цілям, але й перевершує очікування щодо функціональності, зручності та надійності. Система тестування для визначення рівня володіння іноземною мовою готова до використання і може ефективно допомагати користувачам у покращенні їхніх мовних навичок. Загальні результати роботи свідчать про високу якість розробки та відповідність всім сучасним вимогам.

Отже, проведене тестування та аналіз підтвердили, що система є готовою до використання і забезпечує стабільну роботу навіть за умов високого навантаження. Таким чином, розроблена система тестування для визначення рівня володіння іноземною мовою є ефективним інструментом для оцінки та покращення мовних навичок користувачів, забезпечуючи зручність, надійність та доступність.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. English Language Proficiency 2021 [Електронний ресурс] // EF Education First – Режим доступу до ресурсу: https://www.ef.com/assetscdn/WIBIwq6RdJvcD9bc8RMd/cefcom-epi-site/reports/2021/ef-epi-2021-english.pdf.
2. SMART criteria [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/SMART_criteria.
3. LanGeek [Електронний ресурс] // LanGeek – Режим доступу до ресурсу: https://langeek.co/.
4. Duolingo [Електронний ресурс] // Duolingo – Режим доступу до ресурсу: https://uk.duolingo.com/.
5. Test-English [Електронний ресурс] // Test-English – Режим доступу до ресурсу: https://test-english.com/.
6. Гордієнко О. М. Інформаційні системи: аналіз та проектування / О. М. Гордієнко, О. С. Макарчук, О. В. Симоненко., 2023.
7. Language proficiency [Електронний ресурс] // Wikipedia – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Language_proficiency.
8. Гордієнко О. М. Проектування інформаційних систем: методологія та технології / О. М. Гордієнко, О. С. Макарчук, О. В. Симоненко., 2021.
9. Синтез методологій проектування веб-ресурсів освітніх установ [Електронний ресурс] – Режим доступу до ресурсу: https://ekmair.ukma.edu.ua/items/5dcb5371-b07f-45b9-bb21-efe79a1cf187.
10. JavaScript MDN [Електронний ресурс] // Mozilla Developer Network – Режим доступу до ресурсу: https://developer.mozilla.org/uk/docs/Web/JavaScript.

11. Кучеров С. О. Бази даних: основи та проектування / С. О. Кучеров, О. В. Кучеров, О. М. Кучеров., 2023.

12. Гордієнко О. М. Моделювання програмного забезпечення: UML, аналіз та проектування / О. М. Гордієнко, О. С. Макарчук, О. В. Симоненко., 2023.

13. Class Diagram [Електронний ресурс] // Sparx Systems – Режим доступу до ресурсу: https://sparxsystems.com/enterprise_architect_user_guide/16.1/modeling_languages/classdiagram.html.

14. Запорожець О. І. Інженерія програмного забезпечення: UML-моделювання / О. І. Запорожець, О. В. Кучеренко, О. М. Кучеренко., 2021.

15. Як створити базу даних MySQL з phpMyAdmin [Електронний ресурс] // WordPress – Режим доступу до ресурсу: https://developer.wordpress.org/advanced-administration/before-install/creating-database/.

16. Навчальний посібник phpMyAdmin [Електронний ресурс] // phpMyAdmin – Режим доступу до ресурсу: https://www.phpmyadmin.net/docs/.

17. Функціональне тестування: детальний огляд [Електронний ресурс] // Guru99 – Режим доступу до ресурсу: https://www.guru99.com/functional-testing.html.

18. Кучеренко О. В. Функціональне тестування програмного забезпечення: теорія та практика / О. В. Кучеренко., 2022.

19. Техніки функціонального тестування [Електронний ресурс] // ZapTest – Режим доступу до ресурсу: https://www.zaptest.com/uk/топ-10-найкращих-інструментів-та-програ.

20. Нефункціональне тестування: детальний огляд [Електронний ресурс] // Guru99 – Режим доступу до ресурсу: https://www.guru99.com/non-functional-testing.html.

21. Макаров І. М. Навантажувальне тестування та тестування продуктивності: практичний посібник / І. М. Макаров, О. В. Симоненко, О. І. Запорожець., 2021.

22. What is Reliability Testing? A Comprehensive Guide [Електронний ресурс] // Software Testing Help – Режим доступу до ресурсу: https://www.softwaretestinghelp.com/.

23. Методи навантажувального тестування програмного забезпечення [Електронний ресурс] // DevOpsLab – Режим доступу до ресурсу: https://uk.myservername.com/what-is-reliability-testing.

24. What is Failover Testing? A Comprehensive Guide [Електронний ресурс] // Software Testing Help – Режим доступу до ресурсу: https://www.softwaretestinghelp.com/.

25. Cross-Platform Testing: A Beginner's Guide [Електронний ресурс] // QASource – Режим доступу до ресурсу: https://www.qasource.com/.

26. ПОРЯДОК надання домедичної допомоги постраждалим при тепловому ударі [Електронний ресурс] // Міністерство охорони здоров’я України – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/z0356-22#n769.

27. Як підтримувати водний баланс в спеку [Електронний ресурс] // інфографіка від журналу "Здоров'я України" – Режим доступу до ресурсу: https://health-ua.com/newspaper/med_gaz_zu.

28. Волянський А. І. Перша допомога при невідкладних станах / А. І. Волянський, В. М. Фомін..

29. Методи розрахунку індексу теплового комфорту [Електронний ресурс] // веб-сайт Національного університету "Львівська політехніка" – Режим доступу до ресурсу: https://lpnu.ua/.

30. Постанова "Про затвердження Типового положення про службу охорони праці" [Електронний ресурс] // Кабінет Міністрів України – Режим доступу до ресурсу: https://zakon.rada.gov.ua/go/z1526-04.

31. Організація служби охорони праці на підприємстві: правові та практичні аспекти [Електронний ресурс] // журнал \"Охорона праці\" – Режим доступу до ресурсу: https://nmuofficial.com/news/vyjshov-z-druku-pidruchnyk-ohorona-pratsi-v-medychnij-galuzi-pid-redaktsiyeyu-akademika-o-p-yavorovskogo/.

ДОДАТКИ

ДОДАТОК А

Тези конференції

УДК 004.41:371.13:37.06
В. Бурса, І. Мудрик
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

АВТОМАТИЗОВАНА СИСТЕМА ОЦІНКИ РІВНЯ ЗНАННЯ ІНОЗЕМНОЇ МОВИ З ВИКОРИСТАННЯМ ШТУЧНОГО ІНТЕЛЕКТУ

UDC 004.41:371.13:37.06
V. Bursa, I. Mudryk

AUTOMATED LANGUAGE PROFICIENCY ASSESSMENT SYSTEM UTILIZING ARTIFICIAL INTELLIGENCE

Зважаючи на сучасні вимоги до освіти та розвитку, розробка ефективної системи тестування для визначення рівня володіння іноземною мовою є актуальною задачею. Для оцінки рівня володіння мовою існують різні методи, такі як тести, співбесіди та письмові завдання. Однак ці методи часто є суб'єктивними та потребують значних часових витрат. [1]

Виникає питання: чи можна розробити веб-сайт, який використовує комп'ютерне тестування та штучний інтелект (ШІ) для більш об'єктивного та швидкого визначення рівня володіння іноземною мовою?

Метою цієї роботи є розробка веб-сайту, який пропонує користувачам комплексне тестування для визначення їх рівня володіння англійською мовою.

Основна ідея полягає в розподілі тестів за рівнями складності, що дозволить краще визначити поточний рівень володіння мовою. Кожен з чотирьох розділів тесту міститиме відповідно завдання з граматики, словникового запасу, читання та аудіювання. [2]

Очікується, що веб-сайт пропонуватиме користувачам об'єктивну оцінку їх рівня володіння англійською мовою, індивідуальні поради щодо вдосконалення знань та можливість переглянути граматичні правила та виконати граматичні вправи.

Для майбутніх досліджень пропонується розширити функціонал системи, включаючи додаткові види вправ та розширення бази питань. Використання штучного інтелекту у системі дозволить підвищити точність визначення рівня володіння, а також підбирати наступні питання з урахуванням часу відповіді та їх правильності. Такий підхід дозволить зробити систему більш привабливою для користувачів та забезпечити їм більш ефективний спосіб навчання англійської мови.

Література

1. Штучний інтелект та майбутнє тестування мовних навичок. [Ел. ресурс]. − Режим доступу: https://blog.duolingo.com/the-duolingo-english-test-ai-driven-language-assessment/

2. Guidelines for Best Test Development Practices to Ensure Validity and Fairness for International English Language Proficiency Assessments. [Ел. ресурс]. – Режим доступу: https://www.ets.org/pdfs/about/best-practices-validity-fairness-ela.pdf

ДОДАТОК Б

Програмний код

Файл index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>YourEnglish</title>
    <link rel="stylesheet" href="css/style.css" />
  </head>
  <body>
    <hr
      style="
        height: 4px;
        background-color: #6c584c;
        width: 95%;
        margin-top: 80px;
      "
    />
    <h1>Your English</h1>

    <div class="options-btns">
      <a href="./learn_page.html" class="learn-btn">Learn grammar</a>
      <a href="./test_begin.html" class="test-btn">Test your English
level</a>
      <a href="./practice_page.html" class="pract-btn">Practice</a>
    </div>
  </body>
</html>
```

Файл test_begin.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>YourEnglish</title>
    <link rel="stylesheet" href="css/test_begin_style.css" />
  </head>
  <body>
    <a href="./vocabulary.html" class="vocab-btn">Vocabulary</a>
    <a href="./grammar.html" class="gram-btn">Grammar</a>
    <a href="./reading.html" class="read-btn">Reading</a>
    <a href="./listening.html" class="list-btn">Listening</a>
    <a href="./get_the_result.html" class="result-btn">Get the result</a>

    <div class="modal-wrap">
      <div class="modal" id="before-start">
        <button class="modal-close" data-modal="#before-
start">Close</button>
        <p xml:lang="uk">
          Цей тест спрямований на детальне визначення рівня володіння
```

```
          англійською мовою. Він скадається з чотирьох розділів, які
покривають
          основні потреби у використанні англійської мови. <br /><br />
          На проходження перших двох розділів тесту (а саме перевірка
          словникового запасу та граматики) вам буде дано 25 хвилин, у
кожному з
          них буде 15 запитань різного рівня складності. Для перевірки
читання
          вам буде надано 12 різних текстів з одним питанням до кожного
тексту,
          тому уважко читайте кожен з текстів, щоб обрати правильну
відповідь.
          Останнім буде розділ для перевірки сприйняття мови на слух. До
кожного
          з 10 питань буде окрема невелика аудіодоріжка з відповідним
питанням.
          Для проходження останнього розділу вам буде дано 35 хвилин.
          <br /><br />
          Після проходження тесту вам буде показано ваш поточний рівень
          володіння мовою, також буде дано декілька порад щодо того які
теми у
          вас доволі слабкі та як їх покращити.<br /><br />
          Обов'язково пройдіть всі етапи тестування перед отриманням
результату,
          так як в іншому випадку результат може бути неточним.
        </p>
        <button class="start-btn">Start test</button>
      </div>
    </div>

    <script src="js/test_begin_script.js"></script>
  </body>
</html>
```

Файл vocabulary.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>YourEnglish</title>
    <link rel="stylesheet" href="./css/vocabulary_style.css" />
  </head>
  <body>
    <h1>Vocabulary</h1>
    <section class="additional-container">
      <div id="questionNumber"></div>
      <div id="timer"></div>
    </section>
    <div class="question" id="question"></div>
    <form id="options"></form>
    <section class="nav-btns">
      <button onclick="prevQuestion()" id="prevButton" disabled>
        Previous
      </button>
      <button onclick="nextQuestion()" id="nextButton"
disabled>Next</button>
```

```
    </section>
    <div id="result1" style="display: none"></div>
    <div id="timeSpent"></div>
    <button onclick="returnHome()" id="homeButton" style="display: none">
      Next test
    </button>

    <script src="./js/vocabulary_script.js"></script>
  </body>
</html>
```

Файл grammar.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>YourEnglish</title>
    <link rel="stylesheet" href="css/grammar_style.css" />
  </head>
  <body>
    <h1>Grammar</h1>
    <section class="additional-container">
      <div id="questionNumber"></div>
      <div id="timer"></div>
    </section>
    <div class="question" id="question"></div>
    <form id="options"></form>
    <section class="nav-btns">
      <button onclick="prevQuestion()" id="prevButton" disabled>
        Previous
      </button>
      <button onclick="nextQuestion()" id="nextButton"
disabled>Next</button>
    </section>
    <div id="result2" style="display: none"></div>
    <div id="timeSpent"></div>
    <button onclick="returnHome()" id="homeButton" style="display: none">
      Next test
    </button>

    <script src="./js/grammar_script.js"></script>
  </body>
</html>
```

Файл reading.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>YourEnglish</title>
    <link rel="stylesheet" href="./css/reading_style.css" />
  </head>
  <body>
    <h1>Reading</h1>
```

```
    <section class="additional-container">
      <div id="questionNumber"></div>
      <div id="timer"></div>
    </section>
    <div class="text" id="text"></div>
    <div class="question" id="question"></div>
    <form id="options"></form>
    <section class="nav-btns">
      <button onclick="prevQuestion()" id="prevButton" disabled>
        Previous
      </button>
      <button onclick="nextQuestion()" id="nextButton"
disabled>Next</button>
    </section>
    <div id="result3" style="display: none"></div>
    <div id="timeSpent"></div>
    <button onclick="returnHome()" id="homeButton" style="display: none">
      Next test
    </button>

    <script src="./js/reading_script.js"></script>
  </body>
</html>
```

Файл listening.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>YourEnglish</title>
    <link rel="stylesheet" href="css/listening_style.css" />
  </head>
  <body>
    <h1>Listening</h1>
    <section class="additional-container">
      <div id="questionNumber"></div>
      <div id="timer"></div>
    </section>
    <audio id="audio" src="" controls></audio>
    <div class="question" id="question"></div>
    <form id="options"></form>
    <section class="nav-btns">
      <button onclick="prevQuestion()" id="prevButton" disabled>
        Previous
      </button>
      <button onclick="nextQuestion()" id="nextButton"
disabled>Next</button>
    </section>
    <div id="result4" style="display: none"></div>
    <div id="timeSpent"></div>
    <button onclick="returnHome()" id="homeButton" style="display: none">
      Next test
    </button>

    <script src="./js/listening_script.js"></script>
  </body>
```

```
</html>
```

Файл get_the_result.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>YourEnglish</title>
    <link rel="stylesheet" href="./css/get_result_style.css" />
  </head>
  <body>
    <h1>Test Results</h1>
    <table class="result-table">
      <tr>
        <th>Section</th>
        <th>Correct Answers</th>
        <th>Total Questions</th>
      </tr>
      <tr>
        <td>Vocabulary</td>
        <td><span id="result1"></span></td>
        <td>15</td>
      </tr>
      <tr>
        <td>Grammar</td>
        <td><span id="result2"></span></td>
        <td>12</td>
      </tr>
      <tr>
        <td>Reading</td>
        <td><span id="result3"></span></td>
        <td>10</td>
      </tr>
      <tr>
        <td>Listening</td>
        <td><span id="result4"></span></td>
        <td>15</td>
      </tr>
    </table>

    <section class="results-desc">
      <h2>Your English level is: <span id="level"></span></h2>
      <p><strong>Description: </strong><span id="description"></span></p>
      <p><strong>Advices: </strong><span id="advice"></span></p>
    </section>

    <script src="./js/get_result_script.js"></script>
  </body>
</html>
```

Файл learn_page.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
```

```
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>YourEnglish</title>

    <link
      rel="stylesheet"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css"
    />
    <link rel="stylesheet" href="./css/learn_page_style.css" />
  </head>

  <body>
    <div class="main-wrap">
      <div class="sidebar" id="sidebar">
        <div class="menu-toggle" onclick="toggleMenu()">
          <i id="arrow" class="fas fa-chevron-right"></i>
          <span id="menu-text">Меню</span>
        </div>
        <div class="hidden-text">
          <details>
            <summary>Дієслово</summary>
            <ul>
              <li>
                <div class="menu-items"
onclick="showText(1)">Герундій</div>
              </li>
              <li>
                <div class="menu-items"
onclick="showText(2)">Інтфінітив</div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(3)">
                  Неправильні дієслова
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(4)">
                  Спосіб дієслова
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(5)">
                  Дієприкметник
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(6)">
                  Фразові дієслова
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(7)">
                  Умовний спосіб
                </div>
              </li>
              <li>
```

```
                <div class="menu-items" onclick="showText(8)">
                  Дієслово to be
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(9)">Види
дієслів</div>
              </li>
            </ul>
          </details>
          <details>
            <summary>Часи дієслова</summary>
            <ul>
              <li>
                <div class="menu-items" onclick="showText(10)">
                  Future Continuous
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(11)">
                  Future in the Past
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(12)">
                  Future Perfect
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(13)">
                  Future Perfect Continuous
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(14)">
                  Future Simple
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(15)">
                  Have something done
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(16)">
                  Пасивний стан
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(17)">
                  Past Continuous
                </div>
              </li>
              <li>
                <div class="menu-items" onclick="showText(18)">
                  Past Perfect
                </div>
              </li>
```

```
                <li>
                  <div class="menu-items" onclick="showText(19)">
                    Past Perfect Continuous
                  </div>
                </li>
                <li>
                  <div class="menu-items" onclick="showText(20)">Past
Simple</div>
                </li>
                <li>
                  <div class="menu-items" onclick="showText(21)">
                    Present Continuous
                  </div>
                </li>
                <li>
                  <div class="menu-items" onclick="showText(22)">
                    Present Perfect
                  </div>
                </li>
                <li>
                  <div class="menu-items" onclick="showText(23)">
                    Present Perfect Continuous
                  </div>
                </li>
                <li>
                  <div class="menu-items" onclick="showText(24)">
                    Present Simple
                  </div>
                </li>
                <li>
                  <div class="menu-items" onclick="showText(25)">
                    Узгодження часів
                  </div>
                </li>
              </ul>
            </details>
            <details>
              <summary>Модальні дієслова</summary>
              <ul>
                <li>
                  <div lang="en" class="menu-items" onclick="showText(26)">
                    Can/Could
                  </div>
                </li>
                <li>
                  <div lang="en" class="menu-items" onclick="showText(27)">
                    Dare
                  </div>
                </li>
                <li>
                  <div lang="en" class="menu-items" onclick="showText(28)">
                    Had better
                  </div>
                </li>
                <li>
                  <div lang="en" class="menu-items" onclick="showText(29)">
                    Have to
                  </div>
```

```
          </li>
          <li>
            <div lang="en" class="menu-items" onclick="showText(30)">
              May/Might
            </div>
          </li>
          <li>
            <div lang="en" class="menu-items" onclick="showText(31)">
              Must
            </div>
          </li>
          <li>
            <div lang="en" class="menu-items" onclick="showText(32)">
              Need
            </div>
          </li>
          <li>
            <div lang="en" class="menu-items" onclick="showText(33)">
              Shall
            </div>
          </li>
          <li>
            <div lang="en" class="menu-items" onclick="showText(34)">
              Should/Ought to
            </div>
          </li>
          <li>
            <div lang="en" class="menu-items" onclick="showText(35)">
              To be to
            </div>
          </li>
          <li>
            <div lang="en" class="menu-items" onclick="showText(36)">
              Will/Would
            </div>
          </li>
        </ul>
      </details>
    </div>
  </div>
  <div class="content-wrap" id="content-wrap">
    <div id="text1" class="hidden"></div>
    <div id="text2" class="hidden"></div>
    <div id="text3" class="hidden"></div>
    <div id="text4" class="hidden"></div>
    <div id="text5" class="hidden"></div>
    <div id="text6" class="hidden"></div>
    <div id="text7" class="hidden"></div>
    <div id="text8" class="hidden"></div>
    <div id="text9" class="hidden"></div>
    <div id="text10" class="hidden"></div>
    <div id="text11" class="hidden"></div>
    <div id="text12" class="hidden"></div>
    <div id="text13" class="hidden"></div>
    <div id="text14" class="hidden"></div>
    <div id="text15" class="hidden"></div>
    <div id="text16" class="hidden"></div>
    <div id="text17" class="hidden"></div>
```

```
        <div id="text18" class="hidden"></div>
        <div id="text19" class="hidden"></div>
        <div id="text20" class="hidden"></div>
        <div id="text21" class="hidden"></div>
        <div id="text22" class="hidden"></div>
        <div id="text23" class="hidden"></div>
        <div id="text24" class="hidden"></div>
        <div id="text25" class="hidden"></div>
        <div id="text26" class="hidden"></div>
        <div id="text27" class="hidden"></div>
        <div id="text28" class="hidden"></div>
        <div id="text29" class="hidden"></div>
        <div id="text30" class="hidden"></div>
        <div id="text31" class="hidden"></div>
        <div id="text32" class="hidden"></div>
        <div id="text33" class="hidden"></div>
        <div id="text34" class="hidden"></div>
        <div id="text35" class="hidden"></div>
        <div id="text36" class="hidden"></div>
      </div>
    </div>

    <script src="./js/learn_page_script.js"></script>
  </body>
</html>
```

Файл practice_page.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>YourEnglish</title>

    <link rel="stylesheet" href="./css/practice_page_style.css" />
  </head>
  <body>
    <div class="container">
      <div class="sidebar">
        <h2>Grammar Topics</h2>
        <ul id="topic-list">
          <li>
            <a href="#" onclick="showExercise('topic1')">Present Simple</a>
          </li>
          <li>
            <a href="#" onclick="showExercise('topic2')">Present
Continuous</a>
          </li>
          <li><a href="#" onclick="showExercise('topic3')">Past
Simple</a></li>
          <li>
            <a href="#" onclick="showExercise('topic4')">Past
Continuous</a>
          </li>
          <li>
            <a href="#" onclick="showExercise('topic5')">Future Simple</a>
          </li>
```

```
        </ul>
      </div>
      <div class="main-content">
        <div id="exercises">
          <!-- Exercises will be dynamically loaded here -->
        </div>
        <button onclick="checkAnswers()">Check Answers</button>
        <div id="results"></div>
      </div>
    </div>
    <script src="./js/practice_page_script.js"></script>
  </body>
</html>
```

Файл test_begin_script.js

```
document.addEventListener("DOMContentLoaded", function () {
  let modalWrap = document.querySelector(".modal-wrap");
  let modalShown = localStorage.getItem("modalShown");

  if (!modalShown) {
    modalWrap.classList.remove("hide");
    modal.classList.remove("hide");
    localStorage.setItem("modalShown", true);
  }
});

function closeModal() {
  let modalId = this.dataset.modal;
  document.querySelector(modalId).parentElement.classList.add("hide");
  document.querySelector(modalId).classList.add("hide");
}

function closeModalWrap() {
  this.classList.add("hide");
  this.children[0].classList.add("hide");
}

document.querySelectorAll(".modal-close").forEach(function (element) {
  element.onclick = closeModal;
});

document.querySelectorAll(".modal-wrap").forEach(function (element) {
  element.onclick = closeModalWrap;
});
```

Файл vocabulary_script.js

```
const questions = [
  {
    question:
      "The weather was __________, so we decided to have a picnic
indoors.",
    options: ["pleasant", "enjoyable", "delightful", "favorable"],
    answer: "pleasant",
  },
  {
    question: "She has a __________ sense of humor and always makes me
laugh.",
    options: ["keen", "sharp", "witty", "smart"],
```

```
    answer: "witty",
  },
  {
    question:
      "After a long day at work, he __________ to relax by reading a
book.",
    options: ["likes", "prefers", "enjoys", "favors"],
    answer: "prefers",
  },
  {
    question: "The students were __________ for the upcoming exam.",
    options: ["studying", "reviewing", "preparing", "practicing"],
    answer: "preparing",
  },
  {
    question: "The movie was so __________ that I fell asleep halfway
through.",
    options: ["boring", "dull", "uninteresting", "tedious"],
    answer: "boring",
  },
  {
    question: "Despite the __________ weather, they decided to go hiking.",
    options: ["adverse", "harsh", "severe", "extreme"],
    answer: "adverse",
  },
  {
    question:
      "She __________ her career as a teacher before becoming a writer.",
    options: ["followed", "pursued", "chased", "sought"],
    answer: "pursued",
  },
  {
    question:
      "The company's __________ growth over the past year has been
impressive.",
    options: ["stable", "consistent", "steady", "constant"],
    answer: "steady",
  },
  {
    question:
      "He made a __________ effort to learn the language and was fluent in
a few months.",
    options: ["committed", "determined", "dedicated", "resolved"],
    answer: "determined",
  },
  {
    question:
      "The new restaurant in town has received __________ reviews from food
critics.",
    options: ["favorable", "positive", "good", "excellent"],
    answer: "excellent",
  },
  {
    question: "The company is known for its __________ customer service.",
    options: ["exceptional", "outstanding", "remarkable", "extraordinary"],
    answer: "exceptional",
  },
  {
```

```
    question:
      "He is __________ about protecting the environment and volunteers for
clean-up projects regularly.",
    options: ["enthusiastic", "passionate", "dedicated", "committed"],
    answer: "passionate",
  },
  {
    question:
      "The team's __________ effort led to a successful product launch.",
    options: ["collaborative", "cooperative", "collective", "united"],
    answer: "collaborative",
  },
  {
    question:
      "The novel is __________ with suspense, keeping readers on the edge
of their seats.",
    options: ["filled", "loaded", "brimming", "packed"],
    answer: "brimming",
  },
  {
    question: "Her __________ personality makes her a great leader.",
    options: ["charming", "magnetic", "captivating", "charismatic"],
    answer: "charismatic",
  },
];
const maxQuestions = 15;
let currentQuestion = 0;
let correctAnswers1 = 0;
let timer;
let timeLeft = 25 * 60;

function displayQuestion() {
  const questionElement = document.getElementById("question");
  const optionsElement = document.getElementById("options");
  const current = questions[currentQuestion];

  questionElement.textContent = current.question;
  optionsElement.innerHTML = "";

  current.options.forEach((option, index) => {
    const label = document.createElement("label");
    const radio = document.createElement("input");
    radio.type = "radio";
    radio.name = "answer";
    radio.value = option;
    radio.onclick = () => selectAnswer(option);
    label.classList.add("option");
    label.appendChild(radio);
    const labelText = document.createTextNode(option);
    label.appendChild(labelText);
    optionsElement.appendChild(label);
  });

  document.getElementById("prevButton").disabled = currentQuestion === 0;
  document.getElementById("nextButton").disabled = true;

  document.getElementById("timer").textContent = formatTime(timeLeft);
  document.getElementById("questionNumber").textContent = `Question ${
```

```
    currentQuestion + 1
  } of ${maxQuestions}`;
}

function selectAnswer(selected) {
  const current = questions[currentQuestion];

  // Check if the selected answer is correct
  if (selected === current.answer) {
    correctAnswers1++;
  }

  // Enable the Next button
  document.getElementById("nextButton").disabled = false;

  localStorage.setItem("correctAnswers1", correctAnswers1);
}

function nextQuestion() {
  currentQuestion++;
  if (currentQuestion === maxQuestions) {
    clearInterval(timer);
    showResult();
  } else {
    displayQuestion();
  }
}

function prevQuestion() {
  currentQuestion--;
  displayQuestion();
}

function showResult() {
  const resultElement = document.getElementById("result1");
  resultElement.textContent = `You answered ${correctAnswers1} out of
${maxQuestions} questions correctly.`;
  resultElement.style.display = "block";
  document.getElementById("nextButton").style.display = "none";
  document.getElementById("homeButton").style.display = "block";
  document.getElementById("question").style.display = "none";
  document.getElementById("options").style.display = "none";
  document.getElementById("questionNumber").style.display = "none";
  document.getElementById("prevButton").style.display = "none";
  document.getElementById("timer").style.display = "none";
  document.getElementById("timeSpent").textContent = `Time spent:
${formatTime(
    25 * 60 - timeLeft
  )}`;
}

function formatTime(seconds) {
  const minutes = Math.floor(seconds / 60);
  const remainingSeconds = seconds % 60;
  return `${minutes}:${remainingSeconds < 10 ? "0" :
""}${remainingSeconds}`;
}
```

```
function returnHome() {
  window.location.href = "./test_begin.html";
}

timer = setInterval(() => {
  timeLeft--;
  document.getElementById("timer").textContent = formatTime(timeLeft);
  if (timeLeft === 0) {
    showResult();
  }
}, 1000);

displayQuestion();
startTime = new Date();
```

Файл learn_page.js

```
let currentText = 1;

function toggleMenu() {
  const sidebar = document.querySelector(".sidebar");
  sidebar.classList.toggle("hidden");
  const arrow = document.getElementById("arrow");
  const menuText = document.getElementById("menu-text");
  arrow.style.transform = sidebar.classList.contains("hidden")
    ? "rotate(0deg) translateX(-50%)" // Зсув вліво при згортанні
    : "rotate(90deg)";
  menuText.style.display = sidebar.classList.contains("hidden")
    ? "none"
    : "block";

  // Зміщення тексту з правого боку при згортанні/розгортанні меню
  const content = document.querySelector(".content-wrap");
  const hiddenText = document.querySelectorAll(".hidden-text");
  content.style.marginLeft = sidebar.classList.contains("hidden")
    ? "20px"
    : "20px";
  content.style.maxWidth = sidebar.classList.contains("hidden")
    ? "100%"
    : "100%";
  if (sidebar.classList.contains("hidden")) {
    hiddenText.forEach((element) => {
      element.style.display = "none"; // Приховування тексту
    });
  } else {
    hiddenText.forEach((element) => {
      element.style.display = "flex"; // Показ тексту
    });
  }
}

function showText(num) {
  const xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function () {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("content-wrap").innerHTML =
this.responseText;
      currentText = num;
    }
```

```
  };
  xhttp.open("GET", "./php/get_topic.php?id=" + num, true);
  xhttp.send();
}

window.onload = function () {
  showText(1);
  document.getElementById("menu-toggle").addEventListener("click",
toggleMenu);
};
```

Файл fetch_exercises.php

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "english";

// Створення з'єднання
$conn = new mysqli($servername, $username, $password, $dbname);

// Перевірка з'єднання
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Запит для отримання вправ
$sql = "SELECT * FROM exercises";
$result = $conn->query($sql);

$exercises = array();
if ($result->num_rows > 0) {
    // Отримання рядків з набору даних
    while ($row = $result->fetch_assoc()) {
        // Розбір масиву JSON варіантів відповідей
        $answers = json_decode($row['answers']);
        $row['answers'] = $answers;
        $exercises[$row['topic']][] = $row;
    }
}

// Закриття з'єднання
$conn->close();

// Повернення вправ у вигляді JSON
header('Content-Type: application/json');
echo json_encode($exercises);
?>
```

Файл get_topic.php

```
<?php
$servername = "sql313.infinityfree.com";
$username = "if0_36347033";
$password = "NEOyK6mZLXll";
$dbname = "if0_36347033_english";

// Підключення до бази даних
$conn = new mysqli($servername, $username, $password, $dbname);
```

```
$conn->set_charset("utf8mb4");
if ($conn->connect_error) {
    die("Помилка підключення: " . $conn->connect_error);
}

// Отримання ID теми з GET-параметра
if (isset($_GET['id'])) {
    $topicId = $_GET['id'];

    // Вибірка тексту теми з бази даних
    $sql = "SELECT content FROM topics WHERE id=?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("i", $topicId);
    $stmt->execute();
    $result = $stmt->get_result();

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        $content = $row["content"];

        // Декодування HTML-сутностей
        $decoded_content = htmlspecialchars_decode($content);

        echo "<div class='content'>$decoded_content</div>";
    } else {
        echo "Тема не знайдена";
    }
} else {
    echo "Не вказано ID теми";
}

$conn->close();
?>
```

ДОДАТОК В

Диск із кваліфікаційною роботою бакалавра