НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету	**Завідувач кафедри комп'ютерних наук**
_____ Ігор БОЛБОТ	**_____ Белла ГОЛУБ**
(підпис)	(підпис)
«___» _____ 2026 р.	«___» _____ 2026 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Мобільна інтелектуальна система оцінки рівня знань з іноземної мови»

Спеціальність «121 Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення»

Гарант освітньої програми
к.т.н., доцент _____ **Ганна ВАЙГАНГ**
(підпис)

Керівник бакалаврської кваліфікаційної роботи _____ **Ганна ВАЙГАНГ**
к.т.н., доцент (підпис)

Виконав _____ **Артем МЕЛЬНИКОВ**
(підпис)

КИЇВ – 2026

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук
к.т.н., доцент _____ Белла ГОЛУБ
(підпис)
«___» _____ 2025 р.

ЗАВДАННЯ ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

Мельникову Артему Валерійовичу

Спеціальність «121 Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення»

Тема бакалаврської кваліфікаційної роботи
«Мобільна інтелектуальна система оцінки рівня знань з іноземної мови»
затверджена наказом від «19» грудня 2025 р. № 3204 «С»

Термін подання завершеної роботи на кафедру «_22__» травня 2026 р.

Вихідні дані до роботи: опис програмного забезпечення та процесу оцінювання рівня знань з іноземної мови; вимоги до мобільної інтелектуальної системи тестування; структура тестових завдань, дані користувачів і результати тестування; технології HTML, CSS, JavaScript, Node.js, REST API та SQLite; наукові й інформаційні джерела з цифрової освіти та розробки програмного забезпечення.

Перелік питань що розглядаються: системний аналіз предметної області інформаційної системи; проєктування інформаційного та програмного забезпечення; розробка інформаційного та програмного забезпечення; рекомендації щодо впровадження та експлуатації системи

Перелік графічних документів (за необхідності).

Дата видачі завдання «_22_» ___грудня_____ 2025 р.

Керівник бакалаврської кваліфікаційної роботи _____ **Ганна ВАЙГАНГ**
(підпис)

Завдання взяв до виконання _____ **Артем МЕЛЬНИКОВ**
(підпис)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – Application Programming Interface, інтерфейс прикладного програмування

CEFR – Common European Framework of Reference for Languages, система рівнів володіння мовою

CSS – Cascading Style Sheets, каскадні таблиці стилів

DB – Database, база даних

HTML – HyperText Markup Language, мова розмітки гіпертексту

HTTP – HyperText Transfer Protocol, протокол передачі гіпертексту

JSON – JavaScript Object Notation, формат обміну даними

JS – JavaScript, мова програмування

ML – Machine Learning, машинне навчання

MVC – Model-View-Controller, архітектурний шаблон

Node.js – середовище виконання JavaScript на сервері

REST – Representational State Transfer, архітектурний стиль взаємодії

SPA – Single Page Application, односторінковий застосунок

SQL – Structured Query Language, мова запитів до баз даних

SQLite – вбудована система керування базами даних

UML – Unified Modeling Language, уніфікована мова моделювання

UI – User Interface, інтерфейс користувача

UX – User Experience, користувацький досвід

БД – база даних

ІКТ – інформаційно-комунікаційні технології

ІС – інформаційна система

КС – клієнт-сервер

ПЗ – програмне забезпечення

РБД – реляційна база даних

ВСТУП

Сучасний етап розвитку інформаційних технологій характеризується стрімким зростанням ролі мобільних застосунків в різних сферах людської діяльності, серед яких особливе місце займає освітній напрям, орієнтований на забезпечення доступу до знань в будь-який час і з будь-якого місця. Поширення смартфонів, високошвидкісного інтернету та цифрових освітніх ресурсів сприяло формуванню нових підходів до організації навчального процесу, що передбачають інтерактивність, персоналізацію та адаптацію навчального контенту відповідно до індивідуальних особливостей користувача.

Значна увага в освітньому просторі приділяється вивченню іноземних мов, що обумовлено глобалізаційними процесами, розвитком міжнародної співпраці та необхідністю професійної мобільності фахівців в різних галузях діяльності. Традиційні форми контролю знань, які базуються на тестуванні в рамках аудиторних занять, поступово доповнюються цифровими інструментами, здатними забезпечити оперативний зворотний зв’язок, автоматизований аналіз результатів та індивідуалізацію навчального процесу.

Поява мобільних освітніх платформ, орієнтованих на вивчення мов, відкрила нові можливості для поєднання навчання та оцінювання знань в єдиному цифровому середовищі, що дозволяє не лише контролювати рівень підготовки користувача, але й формувати рекомендації щодо подальшого вдосконалення мовних навичок. Важливою особливістю сучасних систем є інтеграція елементів інтелектуального аналізу, які забезпечують обробку результатів тестування, виявлення прогалин в знаннях та адаптацію навчального контенту відповідно до отриманих показників.

Актуальність розробки мобільної інтелектуальної системи оцінки рівня знань з іноземної мови обумовлена потребою в створенні універсального програмного продукту, який поєднує функції тестування, навчання та аналітики результатів у межах єдиного застосунку. Розробка подібної системи дозволяє підвищити ефективність засвоєння матеріалу, забезпечити об’єктивність

оцінювання та створити комфортні умови для самостійного навчання користувачів різного рівня підготовки.

Метою роботи є розробка мобільного застосунку інтелектуального типу, який забезпечує визначення рівня знань з іноземної мови, організацію навчального процесу у вигляді інтерактивних уроків та аналіз результатів тестування з урахуванням індивідуальних особливостей користувача.

Для досягнення поставленої мети необхідно виконати наступні завдання:

1) здійснити аналіз предметної області та визначити вимоги до мобільної системи оцінювання знань;
2) розробити модель функціонування системи з застосуванням засобів UML;
3) спроєктувати архітектуру мобільного застосунку та структуру інформаційної бази;
4) реалізувати основні функціональні модулі, що включають авторизацію, тестування та навчальні компоненти;
5) впровадити алгоритми аналізу результатів тестування;
6) провести тестування програмного продукту та оцінити ефективність його функціонування.

Об’єктом дослідження виступає процес оцінювання рівня знань користувачів з іноземної мови із застосуванням мобільних інформаційних технологій.

Предметом дослідження є методи та засоби розробки мобільних програмних систем, що забезпечують автоматизоване тестування, аналіз результатів та формування рекомендацій для подальшого навчання.

Практична значущість отриманих результатів полягає в можливості використання розробленого мобільного застосунку як інструменту для самостійного вивчення іноземної мови та оцінювання рівня знань в навчальних закладах або в межах неформальної освіти. Реалізація програмного продукту передбачає використання сучасних технологій розробки мобільних застосунків, що забезпечують високу продуктивність, зручність використання та можливість подальшого розширення функціональності.

В процесі виконання роботи застосовуються методи системного аналізу, моделювання інформаційних процесів, проєктування програмних систем та розробки мобільних застосунків з використанням сучасних програмних засобів. Реалізація системи орієнтована на створення зручного інтерфейсу користувача, ефективної логіки обробки даних та надійного механізму збереження інформації.

Структура роботи передбачає послідовне розкриття основних етапів створення програмного продукту, починаючи від аналізу предметної області та завершуючи впровадженням і тестуванням розробленої системи, що забезпечує цілісне уявлення про процес розробки мобільного інтелектуального застосунку для оцінювання знань з іноземної мови.

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області та особливості оцінювання рівня володіння іноземною мовою

Процес оцінювання рівня володіння іноземною мовою в сучасному освітньому середовищі, зазнає суттєвих трансформацій під впливом цифровізації та інтеграції інформаційних технологій у навчальну діяльність. Використання мобільних застосунків та онлайн-платформ забезпечує нові можливості для організації контролю знань, що характеризується автоматизацією перевірки відповідей, адаптацією складності завдань та формуванням аналітичних висновків щодо рівня підготовки користувача. Поступове зміщення акценту з традиційних форм оцінювання на цифрові інструменти обумовлює необхідність детального дослідження особливостей функціонування відповідних систем.

Суть оцінювання мовних знань полягає у визначенні рівня сформованості ключових компетенцій, серед яких виділяються лексичні, граматичні, аудитивні та комунікативні навички. В цифровому середовищі, зазначені компоненти можуть оцінюватися за допомогою інтерактивних тестів, вправ на відповідність, завдань із вибором правильної відповіді, а також шляхом аналізу поведінки користувача під час виконання завдань. Інтеграція мультимедійних елементів, включаючи аудіо та відео, дозволяє значно розширити можливості оцінювання, забезпечуючи перевірку не лише теоретичних знань, але й практичних навичок сприйняття та використання мови [1].

Важливим аспектом цифрового оцінювання є можливість збору та обробки великого обсягу даних про результати виконання завдань, що створює передумови для застосування алгоритмів інтелектуального аналізу. Завдяки цьому, система здатна не лише фіксувати правильність відповідей, але й визначати типові помилки, оцінювати швидкість виконання завдань та формувати індивідуальні рекомендації щодо подальшого навчання. Компоненти оцінювання рівня знань з іноземної мови наведено в таблиці 1.1.

Таблиця 1.1 – Компоненти оцінювання рівня знань з іноземної мови

№	Компонент оцінювання	Характеристика
1	Лексична компетенція	Перевірка словникового запасу та розуміння значень слів
2	Граматична компетенція	Оцінювання правильності побудови речень
3	Аудіювання	Визначення здатності сприймати мову на слух
4	Читання	Аналіз розуміння текстової інформації
5	Письмо	Перевірка навичок формування письмових висловлювань
6	Комунікативні навички	Оцінювання здатності використовувати мову у практиці

Забезпечення повноцінного охоплення всіх зазначених аспектів є важливою умовою отримання об’єктивної картини рівня знань користувача, що особливо актуально для мобільних застосунків, орієнтованих на самостійне навчання [2].

Цифрове середовище характеризується високим рівнем інтерактивності, що позитивно впливає на мотивацію користувачів до проходження тестування та виконання навчальних завдань. Використання гейміфікації, системи досягнень та рейтингових показників дозволяє залучити користувача до регулярної взаємодії з застосунком, що, в свою чергу, сприяє більш ефективному засвоєнню матеріалу. Оцінювання в даних умовах, перестає виконувати лише контрольну функцію та стає складовою навчального процесу.

Значну роль відіграє адаптивність системи оцінювання, яка полягає в зміні рівня складності завдань залежно від результатів користувача. Реалізація адаптивного механізму дозволяє уникнути ситуацій перевантаження або недостатньої складності, забезпечуючи оптимальний рівень навчального навантаження [3]. Впровадження подібних рішень вимагає використання алгоритмів, здатних аналізувати результати в режимі реального часу та формувати відповідну логіку переходу між рівнями складності. Критерії оцінювання результатів тестування наведено в таблиці 1.2.

Таблиця 1.2 – Критерії оцінювання результатів тестування

№	Критерій	Опис застосування
1	Точність відповідей	Частка правильних відповідей від загальної кількості
2	Час виконання	Тривалість проходження тесту
3	Складність завдань	Рівень складності виконаних вправ
4	Частота помилок	Аналіз типових помилок користувача
5	Динаміка результатів	Зміна показників у процесі навчання
6	Рівень засвоєння матеріалу	Інтегральна оцінка знань користувача

Розглянуті критерії дозволяють сформувати комплексну оцінку знань користувача, яка враховує не лише правильність відповідей, але й інші параметри, що характеризують процес навчання. Використання багатофакторного аналізу результатів сприяє підвищенню точності визначення рівня знань та створює передумови для формування персоналізованих рекомендацій [4].

Цифрове середовище оцінювання знань з іноземної мови забезпечує значно ширші можливості порівняно з традиційними підходами, оскільки поєднує автоматизацію процесів, інтерактивність та інтелектуальний аналіз даних.

1.2 Аналіз вимог до мобільного застосунку

Процес формування вимог до мобільної інтелектуальної системи оцінювання рівня знань з іноземної мови передбачає детальне визначення очікуваної поведінки програмного продукту, а також характеристик, які забезпечують його ефективну експлуатацію в умовах реального використання. Коректно сформульовані вимоги виступають основою для подальшого проєктування архітектури системи, визначення її функціональних можливостей та оцінки якості реалізації. В контексті мобільних застосунків освітнього призначення, особливого значення набуває поєднання зручності використання, швидкодії та адаптивності до індивідуальних потреб користувача [5].

Функціональні характеристики системи визначають перелік дій, які має виконувати програмний продукт в процесі взаємодії з користувачем. Ключовими є процеси реєстрації та входу до системи, проходження тестування для визначення рівня підготовки, а також доступ до навчальних матеріалів, що відповідають отриманим результатам. Окрім цього, передбачається реалізація механізмів збереження результатів тестування, відображення статистики успішності та формування рекомендацій щодо подальшого навчання.

Особливістю функціонального наповнення системи виступає інтеграція елементів інтелектуального аналізу, які забезпечують обробку результатів тестування та адаптацію складності навчальних матеріалів. Завдяки використанню відповідних алгоритмів система здатна визначати рівень знань користувача, аналізувати типові помилки та пропонувати завдання, спрямовані на усунення виявлених прогалин. Дана організація функціоналу дозволяє забезпечити персоналізований підхід до навчання та підвищити ефективність засвоєння матеріалу [6]. Функціональні вимоги до мобільної системи оцінювання знань наведено на рисунку 1.3.

Таблиця 1.3 – Функціональні вимоги до мобільної системи оцінювання знань

№	Функціональна можливість	Призначення та опис реалізації
1	Реєстрація та авторизація	Забезпечення доступу користувача до персоналізованого середовища
2	Визначення рівня знань	Проходження тесту для оцінювання початкового рівня
3	Надання навчального контенту	Доступ до уроків відповідно до результатів тестування
4	Збереження результатів	Фіксація відповідей та показників успішності
5	Відображення статистики	Візуалізація результатів та прогресу користувача
6	Формування рекомендацій	Генерація порад щодо подальшого навчання
7	Адаптація складності завдань	Автоматичне коригування рівня складності вправ

Важливим аспектом є узгодженість між окремими модулями системи, що дозволяє досягти цілісності програмного продукту та забезпечити логічну послідовність виконання дій користувача.

Окрім функціонального наповнення, значну роль відіграють нефункціональні характеристики, які визначають якість роботи системи, її надійність та зручність використання. В контексті мобільних застосунків освітнього спрямування, важливо забезпечити швидку обробку запитів, стабільність роботи, захист персональних даних користувачів та можливість масштабування системи у разі збільшення кількості користувачів. Забезпечення відповідних характеристик є необхідною умовою для успішного впровадження програмного продукту [7].

Високий рівень продуктивності системи дозволяє забезпечити миттєву реакцію інтерфейсу на дії користувача, що позитивно впливає на загальне сприйняття застосунку. Надійність функціонування передбачає відсутність збоїв та втрати даних, що особливо важливо для систем, які зберігають результати навчальної діяльності. Захист інформації користувачів реалізується за рахунок використання сучасних протоколів шифрування та механізмів автентифікації, що запобігають несанкціонованому доступу. Нефункціональні вимоги до мобільного застосунку наведено в таблиці 1.4.

Таблиця 1.4 – Нефункціональні вимоги до мобільного застосунку

№	Характеристика	Опис та значення для системи
1	Продуктивність	Забезпечення швидкої обробки запитів та відображення даних
2	Надійність	Стабільна робота без втрати інформації
3	Безпека	Захист персональних даних користувачів
4	Масштабованість	Можливість обслуговування великої кількості користувачів
5	Зручність використання	Інтуїтивно зрозумілий інтерфейс користувача
6	Сумісність	Підтримка різних мобільних платформ

Представлені нефункціональні характеристики визначають якісні параметри системи, що впливають на ефективність її використання у реальних умовах. Забезпечення відповідного рівня якості дозволяє створити конкурентоспроможний програмний продукт, здатний задовольнити потреби користувачів та відповідати сучасним вимогам до мобільних освітніх застосунків.

Формування функціональних і нефункціональних вимог є ключовим етапом розробки мобільної інтелектуальної системи оцінювання знань з іноземної мови, оскільки визначає напрям подальшого проєктування та реалізації програмного забезпечення. Врахування як функціональних можливостей, так і якісних характеристик дозволяє створити ефективний інструмент для навчання та контролю знань, орієнтований на потреби сучасного користувача.

1.3 Формалізація структури та моделювання предметної області

Дослідження предметної області мобільної інтелектуальної системи оцінювання знань з іноземної мови передбачає побудову формалізованого опису її основних сутностей, взаємозв’язків та процесів функціонування, що дозволяє забезпечити цілісне уявлення про логіку роботи програмного продукту. Використання методів моделювання сприяє визначенню ключових компонентів системи, встановленню їх ролей та опису сценаріїв взаємодії користувача з функціональними модулями застосунку. Формалізація структури предметної області створює основу для подальшого проєктування архітектури та реалізації програмного забезпечення.

В рамках розглядуваної системи, основними сутностями виступають користувач, тестове завдання, результат тестування, навчальний модуль та аналітичний компонент, який здійснює обробку отриманих даних. Кожна з зазначених сутностей виконує окрему функцію, що визначає її місце в загальній структурі системи та впливає на формування логіки взаємодії. Взаємозв’язки між

сутностями відображають послідовність виконання дій, починаючи від авторизації користувача і завершуючи отриманням рекомендацій щодо подальшого навчання [8].

Формалізація предметної області передбачає опис потоків даних, що циркулюють в системі під час виконання основних функціональних сценаріїв. Зокрема, дані про відповіді користувача передаються до модуля обробки, де відбувається їх аналіз, після чого результати зберігаються у базі даних та використовуються для формування статистики і рекомендацій. Подібна організація інформаційних потоків забезпечує узгодженість роботи компонентів системи та дозволяє реалізувати механізми адаптації навчального контенту. Основні сутності предметної області системи оцінювання знань наведено в таблиці 1.5.

Таблиця 1.5 – Основні сутності предметної області системи оцінювання знань

№	Сутність	Опис функціонального призначення
1	Користувач	Суб’єкт, що проходить тестування та використовує систему
2	Тестове завдання	Окремий елемент перевірки знань
3	Результат тестування	Дані про правильність відповідей та рівень знань
4	Навчальний модуль	Набір матеріалів для підвищення рівня знань
5	Аналітичний компонент	Модуль обробки результатів та формування рекомендацій
6	База даних	Сховище інформації про користувачів та результати

Кожна сутність виконує визначену роль, а їх сукупність забезпечує реалізацію процесів оцінювання та навчання в одному інформаційному середовищі.

Подальший аналіз предметної області передбачає дослідження сценаріїв взаємодії користувача з системою, які визначають послідовність виконання дій та обмін інформацією між компонентами. Основним сценарієм виступає проходження тестування, що включає авторизацію, вибір рівня складності, виконання завдань та отримання результатів. Іншим важливим сценарієм є

взаємодія з навчальними матеріалами, яка реалізується на основі результатів попереднього оцінювання та передбачає доступ до відповідних уроків.

Значну роль у формалізації процесів відіграє визначення логіки обробки даних, що забезпечує коректну інтерпретацію результатів тестування та формування аналітичних висновків. Використання алгоритмів аналізу дозволяє визначити рівень знань користувача, враховуючи кількість правильних відповідей, складність завдань та динаміку змін показників у процесі навчання. Результати обробки використовуються для побудови рекомендацій, спрямованих на підвищення ефективності засвоєння матеріалу. Основні процеси функціонування системи наведено в таблиці 1.6.

Таблиця 1.6 – Основні процеси функціонування системи

№	Процес	Опис виконання
1	Авторизація користувача	Вхід у систему з використанням облікових даних
2	Проходження тестування	Виконання завдань для визначення рівня знань
3	Обробка результатів	Аналіз відповідей та визначення рівня підготовки
4	Збереження інформації	Запис результатів у базу даних
5	Формування рекомендацій	Генерація порад щодо подальшого навчання
6	Доступ до навчальних матеріалів	Надання уроків відповідно до результатів тестування

Представлені процеси відображають логіку функціонування системи та визначають послідовність взаємодії між її компонентами. Узгодженість виконання зазначених процесів забезпечує коректну роботу застосунку та сприяє досягненню поставленої мети, що полягає у визначенні рівня знань користувача та організації ефективного навчального процесу [9].

Формалізація структури та процесів предметної області дозволяє створити чітке уявлення про функціонування мобільної інтелектуальної системи оцінювання знань з іноземної мови, що є необхідною умовою для подальшого проєктування та реалізації програмного забезпечення. Визначення сутностей, взаємозв’язків та основних процесів забезпечує основу для побудови ефективної

архітектури системи та впровадження механізмів інтелектуального аналізу результатів навчальної діяльності.

1.4 Дослідження існуючих мобільних застосунків для вивчення мов

Аналіз сучасних мобільних застосунків для вивчення іноземних мов дозволяє визначити основні тенденції розвитку цифрових освітніх платформ, а також, виявити функціональні рішення, які можуть бути використані під час розробки власної системи оцінювання знань. Сучасні застосунки орієнтовані на інтерактивність, персоналізацію навчання та використання елементів гейміфікації, що значно підвищує залученість користувачів та ефективність засвоєння матеріалу. Дослідження найбільш популярних платформ дає можливість оцінити їх сильні сторони, визначити недоліки та сформувати вимоги до майбутнього програмного продукту.

Одним з найбільш відомих мобільних застосунків для вивчення іноземних мов є Duolingo, який поєднує навчальні вправи з елементами гри та системою винагород. Головна сторінка мобільного застосунку Duolingo наведена на рисунку 1.1.

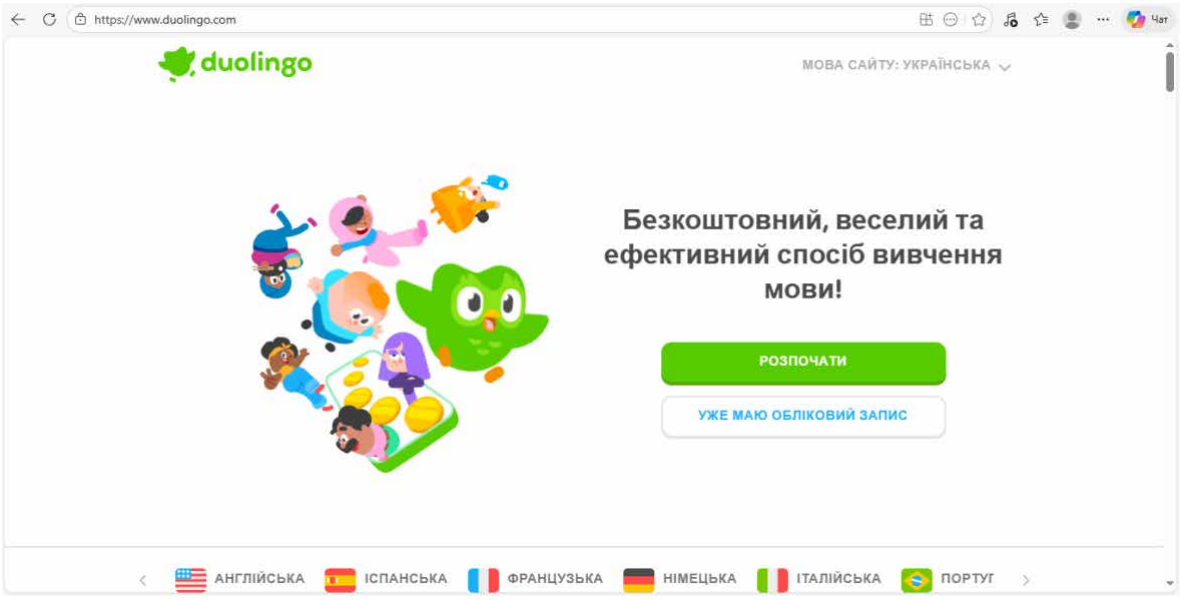

Рис. 1.1 Головна сторінка мобільного застосунку Duolingo

Інтерфейс застосунку характеризується простотою та зрозумілою структурою, що дозволяє користувачам швидко орієнтуватися в доступних функціях. Навчальний процес реалізовано у вигляді коротких уроків, які включають завдання на переклад, аудіювання та вибір правильної відповіді. Особливу увагу приділено адаптації складності завдань залежно від результатів користувача, що забезпечує індивідуалізацію навчального процесу [10].

Незважаючи на високу популярність, функціонал оцінювання рівня знань в даному застосунку має обмежений характер, оскільки основний акцент зроблено на навчальному процесі, а не на глибокому аналізі результатів. Визначення рівня знань здійснюється в спрощеній формі, що обмежує можливості використання застосунку як повноцінного інструменту для об’єктивного тестування.

Іншим прикладом сучасного мобільного рішення є Babbel, який орієнтований на більш структуроване навчання з акцентом на практичне використання мови. Інтерфейс застосунку має більш формалізований вигляд, а навчальні матеріали подаються у вигляді послідовних курсів, що відповідають певним рівням складності. Важливою особливістю є використання реалістичних діалогів та ситуацій, що сприяє розвитку комунікативних навичок користувача [11]. Головна сторінка мобільного застосунку Babbel наведена на рисунку 1.2.

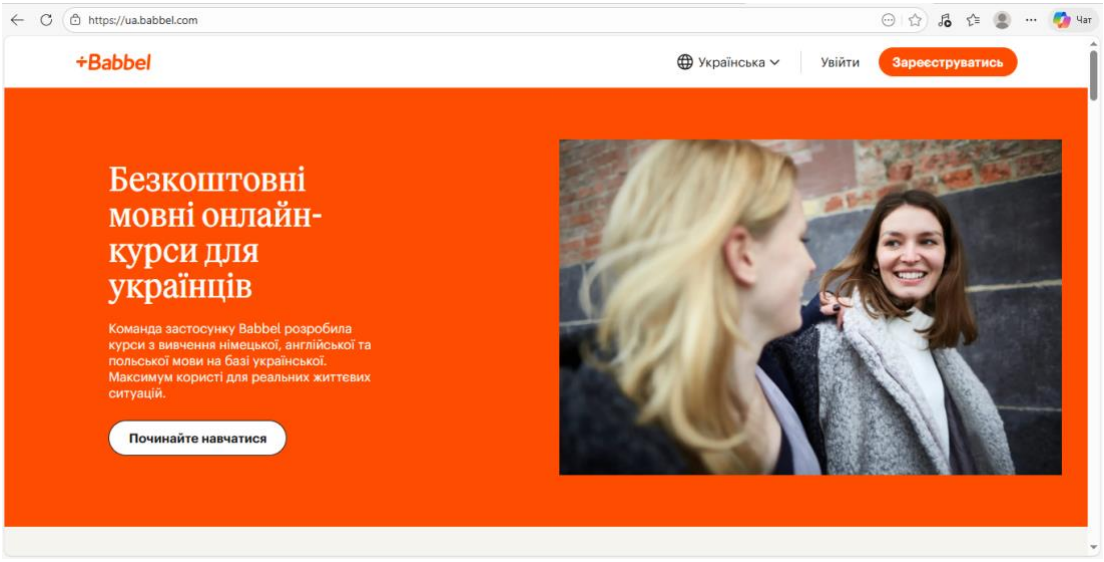

Рис. 1.2 Головна сторінка мобільного застосунку Babbel

Разом з тим, можливості аналізу результатів навчання в даному застосунку, також мають обмеження, оскільки система не передбачає глибокого інтелектуального оцінювання рівня знань, а переважно орієнтована на проходження навчальних курсів за визначеною програмою. Відсутність розвиненої аналітичної підсистеми ускладнює формування персоналізованих рекомендацій.

Застосунок Busuu пропонує інший підхід до організації навчального процесу, поєднуючи автоматизоване навчання з елементами соціальної взаємодії між користувачами. В межах платформи, користувачі мають можливість отримувати зворотний зв’язок від інших учасників, що сприяє покращенню якості навчання та розвитку практичних навичок спілкування [12]. Головна сторінка мобільного застосунку Busuu наведена на рисунку 1.3.

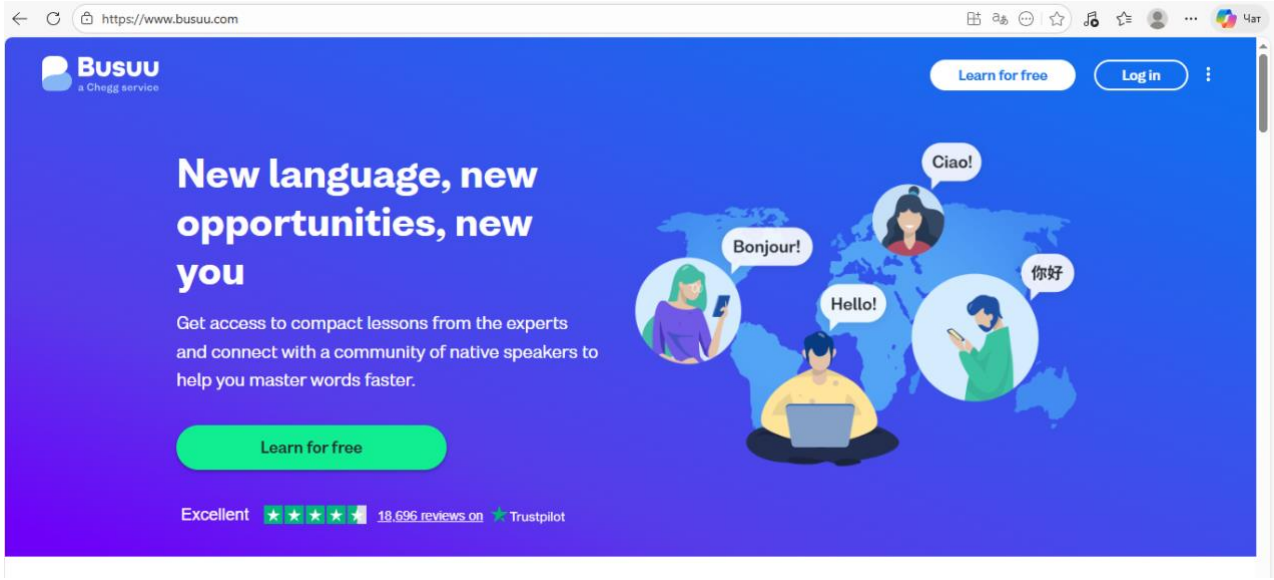

Рис. 1.3 Головна сторінка мобільного застосунку Busuu

Попри наявність елементів взаємодії між користувачами, система оцінювання знань залишається недостатньо формалізованою, що обмежує можливості використання платформи для точного визначення рівня підготовки. Оцінювання результатів базується переважно на виконанні вправ та відгуках інших користувачів, що може впливати на об’єктивність отриманих результатів.

Ще одним прикладом є Memrise, який орієнтований на запам’ятовування лексики за допомогою повторення та асоціативних методів. Інтерфейс застосунку містить значну кількість мультимедійного контенту, що сприяє покращенню сприйняття інформації та підвищенню ефективності навчання [13]. Головна сторінка мобільного застосунку Memrise наведена на рисунку 1.4.

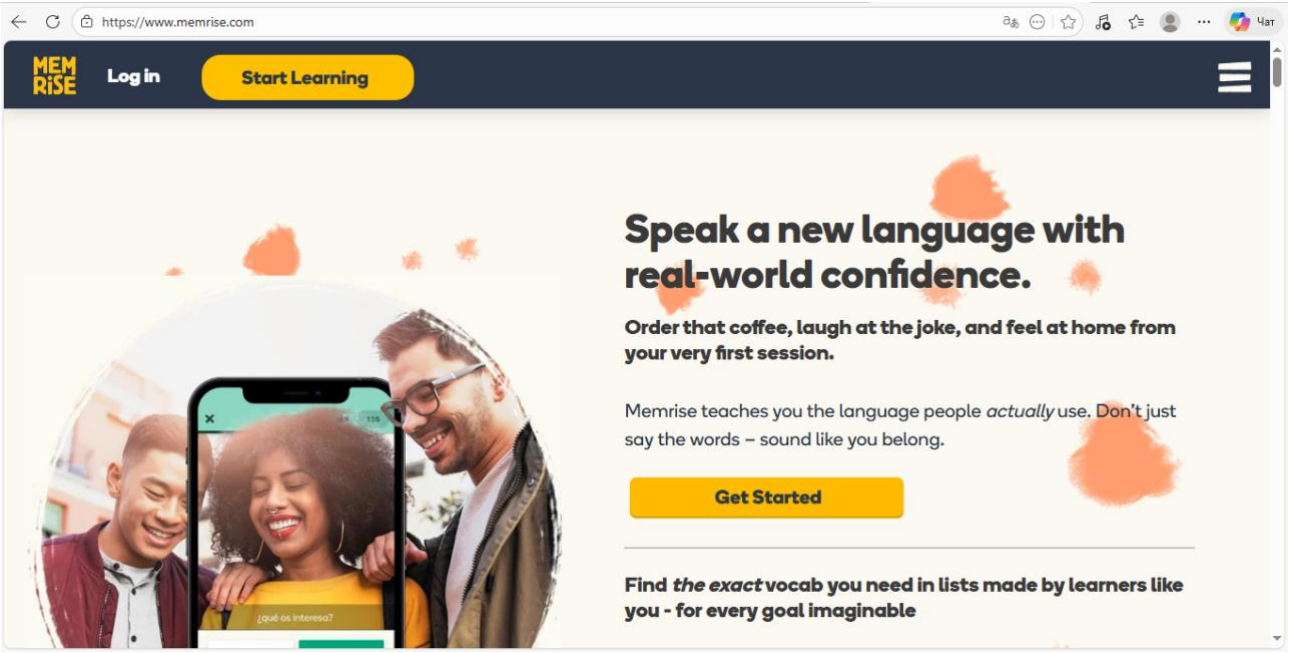

Рис. 1.4 Головна сторінка мобільного застосунку Memrise

Функціонал оцінювання знань в зазначеному застосунку обмежується перевіркою засвоєння окремих слів та фраз, що не дозволяє сформувати повну картину рівня володіння мовою. Відсутність комплексного підходу до оцінювання знижує ефективність використання платформи для цілей контролю знань.

Застосунок Mondly, використовує сучасні технології, включаючи голосове розпізнавання та інтерактивні вправи, що забезпечують більш глибоке залучення користувача до навчального процесу. Інтерфейс застосунку має привабливий дизайн та інтуїтивно зрозумілу структуру, що сприяє комфортному використанню [14]. Головна сторінка мобільного застосунку Mondly наведена на рисунку 1.5.

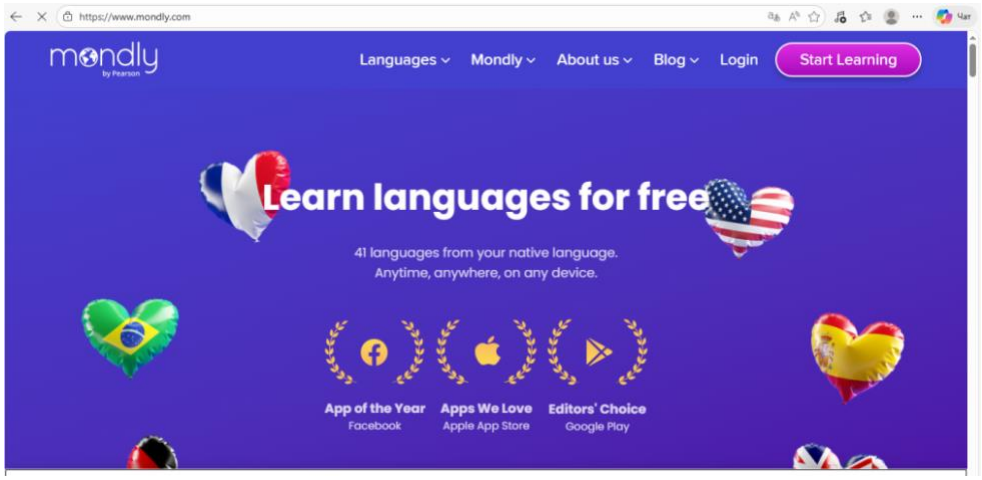

Рис. 1.5 Головна сторінка мобільного застосунку Mondly

Незважаючи на використання сучасних технологічних рішень, система оцінювання знань в даному застосунку також не забезпечує достатньої глибини аналізу результатів, оскільки основний акцент зроблено на інтерактивному навчанні. Відсутність розвинених алгоритмів аналізу обмежує можливості адаптації навчального процесу.

Більшість сучасних мобільних застосунків для вивчення іноземних мов орієнтовані переважно на навчальний процес та підвищення мотивації користувачів, тоді як функціонал об’єктивного оцінювання рівня знань залишається недостатньо розвиненим. Виявлені особливості підтверджують доцільність розробки мобільної інтелектуальної системи, яка поєднує навчання та глибокий аналіз результатів тестування, що дозволить підвищити ефективність контролю знань та забезпечити персоналізований підхід до навчання.

1.5 Постановка завдання

Формування постановки задачі розробки мобільної інтелектуальної системи оцінювання рівня знань з іноземної мови ґрунтується на результатах проведеного аналізу предметної області, дослідження існуючих програмних рішень та визначення функціональних і якісних характеристик майбутнього застосунку. Узагальнення отриманих результатів дозволяє сформулювати

ключові вимоги до системи, яка повинна поєднувати можливості тестування, навчання та аналітичної обробки даних.

Аналіз сучасних мобільних платформ показав, що більшість застосунків орієнтована на організацію навчального процесу з використанням інтерактивних вправ, тоді як питання об’єктивного оцінювання рівня знань користувача реалізовано на обмеженому рівні. Відсутність глибокого аналізу результатів тестування, а також недостатня адаптація навчального контенту відповідно до індивідуальних показників користувача створюють передумови для розробки програмного продукту, який здатний усунути зазначені недоліки та забезпечити більш ефективний підхід до оцінювання знань.

В рамках розроблюваної системи, передбачається реалізація інтегрованого середовища, що забезпечує взаємодію користувача з функціональними модулями застосунку, включаючи механізми реєстрації, проходження тестування, отримання результатів та доступу до навчальних матеріалів. Важливою складовою виступає впровадження алгоритмів інтелектуального аналізу, які дозволяють здійснювати обробку результатів тестування з урахуванням різних параметрів, включаючи точність відповідей, складність завдань та динаміку змін показників в процесі навчання.

Суть задачі полягає в створенні програмної системи, яка забезпечує автоматизоване визначення рівня знань користувача з іноземної мови, а також формує рекомендації щодо подальшого вдосконалення мовних навичок. Реалізація відповідної функціональності, передбачає розробку алгоритмів, здатних аналізувати результати тестування та адаптувати навчальний контент відповідно до індивідуальних потреб користувача. Використання подібних рішень дозволяє підвищити ефективність навчального процесу та забезпечити більш точне оцінювання рівня підготовки.

Значну роль в постановці задачі відіграє визначення структури програмного продукту, яка повинна забезпечувати модульність, масштабованість та зручність подальшого розширення функціональних можливостей. Архітектура системи має передбачати чітке розмежування між

інтерфейсом користувача, логікою обробки даних та механізмами збереження інформації, що сприяє підвищенню надійності та підтримуваності програмного забезпечення.

Практична спрямованість задачі полягає в створенні інструменту, який може використовуватися як в межах формальної освіти, так і для самостійного навчання, забезпечуючи користувачам можливість оцінювати рівень своїх знань та отримувати рекомендації щодо їх покращення. Застосування мобільного формату дозволяє забезпечити доступність системи для широкого кола користувачів, незалежно від їх місцезнаходження та умов навчання.

2 ПРОЕКТУВАННЯ ІНТЕЛЕКТУАЛЬНОЇ МОБІЛЬНОЇ СИСТЕМИ

2.1 Логічна модель даних системи навчання та тестування

Проєктування логічної моделі даних мобільної інтелектуальної системи оцінювання знань з іноземної мови, є ключовим етапом розробки програмного забезпечення, оскільки саме на цьому рівні визначається структура інформаційного середовища, взаємозв'язки між елементами системи та принципи організації збереження даних. Формування коректної моделі дозволяє забезпечити узгодженість обробки інформації, уникнути надмірного дублювання та створити основу для подальшої реалізації програмних компонентів. В контексті мобільного застосунку освітнього призначення, важливим аспектом є підтримка гнучкої структури даних, здатної адаптуватися до змін функціоналу та масштабування системи.

Логічна модель даних відображає сукупність сутностей предметної області, їх атрибутів та зв'язків між ними, що формують цілісну структуру інформаційної системи. Основними елементами виступають користувачі, тестові завдання, результати виконання, навчальні модулі та аналітичні показники, що використовуються для формування рекомендацій. Кожна сутність має набір характеристик, які визначають її властивості та роль в загальній структурі системи [15].

Забезпечення нормалізації даних передбачає поділ інформації на логічно завершені сутності та встановлення між ними зв'язків відповідно до правил реляційної теорії. Подібна організація структури дозволяє мінімізувати надлишковість, підвищити цілісність даних та забезпечити ефективність виконання запитів. Реалізація зазначених принципів створює передумови для стабільної роботи системи навіть за умов значного обсягу інформації. Основні сутності логічної моделі даних наведено на рисунку 2.1.

Таблиця 2.1 – Основні сутності логічної моделі даних

№	Сутність	Опис функціонального призначення
1	Користувач	Зберігання інформації про зареєстрованих користувачів
2	Тест	Набір завдань для визначення рівня знань
3	Питання	Окремий елемент тесту, що містить умову завдання
4	Відповідь	Варіанти відповідей або правильна відповідь на питання
5	Результат	Дані про проходження тестування користувачем
6	Навчальний модуль	Матеріали для підвищення рівня знань
7	Рекомендації	Дані, сформовані на основі аналізу результатів

Наведена структура відображає базовий склад елементів системи, що забезпечують реалізацію функціоналу тестування та навчання. Кожна сутність виконує визначену роль в процесі обробки інформації, а їх взаємодія формує логіку роботи мобільного застосунку. Особливе значення має зв’язок між результатами тестування та навчальними модулями, оскільки саме на основі отриманих даних формується подальша траєкторія навчання користувача.

Побудова логічної моделі передбачає визначення атрибутів кожної сутності, які забезпечують деталізацію збереженої інформації. Атрибути визначають структуру записів в базі даних та використовуються для виконання запитів, аналізу результатів та відображення інформації в користувацькому інтерфейсі. Коректний вибір атрибутів дозволяє забезпечити повноту даних та ефективність їх використання в межах системи.

Значну увагу слід приділити організації зв’язків між сутностями, які визначають залежності та порядок взаємодії між елементами моделі. Наприклад, один користувач може проходити декілька тестів, що формує зв’язок типу «один до багатьох», тоді як кожен тест містить множину питань, які, в свою чергу, пов’язані з варіантами відповідей. Подібна структура дозволяє забезпечити гнучкість моделі та підтримувати складні сценарії взаємодії в межах системи [16]. Атрибути сутностей логічної моделі даних наведено на рисунку 2.2.

Таблиця 2.2 – Атрибути сутностей логічної моделі даних

Сутність	Основні атрибути
Користувач	ID, ім’я, електронна пошта, пароль, рівень знань
Тест	ID, назва, рівень складності
Питання	ID, текст питання, тип завдання
Відповідь	ID, текст відповіді, ознака правильності
Результат	ID, ID користувача, ID тесту, оцінка, час проходження
Навчальний модуль	ID, назва, опис, рівень складності
Рекомендації	ID, ID користувача, текст рекомендації

Представлені атрибути забезпечують необхідний рівень деталізації даних, що дозволяє реалізувати функціональні можливості системи, пов’язані зі збереженням результатів тестування, аналізом прогресу користувача та формуванням персоналізованих рекомендацій. Використання ідентифікаторів забезпечує унікальність записів та підтримку зв’язків між сутностями.

Побудова логічної моделі даних мобільної інтелектуальної системи оцінювання знань з іноземної мови забезпечує основу для ефективної організації інформаційного середовища, підтримки цілісності даних та реалізації функціональних можливостей застосунку. Коректне визначення сутностей, атрибутів та зв’язків дозволяє створити масштабовану та гнучку систему, здатну адаптуватися до змін вимог та забезпечувати високий рівень ефективності навчального процесу.

2.2 Розробка UML-діаграм взаємодії користувачів з системою

Проєктування взаємодії користувачів з мобільною інтелектуальною системою оцінювання знань з іноземної мови потребує формалізованого опису сценаріїв використання, що забезпечує чітке розуміння поведінки системи та логіки виконання операцій. Використання UML-діаграм дозволяє відобразити як загальну структуру взаємодії, так і детальну послідовність обробки запитів, що є важливим для подальшої реалізації програмного забезпечення.

Першим етапом моделювання, є побудова діаграми прецедентів, яка відображає взаємодію користувачів з системою на концептуальному рівні. В досліджуваній системі, виділяються дві ролі, серед яких користувач, що проходить тестування та взаємодіє з навчальним контентом, а також адміністратор, який відповідає за керування системою [17]. Діаграма дозволяє відобразити перелік доступних дій та встановити рамки функціональності системи. Діаграма прецедентів системи оцінювання знань наведена на рис. 2.1.

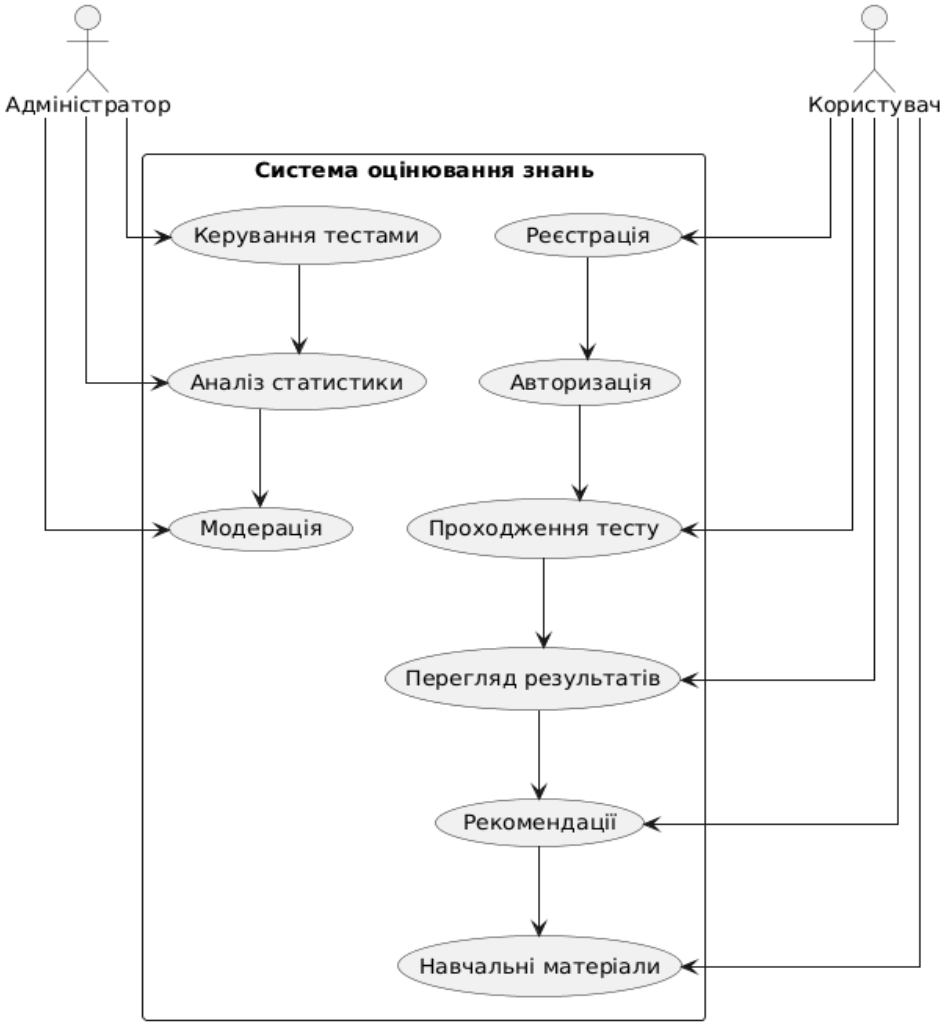

Рис. 2.1 Діаграма прецедентів системи оцінювання знань

Наведена діаграма формує загальне уявлення про функціональні можливості системи та дозволяє визначити основні сценарії взаємодії без деталізації внутрішніх процесів. Вона демонструє розмежування ролей та забезпечує основу для подальшого аналізу логіки функціонування.

Подальший етап передбачає деталізацію процесу взаємодії між компонентами системи, що досягається шляхом побудови діаграми послідовності. Розглядається сценарій проходження тестування, який є центральним елементом функціонування застосунку. Діаграма дозволяє відобразити обмін повідомленнями між користувачем, інтерфейсом, серверною частиною та базою даних [18].

Між попередньою діаграмою та даним етапом необхідно акцентувати увагу на тому, що діаграма прецедентів відображає лише зовнішню взаємодію, тоді як діаграма послідовності дозволяє розкрити внутрішню логіку роботи системи. Саме тому, вона є критично важливою для розуміння того, як відбувається обробка запитів в часовій послідовності та які компоненти беруть участь у виконанні операцій. Діаграма послідовності проходження тестування наведена на рисунку 2.2.

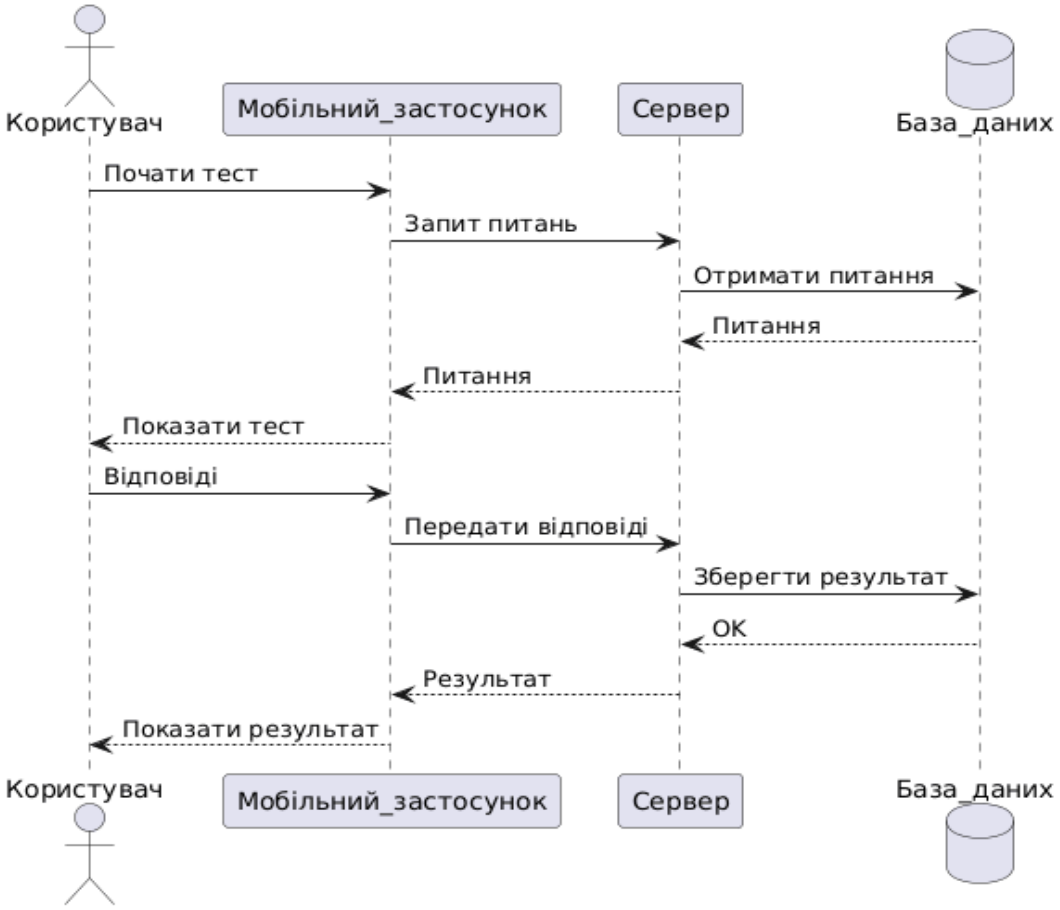

Рис. 2.2 Діаграма послідовності проходження тестування

Розглянута діаграма дозволяє встановити чітку послідовність взаємодії між компонентами системи та визначити точки обробки даних. Вона демонструє, як інформація проходить через різні рівні системи та перетворюється в результат, який отримує користувач. Аналіз даної моделі, дозволяє виявити потенційні вузькі місця в продуктивності та оптимізувати процес обробки запитів [19].

Наступним етапом є побудова діаграми активності, яка дозволяє відобразити алгоритм виконання процесів в системі. В даному випадку розглядається процес оцінювання знань, який включає декілька послідовних етапів, що формують повний цикл взаємодії користувача з системою. Діаграма активності дозволяє не лише описати послідовність дій, але й врахувати альтернативні сценарії розвитку подій. Діаграма активності процесу оцінювання знань наведена на рисунку 2.3.

Рис. 2.3 Діаграма активності процесу оцінювання знань

Перед побудовою діаграми необхідно підкреслити, що алгоритмічне представлення процесів дозволяє значно спростити реалізацію програмної логіки, оскільки кожен етап може бути безпосередньо відображений у вигляді окремого модуля або функції [19].

Побудована діаграма дозволяє відобразити логіку виконання основного процесу системи та визначити послідовність переходів між етапами. Вона також демонструє наявність умовного розгалуження, що дозволяє врахувати альтернативні варіанти розвитку подій під час використання застосунку.

Використання різних типів UML-діаграм забезпечує повне уявлення про функціонування системи на різних рівнях деталізації. Поєднання діаграми прецедентів, послідовності та активності дозволяє охопити як загальну структуру взаємодії, так і детальну логіку виконання процесів, що створює надійну основу для подальшого проєктування архітектури мобільного застосунку.

2.3 Вибір архітектури застосунку

Проєктування архітектури мобільної інтелектуальної системи оцінювання знань з іноземної мови є визначальним етапом розробки програмного забезпечення, оскільки саме на цьому рівні формується структура взаємодії між основними компонентами, визначаються принципи організації обробки даних та забезпечується можливість подальшого розширення функціоналу. Обрана архітектурна модель повинна забезпечувати ефективну роботу застосунку в умовах змінного навантаження, підтримувати адаптивність до різних сценаріїв використання та гарантувати зручність супроводу програмного коду.

В контексті мобільного застосунку освітнього спрямування, важливо забезпечити чітке розмежування між рівнями інтерфейсу, логіки обробки даних та механізмами збереження інформації, що дозволяє підвищити якість розробки та уникнути надмірної складності в структурі програмного продукту. Використання сучасних архітектурних підходів сприяє підвищенню

масштабованості системи, що є особливо важливим у разі збільшення кількості користувачів або розширення функціональних можливостей [20].

В процесі вибору архітектури, було розглянуто декілька підходів до побудови мобільних застосунків, кожен з яких має власні переваги та обмеження. Основна увага приділяється архітектурам, що широко використовуються у розробці мобільних систем, включаючи монолітну модель, клієнт-серверну організацію та багаторівневу архітектуру. Порівняння архітектурних підходів наведено в таблиці 2.3

Таблиця 2.3 – Порівняння архітектурних підходів

Архітектурний підхід	Характеристика	Переваги	Обмеження
Монолітна модель	Уся логіка реалізується в межах одного застосунку	Простота реалізації, швидкий старт розробки	Складність масштабування, важкість підтримки
Клієнт-серверна модель	Розподіл функцій між мобільним клієнтом та сервером	Гнучкість, можливість віддаленої обробки даних	Залежність від мережі, складність синхронізації
Багаторівнева архітектура	Розподіл системи на рівні представлення, логіки та даних	Масштабованість, модульність, зручність підтримки	Вища складність реалізації

Аналіз представлених підходів дозволяє визначити, що для реалізації мобільної системи оцінювання знань найбільш доцільним є використання клієнт-серверної архітектури з елементами багаторівневої організації. Дане поєднання, забезпечує ефективний розподіл навантаження між застосунком та серверною частиною, що дозволяє виконувати обчислювально складні операції на сервері, залишаючи клієнтську частину відповідальною за взаємодію з користувачем [21].

Застосунок виконує роль клієнта, який забезпечує відображення інтерфейсу, обробку дій користувача та взаємодію з серверною частиною через

мережеві запити. Сервер відповідає за обробку запитів, виконання логіки оцінювання знань, збереження даних та формування рекомендацій. Використання бази даних дозволяє забезпечити централізоване зберігання інформації та підтримку цілісності даних.

Важливою складовою архітектури виступає розподіл системи на логічні рівні, що дозволяє забезпечити структурованість програмного коду та полегшити його підтримку. Рівні архітектури застосунку представлено в таблиці 2.4.

Таблиця 2.4 – Рівні архітектури мобільного застосунку

Рівень системи	Основне призначення
Рівень представлення	Взаємодія з користувачем, відображення інтерфейсу
Рівень логіки	Обробка даних, виконання алгоритмів оцінювання
Рівень даних	Збереження та отримання інформації з бази даних

Рівень представлення реалізується у вигляді мобільного інтерфейсу, який забезпечує взаємодію користувача з системою, відображення результатів тестування та доступ до навчальних матеріалів. Важливим аспектом є забезпечення інтуїтивно зрозумілого інтерфейсу, що дозволяє користувачеві швидко орієнтуватися у функціональних можливостях застосунку.

Рівень бізнес-логіки відповідає за реалізацію алгоритмів обробки даних, включаючи аналіз результатів тестування, визначення рівня знань та формування рекомендацій. Саме на цьому рівні, реалізується інтелектуальна складова системи, що забезпечує адаптацію навчального процесу до індивідуальних особливостей користувача.

Рівень доступу до даних забезпечує взаємодію з базою даних, включаючи операції збереження, оновлення та отримання інформації. Використання структурованої бази даних дозволяє забезпечити ефективну організацію інформації та підтримку цілісності даних в системі [22].

Окрім розподілу на рівні, важливим аспектом архітектури виступає організація взаємодії між клієнтом та сервером, яка здійснюється за допомогою

API. Використання REST-підходу дозволяє забезпечити стандартизований обмін даними у форматі JSON, що сприяє сумісності між різними компонентами системи та полегшує інтеграцію з іншими сервісами. Основні компоненти архітектури системи наведено в таблиці 2.5.

Таблиця 2.5 – Основні компоненти архітектури системи

Компонент	Призначення
Мобільний застосунок	Інтерфейс користувача та взаємодія із системою
Сервер	Обробка запитів та виконання бізнес-логіки
База даних	Збереження інформації про користувачів і результати
API	Забезпечення обміну даними між клієнтом і сервером
Аналітичний модуль	Аналіз результатів тестування та формування рекомендацій

Сукупність зазначених компонентів формує цілісну архітектуру системи, яка забезпечує ефективну взаємодію між різними частинами програмного продукту та дозволяє реалізувати всі необхідні функціональні можливості. Обрана архітектурна модель дозволяє досягти високого рівня гнучкості, забезпечити масштабованість та створити основу для подальшого розвитку системи.

Вибір клієнт-серверної архітектури з багаторівневою організацією є обґрунтованим рішенням для розробки мобільної інтелектуальної системи оцінювання знань з іноземної мови, оскільки дозволяє забезпечити ефективну обробку даних, адаптацію функціоналу до потреб користувачів та підтримку високого рівня продуктивності програмного забезпечення.

2.3 Структуризація програмних модулів системи

Проєктування структури програмних модулів мобільної інтелектуальної системи оцінювання знань з іноземної мови передбачає визначення логічного поділу функціональності застосунку на окремі компоненти, кожен з яких виконує чітко визначену роль у межах загальної архітектури. Даний підхід

дозволяє забезпечити зручність розробки, підвищити рівень підтримуваності програмного коду та створити передумови для подальшого розширення функціональних можливостей системи без суттєвого впливу на вже реалізовані компоненти.

Доцільним є використання модульного підходу, який передбачає розподіл системи на незалежні функціональні блоки, що взаємодіють між собою через чітко визначені інтерфейси. Подібна організація дозволяє ізолювати окремі частини системи, спростити процес тестування та забезпечити можливість паралельної розробки різних компонентів. Особливе значення має узгодженість між модулями, що гарантує коректну передачу даних та стабільність функціонування застосунку.

Структура системи формується на основі ключових функціональних можливостей, які були визначені на попередніх етапах проєктування. Кожен модуль відповідає за реалізацію окремого аспекту роботи застосунку, включаючи взаємодію з користувачем, обробку даних, виконання тестування та формування рекомендацій. Основні програмні модулі системи наведено в таблиці 2.6.

Таблиця 2.6 – Основні програмні модулі системи

№	Модуль	Призначення
1	Модуль автентифікації	Реєстрація користувачів та керування доступом
2	Модуль тестування	Формування та проведення тестів
3	Модуль обробки результатів	Аналіз відповідей та визначення рівня знань
4	Модуль навчальних матеріалів	Надання доступу до уроків
5	Модуль рекомендацій	Генерація персоналізованих порад
6	Модуль збереження даних	Робота з базою даних
7	Інтерфейсний модуль	Відображення інформації та взаємодія з користувачем

Наведена структура дозволяє визначити функціональне призначення кожного модуля та встановити межі відповідальності між компонентами системи. Важливим аспектом є взаємодія між модулями тестування, обробки

результатів та рекомендацій, оскільки саме їх узгоджена робота забезпечує реалізацію інтелектуальної складової застосунку.

Модуль автентифікації забезпечує контроль доступу до системи та зберігання інформації про користувачів, включаючи облікові дані та базові параметри профілю. Реалізація цього модуля передбачає використання механізмів захисту даних та перевірки достовірності введеної інформації, що є необхідною умовою для забезпечення безпеки системи.

Модуль тестування відповідає за формування структури тестових завдань, їх відображення в користувацькому інтерфейсі та збір відповідей. Важливим аспектом є підтримка різних типів завдань, що дозволяє оцінювати різні аспекти знання мови, включаючи лексику, граматику та розуміння тексту.

Модуль обробки результатів виконує аналіз отриманих відповідей, визначає рівень знань користувача та формує узагальнені показники успішності. Використання алгоритмів аналізу дозволяє враховувати не лише правильність відповідей, але й інші параметри, що характеризують процес виконання завдань.

Модуль навчальних матеріалів забезпечує доступ до освітнього контенту, який формується відповідно до рівня знань користувача. Організація навчальних матеріалів передбачає їх структуризацію за рівнями складності, що дозволяє забезпечити послідовність навчального процесу.

Модуль рекомендацій формує індивідуальні поради щодо подальшого навчання на основі результатів тестування. Реалізація цього модуля передбачає використання алгоритмів, здатних аналізувати слабкі місця в знаннях користувача та пропонувати відповідні навчальні матеріали.

Модуль збереження даних забезпечує взаємодію з базою даних, включаючи операції збереження результатів тестування, отримання інформації про користувачів та оновлення даних. Ефективна реалізація цього модуля дозволяє забезпечити стабільність роботи системи та підтримку цілісності інформації.

Інтерфейсний модуль відповідає за відображення інформації та забезпечення взаємодії користувача з системою. Взаємодія програмних модулів наведена в таблиці 2.7.

Таблиця 2.7 – Взаємодія програмних модулів

Модуль 1	Модуль 2	Характер взаємодії
Інтерфейсний	Модуль тестування	Передача дій користувача та відображення тестів
Модуль тестування	Модуль обробки результатів	Передача відповідей для аналізу
Модуль обробки результатів	Модуль рекомендацій	Формування рекомендацій на основі результатів
Модуль рекомендацій	Модуль навчальних матеріалів	Надання відповідного навчального контенту
Усі модулі	Модуль збереження даних	Збереження та отримання інформації

Узгодженість роботи модулів забезпечує цілісність функціонування системи та дозволяє реалізувати всі необхідні функціональні можливості.

Структуризація програмних модулів мобільної системи оцінювання знань з іноземної мови забезпечує ефективну організацію програмного забезпечення, підвищує його гнучкість та створює основу для подальшого розвитку. Чіткий розподіл функціональності між модулями дозволяє спростити процес розробки, забезпечити стабільність роботи системи та реалізувати інтелектуальні механізми аналізу результатів навчання.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЗЕПЕЧЕННЯ

3.1 Обґрунтування вибору технологічного стеку розробки

Проєктування програмного забезпечення мобільної інтелектуальної системи оцінювання знань з іноземної мови передбачає обґрунтований вибір технологічного стеку, який забезпечує реалізацію функціональних можливостей, стабільність роботи та зручність подальшого супроводу.

З урахуванням поставлених завдань доцільним є використання веб-орієнтованого технологічного стеку, який базується на поєднанні HTML, CSS та JavaScript для реалізації клієнтської частини системи. Застосування зазначених технологій забезпечує створення інтерфейсу користувача, який працює в браузері та не потребує встановлення додаткового програмного забезпечення. Приведена організація дозволяє забезпечити універсальний доступ до системи з різних пристроїв, включаючи персональні комп’ютери та мобільні пристрої [23].

Для підвищення структурованості інтерфейсу та спрощення розробки доцільно використовувати сучасні бібліотеки, що дозволяють організувати інтерфейс у вигляді компонентів та забезпечити ефективне керування станом застосунку. Компонентний підхід сприяє повторному використанню елементів інтерфейсу та підвищує зрозумілість програмної структури. Використання динамічної взаємодії з користувачем дозволяє реалізувати функціонал тестування, відображення результатів та доступ до навчальних матеріалів без перезавантаження сторінок [24].

Реалізація серверної частини системи з застосуванням середовища виконання JavaScript дозволяє використовувати єдину мову програмування для клієнтської та серверної логіки. Серверна частина відповідає за обробку запитів користувачів, виконання логіки оцінювання знань, формування рекомендацій та взаємодію з базою даних [25].

В якості системи керування базою даних використано SQLite. Використання SQLite дозволяє реалізувати збереження даних в межах одного

файлу без потреби в розгортанні окремого серверу бази даних, що значно полегшує процес розробки та тестування застосунку [26].

Застосування SQLite забезпечує збереження інформації про користувачів, результати тестування, структуру тестових завдань та навчальні матеріали. Вбудований характер цієї системи дозволяє досягти високої швидкості обробки запитів та забезпечити автономність роботи застосунку [27]. Для задач, пов’язаних з прототипуванням та демонстрацією функціоналу, зазначене рішення є повністю достатнім та обґрунтованим.

Вибір технологічного стеку враховує можливість подальшого розвитку системи, що передбачає потенційний перехід до більш складних рішень у разі необхідності масштабування. Архітектура клієнт-серверної взаємодії дозволяє в майбутньому замінити систему збереження даних на більш потужну без кардинальних змін в логіці роботи застосунку [28]. Технологічний стек розробки системи наведено в таблиці 3.1.

Таблиця 3.1 – Технологічний стек розробки системи

Компонент системи	Використана технологія	Призначення
Клієнтська частина	HTML, CSS, JavaScript	Формування інтерфейсу користувача
Інтерфейсна бібліотека	Компонентний підхід (JS)	Організація структури інтерфейсу
Серверна частина	JavaScript (серверне середовище)	Обробка запитів та виконання логіки
API	REST	Обмін даними між клієнтом і сервером
База даних	SQLite	Збереження інформації

Запропонований технологічний стек дозволяє реалізувати всі основні функціональні можливості системи, включаючи автентифікацію користувачів, проведення тестування, аналіз результатів та формування рекомендацій. Використання веб-технологій забезпечує універсальність доступу, тоді як

застосування вбудованої бази даних дозволяє спростити процес розробки та забезпечити стабільну роботу прототипу.

3.2 Реалізація механізму автентифікації та керування користувачами

Забезпечення контролю доступу до функціональних можливостей мобільної інтелектуальної системи оцінювання знань з іноземної мови виступає одним з ключових аспектів розробки програмного забезпечення, оскільки саме механізм автентифікації визначає рівень захисту даних користувачів та стабільність функціонування системи. Реалізація відповідного функціоналу передбачає створення процедур реєстрації, входу до системи, збереження облікових даних та керування доступом до інформаційних ресурсів застосунку.

Механізм автентифікації базується на використанні облікових записів користувачів, які зберігаються в базі даних. Кожен користувач має унікальний набір ідентифікаційних параметрів, що дозволяє однозначно визначити його в системі та забезпечити персоналізацію навчального процесу. До основних атрибутів облікового запису належать ідентифікатор користувача, електронна адреса, пароль та додаткові характеристики, що відображають рівень знань та прогрес навчання.

Процес автентифікації передбачає перевірку введених користувачем даних шляхом їх порівняння з записами, що зберігаються в базі даних. Для підвищення рівня безпеки зберігання паролів здійснюється в зашифрованому вигляді, що дозволяє запобігти несанкціонованому доступу до конфіденційної інформації. Використання механізмів шифрування забезпечує додатковий рівень захисту та відповідає сучасним вимогам до безпеки інформаційних систем [29].

Процес взаємодії користувача з системою включає декілька етапів, починаючи з реєстрації нового облікового запису та завершуючи входом до системи з подальшим доступом до функціоналу застосунку. Кожен етап супроводжується перевіркою коректності введених даних та обробкою

можливих помилок, що забезпечує стабільність роботи системи та зручність використання. Схема процесу автентифікації користувача наведена на рис. 3.1.

Рис. 3.1 Схема процесу автентифікації користувача

Розглянута схема демонструє логіку перевірки облікових даних користувача та визначає основні етапи процесу входу до системи. При успішній автентифікації, користувач отримує доступ до функціональних можливостей застосунку, тоді як у випадку помилки система повідомляє про необхідність повторного введення даних.

Окрім механізму автентифікації, важливим аспектом є організація керування користувачами, що передбачає збереження інформації про їх діяльність, результати тестування та рівень знань. Це дозволяє забезпечити

персоналізацію навчального процесу та формування індивідуальних рекомендацій. Система повинна підтримувати можливість оновлення даних користувача, включаючи зміну пароля, редагування профілю та перегляд історії результатів. Структура даних користувача наведена на в таблиці 3.2.

Таблиця 3.2 – Структура даних користувача

Атрибут	Опис призначення
ID	Унікальний ідентифікатор користувача
Електронна адреса	Використовується для входу до системи
Пароль	Зберігається у зашифрованому вигляді
Рівень знань	Поточний рівень володіння мовою
Історія тестування	Дані про результати виконаних тестів
Прогрес навчання	Інформація про виконані навчальні модулі

Представлена структура дозволяє забезпечити повноцінне збереження інформації про користувача та підтримку функціоналу персоналізації. Збереження історії тестування дозволяє аналізувати динаміку змін результатів, що є важливим для формування рекомендацій та оцінювання ефективності навчального процесу [30].

Важливим аспектом реалізації механізму керування користувачами є обмеження доступу до функціональних можливостей системи залежно від ролі користувача. Зокрема, звичайний користувач має доступ до проходження тестів та перегляду результатів, тоді як адміністратор отримує можливість керування тестовими завданнями та перегляду статистичних даних. Розмежування доступу дозволяє забезпечити безпеку системи та уникнути несанкціонованих змін в структурі даних. Розмежування доступу користувачів наведено в таблиці 3.3.

Таблиця 3.3 – Розмежування доступу користувачів

Роль користувача	Доступні функції
Користувач	Проходження тестів, перегляд результатів, навчання
Адміністратор	Керування тестами, аналіз статистики, модерація даних

Забезпечення коректної реалізації механізму автентифікації та керування користувачами дозволяє створити надійну основу для функціонування системи, що гарантує захист даних та зручність використання. Узгодженість між процесами перевірки доступу та збереження інформації забезпечує стабільність роботи застосунку та підвищує рівень довіри користувачів до системи.

3.3 Створення підсистеми тестування для визначення рівня знань

Розробка підсистеми тестування виступає центральним елементом мобільної інтелектуальної системи оцінювання знань з іноземної мови, оскільки саме вона забезпечує визначення рівня підготовки користувача та формує основу для подальшого навчального процесу. Організація тестування передбачає створення структури завдань, механізмів їх відображення, обробки відповідей та формування результатів, що дозволяє реалізувати повний цикл оцінювання знань у межах програмного продукту.

Функціонування підсистеми тестування базується на використанні набору тестових завдань, які формуються відповідно до визначених рівнів складності та типів перевірки знань. Завдання можуть включати перевірку лексичних знань, граматичних конструкцій, розуміння тексту та здатності до перекладу. Використання різноманітних типів завдань дозволяє забезпечити комплексну оцінку мовних навичок користувача та підвищити точність визначення рівня знань.

Процес проходження тестування передбачає послідовне відображення завдань у користувацькому інтерфейсі, фіксацію відповідей та передачу отриманих даних для подальшої обробки. Важливим аспектом виступає забезпечення зручності взаємодії користувача із системою, що досягається за рахунок чіткої структури тесту, зрозумілого інтерфейсу та можливості навігації між завданнями. Реалізація відповідних механізмів дозволяє мінімізувати кількість помилок під час виконання тесту та підвищити точність отриманих результатів.

Таблиця 3.4 – Типи тестових завдань у системі

Тип завдання	Характеристика
Вибір правильної відповіді	Користувач обирає один варіант із запропонованих
Множинний вибір	Можливість обрати декілька правильних відповідей
Відкрите питання	Введення текстової відповіді користувачем
Встановлення відповідності	Поєднання елементів із різних списків
Переклад	Переклад слова або речення

Наведені типи завдань дозволяють охопити різні аспекти знання мови та забезпечити різноманітність тестування, що позитивно впливає на якість оцінювання. Використання комбінованих типів завдань дозволяє сформувати більш повне уявлення про рівень підготовки користувача.

Організація структури тесту передбачає формування набору завдань, які об’єднуються у логічні блоки відповідно до рівня складності. Кожен тест містить певну кількість завдань, що дозволяє отримати достатній обсяг даних для аналізу результатів. Важливим аспектом є можливість варіювання складності тесту залежно від рівня підготовки користувача, що дозволяє адаптувати процес оцінювання до індивідуальних особливостей.

Процес обробки відповідей користувача передбачає визначення правильності виконання кожного завдання та формування узагальненого результату тестування. Для цього використовується система оцінювання, яка враховує кількість правильних відповідей, складність завдань та інші параметри, що характеризують процес виконання тесту.

Отримані результати зберігаються у базі даних та використовуються для подальшого аналізу. Схема роботи підсистеми тестування наведена на рисунку 3.2.

Наявність циклічної структури дозволяє забезпечити проходження всіх завдань тесту, після чого здійснюється підсумковий аналіз результатів.

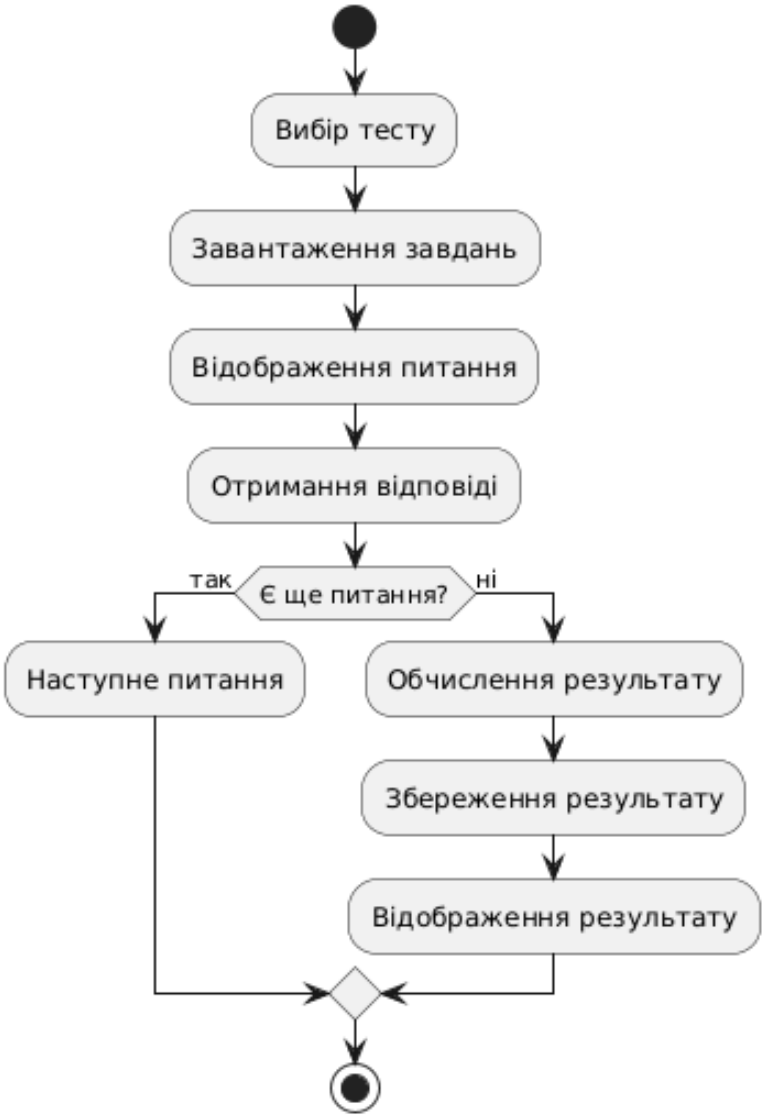

Рис. 3.2 Схема роботи підсистеми тестування

Важливим аспектом реалізації підсистеми виступає збереження результатів тестування, що дозволяє аналізувати прогрес користувача та використовувати отримані дані для формування рекомендацій. Збережена інформація включає кількість правильних відповідей, час виконання тесту та рівень складності завдань, що дозволяє отримати повну картину навчальної діяльності. Структура результатів тестування наведена в таблиці 3.5.

Таблиця 3.5 – Структура результатів тестування

Атрибут	Опис
ID користувача	Ідентифікатор користувача
ID тесту	Ідентифікатор тесту
Кількість правильних	Число правильних відповідей
Загальна кількість	Загальна кількість завдань
Час виконання	Тривалість проходження тесту
Рівень знань	Визначений рівень підготовки

Збереження зазначених параметрів дозволяє здійснювати подальший аналіз результатів та формувати рекомендації щодо навчання. Використання структурованого підходу до збереження даних забезпечує ефективну обробку інформації та підтримку функціональних можливостей системи.

Реалізація гнучкої структури тестів, підтримка різних типів завдань та ефективна обробка результатів дозволяють забезпечити високу якість оцінювання та адаптацію навчального процесу до індивідуальних потреб користувача.

3.4 Розробка навчального контенту та демонстраційних уроків

Розроблення навчального контенту виступає важливою складовою, яка забезпечує не лише перевірку рівня підготовки користувача, але й подальший розвиток мовних навичок. Формування навчальних матеріалів повинно враховувати рівень складності, послідовність подання інформації та можливість адаптації до індивідуальних потреб користувача. Організація демонстраційних уроків дозволяє реалізувати базовий функціонал навчання без необхідності створення великого обсягу контенту.

Навчальний контент формується у вигляді структурованих модулів, кожен з яких орієнтований на розвиток певного аспекту володіння мовою. Дана

організація дозволяє забезпечити логічну послідовність навчання та поступове ускладнення матеріалу.

Кожен навчальний модуль включає теоретичну частину, приклади використання мовних конструкцій та практичні завдання для закріплення матеріалу, що дозволяє забезпечити комплексний підхід до навчання, що поєднує пояснення нового матеріалу та його практичне застосування. Важливим аспектом є інтерактивність навчального процесу, яка реалізується через використання вправ, що передбачають активну участь користувача. Структура навчального модуля наведена в таблиці 3.6.

Таблиця 3.6 – Структура навчального модуля

Елемент модуля	Опис призначення
Теоретичний блок	Пояснення мовних правил та конструкцій
Приклади	Демонстрація використання мовних елементів
Практичні завдання	Виконання вправ для закріплення матеріалу
Контрольні питання	Перевірка засвоєння матеріалу
Підсумок	Узагальнення основних положень

Демонстраційні уроки, орієнтовані на базовий рівень знань та включають теми, пов’язані з повсякденним спілкуванням. Обмежена кількість уроків дозволяє зосередитися на якості реалізації функціоналу та продемонструвати принципи роботи системи без необхідності створення великої бази навчальних матеріалів.

Важливою особливістю є можливість розширення контенту в майбутньому шляхом додавання нових модулів. Схема проходження навчального модуля наведена на рисунку 3.3.

Рис. 3.3 Схема проходження навчального модуля

Наявність можливості повторного виконання завдань дозволяє користувачеві закріпити матеріал та покращити результати навчання.

Система повинна забезпечувати підбір навчальних модулів, які відповідають поточному рівню підготовки, що дозволяє уникнути надмірної складності або недостатнього рівня навчального навантаження.

Окрім цього, важливо забезпечити зручне відображення навчального контенту в користувацькому інтерфейсі, що передбачає чітку структуру матеріалів, використання зрозумілих елементів навігації та адаптацію до різних типів пристроїв. Якісна реалізація інтерфейсу дозволяє зменшити когнітивне навантаження на користувача та підвищити комфорт використання системи. Основні характеристики навчального контенту наведено в таблиці 3.7.

Таблиця 3.7 – Основні характеристики навчального контенту

Характеристика	Опис
Структурованість	Поділ матеріалу на логічні модулі
Адаптивність	Врахування рівня знань користувача
Інтерактивність	Використання вправ та завдань
Доступність	Можливість використання на різних пристроях
Масштабованість	Можливість додавання нових навчальних матеріалів

Використання структурованого та адаптивного підходу дозволяє створити навчальне середовище, яке сприяє розвитку мовних навичок та підвищенню рівня підготовки користувача.

Реалізація структурованих модулів, інтерактивних вправ та адаптивного підбору матеріалів дозволяє створити ефективний інструмент для вивчення іноземної мови у цифровому середовищі.

3.5 Впровадження алгоритмів інтелектуального оцінювання результатів

Реалізація алгоритмів інтелектуального оцінювання результатів виступає ключовим елементом функціонування розроблюваної системи, оскільки саме цей компонент забезпечує перетворення сирих даних тестування в змістовну оцінку рівня знань користувача. Використання алгоритмічного підходу дозволяє враховувати не лише кількість правильних відповідей, але й інші параметри, що характеризують процес виконання завдань, включаючи складність питань, час виконання та динаміку змін результатів.

Базовий рівень оцінювання передбачає розрахунок відсотка правильних відповідей, що дозволяє отримати узагальнений показник успішності. Проте, для підвищення точності визначення рівня знань доцільно використовувати більш складні алгоритми, які враховують вагові коефіцієнти завдань залежно від їх складності. Застосування даного підходу, дозволяє підвищити об’єктивність

оцінювання та забезпечити більш точне відображення рівня підготовки користувача.

Кожному тестовому завданню присвоюється певний рівень складності, що впливає на його вагу у загальному результаті. Завдання вищого рівня складності мають більший вплив на підсумкову оцінку, що дозволяє враховувати не лише кількість правильних відповідей, але й їх якість. Приведена організація оцінювання забезпечує більш гнучкий підхід до визначення рівня знань та дозволяє уникнути ситуацій, коли користувач отримує високий результат за рахунок виконання лише простих завдань. Параметри оцінювання результатів тестування наведено в таблиці 3.8.

Таблиця 3.8 – Параметри оцінювання результатів тестування

Параметр	Опис
Кількість правильних	Число правильно виконаних завдань
Загальна кількість	Загальна кількість питань у тесті
Рівень складності	Вага завдань залежно від складності
Час виконання	Тривалість проходження тесту
Частота помилок	Аналіз типових помилок
Динаміка результатів	Зміна показників у процесі навчання

Використання зазначених параметрів дозволяє сформувати комплексний підхід до оцінювання, що враховує різні аспекти навчальної діяльності користувача. Поєднання кількісних та якісних показників забезпечує більш точне визначення рівня знань та створює основу для формування рекомендацій.

Наступним етапом алгоритмічної обробки є класифікація користувачів за рівнями знань, що здійснюється на основі отриманих результатів. Доцільно використовувати поділ на рівні, що відповідають загальноприйнятим стандартам оцінювання мовної компетенції. Визначення рівня здійснюється шляхом порівняння отриманого результату із встановленими пороговими значеннями. Класифікація рівнів знань користувача наведена в таблиці 3.9.

Таблиця 3.9 – Класифікація рівнів знань користувача

Рівень знань	Діапазон результату (%)	Характеристика рівня
Початковий	0-40	Базове розуміння окремих елементів мови
Середній	41-70	Можливість виконання типових завдань
Високий	71-100	Впевнене володіння мовними навичками

Застосування класифікації дозволяє спростити інтерпретацію результатів тестування та забезпечити зрозуміле представлення рівня знань користувача. Визначений рівень використовується для формування рекомендацій щодо подальшого навчання та підбору відповідного контенту.

Важливим аспектом інтелектуального оцінювання є аналіз помилок, що дозволяє визначити слабкі місця в знаннях користувача. Система повинна фіксувати типи допущених помилок та встановлювати закономірності їх виникнення, що створює основу для формування персоналізованих рекомендацій. Наприклад, часте допущення помилок в граматичних конструкціях може свідчити про необхідність повторного вивчення відповідного матеріалу. Схема інтелектуального оцінювання результатів наведена на рис. 3.4.

Рис. 3.4 Схема інтелектуального оцінювання результатів

Наведена схема демонструє послідовність виконання операцій у межах алгоритму оцінювання та дозволяє визначити ключові етапи обробки даних. Включення етапу аналізу складності завдань та формування рекомендацій забезпечує більш глибокий підхід до оцінювання знань користувача.

Реалізація алгоритмів інтелектуального оцінювання передбачає врахування динаміки змін результатів в процесі навчання, що дозволяє оцінити прогрес користувача та визначити ефективність навчального процесу. Збереження історії результатів створює можливість для побудови аналітичних моделей, що відображають розвиток знань.

4 ВПРОВАДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ СИСТЕМИ ОЦІНКИ РІВНЯ ЗНАНЬ З ІНОЗЕМНОЇ МОВИ

4.1 Проведення тестування мобільного застосунку

Після завершення реалізації програмного продукту, проведено функціональне тестування мобільно-адаптивного застосунку LinguaLevel AI, призначеного для оцінювання рівня знань з англійської мови, збереження результатів, формування рекомендацій і надання навчального контенту.

Тестування проводилось в середовищі браузера на персональному комп’ютері з подальшою перевіркою адаптації інтерфейсу до меншої ширини екрана. Оскільки програмний продукт має мобільно-адаптивну організацію, важливим критерієм перевірки стала здатність інтерфейсу зберігати читабельність, логічне розміщення елементів та зручність керування під час зміни розміру вікна браузера. Стартова форма авторизації та реєстрації користувача наведена на рисунку 4.1.

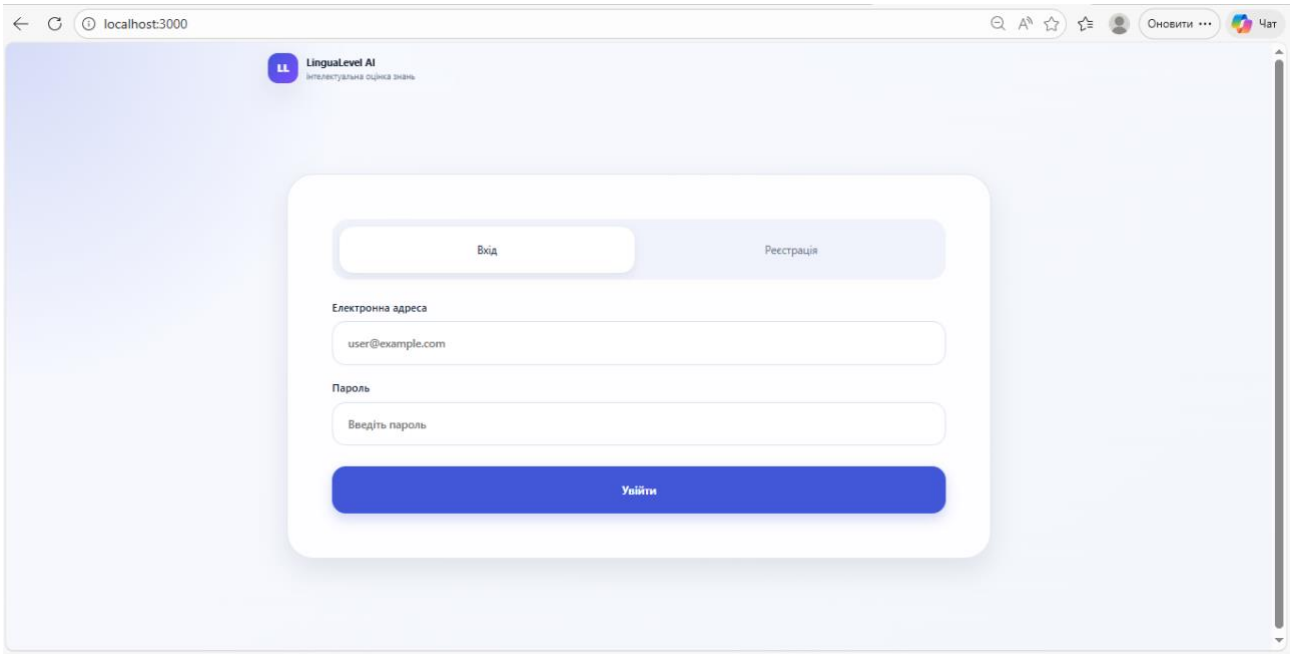

Рис. 4.1 Стартова форма авторизації та реєстрації користувача

Перевірка механізму автентифікації охоплювала реєстрацію нового користувача, вхід з коректними обліковими параметрами, відмову в доступі при неправильному паролі та вихід з системи. В процесі реєстрації вводилися ім’я, електронна адреса та пароль, після чого застосунок створював запис користувача в базі SQLite та відкривав персональне середовище. При повторному вході система перевіряла електронну адресу та пароль, повертала токен доступу і відображала кабінет користувача. За умови помилкового введення пароля, доступ не надавався, що підтвердило працездатність контролю доступу.

Після входу до системи користувач переходив до персонального середовища, де відображалися поточний рівень знань, середній результат, навчальний прогрес, кількість пройдених модулів, блок рекомендацій та перелік останніх результатів. Особливу увагу було приділено перевірці переходів між основними розділами: тестуванням, готовими курсами, демонстраційним online-навчанням та історією результатів. Навігація виконувалася без перезавантаження сторінки, а зміна екранів здійснювалася засобами клієнтської логіки JavaScript з передаванням запитів до серверної частини через REST API. Персональне середовище користувача після входу до системи наведено на рисунку 4.2.

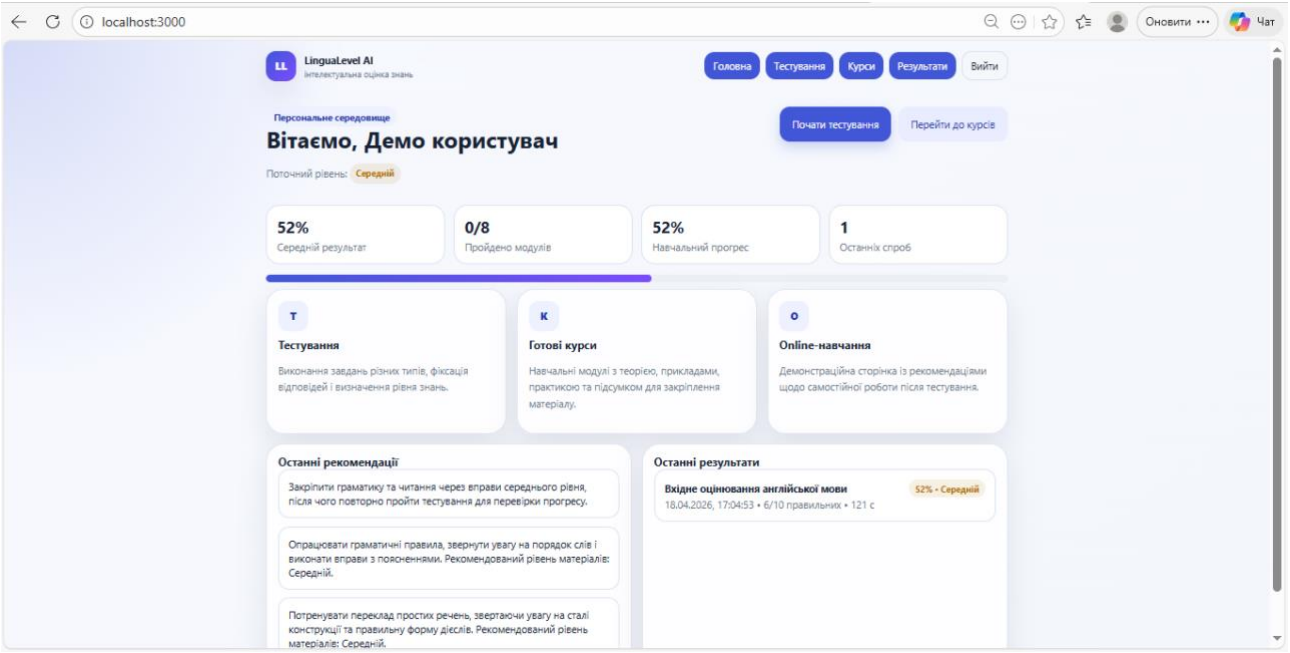

Рис. 4.2 Персональне середовище користувача після входу до системи

Окремо перевірялась підсистема тестування, яка є основним функціональним компонентом розробленого програмного продукту. Система відображала доступні тести, серед яких передбачено вхідне оцінювання англійської мови, перевірку граматики та лексики рівня A1-A2, а також, тест читання та розуміння тексту. Під час запуску тесту, застосунок послідовно подавав питання різних типів, а саме, вибір однієї правильної відповіді, вибір кількох відповідей, відкрите питання, переклад та встановлення відповідності між англійськими словами та українськими значеннями. Перехід між питаннями виконувався кнопками навігації, а прогрес проходження відображався у вигляді індикатора. Виконання тестового завдання у підсистемі оцінювання знань наведено на рисунку 4.3.

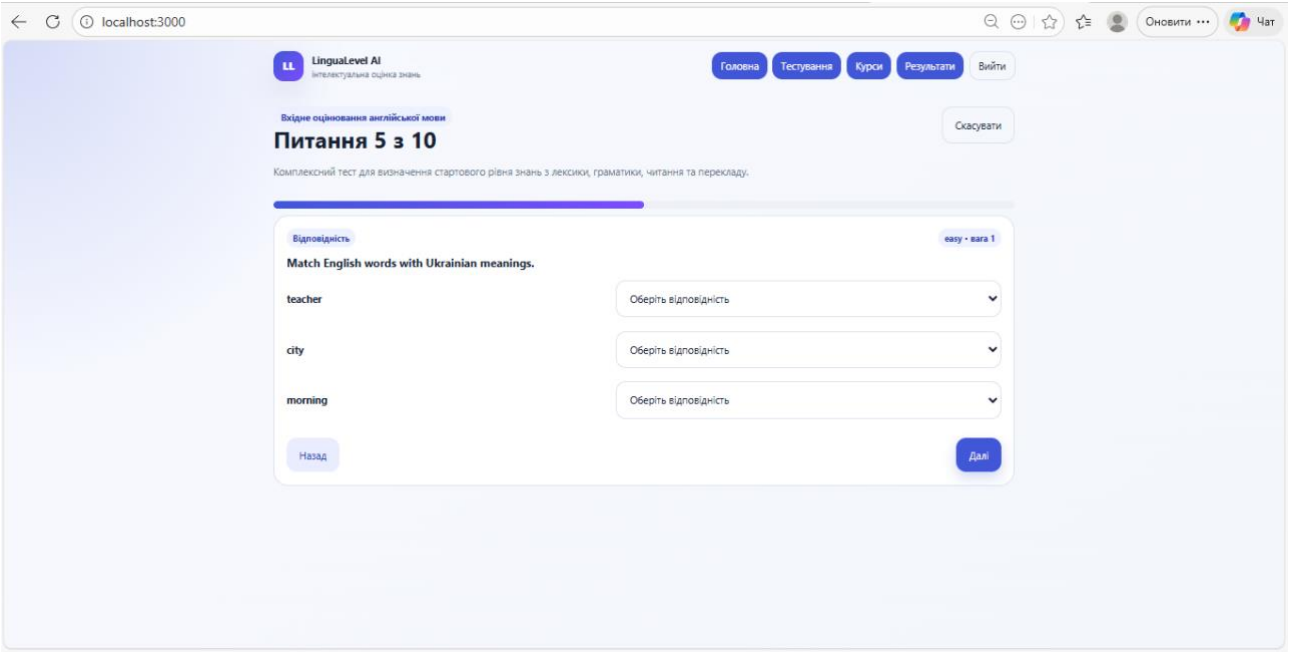

Рис. 4.3 Виконання тестового завдання у підсистемі оцінювання знань

Після завершення тесту, перевірялась робота алгоритму інтелектуального оцінювання. Сервер отримував відповіді користувача у форматі JSON, порівнював їх з правильними значеннями, враховував вагу кожного питання, визначав кількість правильних відповідей, обчислював підсумковий відсотковий результат та встановлював рівень знань. Для низького результату, присвоювався початковий рівень, для середнього показника встановлювався середній рівень, а

для високого відсотку правильних відповідей, система визначала високий рівень володіння матеріалом. Оцінювання враховувало не лише факт правильної відповіді, але й категорію помилки, складність завдання та тривалість проходження тесту.

Під час практичної перевірки встановлено, що застосунок коректно обробляє різні типи відповідей. Для завдань з одним варіантом, система перевіряла ідентифікатор обраної відповіді. Для завдань з кількома правильними варіантами, враховувалася часткова правильність, а вибір зайвих варіантів знижував результат. Для відкритих відповідей та перекладу, проводилась нормалізація введеного тексту, що дозволяло ігнорувати регістр символів та зайві пробіли. Для завдань на відповідність, оцінювалась частка правильно встановлених пар. Практична перевірка підтвердила, що алгоритм не обмежується простим підрахунком правильних відповідей, а формує зважену оцінку результату.

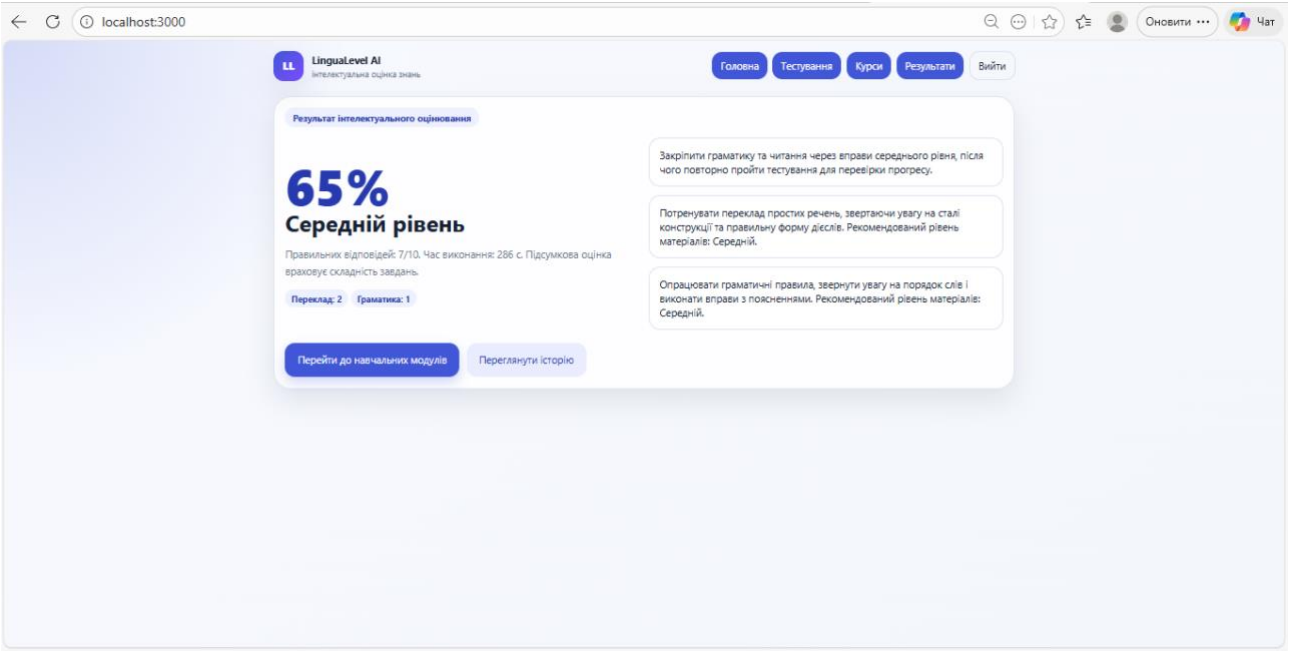

Рис. 4.4 Результат інтелектуального оцінювання після завершення тесту

Важливим етапом перевірки стала оцінка механізму формування рекомендацій. Після завершення тесту, система аналізувала категорії помилок та створювала персоналізовані поради щодо подальшого навчання. При наявності

помилок в лексиці, користувач отримував пораду повторити словниковий матеріал, при труднощах з граматикою, пропонувалось опрацювати правила побудови речень, а при помилках в читанні, рекомендувалась робота з короткими текстами. Окрім змістових рекомендацій, система визначала рівень навчальних матеріалів, які варто опрацювати після отримання результату.

Перевірка навчального блоку показала, що застосунок надає доступ до модулів початкового, середнього та високого рівнів. Кожен модуль містить теоретичний матеріал, приклади, практичні завдання і короткий підсумок. Після натискання кнопки перегляду рекомендованих матеріалів, система відбирала навчальні модулі з урахуванням поточного рівня користувача. Позначення модуля як виконаного оновлювало прогрес і впливало на статистичні показники персонального середовища.

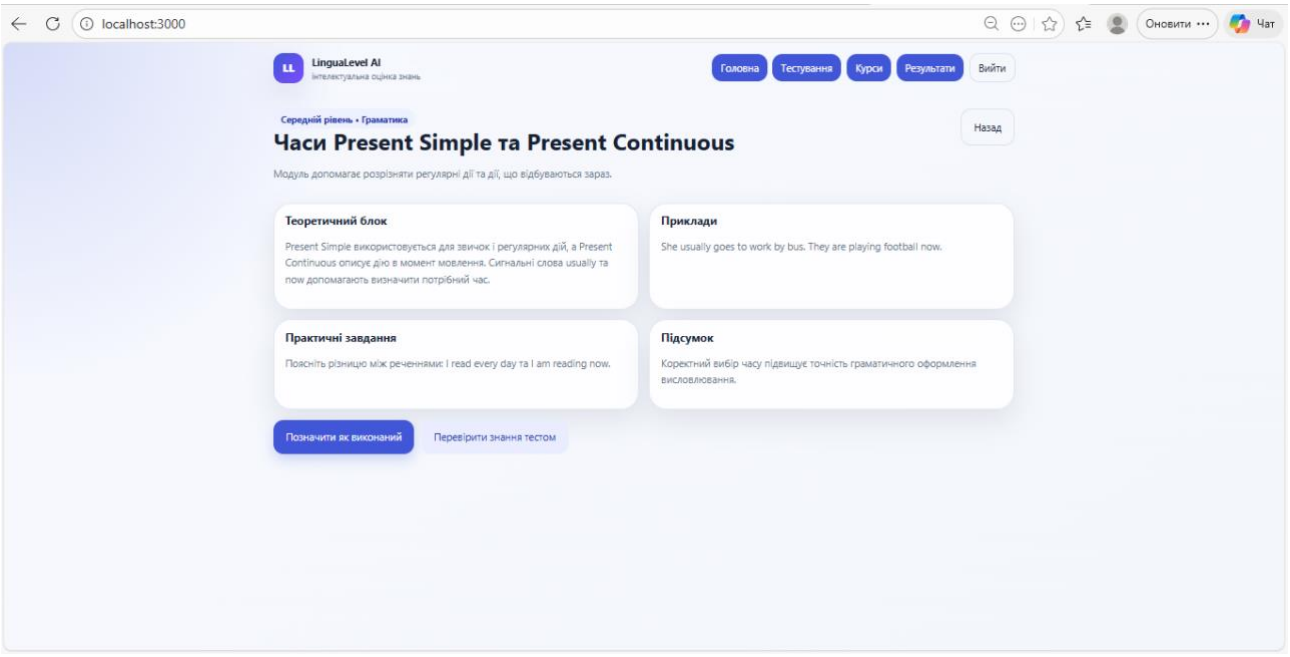

Рис. 4.5 Перегляд навчального модуля після проходження тестування

Додаткова перевірка адміністративного модуля, підтвердила можливість входу під обліковим записом адміністратора, перегляду загальної статистики користувачів, кількості проходжень тестування, середнього балу та переліку доступних тестів. Також, перевірено форму додавання нового питання, яка дозволяє вказати тест, тип завдання, категорію, складність, текст питання,

правильну відповідь і варіанти вибору. Після збереження питання воно ставало доступним у відповідному тесті, що підтвердило працездатність механізму адміністрування контенту. Адміністративна панель керування тестовими завданнями наведена на рисунку 4.6.

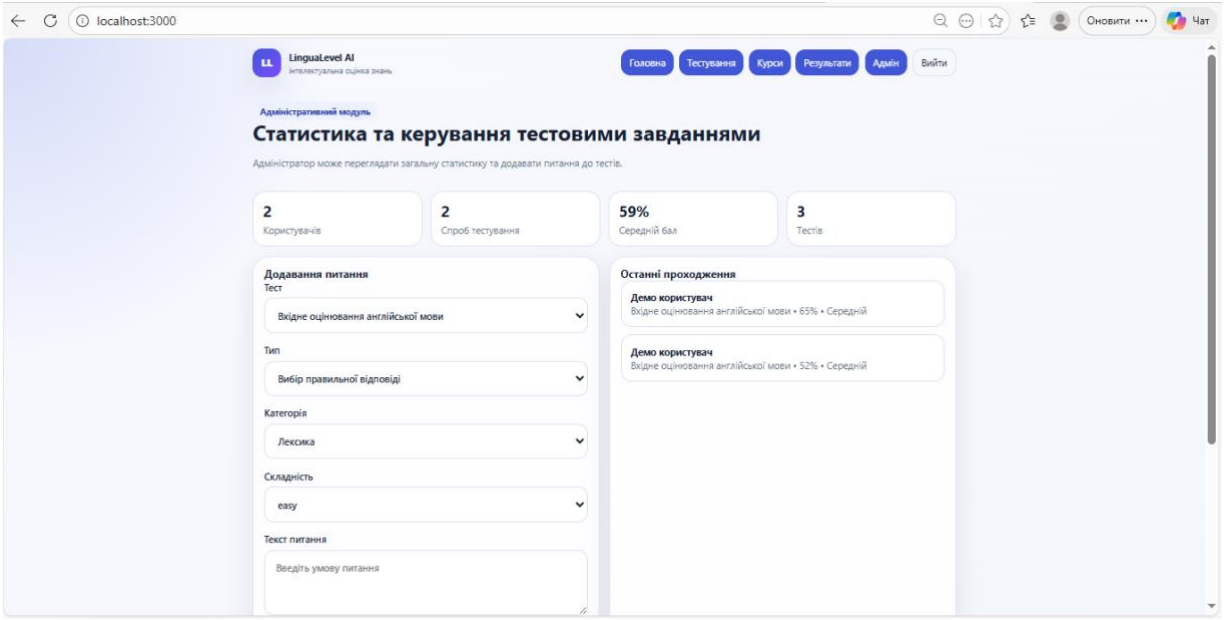

Рис. 4.6 Адміністративна панель керування тестовими завданнями

Під час перевірки збереження інформації було встановлено, що база SQLite коректно фіксує користувачів, тестові питання, відповіді, результати проходження, деталізацію помилок, рекомендації та прогрес навчальних модулів. Після перезапуску сервера? результати не втрачалися, що підтвердило використання постійного файлового сховища. Окремо, перевірено оновлення рівня знань користувача після завершення тестування та зміну навчального прогресу після виконання модуля.

4.2 Аналіз результатів функціонування системи оцінювання

Після перевірки працездатності мобільно-адаптивного застосунку LinguaLevel AI, проведено аналіз результатів функціонування системи оцінювання, спрямований на встановлення коректності визначення рівня знань користувача, адекватності сформованих рекомендацій, стабільності збереження

результатів та відповідності роботи аналітичного модуля поставленим вимогам. Основна увага приділялась не лише факту успішного завершення тесту, але й правильності інтерпретації отриманих відповідей, оскільки саме аналітичний компонент забезпечує інтелектуальну складову програмного продукту.

Система оцінювання функціонує за принципом комплексної обробки результатів, де враховується кількість правильних відповідей, тип виконаного завдання, складність питання, категорія мовної компетентності та час проходження тесту. Внаслідок обробки введених користувачем даних, формується підсумковий відсотковий показник, на основі якого визначається рівень володіння навчальним матеріалом. Окрім загального результату, система аналізує тематичні напрями помилок, що дозволяє сформувати рекомендації не загального характеру, а пов’язані з конкретними прогалинами в знаннях.

Під час контрольного проходження тестування, було перевірено декілька варіантів поведінки користувача, серед яких повністю правильне виконання завдань, частково правильне проходження тесту, наявність помилок в лексичних питаннях, неправильне виконання граматичних завдань та низький результат за більшістю категорій. У всіх випадках система коректно виконувала розрахунок підсумкового балу, зберігала результат у базі SQLite та оновлювала інформацію в персональному кабінеті користувача.

Аналіз результатів показав, що застосунок коректно розмежовує рівні підготовки користувача. При низькій частці правильних відповідей, система відносить користувача до початкового рівня та пропонує базові навчальні матеріали, орієнтовані на засвоєння простих граматичних конструкцій і базової лексики. При середньому результаті, користувач отримує рекомендації щодо закріплення граматичних структур, розширення словникового запасу та роботи з короткими текстами. При високому показнику, система визначає достатній рівень підготовки та пропонує матеріали підвищеної складності, спрямовані на удосконалення навичок читання, розуміння змісту та практичного використання мови.

Підсумковий бал формується на основі зваженого оцінювання відповідей, що дозволяє уникнути надмірного спрощення аналізу. Для завдань з вибором однієї відповіді, результат визначається за правильністю обраного варіанту, для завдань з кількома відповідями, враховується часткова відповідність, для відкритих відповідей виконується текстова нормалізація, а для завдань на відповідність, оцінюється кількість правильно встановлених пар. Завдяки диференційованій логіці перевірки, система забезпечує більш точне відображення реального рівня знань користувача. Узагальнені результати перевірки сценаріїв оцінювання наведено в таблиці 4.1.

Таблиця 4.1 – Узагальнені результати перевірки сценаріїв оцінювання

Варіант проходження тесту	Характер відповідей користувача	Очікувана реакція системи	Отриманий результат
Висока успішність	Більшість відповідей правильна, помилки поодинокі	Визначення високого рівня та добір складніших матеріалів	Рівень визначено коректно, рекомендації відповідають результату
Середня успішність	Правильні відповіді чергуються з помилками	Встановлення середнього рівня та надання матеріалів для закріплення	Результат опрацьовано коректно, поради сформовано за категоріями
Низька успішність	Переважають неправильні відповіді	Визначення початкового рівня та пропозиція базового навчання	Система знизила рівень і запропонувала початкові модулі
Лексичні помилки	Найбільша кількість неправильних відповідей пов’язана зі словником	Формування рекомендацій щодо повторення лексики	Поради спрямовано на розширення словникового запасу
Граматичні помилки	Помилки виникають у завданнях на побудову речень	Рекомендації щодо повторення граматичних правил	Система визначила слабкий напрям і запропонувала відповідний матеріал

Варіант проходження тесту	Характер відповідей користувача	Очікувана реакція системи	Отриманий результат
Помилки читання	Неправильно виконано завдання з розуміння тексту	Рекомендації щодо роботи з короткими текстами	Аналітичний блок сформував поради для покращення читання

Результати перевірки засвідчили, що система не обмежується виведенням загального балу, а формує деталізовану оцінку виконання завдань. Аналітичний блок визначає слабкі напрями підготовки, зберігає інформацію про помилки та використовує її для добору навчального контенту. Дана логіка забезпечує зв’язок між тестуванням та подальшим навчанням, оскільки користувач після отримання результату має змогу перейти до матеріалів, які відповідають його реальним потребам.

Після завершення тесту, система створює запис в базі SQLite, де фіксуються дата проходження, назва тесту, набраний бал, визначений рівень, тривалість виконання та деталізація за категоріями. Після перезапуску сервера, інформація залишалася доступною, що підтвердило стабільність роботи файлового сховища та коректність операцій запису. Історія результатів тестування користувача наведена на рисунку 4.7.

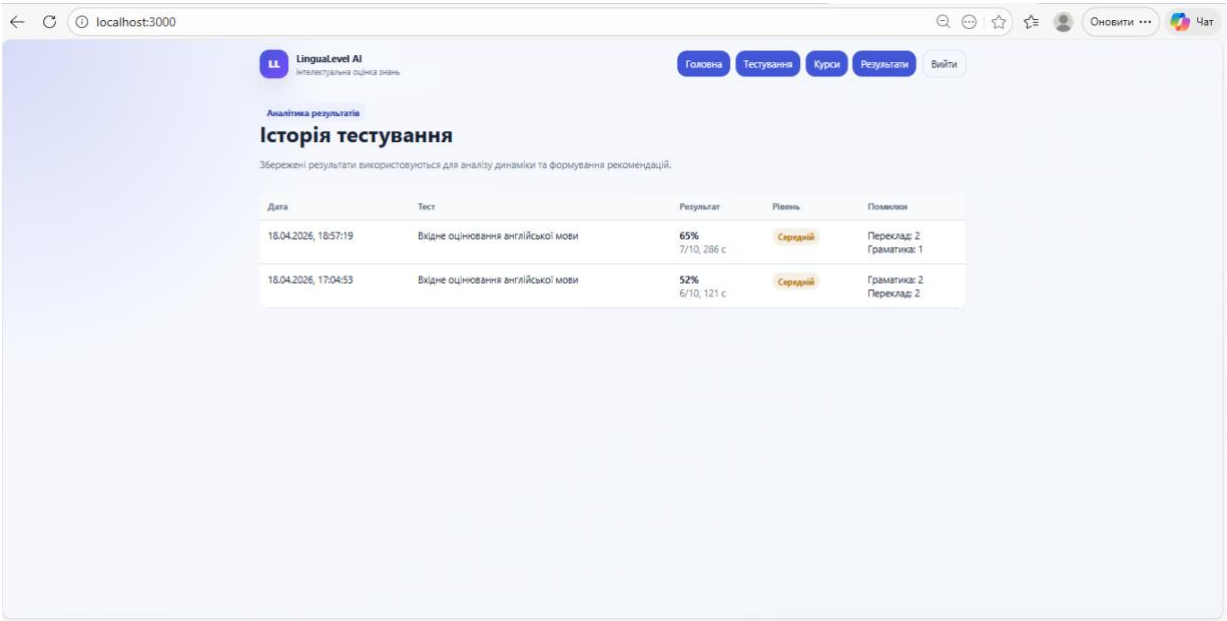

Рис. 4.7 Історія результатів тестування користувача

Аналіз роботи рекомендаційного модуля показав, що сформовані поради мають прикладне значення для користувача. Вони не дублюють підсумковий бал, а пояснюють, які напрями потребують додаткового опрацювання. Якщо користувач допускав помилки в словникових завданнях, система рекомендувала повторити тематичну лексику та перейти до базових вправ. Якщо проблема виникала в граматиці, користувач отримував пропозицію опрацювати правила побудови речень та виконати практичні завдання. Якщо основні труднощі були пов’язані з читанням, система спрямовувала користувача до навчальних текстів з поступовим ускладненням.

Після встановлення рівня знань користувач отримує доступ до матеріалів, які відповідають його підготовці. Початковий рівень пов’язується з базовими темами, середній рівень відкриває матеріали для поглиблення граматики та лексики, а високий рівень, орієнтує користувача на складніші завдання. Завдяки зв’язку між оцінюванням та навчальним контентом програмний продукт виконує не лише контрольну, але й навчально-коригувальну функцію. Відповідність результату тестування навчальним рекомендаціям представлена в таблиці 4.2.

Таблиця 4.2 – Відповідність результату тестування навчальним рекомендаціям

Показник результату	Визначений рівень	Характер рекомендації	Запропонований навчальний контент
Низький бал	Початковий	Повторення базової лексики та простих граматичних структур	Модулі рівня A1
Середній бал	Базово-середній	Закріплення граматики, розширення словника, робота з короткими текстами	Модулі рівня A1–A2
Високий бал	Середній або вищий	Ускладнення навчальних завдань і розвиток навичок читання	Матеріали рівня A2–B1
Виражені граматичні помилки	Потребує корекції граматики	Повторення правил побудови речень	Граматичні уроки з практичними вправами

Показник результату	Визначений рівень	Характер рекомендації	Запропонований навчальний контент
Виражені лексичні помилки	Потребує розширення словника	Опрацювання тематичних слів і фраз	Лексичні модулі
Труднощі з читанням	Потребує тренування розуміння тексту	Виконання завдань на змістове сприйняття	Тексти з питаннями до змісту

Проведений аналіз підтвердив працездатність всіх основних компонентів системи оцінювання. Клієнтська частина забезпечує зручне введення відповідей, серверна частина виконує обробку запитів та розрахунок результатів, база SQLite зберігає навчальну історію, а рекомендаційний модуль, формує індивідуальні поради. Взаємодія між компонентами відбувається послідовно, без втрати введених користувачем даних і без порушення логіки переходів між екранами.

Розроблена система оцінювання виконує заявлене функціональне призначення та забезпечує об’єктивну обробку результатів користувача. Розрахунок балу, визначення рівня знань, виявлення слабких категорій, збереження статистики та формування рекомендацій працюють узгоджено.

4.3 Формування вимог до програмного та апаратного забезпечення

Формування вимог до програмного та апаратного забезпечення є необхідною умовою коректного впровадження мобільно-адаптивної інтелектуальної системи оцінювання знань з англійської мови. Розроблений застосунок LinguaLevel AI, має клієнт-серверну організацію, тому для його стабільної роботи потрібно враховувати параметри серверного середовища, клієнтського пристрою, браузера, файлового сховища бази SQLite та мережевої взаємодії між інтерфейсом та серверною частиною. Вимоги сформовано з урахуванням фактичної реалізації програмного продукту, де клієнтська частина

створена засобами HTML, CSS та JavaScript, серверна логіка працює на Node.js, а збереження інформації здійснюється в базі SQLite.

Особливістю реалізованого програмного продукту є відсутність потреби у встановленні окремого сервера баз даних, що спрощує розгортання демонстраційної версії та зменшує кількість залежностей. База SQLite функціонує у вигляді локального файлу, який зберігає облікові записи користувачів, тестові завдання, відповіді, результати проходження, рекомендації та навчальний прогрес. З огляду на використання вбудованого SQLite-модуля середовища Node.js, важливо забезпечити встановлення актуальної версії Node.js, яка підтримує зазначений механізм роботи з базою.

Для запуску серверної частини, потрібна операційна система з підтримкою Node.js, доступом до локальної файлової системи та можливістю відкриття мережевого порту для локального веб-сервера. В демонстраційному варіанті, застосунок запускається на персональному комп’ютері через команду npm start, після чого інтерфейс відкривається в браузері за адресою localhost:3000. Оскільки сервер та клієнтська частина можуть працювати на одному пристрої, окрема мережева інфраструктура для локального тестування не є обов’язковою. Вимоги до серверного програмного середовища наведено в таблиці 4.3.

Таблиця 4.3 – Вимоги до серверного програмного середовища

Компонент	Мінімальні вимоги	Рекомендовані вимоги	Призначення
Операційна система	Windows 10, Linux або macOS із підтримкою Node.js	Windows 11 або актуальний дистрибутив Linux	Забезпечення запуску серверної частини
Середовище виконання	Node.js 22.5 або новіша версія	Node.js актуальної стабільної гілки	Виконання серверної логіки та REST API
Менеджер пакетів	npm, що встановлюється разом із Node.js	npm останньої доступної версії	Запуск проєкту та обслуговування залежностей

Компонент	Мінімальні вимоги	Рекомендовані вимоги	Призначення
База даних	SQLite у складі проєкту	SQLite із регулярним резервним копіюванням файлу бази	Збереження користувачів, тестів і результатів
Мережевий порт	Доступний порт 3000	Можливість зміни порту через конфігурацію	Обробка HTTP-запитів браузера
Файлова система	Дозвіл на читання й запис у папці проєкту	Окремий каталог із правами резервного копіювання	Збереження бази, журналів і службових файлів

Наведені вимоги забезпечують стабільний запуск серверної частини та коректну роботу всіх модулів, пов’язаних з автентифікацією, тестуванням, аналізом відповідей, формуванням рекомендацій і доступом до навчального контенту. Для навчального або демонстраційного використання, достатньо локального комп’ютера середнього рівня продуктивності, оскільки обсяг операцій є відносно невеликим, а база SQLite не потребує окремого серверного процесу.

Клієнтська частина застосунку працює у веб-браузері, що дозволяє використовувати систему на персональних комп’ютерах, ноутбуках, планшетах і смартфонах. Інтерфейс має адаптивну верстку, тому його елементи автоматично змінюють розміщення залежно від ширини екрана. Для коректного відображення сторінок потрібен браузер з підтримкою сучасних стандартів HTML5, CSS3 і JavaScript. Важливими умовами є ввімкнене виконання JavaScript, підтримка локальних HTTP-запитів і стабільна робота механізмів обробки форм. Вимоги до клієнтського програмного забезпечення наведено в таблиці 4.4.

Таблиця 4.4 – Вимоги до клієнтського програмного забезпечення

Компонент	Мінімальні вимоги	Рекомендовані вимоги	Призначення
Веб-браузер	Google Chrome, Microsoft Edge, Mozilla Firefox або Safari актуальних версій	Оновлений Chrome або Edge	Відображення інтерфейсу застосунку
Підтримка JavaScript	Увімкнене виконання сценаріїв	Увімкнене виконання сценаріїв без блокування локальних запитів	Робота тестів, навігації та форм
Підтримка CSS3	Базова підтримка адаптивної верстки	Повна підтримка flex, grid і медіазапитів	Коректне масштабування інтерфейсу
Доступ до локального сервера	Можливість відкриття адреси `localhost:3000`	Стабільна робота локального браузерного з’єднання	Взаємодія з серверною частиною
Масштаб сторінки	Значення браузера 100 %	Значення браузера 100 % або системна адаптація дисплея	Збереження пропорцій інтерфейсу

З боку користувача, додаткове встановлення спеціалізованого клієнтського програмного забезпечення не потрібне. Доступ до функціоналу здійснюється через браузер, що відповідає принципу універсальності застосунку. Перевагою веб-орієнтованої реалізації є можливість використання системи на різних пристроях без створення окремих інсталяційних пакетів для мобільних платформ.

Апаратні вимоги до серверного пристрою залежать від кількості одночасних користувачів та характеру експлуатації. Для локального тестування або демонстрації, достатньо стандартного персонального комп’ютера, який здатний виконувати Node.js та забезпечувати роботу браузера. Оскільки основні операції системи пов’язані з обробкою невеликих JSON-запитів, перевіркою відповідей та записом результатів в локальну базу, потреба у високопродуктивному обладнанні відсутня. При перенесенні системи на

віддалений сервер, доцільно збільшити обсяг оперативної пам’яті, забезпечити стабільний накопичувач та налаштувати резервне копіювання бази. Вимоги до апаратного забезпечення серверного пристрою наведено в таблиці 4.5.

Таблиця 4.5 – Вимоги до апаратного забезпечення серверного пристрою

Параметр	Мінімальна конфігурація	Рекомендована конфігурація	Обґрунтування
Процесор	Двоядерний процесор із частотою від 1,8 ГГц	Чотириядерний процесор із частотою від 2,4 ГГц	Обробка запитів і виконання серверної логіки
Оперативна пам’ять	4 ГБ	8 ГБ або більше	Стабільна робота Node.js, браузера та системних процесів
Накопичувач	500 МБ вільного місця	2 ГБ або більше з резервом для бази та копій	Розміщення проєкту, бази SQLite і службових файлів
Тип накопичувача	HDD	SSD	Швидший доступ до файлу бази та ресурсів застосунку
Мережевий адаптер	Локальна мережа або внутрішній доступ до `localhost`	Стабільне Ethernet або Wi-Fi з’єднання	Робота клієнт-серверної взаємодії
Дисплей для адміністрування	Роздільна здатність від 1366×768	Full HD або вище	Зручність роботи з панеллю адміністратора

Апаратні вимоги до клієнтського пристрою визначаються насамперед потребою в комфортному відображенні інтерфейсу та швидкому виконанні браузерних сценаріїв. Застосунок може використовуватися на смартфонах, планшетах та персональних комп’ютерах, оскільки адаптивна верстка підтримує різні розміри екранів. Для повноцінного проходження тестування, користувачеві потрібен екран, достатній для читання питань, вибору варіантів відповідей та

перегляду результатів. Вимоги до апаратного забезпечення клієнтського пристрою наведено в таблиці 4.6.

Таблиця 4.6 – Вимоги до апаратного забезпечення клієнтського пристрою

Параметр	Мінімальна конфігурація	Рекомендована конфігурація	Обґрунтування
Тип пристрою	Смартфон, планшет, ноутбук або ПК	Смартфон із сучасним браузером або ноутбук	Доступ до веб-інтерфейсу
Процесор	Мобільний або настільний процесор базового рівня	Багатоядерний процесор середнього рівня	Плавна робота інтерфейсу
Оперативна пам’ять	2 ГБ	4 ГБ або більше	Стабільна робота браузера
Дисплей	Ширина екрана від 360 пікселів	Екран від 6 дюймів або роздільна здатність HD і вище	Зручне проходження тестів
Пристрій введення	Сенсорний екран, клавіатура або миша	Сенсорний екран із клавіатурою браузера	Введення відповідей і навігація
Інтернет-з’єднання	Локальний доступ до сервера або мережеве з’єднання	Стабільне Wi-Fi або мобільне з’єднання	Надсилання відповідей і отримання результатів

Для демонстраційного середовища достатньо локального запуску та обмеженого доступу до папки проєкту. При використанні системи в навчальному закладі або на віддаленому сервері, потрібно забезпечити захищене з’єднання, регулярне створення резервних копій файлу SQLite, обмеження доступу до адміністративного профілю та контроль прав на зміну тестового контенту. Окрему увагу потрібно приділити збереженню паролів в захищеному вигляді, оскільки облікові параметри користувачів належать до чутливої інформації навчального сервісу.

Під час внесення змін до клієнтських файлів, необхідно перезапустити серверну частину та очистити кеш браузера або виконати повне оновлення

сторінки. При зміні структури бази даних, потрібно попередньо створити резервну копію файлу SQLite, щоб уникнути втрати результатів користувачів. Для супроводу системи, бажано зберігати вихідний код в контрольованому каталозі та документувати зміни, пов’язані з тестами, навчальними модулями та алгоритмом оцінювання.

4.4 Надання рекомендацій щодо експлуатації та розвитку системи

Після проведення тестування, аналізу результатів функціонування та визначення вимог до програмного і апаратного забезпечення, сформовано рекомендації щодо експлуатації та подальшого розвитку мобільно-адаптивної інтелектуальної системи оцінювання знань з англійської мови LinguaLevel AI. Рекомендації спрямовані на забезпечення стабільної роботи програмного продукту, збереження результатів користувачів, підтримку актуальності навчального контенту, підвищення точності оцінювання та створення передумов для майбутнього розширення функціональних можливостей.

Експлуатація розробленого застосунку, має ґрунтуватися на коректному запуску серверної частини, перевірці доступності локального веб-інтерфейсу та контролі працездатності бази SQLite. Перед початком роботи, необхідно переконатися, що середовище Node.js встановлене коректно, папка проєкту містить всі службові файли, а порт локального сервера не зайнятий іншими процесами. Після запуску команди npm start, користувач відкриває веб-інтерфейс в браузері та виконує вхід або реєстрацію. При внесенні змін до клієнтської частини, потрібно виконувати повне оновлення сторінки браузера, оскільки кешування файлів CSS і JavaScript, може призвести до відображення попередньої версії інтерфейсу.

Під час практичного використання системи, важливо забезпечити регулярне резервне копіювання файлу бази SQLite, оскільки саме в ньому зберігаються облікові записи, тестові питання, результати проходження, рекомендації та навчальний прогрес. Втрата файлу бази може призвести до

порушення цілісності освітньої статистики, тому копіювання потрібно виконувати перед оновленням структури проєкту, додаванням нових тестів, зміною алгоритму оцінювання або перенесенням застосунку на інший пристрій.

Адміністративна панель надає можливість керувати тестовими завданнями, переглядати статистичні показники та впливати на зміст навчально-оцінювального середовища, тому облікові дані адміністратора не мають передаватися звичайним користувачам. Під час експлуатації, необхідно використовувати складний пароль, обмежувати кількість осіб з правами керування та періодично перевіряти правильність доданих питань, варіантів відповідей та категорій оцінювання. Рекомендації щодо експлуатації програмного продукту наведено в таблиці 4.7.

Таблиця 4.7 – Рекомендації щодо експлуатації програмного продукту

Напрям експлуатації	Рекомендовані дії	Очікуваний результат
Запуск застосунку	Перевіряти встановлення Node.js, запускати сервер через `npm start`, відкривати адресу `localhost:3000`	Стабільне відкриття веб-інтерфейсу та доступ до функцій системи
Оновлення інтерфейсу	Після зміни файлів CSS або JavaScript виконувати перезапуск сервера й повне оновлення сторінки	Коректне відображення нової версії дизайну
Збереження результатів	Створювати резервні копії SQLite-файлу перед оновленням або перенесенням проєкту	Захист навчальної історії від втрати
Адміністрування	Обмежувати доступ до адміністративного профілю та контролювати якість тестових завдань	Підтримка достовірності оцінювання
Робота користувачів	Рекомендувати завершувати тест без закриття вкладки браузера	Запобігання втраті проміжних відповідей
Перевірка після змін	Проводити коротке контрольне тестування після редагування питань або логіки оцінювання	Підтвердження коректності роботи системи

Для ефективного використання користувач має спочатку пройти вхідне оцінювання, після чого ознайомитися з результатом, переглянути рекомендації та перейти до відповідного навчального модуля. Завершення навчального модуля бажано супроводжувати повторним тестуванням, оскільки порівняння попереднього й нового результату дозволяє простежити динаміку навчального прогресу.

Під час роботи з тестами, користувачеві варто уважно читати формулювання завдань, не пропускати питання без потреби та завершувати тест лише після перевірки вибраних відповідей. При виконанні відкритих завдань, потрібно вводити відповідь без зайвих символів, оскільки система виконує текстову нормалізацію, проте змістова неточність може вплинути на результат. Для завдань на відповідність необхідно встановлювати всі пари, адже часткове виконання знижує підсумкову оцінку.

Подальший розвиток системи доцільно спрямувати на розширення змістової бази тестів, збільшення кількості навчальних модулів, удосконалення аналітичного алгоритму та підвищення рівня персоналізації. Наявна реалізація вже забезпечує базовий цикл роботи, проте майбутнє розширення може посилити освітню цінність програмного продукту. Перспективним напрямом є додавання нових типів завдань, пов’язаних з аудіюванням, вимовою, побудовою речень і роботою з розгорнутими письмовими відповідями.

Підвищення точності інтелектуального оцінювання може здійснюватися через глибший аналіз поведінки користувача під час проходження тесту. Крім правильності відповіді, система може враховувати кількість змін вибору, тривалість роздумів над окремим питанням, повторюваність помилок у попередніх спробах і стабільність результату за певний період. Застосування розширених аналітичних показників, дозволить формувати більш точний профіль користувача та надавати рекомендації з урахуванням навчальної динаміки. Перспективні напрями розвитку системи LinguaLevel AI наведено в таблиці 4.8.

Таблиця 4.8 – Перспективні напрями розвитку системи LinguaLevel AI

Напрям розвитку	Зміст удосконалення	Очікуваний ефект
Розширення тестової бази	Додавання нових питань для рівнів A1, A2, B1 і B2	Підвищення точності визначення рівня знань
Додавання аудіювання	Впровадження аудіофрагментів із питаннями до змісту	Оцінювання навичок сприйняття мови на слух
Перевірка вимови	Інтеграція розпізнавання мовлення для аналізу усних відповідей	Розширення контролю комунікативних умінь
Розвиток аналітики	Урахування часу відповіді, повторюваності помилок і навчальної динаміки	Підвищення персоналізації рекомендацій
Розширення адміністративної панелі	Додавання повноцінного редактора тестів і навчальних модулів	Спрощення підтримки контенту
Перенесення на віддалений сервер	Розгортання застосунку для доступу кількох користувачів через мережу	Підготовка до використання у навчальному середовищі
Покращення звітності	Формування графіків прогресу та експорт результатів	Зручніший аналіз навчальних досягнень

Одним з найбільш важливих напрямів розвитку є поглиблення адміністративної частини. Наявна панель дозволяє працювати з тестовими питаннями, однак у перспективі доцільно додати повноцінний редактор навчальних матеріалів, можливість групування користувачів, фільтрацію результатів за періодом, перегляд детальної статистики за окремими категоріями та експорт звітів у форматах, придатних для подальшого використання викладачем. Розширення адміністративного функціоналу дозволить перетворити застосунок з демонстраційного програмного продукту, на більш повноцінний інструмент підтримки навчального процесу.

Після кількох проходжень тестів застосунок може визначати не лише поточний рівень знань, а й характер розвитку користувача, сталі помилки та теми, які потребують повторення. На основі накопиченої історії, можна

формувати послідовність уроків, де кожний наступний матеріал залежить від попереднього результату. Внаслідок такого розширення, користувач отримуватиме персоналізований маршрут навчання, а не окремі поради після завершення тесту.

Подальше покращення користувацького інтерфейсу може бути пов’язане з розширенням мобільної зручності, додаванням темного режиму, збільшенням доступності для користувачів з різними потребами та впровадженням більш наочного подання прогресу. Важливим напрямом є розробка графіків динаміки успішності, індикаторів засвоєння окремих тем і візуального порівняння результатів різних тестових спроб. Візуалізація прогресу підвищує мотивацію користувача та робить результати оцінювання зрозумілішими.

Експлуатація та розвиток LinguaLevel AI мають орієнтуватися на стабільність, безпеку, достовірність оцінювання та поступове нарощування функціональності. Розроблена система вже забезпечує завершений цикл взаємодії користувача з освітнім середовищем, починаючи від реєстрації й завершуючи отриманням рекомендацій після тестування. Запропоновані напрями вдосконалення дозволять підвищити точність аналізу знань, розширити навчальний контент, посилити адміністративні можливості та підготувати програмний продукт до використання в ширшому освітньому середовищі.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи, досягнуто поставленої мету, пов’язаної з розробкою мобільної інтелектуальної системи оцінювання рівня знань з іноземної мови, організацією навчального процесу через інтерактивні матеріали та автоматизованим аналізом результатів користувача. Створений програмний продукт, поєднав функції реєстрації, авторизації, тестування, збереження результатів, визначення рівня підготовки, формування рекомендацій і надання навчального контенту. Практичний результат роботи має прикладне значення, оскільки розроблена система може використовуватися як демонстраційний інструмент для самостійної перевірки знань, навчальної практики та подальшого розвитку цифрового освітнього середовища.

Проведене дослідження предметної області дало змогу визначити особливості цифрового оцінювання мовної підготовки, де важливу роль відіграють інтерактивність, адаптивність, персоналізація та автоматизована обробка навчальних результатів. Було встановлено, що ефективне оцінювання знань з іноземної мови не може обмежуватися простим підрахунком правильних відповідей, оскільки реальний рівень підготовки користувача залежить від лексичних, граматичних, комунікативних і аналітичних умінь.

Аналіз функціональних і нефункціональних вимог, дозволив сформувати основу для побудови програмного продукту, орієнтованого на зручність використання, стабільність роботи, зрозумілу навігацію та захист облікових даних. До складу основних функціональних можливостей увійшли реєстрація користувачів, вхід до системи, проходження тестів, отримання результатів, перегляд рекомендацій і доступ до навчальних модулів. Нефункціональні характеристики охопили продуктивність, надійність, масштабованість, сумісність із різними пристроями та мобільно-адаптивне відображення інтерфейсу.

Під час формалізації предметної області було визначено ключові сутності програмної системи, серед яких користувач, тест, питання, відповідь, результат,

навчальний модуль і рекомендація. Встановлення зв’язків між зазначеними сутностями дало змогу побудувати логічну модель даних, придатну для збереження навчальної історії, аналізу проходжень і формування персоналізованих порад.

Архітектура застосунку була побудована на основі клієнт-серверної моделі з розподілом функцій між рівнем представлення, рівнем бізнес-логіки та рівнем даних. Клієнтська частина відповідає за взаємодію з користувачем, відображення сторінок, форм і результатів тестування. Серверна частина обробляє запити, виконує перевірку облікових даних, розраховує результати, формує рекомендації та забезпечує доступ до бази SQLite.

Вибір технологічного стеку був обґрунтований потребою створити доступний, легкий в запуску та мобільно-адаптивний застосунок. Для клієнтської частини, використано HTML, CSS і JavaScript, що забезпечило побудову веб-інтерфейсу без необхідності встановлення окремого мобільного клієнта. Серверну логіку реалізовано на Node.js, що дало змогу використовувати JavaScript як єдину мову для клієнтської та серверної частин. Для збереження інформації застосовано SQLite, яка не потребує окремого серверу бази даних і підходить для демонстраційного програмного продукту дипломної роботи. Завдяки REST-взаємодії між клієнтом і сервером забезпечено структурований обмін даними у зручному форматі.

Аналіз ефективності функціонування показав, що розроблений програмний продукт відповідає запланованому призначенню. Система дозволяє не тільки пройти тест і отримати бал, а й побачити рівень підготовки, виявлені слабкі сторони, індивідуальні поради та рекомендовані навчальні матеріали. Наявність історії результатів створює передумови для подальшого відстеження прогресу, а різні типи тестових завдань підвищують достовірність оцінювання порівняно з однотипною перевіркою знань. Отриманий результат підтверджує доцільність використання інтелектуальних механізмів аналізу в освітніх мобільно-адаптивних застосунках.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кабінет Міністрів України. Про схвалення Концепції розвитку цифрових компетентностей та затвердження плану заходів з її реалізації : розпорядження від 03.03.2021 № 167-р. URL: https://zakon.rada.gov.ua/go/167-2021-%D1%80
2. European Commission. Digital Education Action Plan 2021-2027. URL: https://education.ec.europa.eu/focus-topics/digital-education/actions
3. Council of Europe. Common European Framework of Reference for Languages: Learning, Teaching, Assessment. Companion Volume. 2020. URL: https://www.coe.int/en/web/common-european-framework-reference-languages/cefr-companion-volume-and-its-language-versions
4. Зимомря І. М., Мойсюк В. А., Тріфан М. С., Унгурян І. К., Яковчук М. В. Модельна навчальна програма «Іноземна мова. 5-9 класи» для закладів загальної середньої освіти. 2021. URL: https://mon.gov.ua/static-objects/mon/sites/1/zagalna%20serednya/Navchalni.prohramy/2021/14.07/Model.navch.prohr.5-9.klas.NUSH-poetap.z.2022/Inozemni.movy.5-9-kl/Inoz.mov.5-9-kl.Zymomrya.ta.in.14.07.pdf
5. UNESCO. AI and Education: Guidance for Policy-Makers. 2021. URL: https://www.unesco.org/en/articles/ai-and-education-guidance-policy-makers
6. Цеслів О. В. Основи програмування та веб-дизайн для студентів економічних спеціальностей : навчальний посібник. Київ : КПІ ім. Ігоря Сікорського, 2020. 150 с.
7. Стець О. В. Інформатика. Частина 2. Технології програмування: комп’ютерні практикуми : навчальний посібник. Київ : КПІ ім. Ігоря Сікорського, 2022. 71 с.
8. Цибульник С. О., Барандич К. С. Технології розроблення програмного забезпечення. Частина 1. Життєвий цикл програмного забезпечення : підручник. Київ : КПІ ім. Ігоря Сікорського, 2022. 270 с.

9. Бунке О. С. Серверні веб-технології : навчальний посібник. Київ : КПІ ім. Ігоря Сікорського, 2023. 109 с.
10. Duolingo. The world’s most popular way to learn. URL: https://www.duolingo.com/
11. Babbel. Learn languages on Android or iOS with Babbel. URL: https://www.babbel.com/mobile
12. Busuu. Learn Languages Online: Start for Free. URL: https://www.busuu.com/
13. Memrise. Learn a language. Memrise is authentic, useful & personalised. URL: https://www.memrise.com/
14. Mondly. Learn Languages Online for Free. URL: https://www.mondly.com/
15. Sommerville I. Engineering Software Products: An Introduction to Modern Software Engineering. Pearson, 2021. URL: https://www.pearson.com/nl/en_NL/higher-education/subject-catalogue/computer-science/Sommerville-Engineering-Software-Products-An-Introduction-to-Modern-Software-Engineering.html
16. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner’s Approach. 9th ed. McGraw-Hill Education, 2020. URL: https://www.mheducation.com/highered/product/software-engineering-a-practitioners-approach-pressman.html
17. Koç H., Erdoğan A. M., Barjakly Y., Peker S. UML Diagrams in Software Engineering Research: A Systematic Literature Review. Proceedings. 2021. Vol. 74(1). URL: https://www.mdpi.com/2504-3900/74/1/13
18. PlantUML. Supported UML Diagrams. URL: https://plantuml.com/
19. Visual Paradigm. Use Case Diagram Tutorial. 2024. URL: https://blog.visual-paradigm.com/use-case-diagram-tutorial/
20. ISO/IEC/IEEE 42010:2022. Software, systems and enterprise — Architecture description. URL: https://www.iso.org/standard/74393.html
21. Brown S. The C4 Model for Visualising Software Architecture. URL: https://c4model.com/

22. Microsoft. Web API Design Best Practices. Azure Architecture Center. URL: https://learn.microsoft.com/en-us/azure/architecture/best-practices/api-design
23. MDN Web Docs. HTML: HyperText Markup Language. URL: https://developer.mozilla.org/en-US/docs/Web/HTML
24. MDN Web Docs. CSS: Cascading Style Sheets. URL: https://developer.mozilla.org/en-US/docs/Web/CSS
25. MDN Web Docs. JavaScript. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript
26. Node.js. SQLite. Node.js Documentation. URL: https://nodejs.org/api/sqlite.html
27. SQLite. Documentation. URL: https://sqlite.org/docs.html (дата звернення: 19.04.2026).
28. SQLite. Appropriate Uses For SQLite. URL: https://sqlite.org/whentouse.html
29. OWASP Foundation. Password Storage Cheat Sheet. URL: https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html
30. OWASP Foundation. Authentication Cheat Sheet. URL: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

ДОДАТОК А

Програмний код серверної частини мобільно-адаптивної системи

```
const http = require('node:http');
const fs = require('node:fs');
const path = require('node:path');
const { PORT, PUBLIC_DIR } = require('./config');
require('./database');
const { handleApi, fail } = require('./routes');

const mimeTypes = {
  '.html': 'text/html; charset=utf-8',
  '.css': 'text/css; charset=utf-8',
  '.js': 'application/javascript; charset=utf-8',
  '.json': 'application/json; charset=utf-8',
  '.svg': 'image/svg+xml',
  '.png': 'image/png',
  '.jpg': 'image/jpeg',
  '.jpeg': 'image/jpeg',
  '.ico': 'image/x-icon'
};

function serveStatic(req, res, pathname) {
  let safePath = decodeURIComponent(pathname);
  if (safePath === '/') safePath = '/index.html';
  const filePath = path.normalize(path.join(PUBLIC_DIR, safePath));

  if (!filePath.startsWith(PUBLIC_DIR)) {
    res.writeHead(403, { 'Content-Type': 'text/plain; charset=utf-8' });
    res.end('Forbidden');
    return;
  }

  fs.readFile(filePath, (error, content) => {
    if (error) {
      fs.readFile(path.join(PUBLIC_DIR,      'index.html'),      (fallbackError,
fallbackContent) => {
        if (fallbackError) {
          res.writeHead(404, { 'Content-Type': 'text/plain; charset=utf-8' });
          res.end('Not found');
          return;
        }
        res.writeHead(200, { 'Content-Type': 'text/html; charset=utf-8' });
        res.end(fallbackContent);
      });
      return;
    }

    const ext = path.extname(filePath).toLowerCase();
    res.writeHead(200, { 'Content-Type': mimeTypes[ext] || 'application/octet-
stream' });
    res.end(content);
  });
}

const server = http.createServer(async (req, res) => {
  const url = new URL(req.url, `http://${req.headers.host || 'localhost'}`);
```

```
  try {
    if (url.pathname.startsWith('/api/')) {
      await handleApi(req, res, url.pathname);
      return;
    }
    serveStatic(req, res, url.pathname);
  } catch (error) {
    console.error(error);
    fail(res, error.message || 'Внутрішня помилка сервера.', 500);
  }
});

server.listen(PORT, () => {
  console.log(`LinguaLevel запущено: http://localhost:${PORT}`);
  console.log('Демо користувач: student@linguaskill.local / Student12345!');
  console.log('Адміністратор: admin@linguaskill.local / Admin12345!');
});
```

ДОДАТОК Б

Програмний код клієнтської логіки застосунку

```
const app = (() => {
  const root = document.getElementById('app');
  const toastEl = document.getElementById('toast');
  const mainNav = document.getElementById('mainNav');
  const adminNav = document.getElementById('adminNav');

  const state = {
    token: localStorage.getItem('lingua_token') || '',
    user: null,
    authMode: 'login',
    currentTest: null,
    testIndex: 0,
    answers: {},
    startedAt: 0
  };

  const categoryNames = {
    vocabulary: 'Лексика',
    grammar: 'Граматика',
    reading: 'Читання',
    translation: 'Переклад',
    matching: 'Відповідність',
    general: 'Загальні навички'
  };

  function escapeHtml(value) {
    return String(value ?? '')
      .replaceAll('&', '&amp;')
      .replaceAll('<', '&lt;')
      .replaceAll('>', '&gt;')
      .replaceAll('"', '&quot;')
      .replaceAll("'", '&#039;');
  }

  function toast(message) {
    toastEl.textContent = message;
    toastEl.classList.remove('hidden');
    clearTimeout(toastEl._timer);
    toastEl._timer = setTimeout(() => toastEl.classList.add('hidden'), 3600);
  }

  async function api(path, options = {}) {
    const headers = { 'Content-Type': 'application/json', ...(options.headers ||
{}) };
    if (state.token) headers.Authorization = `Bearer ${state.token}`;

    const response = await fetch(path, {
      ...options,
      headers,
      body: options.body ? JSON.stringify(options.body) : undefined
    });

    const data = await response.json().catch(() => ({}));
    if (!response.ok || !data.ok) {
      throw new Error(data.message || 'Помилка запиту до сервера.');
```

```
    }
    return data;
  }

  function setAuth(token, user) {
    state.token = token || '';
    state.user = user || null;
    if (token) localStorage.setItem('lingua_token', token);
    else localStorage.removeItem('lingua_token');
    mainNav.classList.toggle('hidden', !state.user);
    adminNav.classList.toggle('hidden', !state.user || state.user.role !==
'admin');
  }

  async function init() {
    if (!state.token) {
      renderAuth();
      return;
    }

    try {
      const data = await api('/api/me');
      setAuth(state.token, data.user);
      renderDashboard();
    } catch {
      setAuth('', null);
      renderAuth();
    }
  }

  function renderAuth() {
    mainNav.classList.add('hidden');
    root.innerHTML = `
      <section class="hero auth-only">
        <div class="panel auth-panel">
          <div class="tabs">
            <button class="${state.authMode === 'login' ? 'active' : ''}"
onclick="app.setAuthMode('login')">Вхід</button>
            <button class="${state.authMode === 'register' ? 'active' : ''}"
onclick="app.setAuthMode('register')">Реєстрація</button>
          </div>
          ${state.authMode === 'login' ? loginForm() : registerForm()}
        </div>
      </section>
    `;
  }

  function loginForm() {
    return `
      <form class="form-grid" onsubmit="app.login(event)">
        <label>Електронна адреса<input name="email" type="email"
placeholder="user@example.com" required></label>
        <label>Пароль<input name="password" type="password" placeholder="Введіть
пароль" required></label>
        <button type="submit">Увійти</button>
      </form>
    `;
  }

  function registerForm() {
    return `
      <form class="form-grid" onsubmit="app.register(event)">
        <label>Ім'я користувача<input name="name" type="text"
placeholder="Наприклад, Андрій" required></label>
```

```
        <label>Електронна адреса<input name="email" type="email"
placeholder="user@example.com" required></label>
        <label>Пароль<input name="password" type="password" placeholder="Не менше
8 символів" required></label>
        <button type="submit">Зареєструватися</button>
      </form>
    `;
  }

  function setAuthMode(mode) {
    state.authMode = mode;
    renderAuth();
  }

  async function login(event) {
    event.preventDefault();
    const form = new FormData(event.target);
    try {
      const data = await api('/api/auth/login', {
        method: 'POST',
        body: { email: form.get('email'), password: form.get('password') }
      });
      setAuth(data.token, data.user);
      toast('Вхід виконано успішно.');
      renderDashboard();
    } catch (error) {
      toast(error.message);
    }
  }

  async function register(event) {
    event.preventDefault();
    const form = new FormData(event.target);
    try {
      const data = await api('/api/auth/register', {
        method: 'POST',
        body: { name: form.get('name'), email: form.get('email'), password:
form.get('password') }
      });
      setAuth(data.token, data.user);
      toast('Обліковий запис створено.');
      renderDashboard();
    } catch (error) {
      toast(error.message);
    }
  }

  async function logout() {
    try { await api('/api/auth/logout', { method: 'POST' }); } catch {}
    setAuth('', null);
    renderAuth();
  }

  function show(view) {
    if (!state.user) return renderAuth();
    if (view === 'dashboard') return renderDashboard();
    if (view === 'tests') return renderTests();
    if (view === 'lessons') return renderLessons();
    if (view === 'results') return renderResults();
    if (view === 'online') return renderOnlineTraining();
    if (view === 'admin') return renderAdmin();
  }

  async function renderDashboard() {
```

```
    try {
      const data = await api('/api/dashboard');
      state.user = data.user;
      const recs = data.recommendations.length
        ?     data.recommendations.map((item)     =>     `<div     class="list-
item">${escapeHtml(item.text)}</div>`).join('')
        : `<div class="empty">Рекомендації з'являться після проходження першого
тестування.</div>`;
      const recent = data.recentResults.length
        ? data.recentResults.map(resultCard).join('')
        : `<div class="empty">Історія результатів поки порожня.</div>`;

      root.innerHTML = `
        <section class="section-head">
          <div>
            <span class="kicker">Персональне середовище</span>
            <h2>Вітаємо, ${escapeHtml(data.user.name)}</h2>
            <p>Поточний          рівень:          <span          class="badge
${levelClass(data.user.knowledgeLevel)}">${escapeHtml(data.user.knowledgeLevel)}
</span></p>
          </div>
          <div class="row-actions">
            <button onclick="app.show('tests')">Почати тестування</button>
            <button  class="secondary"  onclick="app.show('lessons')">Перейти  до
курсів</button>
          </div>
        </section>

        <div class="stat-grid">
          <div     class="stat"><b>${data.stats.averageScore}%</b><span>Середній
результат</span></div>
          <div
class="stat"><b>${data.stats.completedLessons}/${data.stats.totalLessons}</b><sp
an>Пройдено модулів</span></div>
          <div        class="stat"><b>${data.user.progress}%</b><span>Навчальний
прогрес</span></div>
          <div          class="stat"><b>${data.stats.attempts}</b><span>Останніх
спроб</span></div>
        </div>

        <div          class="progress-wrap"><div          class="progress-bar"
style="width:${Number(data.user.progress || 0)}%"></div></div>

        <section class="grid">
          <article class="card clickable" onclick="app.show('tests')">
            <div class="card-icon">T</div><h3>Тестування</h3>
            <p>Виконання завдань різних типів, фіксація відповідей і визначення
рівня знань.</p>
          </article>
          <article class="card clickable" onclick="app.show('lessons')">
            <div class="card-icon">K</div><h3>Готові курси</h3>
            <p>Навчальні модулі з теорією, прикладами, практикою та підсумком для
закріплення матеріалу.</p>
          </article>
          <article class="card clickable" onclick="app.show('online')">
            <div class="card-icon">O</div><h3>Online-навчання</h3>
            <p>Демонстраційна сторінка із рекомендаціями щодо самостійної роботи
після тестування.</p>
          </article>
        </section>

        <section class="grid two" style="margin-top:1rem">
          <div          class="panel"><h3>Останні          рекомендації</h3><div
class="list">${recs}</div></div>
```

```
          <div           class="panel"><h3>Останні           результати</h3><div
class="list">${recent}</div></div>
        </section>
      `;
    } catch (error) {
      toast(error.message);
    }
  }

  function levelClass(level) {
    if (level === 'Високий') return 'success';
    if (level === 'Середній') return 'warning';
    if (level === 'Початковий') return 'danger';
    return '';
  }

  function resultCard(result) {
    return `
      <div class="list-item">
        <div class="row-actions" style="justify-content:space-between">
          <b>${escapeHtml(result.testTitle)}</b>
          <span                                                    class="badge
${levelClass(result.knowledgeLevel)}">${result.weightedScore}%                  •
${escapeHtml(result.knowledgeLevel)}</span>
        </div>
        <span   class="muted">${new   Date(result.createdAt).toLocaleString('uk-
UA')}     •     ${result.correctCount}/${result.totalCount}     правильних     •
${result.timeSeconds} с</span>
      </div>
    `;
  }

  async function renderTests() {
    try {
      const data = await api('/api/tests');
      root.innerHTML = `
        <section class="section-head">
          <div>
            <span class="kicker">Підсистема тестування</span>
            <h2>Визначення рівня знань</h2>
            <p>Оберіть тест, виконайте завдання та отримайте результат із
рекомендаціями.</p>
          </div>
        </section>
        <div class="grid">
          ${data.tests.map((test) => `
            <article class="card clickable" onclick="app.startTest(${test.id})">
              <div class="card-icon">${test.questionCount}</div>
              <h3>${escapeHtml(test.title)}</h3>
              <p>${escapeHtml(test.description)}</p>
              <span        class="badge">${escapeHtml(test.difficulty)}        •
${test.questionCount} питань</span>
            </article>
          `).join('')}
        </div>
      `;
    } catch (error) {
      toast(error.message);
    }
  }

  async function startTest(testId) {
    try {
      const data = await api(`/api/tests/${testId}`);
```

```
      state.currentTest = data;
      state.testIndex = 0;
      state.answers = {};
      state.startedAt = Date.now();
      renderTestQuestion();
    } catch (error) {
      toast(error.message);
    }
  }

  function renderTestQuestion() {
    const test = state.currentTest;
    const question = test.questions[state.testIndex];
    const progress = Math.round(((state.testIndex + 1) / test.questions.length) *
100);
    root.innerHTML = `
      <section class="section-head">
        <div>
          <span class="kicker">${escapeHtml(test.test.title)}</span>
          <h2>Питання ${state.testIndex + 1} з ${test.questions.length}</h2>
          <p>${escapeHtml(test.test.description)}</p>
        </div>
        <button class="ghost" onclick="app.show('tests')">Скасувати</button>
      </section>
      <div            class="progress-wrap"><div            class="progress-bar"
style="width:${progress}%"></div></div>
      <div class="question-box">
        <div class="row-actions" style="justify-content:space-between">
          <span        class="badge">${categoryNames[question.category]        ||
question.category}</span>
          <span    class="badge">${escapeHtml(question.difficulty)}    •    вага
${question.weight}</span>
        </div>
        <h3 style="margin-top:1rem">${escapeHtml(question.text)}</h3>
        ${questionInput(question)}
        <div class="row-actions" style="margin-top:1.2rem; justify-content:space-
between">
          <button         class="secondary"         onclick="app.prevQuestion()"
${state.testIndex === 0 ? 'disabled' : ''}>Назад</button>
          ${state.testIndex + 1 === test.questions.length
            ? `<button onclick="app.submitTest()">Завершити тест</button>`
            : `<button onclick="app.nextQuestion()">Далі</button>`}
        </div>
      </div>
    `;
  }

  function questionInput(question) {
    const current = state.answers[String(question.id)];
    if (question.type === 'single') {
      return `<div class="options">${question.answers.map((answer) => `
        <label class="option">
          <input   type="radio"   name="q${question.id}"   ${Number(current)   ===
Number(answer.id)  ?  'checked'  :  ''}  onchange="app.setAnswer('${question.id}',
${answer.id})">
          <span>${escapeHtml(answer.text)}</span>
        </label>`).join('')}</div>`;
    }

    if (question.type === 'multiple') {
      const selected = Array.isArray(current) ? current.map(Number) : [];
      return `<div class="options">${question.answers.map((answer) => `
        <label class="option">
```

```
          <input    type="checkbox"    ${selected.includes(Number(answer.id))    ?
'checked' : ''} onchange="app.toggleMultiple('${question.id}', ${answer.id})">
          <span>${escapeHtml(answer.text)}</span>
        </label>`).join('')}</div>`;
    }

    if (question.type === 'open' || question.type === 'translation') {
      return `<div class="options"><input value="${escapeHtml(current || '')}"
placeholder="Введіть відповідь" oninput="app.setAnswer('${question.id}',
this.value)"></div>`;
    }

    if (question.type === 'matching') {
      const pairs = question.metadata?.pairs || [];
      const rights = [...new Set(pairs.map((pair) => pair.right))].sort(() => 0.5
- Math.random());
      const objectValue = current && typeof current === 'object' ? current : {};
      return `<div class="options">${pairs.map((pair) => `
        <div class="match-row">
          <b>${escapeHtml(pair.left)}</b>
          <select                    onchange="app.setMatching('${question.id}',
'${escapeAttr(pair.left)}', this.value)">
            <option value="">Оберіть відповідність</option>
            ${rights.map((right)    =>    `<option    value="${escapeAttr(right)}"
${objectValue[pair.left]       ===       right       ?       'selected'       :
''}>${escapeHtml(right)}</option>`).join('')}
          </select>
        </div>`).join('')}</div>`;
    }

    return `<div class="empty">Тип питання не підтримується інтерфейсом.</div>`;
  }

  function escapeAttr(value) {
    return    String(value    ??    '').replaceAll('&',    '&amp;').replaceAll('"',
'&quot;').replaceAll("'",    '&#039;').replaceAll('<',    '&lt;').replaceAll('>',
'&gt;');
  }

  function setAnswer(questionId, value) {
    state.answers[String(questionId)] = value;
  }

  function toggleMultiple(questionId, answerId) {
    const key = String(questionId);
    const        arr        =        Array.isArray(state.answers[key])        ?
[...state.answers[key]].map(Number) : [];
    const index = arr.indexOf(Number(answerId));
    if (index >= 0) arr.splice(index, 1);
    else arr.push(Number(answerId));
    state.answers[key] = arr;
  }

  function setMatching(questionId, left, right) {
    const key = String(questionId);
    const current = state.answers[key] && typeof state.answers[key] === 'object'
? state.answers[key] : {};
    state.answers[key] = { ...current, [left]: right };
  }

  function nextQuestion() {
    if (state.currentTest && state.testIndex < state.currentTest.questions.length
- 1) {
      state.testIndex += 1;
```

```
      renderTestQuestion();
    }
  }

  function prevQuestion() {
    if (state.currentTest && state.testIndex > 0) {
      state.testIndex -= 1;
      renderTestQuestion();
    }
  }

  async function submitTest() {
    try {
      const unanswered = state.currentTest.questions.filter((q) => !state.answers[String(q.id)] || state.answers[String(q.id)] === '').length;
      if (unanswered && !confirm(`Залишилось питань без відповіді: ${unanswered}. Завершити тест?`)) return;

      const data = await api(`/api/tests/${state.currentTest.test.id}/submit`, {
        method: 'POST',
        body: { answers: state.answers, startedAt: state.startedAt }
      });
      renderTestResult(data.result);
    } catch (error) {
      toast(error.message);
    }
  }

  function renderTestResult(result) {
    const weak = result.weakCategories.length
      ? result.weakCategories.map((item) => `<span class="badge">${categoryNames[item.category] || item.category}: ${item.count}</span>`).join(' ')
      : '<span class="badge success">Типових помилок не виявлено</span>';

    root.innerHTML = `
      <section class="panel">
        <span class="kicker">Результат інтелектуального оцінювання</span>
        <div class="grid two" style="align-items:center; margin-top:1rem">
          <div>
            <div class="result-score">${result.weightedScore}%</div>
            <h2>${escapeHtml(result.knowledgeLevel)} рівень</h2>
            <p>Правильних відповідей: ${result.correctCount}/${result.totalCount}. Час виконання: ${result.timeSeconds} с. Підсумкова оцінка враховує складність завдань.</p>
            <div>${weak}</div>
          </div>
          <div class="list">
            ${result.recommendations.map((text) => `<div class="list-item">${escapeHtml(text)}</div>`).join('')}
          </div>
        </div>
        <div class="row-actions" style="margin-top:1.3rem">
          <button onclick="app.show('lessons')">Перейти до навчальних модулів</button>
          <button class="secondary" onclick="app.show('results')">Переглянути історію</button>
        </div>
      </section>
    `;
  }

  async function renderLessons() {
```

```
    try {
      const data = await api('/api/lessons');
      const levels = ['Початковий', 'Середній', 'Високий'];
      root.innerHTML = `
        <section class="section-head">
          <div>
            <span class="kicker">Навчальний контент</span>
            <h2>Готові курси та демонстраційні уроки</h2>
            <p>Модулі містять теоретичний блок, приклади, практичні завдання, контрольні питання та підсумок.</p>
          </div>
          <button onclick="app.renderRecommendedLessons()">Показати рекомендовані</button>
        </section>
        ${levels.map((level) => {
          const group = data.lessons.filter((lesson) => lesson.level === level);
          if (!group.length) return '';
          return `<h3 style="margin:1.1rem 0 0.7rem">${level} рівень</h3><div class="grid">${group.map(lessonCard).join('')}</div>`;
        }).join('')}
      `;
    } catch (error) {
      toast(error.message);
    }
  }

  async function renderRecommendedLessons() {
    try {
      const data = await api('/api/lessons?recommended=1');
      root.innerHTML = `
        <section class="section-head">
          <div>
            <span class="kicker">Адаптивний підбір</span>
            <h2>Рекомендовані навчальні модулі</h2>
            <p>Матеріали підібрано відповідно до поточного рівня користувача після останнього тестування.</p>
          </div>
          <button class="secondary" onclick="app.show('lessons')">Усі курси</button>
        </section>
        <div class="grid">${data.lessons.length ? data.lessons.map(lessonCard).join('') : '<div class="empty">Спочатку пройдіть тестування для визначення рівня.</div>'}</div>
      `;
    } catch (error) {
      toast(error.message);
    }
  }

  function lessonCard(lesson) {
    return `
      <article class="card clickable" onclick="app.openLesson(${lesson.id})">
        <div class="card-icon">${lesson.completed ? '✓' : 'L'}</div>
        <h3>${escapeHtml(lesson.title)}</h3>
        <p>${escapeHtml(lesson.description)}</p>
        <div class="row-actions">
          <span class="badge ${levelClass(lesson.level)}">${escapeHtml(lesson.level)}</span>
          <span class="badge">${categoryNames[lesson.category] || lesson.category}</span>
        </div>
      </article>
    `;
  }
```

```
  async function openLesson(lessonId) {
    try {
      const data = await api(`/api/lessons/${lessonId}`);
      const lesson = data.lesson;
      root.innerHTML = `
        <section class="section-head">
          <div>
            <span     class="kicker">${escapeHtml(lesson.level)}     рівень     •
${categoryNames[lesson.category] || lesson.category}</span>
            <h2>${escapeHtml(lesson.title)}</h2>
            <p>${escapeHtml(lesson.description)}</p>
          </div>
          <button class="ghost" onclick="app.show('lessons')">Назад</button>
        </section>
        <section class="grid two">
          <article                                    class="panel"><h3>Теоретичний
блок</h3><p>${escapeHtml(lesson.theory)}</p></article>
          <article
class="panel"><h3>Приклади</h3><p>${escapeHtml(lesson.examples)}</p></article>
          <article                                      class="panel"><h3>Практичні
завдання</h3><p>${escapeHtml(lesson.practice)}</p></article>
          <article
class="panel"><h3>Підсумок</h3><p>${escapeHtml(lesson.summary)}</p></article>
        </section>
        <div class="row-actions" style="margin-top:1rem">
          <button onclick="app.completeLesson(${lesson.id})" ${lesson.completed ?
'disabled'  :  ''}>${lesson.completed  ?  'Модуль  уже  виконано'  :  'Позначити  як
виконаний'}</button>
          <button class="secondary" onclick="app.show('tests')">Перевірити знання
тестом</button>
        </div>
      `;
    } catch (error) {
      toast(error.message);
    }
  }

  async function completeLesson(lessonId) {
    try {
      await api(`/api/lessons/${lessonId}/complete`, { method: 'POST' });
      toast('Навчальний модуль позначено як виконаний.');
      openLesson(lessonId);
    } catch (error) {
      toast(error.message);
    }
  }

  async function renderResults() {
    try {
      const data = await api('/api/results');
      root.innerHTML = `
        <section class="section-head">
          <div>
            <span class="kicker">Аналітика результатів</span>
            <h2>Історія тестування</h2>
            <p>Збережені  результати  використовуються  для  аналізу  динаміки  та
формування рекомендацій.</p>
          </div>
        </section>
        ${data.results.length ? `
          <div class="table-wrap">
            <table>
```

```
<thead><tr><th>Дата</th><th>Тест</th><th>Результат</th><th>Рівень</th><th>Помилки</th></tr></thead>
              <tbody>
                ${data.results.map((result) => `
                  <tr>
                    <td>${new Date(result.createdAt).toLocaleString('uk-UA')}</td>
                    <td>${escapeHtml(result.testTitle)}</td>
                    <td><b>${result.weightedScore}%</b><br><span class="muted">${result.correctCount}/${result.totalCount}, ${result.timeSeconds} с</span></td>
                    <td><span class="badge ${levelClass(result.knowledgeLevel)}">${escapeHtml(result.knowledgeLevel)}</span></td>
                    <td>${result.weakCategories.length ? result.weakCategories.map((item) => `${categoryNames[item.category] || item.category}: ${item.count}`).join('<br>') : '—'}</td>
                  </tr>`).join('')}
              </tbody>
            </table>
          </div>` : '<div class="empty">Поки немає збережених результатів.</div>'}
      `;
    } catch (error) {
      toast(error.message);
    }
  }

  function renderOnlineTraining() {
    root.innerHTML = `
      <section class="section-head">
        <div>
          <span class="kicker">Online-навчання</span>
          <h2>Демонстраційний режим самостійної роботи</h2>
          <p>Сторінка показує, як користувач може продовжити навчання після отримання результату тестування.</p>
        </div>
      </section>
      <div class="grid">
        <article class="card"><div class="card-icon">1</div><h3>Коротке заняття</h3><p>Виберіть рекомендований навчальний модуль і повторіть теорію протягом 10–15 хвилин.</p></article>
        <article class="card"><div class="card-icon">2</div><h3>Практична вправа</h3><p>Виконайте завдання з модуля, звертаючи увагу на категорії, у яких були помилки.</p></article>
        <article class="card"><div class="card-icon">3</div><h3>Повторне тестування</h3><p>Після проходження матеріалу поверніться до тесту та перевірте зміну результату.</p></article>
      </div>
    `;
  }

  async function renderAdmin() {
    try {
      const data = await api('/api/admin/stats');
      root.innerHTML = `
        <section class="section-head">
          <div>
            <span class="kicker">Адміністративний модуль</span>
            <h2>Статистика та керування тестовими завданнями</h2>
            <p>Адміністратор може переглядати загальну статистику та додавати питання до тестів.</p>
          </div>
```

```
        </section>
        <div class="stat-grid">
          <div
class="stat"><b>${data.stats.users}</b><span>Користувачів</span></div>
          <div          class="stat"><b>${data.stats.resultsCount}</b><span>Спроб
тестування</span></div>
          <div     class="stat"><b>${data.stats.averageScore}%</b><span>Середній
бал</span></div>
          <div class="stat"><b>${data.tests.length}</b><span>Тестів</span></div>
        </div>
        <section class="grid two">
          <div class="panel">
            <h3>Додавання питання</h3>
            <form class="form-grid" onsubmit="app.adminAddQuestion(event)">
              <label>Тест<select name="testId">${data.tests.map((t) => `<option
value="${t.id}">${escapeHtml(t.title)}</option>`).join('')}</select></label>
              <label>Тип<select     name="type"><option     value="single">Вибір
правильної            відповіді</option><option            value="open">Відкрите
питання</option><option value="translation">Переклад</option></select></label>
              <label>Категорія<select                    name="category"><option
value="vocabulary">Лексика</option><option
value="grammar">Граматика</option><option
value="reading">Читання</option><option
value="translation">Переклад</option></select></label>
              <label>Складність<select                  name="difficulty"><option
value="easy">easy</option><option          value="medium">medium</option><option
value="hard">hard</option></select></label>
              <label>Текст  питання<textarea  name="text"  placeholder="Введіть
умову питання"></textarea></label>
              <label>Правильна   відповідь   для   відкритого   питання   або
перекладу<input name="correctText" placeholder="Можна вказати декілька варіантів
через |"></label>
              <label>Варіанти  для  single  у  форматі:  текст*  для  правильної
відповіді<textarea                                          name="answers"
placeholder="book*\ncity\nwindow"></textarea></label>
              <button type="submit">Додати питання</button>
            </form>
          </div>
          <div class="panel">
            <h3>Останні проходження</h3>
            <div class="list">
              ${data.latest.length ? data.latest.map((item) => `
                <div class="list-item">
                  <b>${escapeHtml(item.userName)}</b><br>
                  <span      class="muted">${escapeHtml(item.testTitle)}      •
${item.weightedScore}% • ${escapeHtml(item.knowledgeLevel)}</span>
                </div>`).join('')       :       '<div       class="empty">Немає
результатів.</div>'}
            </div>
          </div>
        </section>
      `;
    } catch (error) {
      toast(error.message);
    }
  }

  async function adminAddQuestion(event) {
    event.preventDefault();
    const form = new FormData(event.target);
    const type = form.get('type');
    const rawAnswers = String(form.get('answers') || '').split('\n').map((line)
=> line.trim()).filter(Boolean);
    const answers = rawAnswers.map((line) => ({
```

```
      text: line.replace(/\*$/, '').trim(),
      correct: line.endsWith('*')
    }));

    try {
      await api('/api/admin/questions', {
        method: 'POST',
        body: {
          testId: Number(form.get('testId')),
          type,
          category: form.get('category'),
          difficulty: form.get('difficulty'),
          weight: form.get('difficulty') === 'hard' ? 1.8 : form.get('difficulty')
=== 'medium' ? 1.4 : 1,
          text: form.get('text'),
          correctText: form.get('correctText'),
          answers: type === 'single' ? answers : []
        }
      });
      toast('Питання додано.');
      renderAdmin();
    } catch (error) {
      toast(error.message);
    }
  }

  return {
    init,
    show,
    setAuthMode,
    login,
    register,
    logout,
    startTest,
    setAnswer,
    toggleMultiple,
    setMatching,
    nextQuestion,
    prevQuestion,
    submitTest,
    openLesson,
    completeLesson,
    renderRecommendedLessons,
    adminAddQuestion
  };
})();

window.app = app;
app.init();
```

ДОДАТОК В

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

Декларація про академічну доброчесність та використання ШІ

Я, Мельников Артем Валерійович, здобувач НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

1. Моя бакалаврська кваліфікаційна робота на тему «Мобільна інтелектуальна система оцінки рівня знань з іноземної мови» є результатом моєї особистої праці.
2. Усі запозичення з текстів інших авторів мають відповідні посилання згідно з чинними стандартами.
3. Щодо використання інструментів ШІ зазначаю, що (обрати необхідне):

- [ ] інструменти генеративного ШІ не використовувалися.
- [ ] інструменти ШІ (вказати назву: ChatGPT, Gemini, Claude, DeepL, _____інші (вказати назву)) були використані для:
  - [ ] перекладу іншомовних джерел;
  - [ ] стилістичного редагування та перевірки граматики;
  - [ ] допомоги у написанні/відлагодженні програмного коду;
  - [ ] інше: _____.

Я розумію, що надання недостовірної інформації може призвести до скасування результатів захисту та відрахування з числа здобувачів НУБіП України.

« » 2026 р. _____ **Артем МЕЛЬНИКОВ**
(підпис)