Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розробка мобільного застосунку для вивчення слів
на основі генеративного штучного інтелекту
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-21-1
Тетяна Грищенко
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект
(повна назва освітньої програми)
Керівник доц. Анастасія Дейнеко
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Грищенко Тетяні Борисівні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка мобільного застосунку для вивчення слів на основі генеративного штучного інтелекту

затверджена наказом університету від 19 травня 20 25 р. № 378Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20 червня 20 25 р.
3. Вихідні дані до роботи офіційні документації до React, Mantine UI, TypeScript, Nest.js, Firebase, OpenAI API, Auth0 та Google Cloud Platform.

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Проектування системи
3) Програмна реалізація

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	19.05.2025	виконано
2	Визначення проблеми	19.05.2025	виконано
3	Аналіз існуючих аналогів	20.05.2025	виконано
4	Розробка компонентів застосунку	21.05.2025	виконано
5	Реалізація застосунку	22.05.2025	виконано
6	Оформлення пояснювальної записки	01.06.2025	виконано
7	Захист	20.06.2025	виконано

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Анастасія Дейнеко
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 91 с., 25 рис., 1 дод., 12 джерел.

ГЕНЕРАЦІЯ СЛОВНИКОВИХ НАБОРІВ, МОБІЛЬНИЙ ЗАСТОСУНОК, ПЕРСОНАЛІЗОВАНЕ НАВЧАННЯ, ПРОГРЕС НАВЧАННЯ, СЛОВНИКОВИЙ ЗАПАС, GPT-4, LANGUAGE MODEL, OPENAI API.

Об’єктом дослідження є процес вивчення іноземної лексики за допомогою цифрових технологій у мобільному середовищі.

Предметом дослідження є методи персоналізованого навчання лексики з використанням штучного інтелекту в мобільних застосунках.

Метою роботи є розробка мобільного застосунку для вивчення слів, який генерує словникові набори, дозволяє практикувати слова у контексті та адаптує навчання до рівня користувача за допомогою OpenAI API.

У дослідженні застосовано методи порівняльного аналізу існуючих рішень, функціонального моделювання системи, генерації навчального контенту з використанням великої мовної моделі та прототипування інтерфейсів користувача.

У результаті виконаної роботи створено мобільний застосунок із функціями генерації словникових наборів, практичного режиму з перевіркою речень, ведення прогресу користувача та адаптації контенту за допомогою ШІ, що дозволяє підвищити ефективність лексичного навчання.

Результати роботи можуть бути використані в індивідуальному мовному навчанні, у супровід шкільних або онлайн-курсів, а також у проектах, пов’язаних із цифровою освітою. Застосунок має перспективу впровадження в EdTech-індустрії, мовних школах і мобільних платформах самоосвіти.

ABSTRACT

Bachelor’s thesis contains: 91 pp., 25 fig., 1 ann., 12 references.

GENERATION OF VOCABULARY SETS, GPT-4, LANGUAGE MODEL, LEARNING PROGRESS, MOBILE APPLICATION, OPENAI API, PERSONALIZED LEARNING, VOCABULARY.

The object of the research is the process of learning foreign vocabulary through digital technologies in a mobile environment.

The subject of the research is methods of personalized vocabulary learning using artificial intelligence in mobile applications.

The aim of the work is to develop a mobile application for learning words that generates vocabulary sets, enables contextual word practice, and adapts learning to the user’s level using the OpenAI API.

The study applies methods of comparative analysis of existing solutions, functional system modeling, educational content generation using a large language model, and user interface prototyping.

As a result of the work, a mobile application was developed with features for generating vocabulary sets, a practical mode with sentence validation, user progress tracking, and AI-based content adaptation, which improves the effectiveness of vocabulary learning.

The outcomes of the project can be used in individual language learning, as a supplement to school or online courses, and in initiatives related to digital education. The application has potential for adoption in the EdTech industry, language schools, and self-education mobile platforms.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект;

API – Application Programming Interface – прикладний програмний інтерфейс;

GPT – Generative Pre-trained Transformer – генеративний попередньо натренований трансформер;

HTTP – Hypertext Transfer Protocol – протокол передачі гіпертексту;

IT – Information Technology – інформаційні технології;

JWT – JSON Web Token – вебтокен у форматі JSON;

MVP – Minimum Viable Product – мінімально життєздатний продукт;

SPA – Single Page Application – односторінковий застосунок;

SQL – Structured Query Language – мова структурованих запитів;

UI – User Interface – інтерфейс користувача;

UX – User Experience – користувацький досвід.

ВСТУП

У сучасних умовах глобалізації та цифровізації знання іноземних мов набуває особливої ваги. Англійська мова виконує роль універсального інструменту спілкування в освіті, науці, бізнесі, технологіях та повсякденному житті. Вивчення лексики є основою для формування мовної компетентності, адже саме словниковий запас дозволяє будувати фрази, розуміти тексти, брати участь у діалогах. Однак традиційні методи засвоєння лексики, засновані на зазубрюванні списків слів і перекладів, демонструють обмежену ефективність. Вони не враховують індивідуальні особливості учня, не забезпечують контекстуального використання слів та швидко призводять до втрати мотивації.

У відповідь на ці виклики зростає популярність цифрових рішень, зокрема мобільних застосунків для вивчення слів. Вони забезпечують гнучкість навчального процесу, можливість навчатися у зручний час, використовують мультимедійний контент і елементи гейміфікації. Разом із цим більшість таких застосунків залишаються обмеженими у функціональності, орієнтуючись на однакові шаблони вивчення лексики для всіх користувачів. Вони не надають персоналізованих рекомендацій, не дозволяють ефективно практикувати використання нових слів у власних висловлюваннях і, як правило, не використовують потенціал сучасних технологій штучного інтелекту.

Одним із найбільш перспективних напрямів у сфері цифрової освіти є використання генеративного штучного інтелекту. На основі великих мовних моделей стало можливим створювати персоналізований навчальний контент у режимі реального часу. Генеративний ШІ здатен формувати словникові набори за заданими критеріями, пояснювати значення слів у контексті, перевіряти правильність створених речень, а також давати зворотний зв’язок, який наближений до коментарів реального викладача. Це

відкриває нові горизонти для побудови системи лексичного навчання, яка є не лише адаптивною, а й по-справжньому інтерактивною.

Актуальність даної роботи зумовлена необхідністю створення інноваційного мобільного інструменту для вивчення слів, який поєднує переваги цифрових технологій, гнучкість мобільних платформ та інтелектуальні можливості генеративного ШІ. Такий підхід дозволяє подолати обмеження традиційних методів і запропонувати користувачу новий рівень якості мовного навчання. Особливо важливо це для самостійного навчання, коли відсутній постійний контакт із викладачем, і саме система має виконувати роль наставника, асистента та критика.

Метою роботи є розробка мобільного застосунку, який забезпечує інтелектуальне вивчення лексики через автоматичну генерацію словникових наборів, контекстне пояснення значень слів та практичні завдання з перевіркою речень. Реалізація передбачає створення адаптивного інтерфейсу, системи обліку навчального прогресу, режимів навчання і практики, а також інтеграцію з OpenAI API для реалізації ШІ-функціоналу. Застосунок має бути зручним для користувача, ефективним у плані засвоєння лексики і технічно стійким до масштабування.

Результати роботи можуть бути застосовані у сфері індивідуального мовного навчання, у супровід мовних курсів, а також як допоміжний інструмент у загальноосвітніх закладах. Крім того, підхід, розроблений у межах цієї роботи, може бути адаптований для інших мов або розширений до граматичних конструкцій. Такий продукт також має потенціал для комерційного впровадження серед освітніх стартапів, мобільних платформ для самонавчання і EdTech-компаній, які прагнуть впроваджувати штучний інтелект у процеси навчання.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Актуальні тенденції цифрової лексичної освіти

Сучасна цифрова епоха суттєво змінила підходи до вивчення мов, зокрема у сфері засвоєння лексики. Раніше основною формою роботи з новими словами були друковані словники, підручники та списки лексики, що супроводжувалися ручним запам’ятовуванням. Однак з розвитком інформаційних технологій акценти змістилися у бік інтерактивності, доступності та персоналізації навчального процесу. Застосунки для вивчення слів стали важливим інструментом у формуванні мовної компетентності, що дає змогу враховувати індивідуальні особливості кожного користувача.

У фокусі цифрової лексичної освіти нині перебуває створення середовищ [2], які забезпечують активну участь користувача у навчальному процесі. Застосунки дедалі частіше використовують адаптивні алгоритми, що дозволяють будувати персоналізовані маршрути вивчення лексики відповідно до рівня підготовки, інтересів і попереднього досвіду користувача. Це сприяє ефективнішому запам’ятовуванню слів завдяки повторенню у потрібний момент, гейміфікації та контекстуалізації нового матеріалу.

Іншим ключовим напрямом розвитку є залучення інтелектуальних систем до підтримки користувача під час навчання. Генеративні моделі штучного інтелекту, зокрема великі мовні моделі, відкривають нові можливості в автоматичному створенні навчального контенту, перевірці правильності виконаних завдань і формуванні індивідуальних рекомендацій. Це дозволяє не тільки автоматизувати частину процесів, а й підвищити якість лексичного матеріалу завдяки контекстно релевантним прикладам і поясненням.

У межах загальних тенденцій цифрової трансформації освіти, лексичне навчання також отримує новий імпульс завдяки мобільності, хмарним обчисленням і доступу до потужних API. Це забезпечує безперервність навчання, коли користувач має змогу навчатися у зручному для себе форматі і в будь-який час. Таким чином, цифрова лексична освіта продовжує динамічно розвиватися, об’єднуючи класичні освітні підходи з можливостями сучасних технологій.

1.1.1 Зростання ролі мобільних застосунків у мовному навчанні

Останні роки [1] характеризуються стрімким зростанням популярності мобільних застосунків у сфері мовної освіти. Завдяки широкому розповсюдженню смартфонів та розвитку технологій доступу до інтернету, користувачі отримали можливість навчатися в зручному для себе ритмі, не прив’язуючись до класичних освітніх форматів. Це особливо актуально для вивчення слів, яке вимагає частого повторення, індивідуального темпу і зручного способу перевірки засвоєного матеріалу.

Мобільні застосунки демонструють зростання інтерактивності завдяки використанню мультимедійного контенту, гейміфікації та елементів адаптивного навчання. Користувачі можуть переглядати значення слів, слухати вимову, застосовувати слова у реченнях та проходити тести для закріплення знань. Така форма подачі матеріалу створює ефект занурення, який є ключовим чинником у формуванні довготривалої пам’яті.

Згідно з аналітичними даними, кількість користувачів мобільних мовних застосунків з кожним роком зростає. Це свідчить про підвищення довіри до таких інструментів серед широких верств населення. Далі можна побачити динаміку зростання використання подібних рішень у період з 2019 по 2023 рік (рисунок 1.1). Як видно, кількість користувачів не лише стабільно зростає, а й демонструє прискорення темпів приросту.

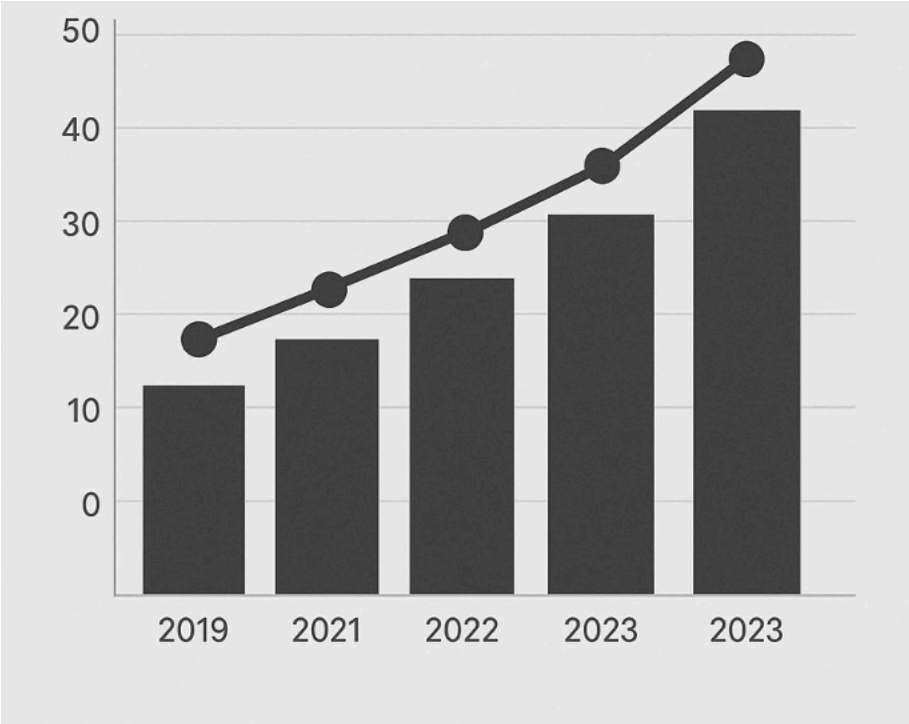

Рисунок 1.1 – Динаміка зростання впливу мобільних застосунків у процесі мовного навчання за останні рока

Такий тренд є прямим свідченням зростання ролі мобільних застосунків у сучасній мовній освіті. Вони стали не лише доповненням до традиційних методів, а й повноцінною альтернативою, здатною охопити більшу аудиторію, забезпечити гнучкість та забезпечити високу якість навчання лексики завдяки технологічним інноваціям.

1.1.2 Персоналізоване навчання як сучасний тренд

Персоналізоване навчання передбачає адаптацію освітнього контенту до індивідуальних особливостей кожного користувача. У контексті вивчення лексики це означає врахування рівня володіння мовою, інтересів, частоти повторення, прогресу у засвоєнні та навіть особистих цілей, таких як підготовка до подорожі або іспиту. Такий підхід дозволяє ефективніше структурувати навчальний процес, уникати перевантаження або навпаки –

недостатньої інтенсивності, що характерно для традиційних статичних методик.

Мобільні застосунки реалізують персоналізацію через динамічне оновлення навчальних траєкторій на основі аналітики дій користувача. Наприклад, якщо система бачить, що користувач не засвоює певний тип слів, вона може запропонувати інші приклади, змінити спосіб подачі матеріалу або збільшити частоту повторення. Окремі застосунки також дозволяють вказати тематику, яка цікава користувачу, обрати бажану складність та стиль англійської мови – британський або американський акцент.

У контексті використання генеративного штучного інтелекту персоналізація виходить на новий рівень. Завдяки мовним моделям можна автоматично створювати словникові набори, враховуючи обрані теми, рівень підготовки та навіть побажання щодо контексту використання слів. Це забезпечує не просто адаптацію до користувача, а дійсно індивідуальний підхід, коли навчальний контент створюється унікально під запит конкретної людини, уникаючи шаблонності та повторюваності.

1.1.3 Вплив генеративного ШІ на методики засвоєння лексики

Використання генеративного штучного інтелекту у лексичному навчанні відкриває нові можливості у побудові індивідуального освітнього досвіду. Завдяки таким моделям, як GPT, користувачі можуть отримувати не лише готові списки слів, а й приклади речень, пояснення значень у контексті, синонімічні ряди та зворотний зв’язок щодо власних висловлювань. Це перетворює статичне навчання у живий діалог із системою, що здатна моделювати роль викладача, коректора і тьютора одночасно.

Залучення генеративного ШІ також підвищує ефективність зворотного зв’язку, дозволяючи перевіряти створені речення, оцінювати

граматику, відповідність контексту і навіть лексичну складність. Користувач може експериментувати з мовними структурами без страху зробити помилку, адже система пояснює недоліки і пропонує варіанти виправлень. Це особливо корисно для тих, хто навчається самостійно, без постійної підтримки викладача. Далі можна побачити перелік основних переваг (рисунок 1.2), які забезпечує використання ШІ в освітньому контексті.

Рисунок 1.2 – Переваги використання ШІ у застосунку для вивчення слів

Серед ключових переваг варто виділити гнучкість масштабування, що дозволяє легко адаптувати систему для будь-якої кількості користувачів. Завдяки даним, які генеруються під час навчання, можна отримати аналітику щодо успішності засвоєння, інтересів користувачів і типових помилок, що дозволяє постійно вдосконалювати освітній контент. Системи зі штучним інтелектом також підвищують доступність навчання, знижуючи бар'єри, пов'язані з вартістю курсів, фізичною присутністю викладача або нестачею якісного контенту. Ще одним важливим фактором є зростання мотивації до навчання. Користувачі, які взаємодіють із «розумним» застосунком, відчувають більший інтерес і залученість, оскільки отримують не шаблонну інформацію, а відповіді на свої конкретні запити.

1.2 Огляд існуючих мобільних застосунків для вивчення слів

Сучасний ринок мобільних застосунків для вивчення іноземних мов представлений великою кількістю рішень, які орієнтуються на різні аспекти навчального процесу. Серед них є застосунки, що зосереджуються на граматиці, аудіюванні, читанні або письмі, однак саме вивчення лексики залишається однією з найпопулярніших категорій. Це зумовлено тим, що знання слів є базовою складовою мовної компетентності, без якої неможливо будувати ані розуміння, ані висловлювання.

Більшість таких застосунків працюють за принципом створення словникових наборів і їх поступового засвоєння через повторення, асоціації або гейміфікаційні елементи. Вони часто пропонують готові списки слів з перекладом, прикладами та аудіосупроводом, а також систему тестування для перевірки знань. У ряді випадків використовується технологія адаптивного навчання, яка дозволяє змінювати темп та складність відповідно до результатів користувача.

Різноманітність підходів до реалізації функціоналу в конкурентних рішеннях дозволяє виділити декілька напрямів розвитку: тематичне навчання, вивчення слів у контексті, створення користувацьких наборів, використання штучного інтелекту для аналізу прогресу та автоматичного формування завдань. Кожен застосунок має свої сильні сторони, які роблять його привабливим для певної категорії користувачів, однак водночас більшість з них мають обмеження у персоналізації або уніфікований підхід до побудови контенту.

У зв’язку з цим актуальним є проведення огляду найбільш популярних застосунків для вивчення слів, порівняння їх функціональних можливостей, інтерфейсів, підходів до оцінювання результатів і надання зворотного зв’язку. Такий аналіз дозволяє виявити типові рішення, що повторюються у більшості продуктів, а також знайти ніші, які залишаються

недостатньо охопленими та можуть бути покриті за допомогою нових технологічних підходів.

1.2.1 Найпопулярніші застосунки

На сучасному ринку існує значна кількість мобільних застосунків, спеціалізованих на вивченні лексики. Деякі з них набули популярності завдяки оригінальному підходу до побудови вправ або гейміфікованій формі подачі матеріалу. Наприклад, застосунки Vocabulary.com, WordUp, Word of the Day, Words with Friends та 7 Little Words забезпечують користувачам різні способи засвоєння нових слів, від щоденних викликів до ігрових форматів із друзями. Їх основна мета – зробити вивчення слів регулярним, цікавим та мотивуючим. Далі можна побачити приклади таких застосунків, які орієнтовані саме на вивчення лексики (рисунок 1.3).

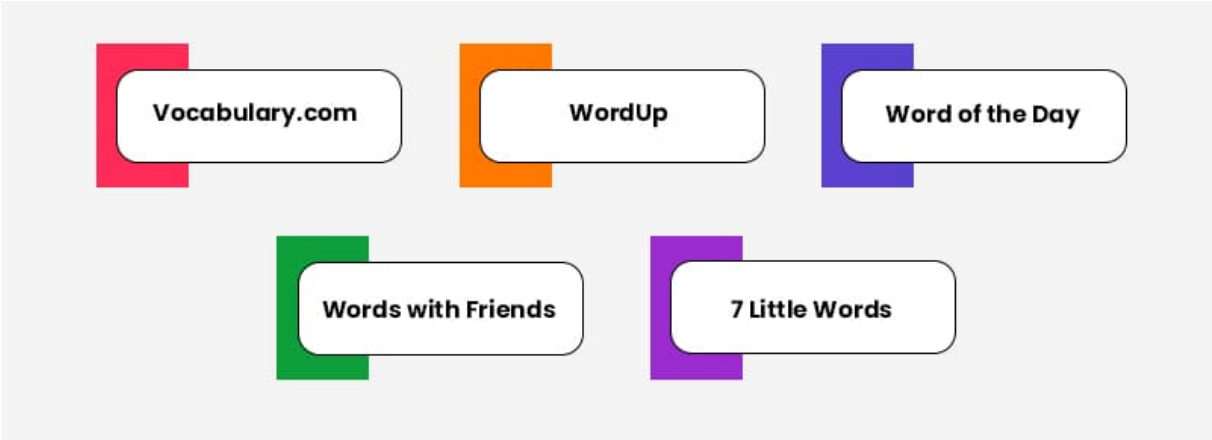

Рисунок 1.3 – Найпопулярніші застосунки для вивчення іноземних слів

Vocabulary.com поєднує словникову базу з системою практичних завдань, яка автоматично адаптується під рівень користувача. WordUp робить акцент на найбільш уживаних у реальному житті словах, підкріплюючи їх прикладами з фільмів та серіалів. Word of the Day фокусується на щоденному поповненні словникового запасу через подання

одного слова в день разом з поясненнями та прикладами. Це дозволяє формувати звичку до регулярного навчання без перевантаження.

Words with Friends та 7 Little Words орієнтовані більше на гейміфікацію – вони залучають користувачів через ігровий процес, де для досягнення результатів потрібно активно використовувати знання англійської лексики. Такий підхід виявився ефективним для користувачів, які не прагнуть формального навчання, але при цьому мають мотивацію покращити свою мовну інтуїцію через гру. Ці застосунки сприяють пасивному засвоєнню лексики через багаторазове повторення та взаємодію.

Серед великих гравців ринку особливо вирізняються Duolingo, Memrise та Quizlet. Ці платформи охоплюють ширший спектр мовного навчання, але кожна з них приділяє велику увагу саме лексиці. Duolingo відома своїм інтерфейсом із короткими вправами, що стимулюють регулярну практику. Memrise використовує методику асоціативного запам’ятовування та приклади з живої мови. Quizlet надає гнучкий інструментарій для створення та використання власних або спільних словникових наборів. Далі можна побачити логотипи цих провідних застосунків (рисунок 1.4).

Рисунок 1.4 – Великі гравці на ринку

Застосунки такого рівня стали стандартом де-факто у багатьох навчальних закладах і серед самостійних учнів. Вони забезпечують доступність, багатофункціональність та інтеграцію з іншими платформами, що дозволяє користувачам зручно організовувати процес навчання. Завдяки широкому охопленню та постійним оновленням, ці продукти часто слугують відправною точкою для подальшого порівняння нових рішень на ринку мобільної освіти.

Узагальнюючи, можна сказати, що попит на застосунки для вивчення слів формує велику екосистему продуктів, які конкурують між собою за функціональністю, зручністю та здатністю мотивувати користувача до систематичного навчання. Аналіз таких застосунків дає змогу виявити не лише успішні практики, а й недоліки, які можуть бути усунені за допомогою інноваційних рішень, таких як інтеграція генеративного штучного інтелекту.

1.2.2 Порівняльна характеристика функціоналу

Функціональні можливості є ключовим чинником при виборі користувачем застосунку для вивчення іноземної лексики. У більшості випадків сучасні рішення не обмежуються лише демонстрацією списку слів із перекладом.

Вони прагнуть охопити повний цикл навчання: від першого ознайомлення зі словом до його активного вживання у мовленні. Саме тому розробники приділяють особливу увагу створенню багаторівневої системи навчання, яка поєднує базу даних слів, вправи, тестування, адаптацію та мотиваційні елементи. Найбільш поширені компоненти функціоналу включають словникову базу, флеш-картки, інтерактивні тести, персоналізовані рекомендації, мовні ігри, систему відстеження прогресу та соціальні можливості.

Далі можна побачити типову структуру функціонального наповнення таких застосунків (рисунок 1.5). Ці складові формують ядро цифрової платформи для вивчення слів, дозволяючи охопити як пасивне, так і активне запам’ятовування, а також контроль результатів навчання.

Рисунок 1.5 – Функціонал сучасного додатку для вивчення слів

Багато застосунків роблять акцент саме на флеш-картках, як зручному та візуально простому способі запам’ятовування нової лексики. Інтерактивні вікторини допомагають оцінити знання в умовах, наближених до реального використання мови. Персоналізація ж дозволяє адаптувати навчальний контент під стиль засвоєння, який найкраще підходить конкретному користувачу. Завдяки ігровим механікам та соціальним елементам, як-от таблиці лідерів або челенджі з друзями, зростає мотивація до регулярного навчання.

Проте не всі продукти реалізують повний набір цих функцій. Частина застосунків має обмежену базу лексики або не підтримує формування власних словників. Інші – не забезпечують зворотного зв’язку або не мають механізмів персоналізації. Порівняльний аналіз функціоналу дозволяє не лише оцінити переваги окремих рішень, а й сформулювати перелік очікуваних можливостей для нового продукту, який має бути конкурентоспроможним на ринку.

1.2.3 Визначення обмежень і недоліків конкурентів

Незважаючи на широкий вибір мобільних застосунків для вивчення слів, більшість із них мають подібну структуру навчання та обмежений набір інструментів персоналізації. Типовою проблемою є використання статичних словникових баз, які не враховують потреби конкретного користувача. Навіть якщо користувач обирає певну тему або рівень складності, самі слова залишаються універсальними для всіх, що знижує ефективність і мотивацію до навчання.

Ще одним поширеним недоліком є недостатній рівень інтерактивності. У багатьох додатках відсутні повноцінні механізми практичного використання вивчених слів, наприклад, формування речень або перевірка розуміння контексту. Тести з варіантами відповідей не забезпечують глибокого опрацювання лексики, а лише створюють ілюзію прогресу. Також значна частина продуктів не підтримує перевірку граматики або стилістики, що є важливою умовою для ефективного мовного розвитку.

Додатковою проблемою є відсутність адаптації до темпу навчання. Більшість застосунків працюють за фіксованим сценарієм, не враховуючи того, що різні користувачі засвоюють інформацію з різною швидкістю. У результаті деякі користувачі відчувають перенавантаження, тоді як інші втрачають інтерес через повільне просування. Лише окремі продукти реалізують алгоритми, які враховують частоту помилок, час відповіді або рівень упевненості, щоб коригувати подальший контент.

Крім того, багато застосунків обмежені в мовній підтримці, акценті або культурному контексті. Наприклад, слова часто подаються без урахування відмінностей між британським та американським варіантами англійської. Також рідко враховується мета навчання – підготовка до тесту, подорож, професійне спілкування тощо.

1.3 Аналіз потреб і поведінки цільової аудиторії

Ефективне проектування мовного застосунку потребує глибокого розуміння потреб і очікувань цільової аудиторії. Користувачі відрізняються за віком, рівнем мовної підготовки, мотивацією до навчання та доступом до ресурсів. Це впливає як на бажаний функціонал, так і на формат подачі навчального матеріалу. Розуміння цих відмінностей є ключовим для створення зручного та корисного інтерфейсу.

Серед основних факторів, які визначають вибір користувача, варто виділити зручність навігації, простоту інтерфейсу, адаптивність матеріалу та наявність візуального супроводу. Багато користувачів очікують можливості швидкого старту без довгого налаштування, а також бажають бачити чіткий прогрес і мотиваційні елементи. Застосунки, що не враховують ці аспекти, ризикують втратити аудиторію на ранніх етапах використання.

Важливим чинником є й очікування щодо персоналізації. Користувачі хочуть відчувати, що система працює саме під їхні цілі – наприклад, підготовку до подорожі або іспиту. Це стосується не лише підбору слів, а й тону, складності речень, акценту та темпу подачі. Особливо це актуально для користувачів, які мають обмежений час і хочуть результатів швидко.

Окрему увагу слід приділити тому, як користувачі сприймають роль штучного інтелекту в процесі навчання. Для багатьох важливо, щоб взаємодія з системою була не лише точною, а й людяною, зрозумілою та без стресу. Саме тому аналіз очікувань користувачів щодо використання ШІ в освіті має суттєве значення для формування відповідного функціоналу.

1.3.1 Визначення цільової аудиторії

Цільова аудиторія мобільного застосунку для вивчення слів охоплює широкий спектр користувачів, однак основною групою залишаються особи

віком від 16 до 35 років, які активно використовують цифрові технології у повсякденному житті. Це школярі старших класів, студенти, молоді спеціалісти та всі, хто прагне покращити володіння іноземною мовою для особистого чи професійного розвитку. Вони цінують мобільність, швидкий доступ до інформації та можливість навчатися у зручному темпі.

До другої за чисельністю категорії належать дорослі користувачі, які вивчають мову з практичних міркувань – наприклад, для роботи за кордоном, подорожей або еміграції. Для них важливо, щоб застосунок був простим у використанні, інтуїтивно зрозумілим і пропонував реальні приклади застосування слів у побутових чи професійних ситуаціях. Такі користувачі часто мають обмежений час для навчання і цінують функціонал, який дозволяє швидко досягати результатів.

Окрему нішу формують учні мовних шкіл та слухачі курсів, які використовують мобільні застосунки як додатковий інструмент для закріплення знань. У цьому випадку важливо забезпечити інтеграцію з навчальним планом або можливість створення індивідуальних наборів слів. Часто ці користувачі слідують рекомендаціям викладача, тому зручний обмін контентом і керування прогресом стають вагомими перевагами.

Крім того, до потенційної аудиторії входять підлітки та навіть молодші користувачі, які починають знайомство з іноземною мовою за допомогою мобільних пристроїв. Для них ключову роль відіграє гейміфікація, яскравий дизайн та елементи гри. Таким чином, цільова аудиторія є різнорідною, а отже застосунок має бути гнучким і адаптивним до різних освітніх потреб, вікових особливостей і мотиваційних чинників.

1.3.2 Очікування користувачів щодо функціоналу та UX

Користувачі сучасних мобільних застосунків очікують простого та інтуїтивного інтерфейсу, який дозволяє одразу зрозуміти, як розпочати навчання. Наявність чіткої структури, зручної навігації та мінімальної

кількості дій для доступу до основного функціоналу є базовою вимогою для позитивного досвіду взаємодії. Багато хто хоче мати змогу почати вивчення вже з головного екрану, без довгого заповнення профілю чи перегляду інструкцій.

Ще одне поширене очікування – це персоналізоване навчання. Користувачі хочуть, щоб застосунок адаптувався під їх рівень знань, вибрану тему, інтереси та цілі. Можливість самостійно обирати складність, акцент, кількість слів на день або формат вправ сприймається як важлива перевага. Інтерфейс повинен дозволяти легко редагувати ці налаштування та швидко змінювати режим навчання без потреби повертатися до головного меню.

Функціональність також повинна включати інструменти для моніторингу прогресу. Більшість користувачів хоче бачити результат своєї роботи у вигляді графіків, відсотків або візуальних індикаторів, що стимулює їх продовжувати навчання. Корисними є також нагадування, досягнення, рейтинги або інші мотиваційні елементи, які підтримують регулярну активність і створюють відчуття поступу.

Окрім навчального змісту, користувачі високо оцінюють швидкість роботи застосунку, стабільність, можливість офлайн-доступу до словників і захист персональних даних. Важливо, щоб інтерфейс був адаптований під різні пристрої та працював без збоїв. Відсутність реклам, нав’язливих повідомлень або складної реєстрації також формує позитивний досвід і підвищує лояльність до продукту.

1.3.3 Особливості використання ШІ у навчальному процесі

Інтеграція штучного інтелекту в мовні застосунки відкриває нові можливості, які безпосередньо впливають на досвід користувача. Завдяки ШІ системи можуть не лише показувати слова та приклади, а й взаємодіяти з користувачем у формі діалогу, аналізувати введений текст, давати

рекомендації та адаптувати складність завдань. Такий підхід значно покращує залучення до навчального процесу, роблячи його гнучким та динамічним.

Користувачі очікують від ШІ не просто автоматизації, а розумного супроводу, який здатен замінити викладача у багатьох аспектах. Наприклад, можливість перевірити своє речення, отримати пояснення помилок, побачити варіанти покращення або поставити запитання про значення слова є бажаними для більшості. Далі можна побачити основні функції, які можуть реалізовуватись за допомогою ШІ (рисунок 1.6) в освітньому середовищі.

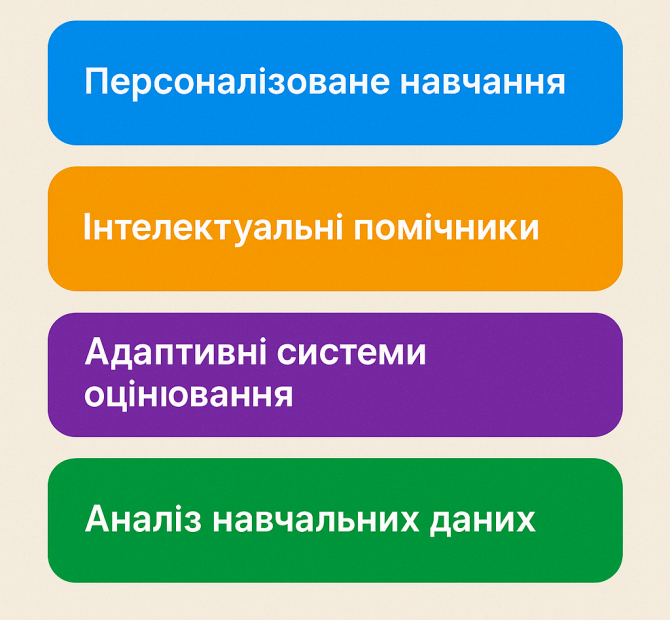

Рисунок 1.6 – Можливості ШІ у навчальному процесі

Особливо високо користувачі оцінюють інструменти, які дозволяють перевірити контекст правильності використання слова, отримати синоніми чи приклади вживання з реального життя. Це забезпечує глибше розуміння і запам’ятовування, ніж стандартне зазубрювання перекладів. Крім того, ШІ може забезпечити більш гнучку структуру навчання, підлаштовуючись під прогрес користувача у реальному часі.

Однак водночас важливо, щоб такі системи залишалися прозорими і зрозумілими. Користувачі не завжди довіряють алгоритмам, які не пояснюють своїх рішень. Тому інтерфейс має бути розроблений так, щоб взаємодія з ШІ не викликала труднощів, а відчувалась як підтримка. За умови правильної реалізації ШІ стає не лише інструментом, а й повноцінним цифровим наставником у вивченні іноземної лексики.

1.4 Визначення ключових функцій застосунку

Ключові функції мобільного застосунку для вивчення слів визначаються його загальною концепцією, орієнтованою на персоналізоване, гнучке та ефективне навчання. Застосунок має не лише відображати словникові набори, а й підтримувати повний цикл опрацювання лексики: від генерації до активного використання. Основу функціоналу становлять інструменти для створення, перегляду, вивчення та практичного засвоєння нових слів.

Важливою особливістю є використання генеративного штучного інтелекту для формування індивідуальних словникових колекцій. Користувач задає параметри, а система автоматично створює набір слів з поясненнями, перекладом і прикладами. Це дозволяє уникнути шаблонного підходу та врахувати особисті мовні цілі. ШІ також відіграє роль в оцінюванні результатів навчання, аналізуючи відповіді користувача у практичному режимі.

Ще одним важливим елементом є перегляд та повторення вивчених слів. Для ефективного запам’ятовування система повинна підтримувати інтервальне повторення, а також давати можливість повернутися до складних або не засвоєних слів. Застосунок має відображати прогрес користувача і давати йому мотиваційні сигнали щодо досягнень.

Окрему роль відіграє практичний режим, у якому користувач застосовує слова в реченнях, отримуючи зворотний зв’язок від ШІ. Такий

формат не лише перевіряє знання, а й формує навичку реального використання мови. Загалом функціональність застосунку має бути спрямована на створення гнучкого, адаптивного середовища, яке дозволяє кожному користувачу будувати власну траєкторію вивчення лексики.

1.4.1 Генерація словникових наборів

Генерація словникових наборів є однією з основних функцій у мобільному застосунку, орієнтованому на персоналізоване вивчення лексики. Вона дозволяє користувачеві швидко отримати набір слів відповідно до заданих параметрів без необхідності самостійного пошуку або складання списків. Такий підхід значно економить час та забезпечує цілеспрямоване навчання, що відповідає конкретній меті або ситуації користувача.

Процес генерації починається з того, що користувач обирає тему словника, наприклад, «подорожі», «бізнес» або «технології». Додатково можна вказати рівень складності, тип англійської (британська або американська), а також додаткові побажання щодо контексту використання слів. Ці параметри формують запит до генеративної мовної моделі, яка створює відповідний список лексичних одиниць із супровідною інформацією.

На відміну від статичних словників, така система дозволяє формувати нові унікальні набори, які відповідають поточному рівню знань та цілям користувача. Генерація відбувається в режимі реального часу, а зміст кожного набору залежить від вхідних даних. Це робить навчання максимально релевантним і дозволяє уникнути зайвої інформації, яка не відповідає запиту.

Кожне згенероване слово супроводжується поясненням англійською мовою, українським перекладом, синонімами та прикладом уживання в реченні. Така багатокомпонентна структура підвищує глибину засвоєння та

допомагає зрозуміти значення слова у конкретному контексті. Користувач має змогу ознайомитися зі словом ще до початку його вивчення у практичному режимі.

Після створення словникового набору він зберігається у профілі користувача та відображається на головному екрані. Надалі користувач може перейти до його вивчення, повторення або редагування. Це дозволяє організувати навчальний процес за тематичними блоками і легко повертатися до потрібного матеріалу в будь-який момент.

Загалом функція генерації словникових наборів забезпечує персоналізацію навчального контенту та створює гнучкий механізм для початку вивчення нової лексики. Завдяки використанню генеративного ШІ цей процес є не лише швидким, а й інтелектуально орієнтованим, що відрізняє такий підхід від традиційних способів складання словників.

1.4.2 Перегляд, вивчення та повторення лексики

Перегляд, вивчення та повторення лексики становлять основний етап засвоєння словникового матеріалу після його генерації. Цей функціонал мобільного застосунку повинен бути інтуїтивно зрозумілим і зручним для користувача, щоб сприяти регулярному навчанню та підтримувати високий рівень залученості. Основна мета цього етапу – забезпечити ефективне запам’ятовування слів через поєднання візуальної, текстової та аудіальної інформації.

Кожне слово у словниковому наборі подається у розгорнутому вигляді: англійське написання, переклад українською, пояснення англійською мовою, список синонімів і приклад речення. Це дозволяє користувачеві зрозуміти значення слова не лише на формальному рівні, а й у контексті. Окрім того, можливість прослуховування вимови сприяє розвитку фонетичної пам’яті та правильному відтворенню мовлення.

У режимі вивчення користувач поступово проходить через увесь словниковий набір, взаємодіючи зі словами в різних форматах. Наприклад, можна гортати картки, відкривати переклади лише після ознайомлення з визначенням, або порівнювати синоніми. Такий підхід дозволяє активізувати пізнавальні процеси та сформувати глибше розуміння лексики.

Окреме значення має механізм повторення. Застосунок має підтримувати принцип інтервального повторення, який базується на когнітивних моделях довготривалої пам’яті. Система автоматично визначає, які слова слід повторити на основі поведінки користувача – тривалості перегляду, частоти помилок або кількості звернень до перекладу. Це дозволяє зосередитися на складних для запам’ятовування елементах і уникнути надлишкових повторень простих слів.

Важливо, щоб користувач мав змогу вручну позначати слова як вивчені або складні. Також варто реалізувати фільтри для перегляду лише нових, вивчених або тих, що потребують повторення. Така гнучкість дозволяє адаптувати навчальний процес до власного стилю засвоєння інформації та уникати одноманітності у подачі матеріалу.

Таким чином, функціонал перегляду, вивчення та повторення слів виконує не лише технічну, а й педагогічну роль, формуючи звичку до системного навчання. Його ефективність значною мірою залежить від зручності інтерфейсу, гнучкості налаштувань і здатності підтримувати баланс між повторенням і новим матеріалом.

1.4.3 Практичний режим із перевіркою OpenAI

Практичний режим із перевіркою OpenAI є ключовим компонентом застосунку, який дозволяє користувачеві не лише пасивно ознайомлюватися з лексикою, а й активно її застосовувати. Цей режим спрямований на формування навичок контекстного використання слів у власному мовленні. Основна ідея полягає в тому, що користувач створює речення із заданим

словом, після чого отримує зворотний зв’язок на основі аналізу генеративної моделі ШІ.

Після активації практичного режиму система пропонує користувачеві слово з активного словникового набору. Завдання полягає в тому, щоб скласти граматично правильне речення, яке демонструє розуміння значення і доречність вживання цього слова. Такий підхід дозволяє перевірити не лише факт знання перекладу, а й здатність впевнено використовувати слово в реальному контексті.

Після введення речення активується перевірка з використанням OpenAI. Модель аналізує текст за кількома критеріями: правильність граматики, відповідність змісту до значення слова, стилістична природність конструкції. Користувач отримує коментар, у якому пояснюється, чи правильно використано слово, які є можливі покращення та чому запропоноване речення вважається коректним або помилковим.

Важливою перевагою такого підходу є персоніфікований зворотний зв’язок. Модель може запропонувати альтернативні варіанти речення, пояснити відмінності між схожими словами або акцентувати увагу на контексті. Це робить навчання більш гнучким і корисним, адже користувач не просто отримує оцінку, а й розуміє причини помилок та способи їх виправлення.

Додатково реалізуються допоміжні функції: кнопка перегляду прикладу, щоб побачити зразкове речення із цим словом, кнопка переходу до наступного слова або можливість спробувати ще раз. Користувач також може зберігати складні речення для подальшого перегляду або повторного опрацювання. Це створює умови для формування активного словника, який справді використовується у мовленні. Загалом практичний режим із перевіркою OpenAI поєднує елементи навчання, самоперевірки та діалогу з інтелектуальною системою. Такий формат не лише покращує рівень володіння лексикою, а й формує впевненість у власних мовних навичках.

1.4.4 Ведення прогресу навчання та персональні налаштування

Ведення прогресу навчання є важливим елементом застосунку, який дозволяє користувачеві контролювати власні досягнення та оцінювати ефективність навчального процесу. Основна мета цього функціоналу – створити прозору систему відстеження результатів, яка мотивує до регулярної роботи з лексикою. Візуальні індикатори, такі як прогрес-бари, відсотки засвоєння або кількість вивчених слів, сприяють формуванню відчуття успіху та досягнення мети.

Застосунок має зберігати інформацію про кожне слово – нове, вивчене, потребує повторення. На основі цієї інформації формуються добірки для повторення та рекомендації щодо подальших дій. Також доцільно реалізувати щоденні або щотижневі статистики, які показують кількість нових слів, виконаних практичних завдань та рівень активності користувача. Це допомагає побачити загальну динаміку навчання та вчасно виявити спад мотивації.

Окрім автоматичних індикаторів прогресу, важливо надати користувачеві інструменти для персоналізації. Це може включати можливість позначати слова як складні, додавати власні позначки або нотатки, обирати режим складності практичних завдань. Застосунок має бути гнучким до індивідуального стилю навчання і дозволяти користувачеві впливати на структуру процесу. Таким чином, ведення прогресу не лише виконує функцію обліку, а й слугує інструментом підтримки мотивації та організації навчального процесу.

1.5 Постановка задачі

З огляду на аналіз предметної галузі, сучасні тенденції цифрової освіти та обмеження існуючих рішень, постає необхідність створення інноваційного мобільного застосунку для вивчення слів, який поєднує

персоналізацію, гнучкість та можливості генеративного штучного інтелекту. Такий застосунок має відповідати очікуванням користувачів щодо адаптивного контенту, інтерактивності та зручності у користуванні. Особливу увагу слід приділити створенню інструментів для активного засвоєння лексики у контексті.

Основна задача полягає у розробці мобільного застосунку, що дозволяє користувачеві створювати індивідуальні словникові набори, вивчати слова з контекстом та практикувати їх використання у власних реченнях із автоматичною перевіркою. Функціонал має бути побудований навколо генеративного ШІ, який відповідає за створення контенту, адаптацію під рівень користувача та оцінку мовного вводу. Це дозволить забезпечити глибший рівень залученості до навчального процесу та підвищити ефективність засвоєння матеріалу.

У межах реалізації необхідно також забезпечити зберігання навчальних даних, ведення прогресу користувача та реалізацію гнучкої архітектури, здатної масштабуватися відповідно до кількості активних користувачів. Інтерфейс повинен бути зрозумілим і доступним, а ключові функції – легко доступними без додаткових інструкцій або тривалого навчання. Застосунок повинен підтримувати теми, рівні складності, варіанти акцентів та інші параметри, які впливають на персоналізацію досвіду.

Таким чином, задача полягає у створенні повноцінного освітнього інструменту нового покоління, який використовує потенціал генеративного ШІ не лише як допоміжну технологію, а як центральний компонент навчальної логіки. Рішення має бути орієнтоване на практичне використання в реальних умовах, враховувати потреби цільової аудиторії та забезпечувати високу якість лексичної підготовки на всіх етапах навчального процесу.

2 ПРОЕКТУВАННЯ СИСТЕМИ

2.1 Аналіз вимог до персоналізованого лексичного середовища

Проектування мобільного застосунку для вивчення слів на основі генеративного штучного інтелекту вимагає ретельного аналізу технічних, функціональних та нефункціональних вимог. Урахування цих вимог є визначальним для ефективної реалізації системи, її масштабованості, зручності у використанні та відповідності очікуванням цільової аудиторії. У цьому контексті особливу роль відіграє персоналізація навчального процесу, яка передбачає адаптацію вмісту під конкретного користувача з урахуванням його мовного рівня, навчальної мети та темпу засвоєння матеріалу.

Функціональні вимоги застосунку охоплюють основні можливості, які повинні бути реалізовані у системі. Насамперед це генерація словникових наборів на основі параметрів, заданих користувачем. Такі параметри включають тему, наприклад, бізнес, подорожі, наука, рівень складності, початковий, середній, просунутий, бажаний акцент, британський або американський. Користувач очікує, що набір буде унікальним, логічно пов’язаним та релевантним, а слова – подані з перекладом, поясненням, прикладами речень і, за можливості, синонімами.

Інша критично важлива функція – можливість вивчення, перегляду і повторення лексики у зручному інтерфейсі. Користувач повинен мати змогу переглядати слова з будь-якого словника, відмічати слова як вивчені або складні, бачити статус кожного елемента (нове, вивчене, потребує повторення), а також отримувати рекомендації щодо того, що повторювати далі. Додатково, система повинна забезпечувати сортування слів за темами, ступенем засвоєння або останнім часом активності.

Особливе місце у функціональній частині займає практичний режим, в якому користувачеві пропонується створювати речення з використанням

вивчених слів. Після введення речення система повинна передати його до OpenAI API для перевірки, отримати відповідь, інтерпретувати її та надати користувачеві зворотний зв’язок. У цьому режимі має бути передбачено кілька варіантів взаємодії – перегляд прикладу, спроба ще раз, перехід до наступного слова, збереження складних речень для подальшого опрацювання. Важливо, щоб усі ці дії не переривали навчальний процес і відбувалися швидко та безпомилково.

Користувач повинен мати власний профіль, у якому зберігатимуться усі створені словники, налаштування, прогрес у вивченні, результати практичних перевірок. Аутентифікація має бути реалізована через надійний сервіс (у даному випадку Auth0), який дозволяє забезпечити безпеку, одночасно надаючи можливість швидкого входу за допомогою соцмереж. Після входу користувач потрапляє на головний екран із доступом до всіх функцій – створення словника, перегляду наявних колекцій, перегляду прогресу, переходу до практики або налаштувань.

Серед нефункціональних вимог першочергове значення має стабільність, продуктивність і масштабованість системи. Застосунок повинен працювати без збоїв навіть у випадку великої кількості одночасних запитів до API, швидко обробляти дані користувача, зберігати інформацію у хмарному середовищі (Firestore) і надавати доступ до неї у режимі реального часу. Потрібно забезпечити кешування результатів, обробку помилок при взаємодії з OpenAI та fallback-механізми, які дозволяють продовжити роботу навіть у випадку недоступності зовнішнього сервісу.

Інтерфейс застосунку має бути адаптивним до мобільних пристроїв з різними розмірами екранів. Його структура повинна бути побудована таким чином, щоб користувач легко орієнтувався у вмісті, швидко переходив між режимами, змінював налаштування, переглядав звіти про успішність. Колірна палітра, шрифти, кнопки, анімації та розмітка мають бути гармонійними, відповідати сучасним UI/UX-стандартам та не перевантажувати користувача.

З точки зору безпеки система повинна гарантувати збереження персональних даних, обмеження доступу до чужого контенту та використання лише авторизованих запитів при доступі до серверних функцій. Для цього слід реалізувати захист API-запитів, автентифікацію на базі токенів, обмеження прав доступу до словників інших користувачів, а також логування критичних подій.

Ще одним аспектом, який формує вимоги до системи, є очікування користувача щодо прозорого зворотного зв’язку. Під час практичного режиму користувач повинен чітко розуміти, чому речення було визнано правильним або неправильним, що саме потребує покращення та як уникнути подібних помилок у майбутньому. Усі відповіді ШІ мають бути стислими, логічними, не суперечити одна одній. Водночас система повинна зберігати результати практики для повторного перегляду і саморефлексії.

Загалом вимоги до системи формуються на перетині освітніх задач, технологічних можливостей і очікувань кінцевого користувача. Для досягнення високої якості кінцевого продукту необхідно забезпечити баланс між функціональністю, швидкістю обробки, зручністю використання і надійністю. Проектування системи повинно враховувати можливість розширення функціоналу в майбутньому, зокрема додавання нових мов, типів вправ, візуалізацій, аудіо-аналізу та інших елементів, що сприятимуть поглибленому вивченню лексики.

2.2 Формалізація сценаріїв користувацької взаємодії

Формалізація сценаріїв користувацької взаємодії є важливим етапом проектування, що дозволяє описати логіку дій користувача та системи у відповідь. Для забезпечення повноти та послідовності поведінки застосунку використовуються діаграми послідовності, які демонструють обмін повідомленнями між учасниками процесу: користувачем, мобільним застосунком, серверною частиною, базою даних і зовнішніми API. Такі

сценарії дозволяють ідентифікувати критичні точки взаємодії та спроектувати систему таким чином, щоб забезпечити плавний і передбачуваний досвід користування.

Один із базових сценаріїв – генерація нового словникового набору. Користувач (рисунок 2.1) обирає параметри тематики, рівня складності та акценту, після чого мобільний застосунок надсилає запит до серверної частини. Сервер, у свою чергу, звертається до OpenAI API для генерації відповідного списку слів. Після отримання результатів бекенд формує структуру словника і повертає її на клієнт. Далі користувач бачить готову колекцію для вивчення.

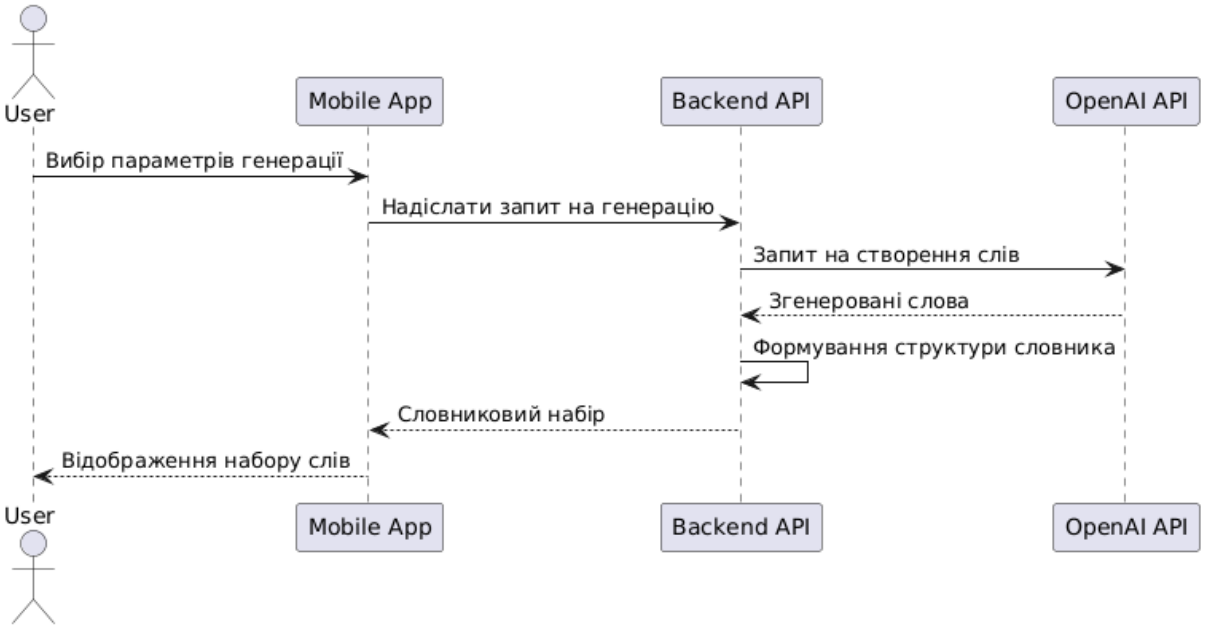

Рисунок 2.1 – Генерація нового словникового набору

Наступний сценарій передбачає вивчення слів із вже згенерованого словника. Користувач відкриває словникову колекцію, застосунок надсилає запит до бази даних, отримує відповідний набір слів, після чого відображає кожне слово з поясненням, перекладом і прикладом речення. Користувач переглядає слово, натискає на кнопку для переходу до наступного, і цикл

повторюється до завершення набору. Цей сценарій є типовим для режиму Learn і дозволяє реалізувати лінійне ознайомлення з новою лексикою.

Детальний опис послідовності дій під час вивчення слів відображено у відповідній діаграмі (рисунок 2.2). Важливо, що на цьому етапі не відбувається жодної генерації нового контенту – система працює з уже збереженою інформацією. Усі запити до Firestore реалізуються з урахуванням унікального ідентифікатора користувача та обраної колекції.

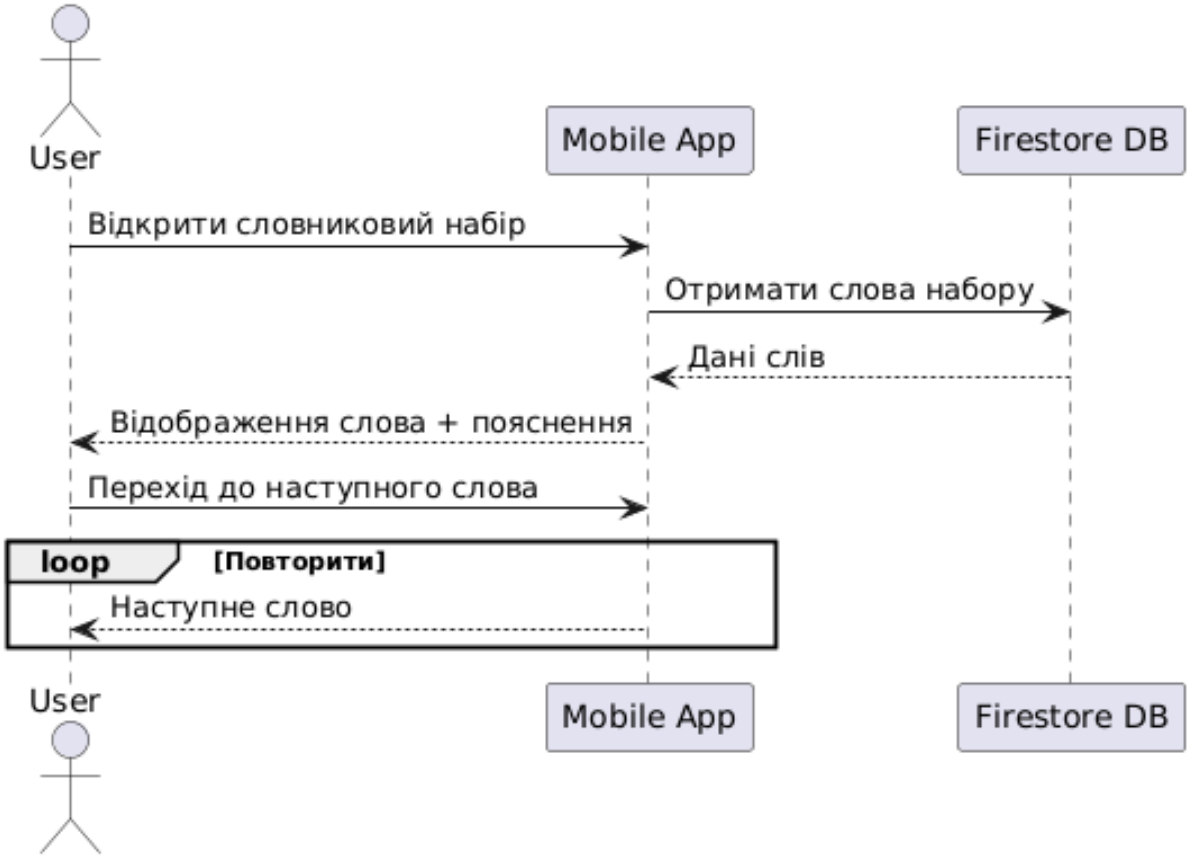

Рисунок 2.2 – Вивчення слів із набору

Ще один важливий сценарій пов’язаний із практичним застосуванням вивчених слів. У цьому режимі користувач самостійно складає речення з заданим словом. Мобільний застосунок передає речення на сервер разом із контекстом, а сервер виконує запит до OpenAI API для аналізу. Відповідь від ШІ включає оцінку правильності, виявлені помилки та рекомендації.

Отримані дані відображаються користувачеві у вигляді текстового зворотного зв’язку.

Деталізований обмін між клієнтом, сервером і OpenAI зображено у наступній діаграмі (рисунок 2.3). Цей сценарій важливий не лише як елемент практики, а й як джерело навчальної аналітики. Всі введені речення та результати їх перевірки можуть зберігатися у базі даних для формування статистики успішності користувача, рекомендацій щодо повторення та персоналізації майбутніх наборів.

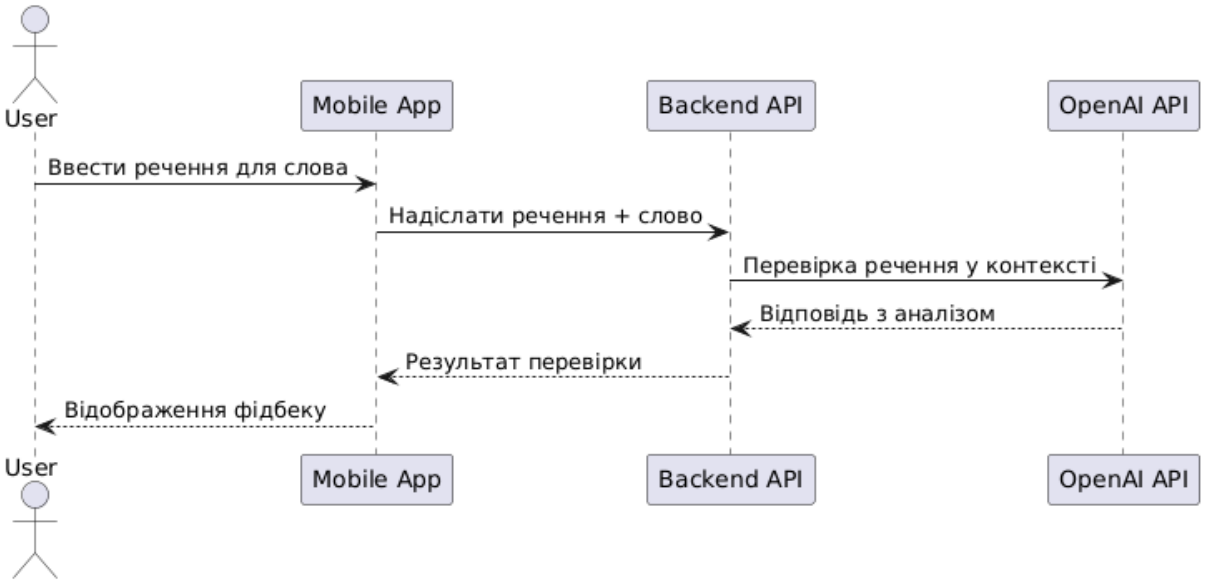

Рисунок 2.3 – Практичний режим – перевірка речення

Сценарій повторення слів базується на обліку прогресу користувача. Після активації режиму Revise мобільний застосунок звертається до бекенду з запитом на отримання слів, які були вивчені з недостатньою впевненістю або давно не повторювалися. Сервер аналізує записи в базі даних, формує список відповідних слів і повертає його клієнту. Далі користувач переглядає слова одне за одним у повторюваному циклі. Цей сценарій показано на діаграмі, яка ілюструє логіку вибірки слів для повторення та відображення їх користувачеві (рисунок 2.4).

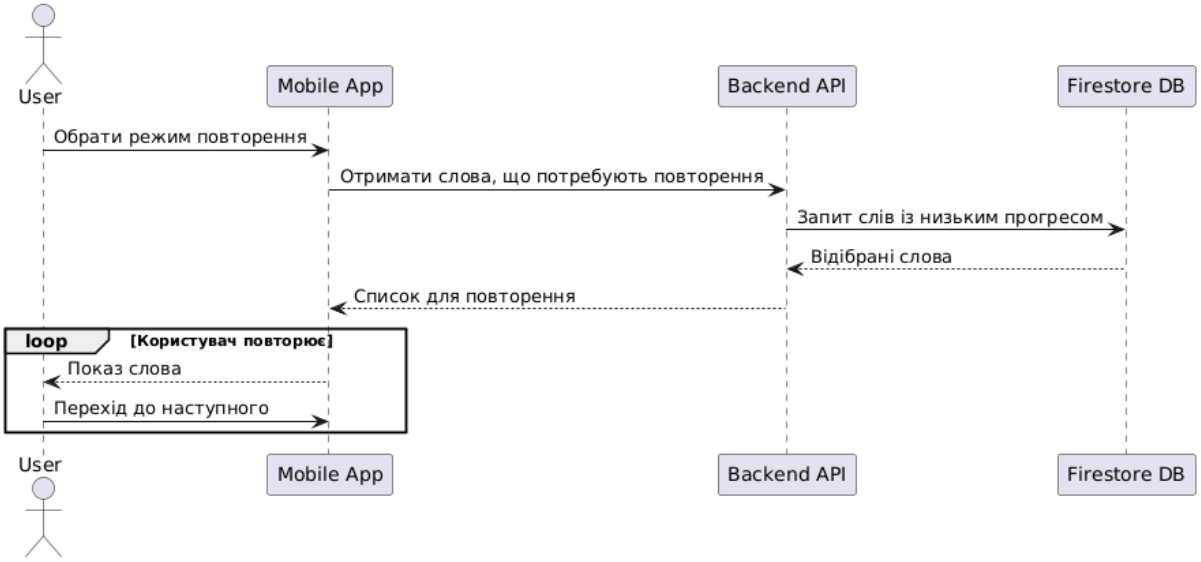

Рисунок 2.4 – Повторення вивчених слів

Кожен із розглянутих сценаріїв відображає певний етап навчальної взаємодії в межах мобільного застосунку. Вони охоплюють повний цикл роботи користувача: починаючи з генерації словникового контенту, далі – вивчення слів у лінійному форматі, активне застосування лексики у практичному режимі з аналізом правильності, та завершуючи механізмами повторення для закріплення знань. Такий підхід дозволяє моделювати не фрагментарну, а цілісну освітню траєкторію, що є особливо важливим для самостійного навчання без участі викладача.

Формалізація цих сценаріїв у вигляді діаграм послідовності дозволяє точно визначити порядок обміну повідомленнями між компонентами системи, а також виявити потенційні вузькі місця у логіці взаємодії. Наприклад, видно, у яких точках система залежить від відповіді стороннього API, коли потрібно реалізувати обробку помилок, де необхідне кешування результатів, а також як зберігаються дані користувача в базі. Це створює умови для проектування надійної та масштабованої архітектури, що забезпечить стабільну роботу навіть за великої кількості одночасних користувачів.

Крім того, такі сценарії слугують основою для тестування системи. На їх основі можуть бути розроблені автоматизовані тести для перевірки коректності реакцій застосунку у відповідь на дії користувача. Вони також важливі для узгодження комунікації між командами розробки фронтенду, бекенду та інтеграції OpenAI, адже дають чітке уявлення про те, які дані передаються між модулями, в якому форматі і на якому етапі. Таким чином, сценарії користувацької взаємодії є критично важливим інструментом на етапі системного проектування.

2.3 Архітектурне моделювання застосунку з компонентом ШІ

Архітектурне моделювання системи дозволяє формалізувати основні компоненти мобільного застосунку, визначити їх функціональне призначення та взаємозв’язки. Побудова архітектури базується на принципах модульності, розподіленої обробки даних і зовнішньої інтеграції з сервісами штучного інтелекту та базами даних. Це забезпечує масштабованість, гнучкість та можливість незалежної розробки окремих частин системи.

Загальна структура застосунку передбачає поділ на три рівні: мобільний клієнт, серверна частина (Backend API) та зовнішні сервіси. На рівні клієнта зосереджена логіка взаємодії з користувачем: формування запитів, відображення результатів, збір введених даних. На серверному рівні обробляються бізнес-процеси, взаємодія з OpenAI та Firebase Firestore. Зовнішні сервіси включають OpenAI API для генерації слів і перевірки речень, Firebase для зберігання та Auth0 для автентифікації. Загальний вигляд архітектури представлено нижче (рисунок 2.5).

Мобільний застосунок містить декілька основних менеджерів: менеджер словників, менеджер практики, менеджер автентифікації, а також інтерфейс користувача. Менеджер словників відповідає за ініціацію генерації нових наборів слів, надсилання параметрів на сервер та отримання

сформованих колекцій. Менеджер практики працює з введеними реченнями користувача, надсилає їх на перевірку та отримує зворотний зв’язок. Менеджер автентифікації керує токенами доступу через Auth0.

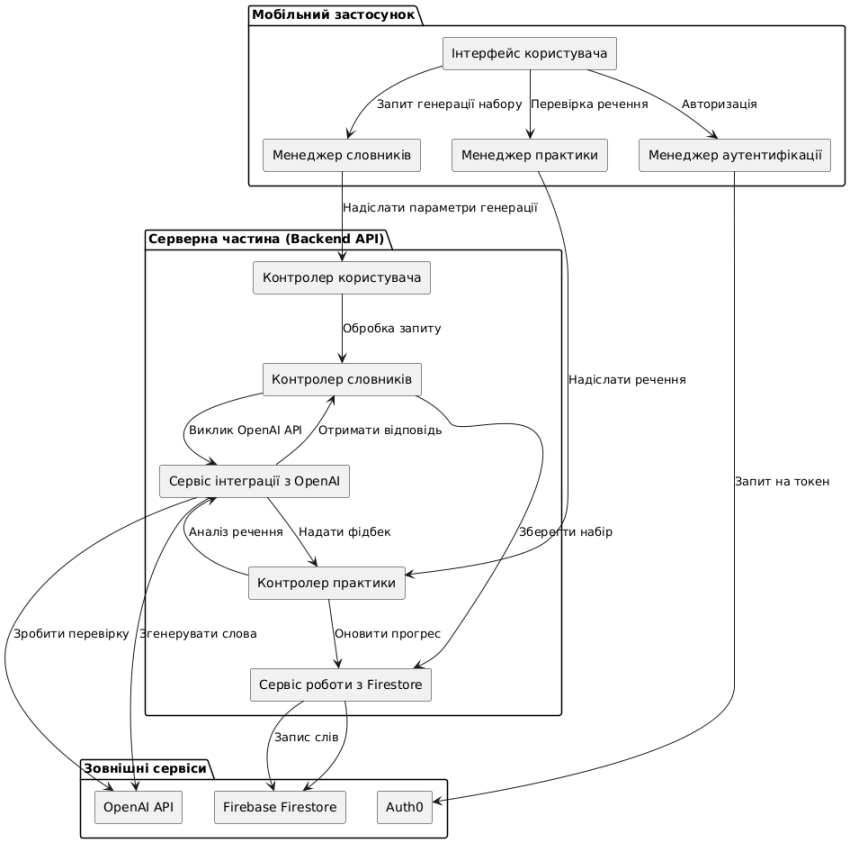

Рисунок 2.5 – Архітектурна діаграма компонентів застосунку

Серверна частина поділяється на контролери (користувача, словників, практики) та сервісні модулі. Контролери обробляють запити з клієнта, перевіряють авторизацію, формують структуру запитів до внутрішніх сервісів і передають результати назад. Вони не містять складної логіки – вона реалізується у сервісах, які відповідають за інтеграцію з OpenAI та Firestore. Наприклад, сервіс інтеграції з OpenAI приймає параметри генерації, звертається до API, а потім формує відповідь у зручному для клієнта форматі.

Сервіс OpenAI працює в двох напрямах: генерація словникових наборів та перевірка речень у практичному режимі. У першому випадку він

формує prompt для генерації лексики за заданими критеріями, у другому – передає речення для аналізу контексту, стилістики та граматики. Усі відповіді OpenAI обробляються сервером, щоб уніфікувати формат і виключити помилки перед показом користувачу.

Збереження даних реалізовано через окремий сервіс Firestore, який відповідає за взаємодію з хмарною базою Firebase. Усі словникові набори, речення, результати перевірки та прогрес користувача зберігаються під його унікальним ідентифікатором. Це дозволяє забезпечити відновлення навчального контексту після перезавантаження або зміни пристрою, а також забезпечити розмежування доступу між користувачами.

Зовнішні сервіси відіграють роль окремих вузлів, що мають чітке функціональне призначення. Наприклад, OpenAI відповідає за генеративну логіку, Firestore – за зберігання та читання даних, а Auth0 – за безпечну аутентифікацію. Завдяки цьому архітектура застосунку залишається чистою, а система – розширюваною. Наприклад, при необхідності можна замінити OpenAI на інший генератор або перенести базу на альтернативне рішення без істотної перебудови логіки.

Таким чином, побудована архітектура підтримує чіткий поділ відповідальностей, забезпечує логічну ізоляцію рівнів і модулів, та створює основу для стабільної і масштабованої реалізації мобільного застосунку з підтримкою генеративного штучного інтелекту. Вона також враховує можливість подальшого розширення – зокрема, впровадження нових мов, типів вправ або додаткових механізмів перевірки.

2.4 Концепція структури динамічного словника

Структура динамічного словника є ключовим компонентом застосунку, що визначає, як саме зберігаються, організовуються та використовуються лексичні одиниці. На відміну від фіксованих словників, які мають жорстко заданий набір слів, динамічний словник формується

індивідуально для кожного користувача, залежно від його запиту, мовного рівня, тематики та прогресу. Завдяки цьому підхід стає персоналізованим, що підвищує ефективність навчання і зацікавленість користувача.

Основною одиницею словника є слово, яке представлено як структура з кількох полів. До цієї структури входять: англійське написання слова, його український переклад, коротке англомовне пояснення (definition), приклад вживання у реченні, список синонімів, а також поле з аудіофайлом для прослуховування вимови. Кожне слово також має унікальний ідентифікатор, дату створення, статус вивчення (нове, вивчене, складне), та поле для зберігання індивідуальних позначок користувача.

Словник організовано у вигляді набору – колекції, яка створюється в результаті генерації за допомогою OpenAI API. Колекція містить список слів, пов’язаних спільною темою або контекстом. Наприклад, набір на тему «подорожі» включатиме лексику, яка зустрічається у відповідних життєвих ситуаціях. Це дозволяє формувати логічно зв’язані групи слів, які легше запам’ятовуються. Кожна колекція зберігається окремо і може бути доступна для редагування або перегляду.

Оскільки словники динамічні, важливо передбачити механізм оновлення. Якщо користувач додає слово вручну або генерує нову версію колекції з цієї ж теми, система повинна вміти об’єднувати, оновлювати або створювати нові екземпляри без втрати вже засвоєного прогресу. Для цього передбачено поле версійності, а також флаг, що позначає, чи є слово унікальним для кількох наборів.

Кожне слово має зв’язок із користувачем через окрему сутність – таблицю прогресу. У ній зберігається інформація про те, коли останній раз слово було переглянуто, скільки разів воно було повторено, у яких реченнях використано, та з яким результатом проходили практичні вправи. Така структура дозволяє системі адаптувати повторення слів на основі реальної активності користувача.

З технічної точки зору структура словника реалізована у форматі документів у Firestore. Кожна колекція є окремим документом, що містить масив слів з вкладеними об’єктами. Всі зміни, додавання або оновлення виконуються транзакційно, що дозволяє уникнути втрати даних або конфліктів при паралельній роботі. Враховуючи обмеження NoSQL-моделі, усі вкладені структури уніфіковані за схемою, щоб забезпечити узгодженість при обробці запитів з клієнта.

Динамічна структура також дозволяє будувати рекомендаційні модулі. Наприклад, система може пропонувати користувачеві слова з інших тем, які є близькими до вже засвоєних, або автоматично включати слова, які часто повторюються у практичних реченнях. Це створює додатковий адаптивний шар, коли словник живе і змінюється разом із прогресом користувача.

Таким чином, концепція динамічного словника базується на персоналізації, гнучкості та контекстній релевантності. Вона дозволяє забезпечити навчання у зручній і логічній формі, підтримує інтеграцію з іншими модулями системи (генерація, практика, статистика) та створює фундамент для масштабування застосунку без втрати якості контенту.

2.5 Проектування хмарної моделі зберігання даних

Зберігання даних у мобільному застосунку з використанням генеративного штучного інтелекту потребує побудови гнучкої, масштабованої та доступної структури. Оскільки застосунок орієнтований на велику кількість користувачів, кожен з яких має індивідуальний словниковий простір, було обрано хмарну модель зберігання даних на основі Firestore – масштабованої NoSQL-бази даних, що є частиною екосистеми Google Firebase.

Firestore дозволяє зберігати дані у вигляді колекцій і документів. Кожен користувач має свою власну колекцію, всередині якої розміщено

документи зі словниковими наборами, інформацією про прогрес, практичні речення, та інші навчальні дані. Такий підхід забезпечує логічне групування даних та дозволяє реалізувати ефективні фільтри, запити та сортування за допомогою вбудованого API.

Кожен словниковий набір зберігається як окремий документ, у якому міститься масив об’єктів слів. Ці об’єкти включають не лише основну лексичну інформацію, а й метадані: дату створення, статус вивчення, рейтинг складності, зв’язок із практичними реченнями. Додатково кожен набір містить параметри генерації, що були задані при його створенні, що дозволяє легко повторити запит або адаптувати його для нової генерації.

Прогрес користувача зберігається окремо від словника – у колекції progress, де кожне слово має унікальний запис. Такий підхід дає змогу централізовано оцінювати навчальну динаміку, формувати звіти, надавати рекомендації щодо повторення. Система також може формувати історію активності: дата останнього перегляду, кількість повторень, результат у практичному режимі. Це дозволяє реалізувати механізм інтервального повторення без зовнішніх трекерів.

Практичні речення, які користувач створює під час вправ, зберігаються в окремій колекції practice_results. Кожен запис містить текст речення, ідентифікатор пов’язаного слова, дату створення, та зворотний зв’язок, отриманий від OpenAI. Це дає змогу не лише зберігати приклади для подальшого аналізу, а й формувати рекомендації для користувача, який хоче переглянути власні помилки або повернутись до складних тем.

Загалом, модель зберігання на базі Firestore дозволяє реалізувати персоналізоване навчання в масштабованому середовищі. Вона підтримує структуру динамічного словника, історію практики, гнучкий облік прогресу і захищений доступ. Така архітектура добре поєднується з іншими компонентами застосунку та створює основу для стабільної та ефективної роботи системи незалежно від кількості користувачів.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Опис застосованих технологій у застосунку

Для реалізації мобільного застосунку було використано сучасний технологічний стек, орієнтований на продуктивність, масштабованість та інтеграцію з хмарними сервісами. Архітектура побудована на принципі поділу відповідальностей між клієнтською частиною, серверною логікою та зовнішніми інтелектуальними сервісами. Такий підхід забезпечує гнучкість у розробці, легкість у розширенні функціоналу та стабільну роботу в умовах зростання кількості користувачів. Також дає змогу легко оновлювати окремі модулі, інтегрувати нові функції та підтримувати високу продуктивність незалежно від кількості користувачів.

Клієнтська частина застосунку реалізована з використанням компонентного фреймворку, який дозволяє створювати адаптивні інтерфейси з високою швидкістю взаємодії. Використання сучасних бібліотек інтерфейсу, маршрутизації та форм сприяє гнучкому проектуванню навчальних сценаріїв і зручному керуванню станом користувацьких даних. Аутентифікація користувачів реалізована через зовнішній сервіс, що забезпечує високий рівень безпеки без необхідності самостійного обслуговування облікових записів.

Серверна частина побудована з використанням фреймворку, який орієнтований на масштабовану розробку вебсервісів. Вона обробляє запити клієнта, взаємодіє з базою даних, а також забезпечує інтеграцію із зовнішнім генеративним API для створення навчального контенту. Уся система розгорнута у хмарному середовищі з використанням контейнеризації, що дає змогу досягти високої гнучкості у розгортанні, оновленні та обслуговуванні застосунку.

3.1.1 Мова програмування

У якості основної мови програмування для реалізації як клієнтської, так і серверної частини застосунку було обрано TypeScript [4]. Це надбудова над JavaScript, яка додає статичну типізацію та дозволяє виявляти помилки ще на етапі компіляції. Завдяки цьому розробка стає більш контрольованою, а код – зрозумілішим і надійнішим у масштабному проєкті з багатьма модулями.

Однією з ключових переваг TypeScript є підтримка суворої типізації та можливість явно описувати структуру об’єктів, функцій і компонентів. У контексті застосунку для вивчення лексики це особливо корисно при роботі з моделями словникових наборів, структурами прогресу користувача, параметрами генерації та результатами зворотного зв’язку. Завдяки типам зменшується ймовірність помилок при передачі даних між клієнтом, сервером і зовнішніми API.

Ще одним аргументом на користь TypeScript є його сумісність із сучасними бібліотеками та фреймворками, що використовуються у проекті. Мова повністю підтримується як React [5], так і Nest.js [6], що дозволяє писати єдиний уніфікований код як для фронтенду, так і для бекенду. Це також спрощує командну роботу, оскільки одна мова дозволяє уникати проблем при інтеграції різних частин системи.

Застосування TypeScript також покращує підтримку автодоповнення, рефакторингу та навігації по коду в середовищі розробки. Це підвищує продуктивність і зменшує витрати часу на пошук і усунення помилок. У довгостроковій перспективі використання цієї мови сприяє масштабуванню проєкту, полегшує адаптацію нових учасників команди та покращує читаність коду, що є критично важливим у складних системах зі складною логікою.

3.1.2 Фреймворки та бібліотеки

У клієнтській частині застосунку було використано фреймворк React, який дозволяє створювати швидкі та інтерактивні інтерфейси на основі компонентної архітектури. Його гнучкість та активна екосистема роблять його одним із найпопулярніших інструментів для побудови односторінкових застосунків. Завдяки React розробка інтерфейсу стала модульною, що дозволяє легко масштабувати, повторно використовувати та тестувати окремі частини UI.

Для побудови стилізованих і адаптивних компонентів було застосовано бібліотеку Mantine UI [7]. Вона забезпечує готові компоненти для створення форм, карток, таблиць, вкладок, а також підтримує теми, модальні вікна й інші важливі для користувача елементи. Використання цієї бібліотеки дозволило суттєво скоротити час на верстку та зосередитись на логіці навчального процесу.

Для управління маршрутизацією у клієнтському застосунку використовується бібліотека React Router, яка дає змогу створювати навігацію між екранами без перезавантаження сторінки. Це забезпечує плавний користувацький досвід і дозволяє організувати логічну структуру вкладених маршрутів, необхідних для розмежування режимів генерації, вивчення, практики та перегляду статистики.

Також активно використовується бібліотека react-hook-form, яка спрощує роботу з формами, включно з валідацією, відстеженням стану полів та обробкою помилок. Це особливо актуально для функціоналу генерації словникових наборів, де користувач обирає параметри теми, складності та кількості слів. Усі ці бібліотеки тісно інтегруються одна з одною, формуючи потужний клієнтський стек, який дозволяє створити зручний, динамічний та масштабований інтерфейс.

3.2 Програмна реалізація застосунку

Програмна реалізація мобільного застосунку передбачає створення повноцінного функціонального середовища для вивчення лексики, що включає інтерфейс користувача, логіку обробки даних і взаємодію зі сторонніми сервісами. Уся система побудована з урахуванням вимог до зручності, продуктивності та розширюваності. Особливу увагу приділено забезпеченню коректної взаємодії між компонентами.

Застосунок реалізовано з використанням сучасних вебтехнологій та підходів до проектування. Компонентна структура клієнтської частини, розділення серверної логіки та хмарна інфраструктура дозволили створити масштабоване рішення з чітко визначеною структурою. У межах реалізації були враховані сценарії генерації, перегляду, вивчення, повторення та перевірки слів у контексті.

У наступних підрозділах буде детально описано реалізацію ключових екранів, логіку обробки запитів, роботу з даними користувача та інтеграцію з OpenAI API [8] для формування динамічного навчального контенту. Такий підхід дозволяє гнучко керувати словниковими наборами, персоналізувати процес навчання та забезпечувати користувачеві інтуїтивно зрозумілий досвід взаємодії із системою.

3.2.1 Програмна реалізація аутентифікації

Функціональність створення нового користувача реалізована у вигляді асинхронного методу, який спочатку завантажує аватар користувача до хмарного сховища Google Cloud [9]. Далі здійснюється перевірка наявності користувача з вказаним ідентифікатором auth0Id у базі даних Firestore [10]. Якщо такого користувача не існує, транзакцією створюється новий запис із відповідними полями, як показано у лістингу 3.1.

Лістинг 3.1 – Програмний код, що створює нового користувача в базі Firestore після завантаження його аватару до Google Cloud Storage, за умови, що користувач із таким auth0Id ще не існує

```
export class AuthService {
  constructor(private firebaseService: any) {}

  async createAuth0User({ auth0Id, email, name, avatar }: any): Promise<void> {
    const db = this.firebaseService.getFirestore();
    const ref = db.collection('users').doc();
    const buffer = await (await fetch(avatar)).arrayBuffer();

    const fileName = `avatars/${Buffer.from(email).toString('base64')}.jpg`;
    const fileUrl = `https://storage.googleapis.com/bucket/${fileName}`;
    await getStorage().bucket().file(fileName).save(Buffer.from(buffer), { public: true }
);

    await firestore().runTransaction(async (tx) => {
      const existing = await tx.get(firestore().collection('users').where('auth0Id', '==', auth0Id));
      if (!existing.empty) throw new Error('User already exists');
      tx.create(ref, { auth0Id, email, name, avatar: fileUrl, collections: [], advancedPracticeMode: false });
    });
  }
}
```

3.2.2 Програмна реалізація API

У лістингу 3.2 представлено програмний код, що відповідає за ініціалізацію клієнта OpenAI за допомогою API-ключа, який зчитується з конфігурації або змінних середовища. Такий підхід дозволяє гнучко адаптувати застосунок до різних середовищ виконання без жорсткого зв’язування з конкретними значеннями. Об’єкт OpenAI створюється один раз при ініціалізації сервісу, що підвищує ефективність повторного використання.

Лістинг 3.2 – Програмний код, що ініціалізує клієнт OpenAI з використанням ключа API, збереженого у змінних середовища, і надає до нього доступ через сервіс

```
export class OpenAIService {
  private openAI;
  constructor(config: any) {
    this.openAI = new OpenAI({ apiKey:
config.get('OPENAI_API_KEY') || process.env.OPENAI_API_KEY });
  }

  getOpenAI() {
    return this.openAI;
  }
}
```

Метод getOpenAI надає доступ до ініціалізованого екземпляра клієнта OpenAI з інших частин програми. Завдяки цьому інтерфейси генерації слів і перевірки речень можуть використовувати мовну модель без необхідності повторної ініціалізації. Структура сервісу є простою, але критично важливою для інтеграції штучного інтелекту в навчальний процес застосунку.

3.2.3 Програмна реалізація користувача

У лістингу 3.3 наведено визначення типу користувача, що використовується для формалізації структури даних у системі. Тип містить основні поля ідентифікації, зокрема auth0Id, а також інформацію про ім’я, електронну пошту та аватар. Крім того, визначено масив collections для збереження словникових наборів і булеве поле advancedPracticeMode, яке вказує на активність розширеного режиму практики.

Лістинг 3.3 – Програмний код, що описує структуру типу користувача, який містить ідентифікатори, персональні дані, аватар, словникові колекції та налаштування режиму практики

```
export type User = {
  id: string;
  auth0Id: string;
  name: string;
  email: string;
  avatar: string;
  collections: any[];
  advancedPracticeMode: boolean;
};
```

У лістингу 3.4 подано код сервісу, що забезпечує пошук користувача в базі Firestore за унікальним ідентифікатором auth0Id, а також оновлення його налаштувань. Метод getUserByAuth0Id повертає об’єкт користувача, якщо відповідний запис існує, і null у протилежному випадку. Метод updateSettings виконує оновлення поля advancedPracticeMode, що дає змогу користувачеві змінювати режим практики у своїх персональних налаштуваннях.

Лістинг 3.4 – Програмний код, що реалізує сервіс для отримання користувача за auth0Id та оновлення його налаштувань у базі Firestore.

```
export class UserService {
  constructor(private firebase: any) {}

  async getUserByAuth0Id(id: string): Promise<any | null>
{
    const snap = await
this.firebase.getFirestore().collection('users').where('auth0I
d', '==', id).limit(1).get();
    return snap.empty ? null : { id: snap.docs[0].id,
...snap.docs[0].data() };
  }

  async updateSettings({ auth0Id, advancedPracticeMode }:
{ auth0Id: string; advancedPracticeMode: boolean }) {
    const snap = await
this.firebase.getFirestore().collection('users').where('auth0I
d', '==', auth0Id).limit(1).get();
    if (!snap.empty) await snap.docs[0].ref.update({
advancedPracticeMode });
  }
}
```

3.2.4 Програмна реалізація словника

У лістингу 3.5 описано типи даних WordsCollection і Word, які використовуються для зберігання словникових наборів у застосунку. Кожна колекція містить ідентифікатор, назву та масив слів, що представлені типом Word. Структура слова включає базову лексику, переклад, список синонімів, приклади вживання та пояснення значення, що дозволяє повноцінно формувати навчальний контент.

Лістинг 3.5 – Програмний код, що описує типи даних для словникової колекції та окремого слова, включаючи переклад, синоніми, приклади вживання та пояснення

```
export type WordsCollection = {
  id: string;
  name: string;
  words: Word[];
};
export type Word = {
  id: string;
  name: string;
  translation: string;
  synonyms: string[];
  examples: string[];
  explanation: string;
};
```

У лістингу 3.6 подано реалізацію сервісу WordService, який відповідає за основну логіку роботи зі словниковими колекціями користувача. Метод getWordsCollections здійснює пошук колекцій у базі даних Firestore за ідентифікатором користувача auth0Id. Метод generateWordsCollection створює нову колекцію слів, формуючи запит до OpenAI на основі заданої теми, рівня, акценту та приміток, після чого результат зберігається в колекції користувача.

Окрему функцію виконує метод verifyExampleSentence, який перевіряє введене користувачем речення на коректне використання заданого слова. Для цього формується відповідний запит до мовної моделі OpenAI з урахуванням пояснення слова та активності режиму розширеної практики. Отримана відповідь аналізується у форматі JSON та використовується для надання зворотного зв’язку в інтерфейсі практики.

Лістинг 3.6 – Програмний код, що реалізує сервіс роботи зі словниковими колекціями: отримання колекцій користувача, генерація нової колекції через OpenAI, перевірка прикладів речень та збереження результатів

```
export class WordService {
  constructor(private openai: any, private firebase: any)
  async getWordsCollections({ auth0Id }: { auth0Id: string)
    const snap = await
this.firebase.getFirestore().collection('users').where('auth0I
d', '==', auth0Id).limit(1).get();
    if (snap.empty) throw new Error('User not found');
    return (snap.docs[0].data() as any).collections;
  }

  async generateWordsCollection({ auth0Id, accent, level,
notes, topic }: any) {
    const userSnap = await
this.firebase.getFirestore().collection('users').where('auth0I
d', '==', auth0Id).limit(1).get();
    if (userSnap.empty) throw new Error('User not found');
    const user = { id: userSnap.docs[0].id,
...userSnap.docs[0].data() };
    const prompt = `Generate a JSON with words for topic
${topic}, level ${level}, accent ${accent}. Notes: ${notes}.`;
    const res = await
this.openai.getOpenAI().chat.completions.create({
      model: 'gpt-4o',
      messages: [{ role: 'user', content: prompt }],
    });
    const json =
JSON.parse(res.choices[0].message.content.replace(/```json|```
/g, ''));
    const collection = { ...json.collection, id: v4(),
words: json.collection.words.map(w => ({ ...w, id: v4() })) };
```

Продовження лістингу 3.6

```
    this.firebase.getFirestore().collection('users').doc(user.
id).update({
          collections: [...user.collections, collection],
        });

        return collection;
      }

      async verifyExampleSentence({ auth0Id, sentence,
collectionId, wordId }: any) {
        const snap = await
this.firebase.getFirestore().collection('users').where('auth0I
d', '==', auth0Id).limit(1).get();
        if (snap.empty) throw new Error('User not found');
        const user = snap.docs[0].data();
        const word = user.collections.find((c: any) => c.id ===
collectionId)?.words.find((w: any) => w.id === wordId);
        if (!word) throw new Error('Word not found');

        const prompt = `Check if the word "${word.name}" with
meaning "${word.explanation}" is used properly in: ${sentence}.
Advanced: ${user.advancedPracticeMode}`;
        const res = await
this.openai.getOpenAI().chat.completions.create({
          model: 'gpt-4o',
          messages: [{ role: 'user', content: prompt }],
        });

        return
JSON.parse(res.choices[0].message.content.replace(/```json|```
/g, ''));
      }
    }
```

3.2.5 Програмна реалізація інтерфейсу головного екрану

У лістингу 3.7 представлено реалізацію головного екрана користувача, на якому відображаються всі наявні словникові колекції. Кожна колекція представлена у вигляді інтерактивної картки з назвою, кількістю слів та індикатором прогресу, а також з можливістю переходу до детального перегляду. Крім того, у верхній частині сторінки передбачено кнопку для створення нової колекції, що забезпечує швидкий доступ до відповідної функціональності.

Лістинг 3.7 – Програмний код, що реалізує інтерфейс головного екрана користувача для перегляду та створення словникових колекцій

```
export const WordsCollectionsListPage = () => {
  const navigate = useNavigate();
  const { data: collections, isLoading, isError } = useGetCollectionsQuery();

  return (
    <>
      <PageHeader
        title="Collections"
        rightSection={<ActionIcon onClick={() => navigate("/collections/new")}><IconPlus size={24} /></ActionIcon>}
      />
      <PageBody>
        {isLoading ? (
          <LoadingOverlay visible />
        ) : isError || !collections ? (
          <>Error</>
        ) : collections.length === 0 ? (
          <Text size="xl" c="dimmed" ta="center">Empty</Text>
```

Продовження лістингу 3.7

```
            collections.map((c) => (
              <Paper      key={c.id}      onClick={()      =>
navigate(`/collections/${c.id}`)} withBorder>
                <Group justify="space-between">
                  <Stack>
                    <Text>{c.name}</Text>
                    <Group>
<Text size="xs">{c.words.length} words</Text>
                      <Progress value={randomInt(10, 90)} />
                    </Group>
                  </Stack>
<ActionIcon><IconArrowRight size={16} /></ActionIcon>
                </Group>
              </Paper>
            ))
          )}
        </PageBody>
      </>
    );
};
```

3.2.6 Програмна реалізація інтерфейсу створення словника

У лістингу 3.8 реалізовано інтерфейс сторінки створення нового словникового набору, який передбачає введення теми, рівня володіння мовою, акценту та додаткових приміток. Форма побудована з використанням бібліотеки react-hook-form і валідується за допомогою схеми, що дозволяє забезпечити коректність введених даних. Після заповнення форми та підтвердження натисканням кнопки Create запускається процедура генерації колекції, яка після успішного створення відкривається у відповідному інтерфейсі.

Лістинг 3.8 – Програмний код, що реалізує інтерфейс створення нового словникового набору

```
export const CreateCollectionPage = () => {
  const navigate = useNavigate();
  const [generateCollection, { isLoading }] = useGenerateCollectionMutation();
  const { control, handleSubmit, formState: { errors } } = useForm({
    resolver: zodResolver(schema),
    defaultValues: { topic: "", level: null, accent: null, notes: "" },
  });

  const onSubmit = async (data: FormState) => {
    const res = await generateCollection(data).unwrap();
    if (res.id) navigate(`/collections/${res.id}`);
  };

  return (
    <>
      <PageHeader title="New Collection" leftSection={<BackButton onClick={() => navigate("/collections")} />} />
      <PageBody>
        <Stack p="md">
          <Controller name="topic" control={control} render={({ field }) => (
            <Input.Wrapper label="Topic" error={errors.topic?.message}>
              <Input placeholder="Business-related" {...field} readOnly={isLoading} />
            </Input.Wrapper>
          )} />
          <Controller name="notes" control={control} render={({ field }) => (
```

Продовження лістингу 3.8

```
                <Textarea       label="Additional       notes"
placeholder="Include more phrasal verbs"
               {...field} />
               )} />
               <Button         onClick={handleSubmit(onSubmit)}
loading={isLoading}>
              Create</Button>
             </Stack>
           </PageBody>
         </>
       );
     };
```

3.2.7 Програмна реалізація інтерфейсу перегляду словника

У лістингу 3.9 представлено реалізацію інтерфейсу перегляду словникової колекції, що дозволяє користувачеві ознайомитися зі списком доступних слів, їх короткими поясненнями та перейти до режиму навчання. Кожне слово відображається у вигляді окремої картки, і при натисканні на неї відкривається детальний перегляд із додатковою інформацією, такою як переклад, синоніми та приклади вживання. Це забезпечує швидку навігацію по контенту та дає змогу гнучко обирати, з яким матеріалом працювати.

Інтерфейс адаптується залежно від стану вибору слова: у разі неактивного вибору відображається загальний список, а при відкритті слова відобраєтся деталізована інформація. Крім того, реалізовано кнопку для переходу до навчального режиму по обраній колекції, що інтегрує цей екран у загальну логіку навчального процесу. Такий підхід підвищує зручність користування застосунком та дозволяє ефективно поєднувати перегляд і навчання.

Лістинг 3.9 – Програмний код, що реалізує інтерфейс перегляду словникової колекції

```
export const WordsCollectionPage = () => {
  const navigate = useNavigate();
  const { id } = useParams<{ id: string }>();
  const [selectedWord, setSelectedWord] = useState<Word | null>(null);
  const { data: collection, isLoading, isError } = useGetCollectionQuery(id!);

  if (isLoading) return <LoadingOverlay visible />;
  if (isError || !collection) return <Text>Error</Text>;

  const header = (
    <PageHeader
      title={collection.name}
      leftSection={<BackButton onClick={() => selectedWord ? setSelectedWord(null) : navigate("/collections")} />}
      rightSection={<ActionIcon><IconPlus size={24} /></ActionIcon>}
    />
  );
  return (
    <>
      {header}
      <PageBody>
        {selectedWord ? (
          <Stack p="md">
            <Title ta="center">{selectedWord.name}</Title>
            <Paper><Text>Explanation: {selectedWord.explanation}</Text></Paper>
            <Paper><Text>Translation: {selectedWord.translation}</Text></Paper>
            <Paper><Text>Synonyms: {selectedWord.synonyms.join(", ")}</Text></Paper>
```

Продовження лістингу 3.9

```
                <Paper>
                  <Text>Examples:</Text>
                  <List type="ordered">
                    {selectedWord.examples.map((ex,    i)    =>
<List.Item key={i}>{ex}</List.Item>)}
                  </List>
                </Paper>
              </Stack>
            )
            :
            (
              <Stack p="md">
                <Progress value={randomInt(10, 90)} />
                <Button             onClick={()             =>
navigate(`/learn/${collection.id}`)}>Learn</Button>
                {collection.words.map((word) => (
                  <Paper    key={word.id}    onClick={()    =>
setSelectedWord(word)}>
                    <Text>{word.name}</Text>
                    <Text                              size="xs"
c="dimmed">{word.explanation}</Text>
                  </Paper>
                )
               )
              }
              </Stack>
            )
           }
          </PageBody>
        </>
      );
    };
```

3.2.8 Програмна реалізація інтерфейсу вивчення слів

У лістингу 3.10 показано реалізацію інтерфейсу, що відповідає за вибір режиму навчання в застосунку. Користувачеві пропонуються три основні варіанти: розпочати нове навчання, повторити раніше вивчений матеріал або самостійно обрати колекцію для тренування. Усі колекції автоматично групуються відповідно до їх позиції у списку, що дозволяє рівномірно розподілити доступний контент між категоріями «To learn», «To revise» і «Last learned», при цьому кожна група представлена у вигляді карток зі статистикою та індикатором прогресу.

Лістинг 3.10 – Програмний код, що реалізує інтерфейс вибору режиму навчання в застосунку

```
export const LearnPage = () => {
  const navigate = useNavigate();
  const { data: collections, isLoading, isError } = useGetCollectionsQuery();
  if (isLoading) return <LoadingOverlay visible />;
  if (isError || !collections) return <Text>Error</Text>;
  const pickEveryNth = (arr, n, i = 0) => arr.filter((_, idx) => (idx - i) % n === 0 && idx >= i);
  const learn = pickEveryNth(collections, 3, 0);
  const revise = pickEveryNth(collections, 3, 1);
  const last = pickEveryNth(collections, 3, 2);
  return (
      <PageHeader title="Learn" />
      <PageBody>
        <Stack p="md">
          <Stack>
            <Button onClick={() => navigate(`/learn/${learn[0]?.id}`)}>Learn</Button>
            {revise[0] && <Button onClick={() => navigate(`/learn/${revise[0].id}`)}>Revise</Button>}
```

Продовження лістингу 3.10

```
                <Button              onClick={()              =>
navigate("/learn/pick")}>Pick</Button>
              </Stack>
              {[{ label: "To learn", list: learn }, { label:
"To revise", list: revise }, { label: "Last learned", list:last}]
                .filter((g) => g.list.length)
                .map((group) => (
                  <Card key={group.label} p="md">
                    <Text fw={500}>{group.label}</Text>
                    <Stack>
                      {group.list.map((col) => (
                        <Paper  key={col.id}  onClick={()  =>
navigate(`/learn/${col.id}`)} withBorder>
                          <Group justify="space-between">
                            <Stack>
                              <Text>{col.name}</Text>
                              <Group>
                                <Text
size="xs">{col.words.length} words</Text>
                          <Progress value={randomInt(10, 90)} />
                              </Group>
                            </Stack>
                          </Group>
                        </Paper>
                      ))}
                    </Stack>
                  </Card>
                ))}
            </Stack>
          </PageBody>
        </>
      );
    };
```

У лістингу 3.11 представлено інтерфейс вибору словникової колекції, який дозволяє користувачеві самостійно обрати набір слів для проходження навчання. Усі наявні колекції відображаються у вигляді інтерактивних карток, що містять назву, кількість слів та індикатор прогресу, а також кнопку для переходу до відповідного навчального режиму. Такий підхід забезпечує гнучкість у навчальному процесі та дозволяє користувачеві орієнтуватися в контенті відповідно до власних потреб.

Лістинг 3.11 – Програмний код, що реалізує інтерфейс вибору словникової колекції для навчання

```
export const PickCollectionPage = () => {
  const navigate = useNavigate();
  const { data: collections, isLoading, isError } = useGetCollectionsQuery();

  const header = <PageHeader title="Pick collection" leftSection={<BackButton onClick={() => navigate("/learn")} />} />;

  if (isLoading) return <>{header}<PageBody><LoadingOverlay visible /></PageBody></>;
  if (isError || !collections) return <>{header}<PageBody>Error</PageBody></>;

  return (
    <>
      {header}
      <PageBody>
        <Stack p="md">
          {collections.map((c) => (
            <Paper key={c.id} onClick={() => navigate(`/learn/${c.id}`)} withBorder shadow="md" bg="dark.6">
              <Group justify="space-between">
                <Stack>
```

Продовження лістингу 3.11

```
                        <Group>
                          <Text                          size="xs"
c="dimmed">{c.words.length} words</Text>
                          <Progress value={randomInt(10, 90)} />
                        </Group>
                      </Stack>
                      <ActionIcon          radius="xl"><IconBook
size={16} /></ActionIcon>
                    </Group>
                  </Paper>
              </Stack>
            </PageBody>
        );
      };
```

3.2.9 Програмна реалізація інтерфейсу практики

У лістингу 3.12 реалізовано інтерфейс режиму практики, в якому користувач покроково опрацьовує слова зі словникової колекції. На кожному кроці користувач може переглянути детальну інформацію про слово (пояснення, переклад, синоніми та приклади) або перейти безпосередньо до практики. У режимі практики користувач вводить приклад речення з використанням слова, після чого за допомогою OpenAI здійснюється автоматична перевірка правильності вживання.

Система генерує відповідь у вигляді зворотного зв’язку, який враховує контекст або складність речення, залежно від увімкненого режиму. Після перевірки користувач може перейти до наступного слова або повторно ознайомитися з деталями поточного. Така інтерактивна взаємодія з лексикою дозволяє не лише вивчати нові слова, а й одразу закріплювати їх через контекст, що підвищує ефективність засвоєння.

Лістинг 3.12 – Програмний код, що реалізує інтерфейс режиму практики у процесі вивчення слів

```
export const LearnCollectionView = ({ collection }) => {
  const [index, setIndex] = useState(0);
  const [sentence, setSentence] = useState("");
  const [mode, setMode] = useState("initial");
  const [verify, { isLoading }] = useVerifySentenceMutation();
  const [result, setResult] = useState(null);
  const word = collection.words[index];
  if (index >= collection.words.length)
    return (
      <Stack align="center" p="md">
        <Title>Congratulations!</Title>
        <Text>You have learned all the words</Text>
        <Button onClick={() => navigate("/learn")}>Learn more</Button>
      </Stack>
    );
      {mode === "details" && (
        <Stack>
          <Title ta="center">{word.name}</Title>
          <Text>Explanation: {word.explanation}</Text>
          <Text>Translation: {word.translation}</Text>
          <Text>Synonyms: {word.synonyms.join(", ")}</Text>
          <Text>Examples: {word.examples.join("; ")}</Text>
          <Group>
            <Button onClick={() => setMode("practice")}>Practice</Button>
            <Button onClick={next}>Next</Button>
          </Group>
        </Stack>
      )}
```

Продовження лістингу 3.12

```
        {mode === "practice" && (
          <Stack>
            <Title ta="center">{word.name}</Title>
            <Textarea
              value={sentence}
              onChange={(e)                                    =>
setSentence(e.currentTarget.value)}
              label="Example sentence"
              required
            />
            {result && (
              result.isCorrect ? (
                <Alert   color="green"   title="Correct">Well
done!</Alert>
              ) : (
                <Alert color="red" title="Incorrect">
                  {result.reason === "simplicity" ? "Sentence
is too simple" : result.explanation}
                </Alert>
              )
            )}
            <Button                          onClick={submit}
loading={isLoading}>Verify</Button>
            <Group>
              <Button             onClick={()             =>
setMode("details")}>Show</Button>
              <Button onClick={next}>Next</Button>
            </Group>
          </Stack>
        )}
      </Stack>
    );
  };
```

3.2.10 Програмна реалізація інтерфейсу профіля користувача

У лістингу 3.13 показано реалізацію інтерфейсу сторінки профілю користувача, яка відображає основну інформацію про поточного користувача, включаючи аватар, ім’я та електронну пошту. Також передбачені кнопки для переходу до сторінки налаштувань, умовної секції безпеки та виходу з облікового запису через Auth0. Завдяки компактному компонуванню та логічній структурі сторінка забезпечує швидкий доступ до персональних параметрів та керування сесією.

Лістинг 3.13 – Програмний код, що реалізує інтерфейс сторінки профілю користувача

```
export const AccountPage = () => {
  const { logout } = useAuth0();
  const user = useCurrentUser();
  const navigate = useNavigate();

  return (
    <>
      <PageHeader title="Account" />
      <PageBody>
        <Stack p="md">
          <Paper withBorder shadow="md">
            <Group>
              <Avatar src={user.avatar} size={70} />
              <Stack gap={0}>
                <Text size="lg">{user.name}</Text>
                <Text c="dimmed">{user.email}</Text>
              </Stack>
            </Group>
          </Paper>
          <Button onClick={() => navigate("/settings")}
leftSection={<IconSettings />}>Settings</Button>
```

Продовження лістингу 3.13

```
                <Button           leftSection={<IconShieldFilled
/>}>Security</Button>
                <Button
                  color="red"
                  onClick={()   =>   logout({   logoutParams:   {
returnTo: window.location.origin } })}
                  leftSection={<IconLogout />}
                >
                  Logout
                </Button>
              </Stack>
            </PageBody>
          </>
          }
        );
      };
```

У лістингу 3.14 реалізовано інтерфейс сторінки налаштувань користувача, який дозволяє керувати розширеним режимом практики. Основним елементом є перемикач типу Switch, що вмикає або вимикає вимогу до створення складних речень при перевірці. Початкове значення встановлюється на основі поточного стану користувача, отриманого з контексту, а зміна стану активує кнопку Apply для збереження змін.

Функція apply викликає асинхронний запит на оновлення параметрів користувача в базі даних через відповідний API. Таким чином, інтерфейс забезпечує простий і зрозумілий механізм керування персоналізованими параметрами навчання, що впливають на логіку перевірки речень у режимі практики. Зручне розміщення елементів та чіткий опис функціональності сприяють позитивному користувацькому досвіду.

Лістинг 3.14 – Програмний код, що реалізує інтерфейс сторінки налаштувань користувача

```
export const SettingsPage = () => {
  const navigate = useNavigate();
  const user = useCurrentUser();
  const [value, setValue] = useState(user.advancedPracticeMode);
  const [update, { isLoading }] = useUpdateSettingsMutation();
  const apply = async () => {
    await update({ advancedPracticeMode: value });};
  return (
<PageHeader title="Settings" leftSection={<BackButton onClick={() => navigate("/account")} />} />
      <PageBody>
        <Stack p="md" justify="space-between" h="100%">
          <Switch
            size="lg"
            checked={value}
            onChange={() => setValue((v) => !v)}
            label="Advanced practice"
description="Require complex sentences in practice"
            labelPosition="left"
styles={{ body: { justifyContent: "space-between" } }}
          />
          {value !== user.advancedPracticeMode && (
            <Button size="lg" loading={isLoading} onClick={apply}>Apply</Button>
          )}
        </Stack>
      </PageBody>
    </>
  );
};
```

3.3 Опис екранів застосунку

Пілся реалізації саме час зробити огляд на основні екрани мобільного застосунку, що реалізовані в рамках проекту для підтримки повноцінного процесу вивчення лексики. Кожен екран виконує окрему функцію, пов’язану з управлінням словниковими наборами, навчанням або взаємодією з профілем користувача. Загальна структура інтерфейсу побудована таким чином, щоб забезпечити інтуїтивну навігацію між основними розділами.

Екрани згруповано відповідно до логіки використання: починаючи від перегляду та створення колекцій, далі переходячи до режимів навчання, практики та перевірки прикладів речень. Окремо виділено блоки, пов’язані з керуванням профілем та персональними налаштуваннями користувача. Інтерфейс побудований із дотриманням принципів адаптивності та доступності.

У кожному з підрозділів буде подано короткий опис призначення відповідного екрану, основні функціональні елементи, а також візуальне представлення у вигляді скріншотів. Це дозволить краще зрозуміти структуру застосунку, взаємозв’язки між елементами та загальний підхід до побудови користувацького досвіду.

3.3.1 Екран входу у застосунок

Екран входу (рисунок 3.1) є першим кроком взаємодії користувача із застосунком і виконує функцію авторизації. Його інтерфейс побудовано максимально лаконічно, щоб зосередити увагу на основній дії – натисканні кнопки для входу. Застосунок використовує інтеграцію з сервісом Auth0, що забезпечує безпечний доступ без необхідності створення локального облікового запису.

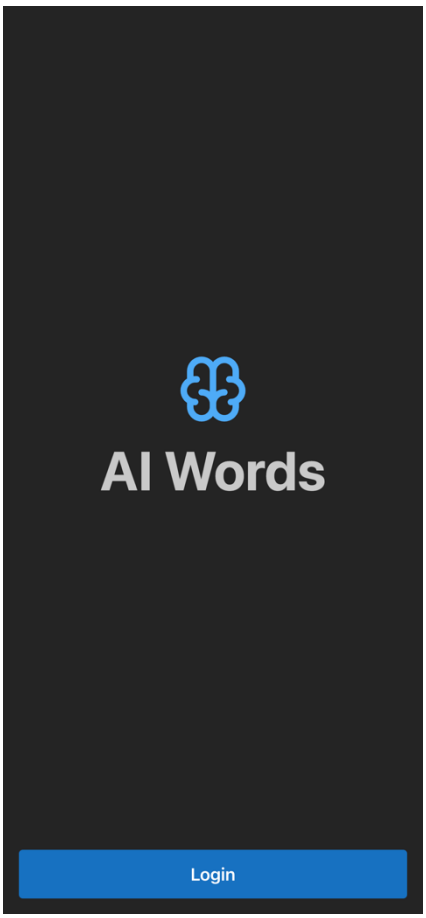

Рисунок 3.1 – Екран входу у застосунок

Графічне оформлення включає логотип і назву системи, що допомагає закріпити її пізнаваність та створює відчуття завершеності. Центральне розташування елементів дозволяє легко орієнтуватися в інтерфейсі навіть на невеликих екранах мобільних пристроїв. Завдяки мінімалізму, екран не перевантажений інформацією та сприяє швидкому старту роботи.

3.3.2 Головний екран застосунку

Головний екран застосунку (рисунок 3.2) слугує стартовою точкою для навігації між словниковими колекціями користувача. На ньому відображаються всі наявні колекції у вигляді списку з назвами, кількістю слів та індикаторами прогресу засвоєння. Завдяки цьому користувач одразу

бачить структуру свого навчального контенту й може швидко оцінити, які теми вже опрацьовані, а які потребують уваги.

Інтерфейс також містить кнопку для створення нової колекції, розташовану у верхньому правому куті, що робить функціональність розширення словника легко доступною. У нижній частині екрана розташована панель навігації, яка дозволяє перемикатися між іншими основними розділами застосунку, такими як навчання чи профіль користувача.

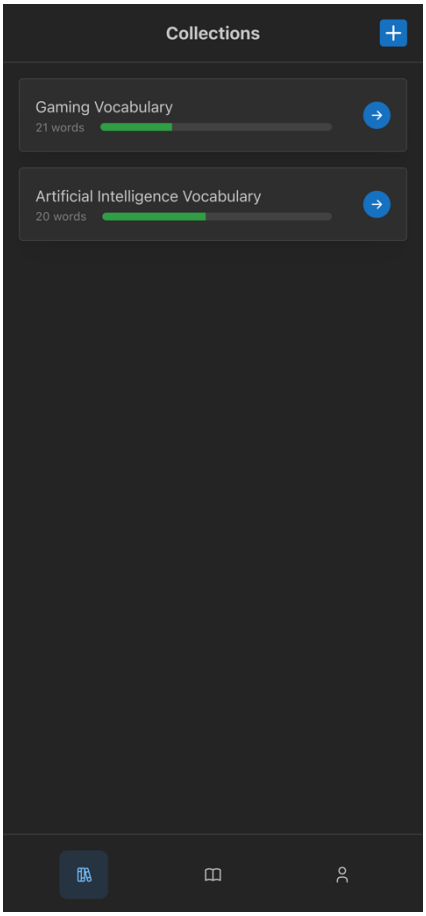

Рисунок 3.2 – Головний екран застосунку

Така структура підвищує зручність використання та дозволяє швидко виконувати основні дії. Компактне й водночас інформативне оформлення екрана сприяє зосередженому управлінню навчальним процесом.

3.3.3 Екран створення словникового набору

Далі можна побачити екран створення словникового набору, який містить усі необхідні компоненти для створення колекції та приклади заповнення полів (рисунок 3.3). Завдяки такому підходу забезпечується висока зручність для користувача та гнучкість у налаштуванні майбутніх словникових наборів.

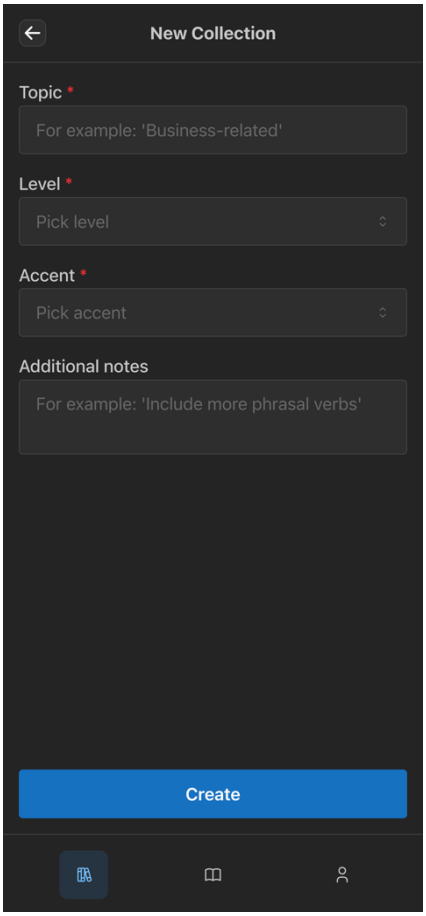

Рисунок 3.3 – Екран створення словникового набору

Екран створення словникового набору призначений для формування користувацьких колекцій слів відповідно до індивідуальних запитів. Усі необхідні поля згруповані вертикально та мають чіткі підказки, що допомагає користувачеві швидко зорієнтуватися у введенні даних. Для

кожної колекції потрібно вказати тему, рівень володіння мовою, акцент, а також за бажанням додати додаткові примітки.

Рівень і акцент обираються з випадаючих списків, що зменшує ймовірність помилок під час введення та стандартизує формат даних. Окреме поле для приміток дозволяє користувачеві надати уточнення, які вплинуть на логіку генерації контенту за допомогою штучного інтелекту. Кнопка створення розташована в нижній частині екрана, що відповідає загальним стандартам мобільного UX.

3.3.4 Екран перегляду словника

Екран перегляду словника дозволяє користувачеві ознайомитися зі змістом конкретної словникової колекції, яку він створив або обрав для навчання. Кожне слово представлено у вигляді інтерактивної картки з коротким поясненням, що допомагає швидко зорієнтуватися у вмісті. Візуальна індикація прогресу за допомогою кольору підсилює сприйняття того, які слова вже опрацьовані.

У верхній частині екрана знаходиться назва поточної колекції та кнопка переходу до режиму навчання, що забезпечує логічну навігацію в межах одного сценарію користування. Карточки з новими словами мають нейтральне забарвлення, а вивчені – підсвічуються зеленим, що дозволяє користувачеві оцінити свій прогрес навіть без переходу до детального перегляду кожного слова.

Далі можна побачити приклад візуального подання цього інтерфейсу, де відображено набір термінів із теми штучного інтелекту з різними станами опрацювання (рисунок 3.4). Такий підхід до реалізації сприяє зручному управлінню вмістом і підтримує мотивацію користувача шляхом наочного відстеження успіхів.

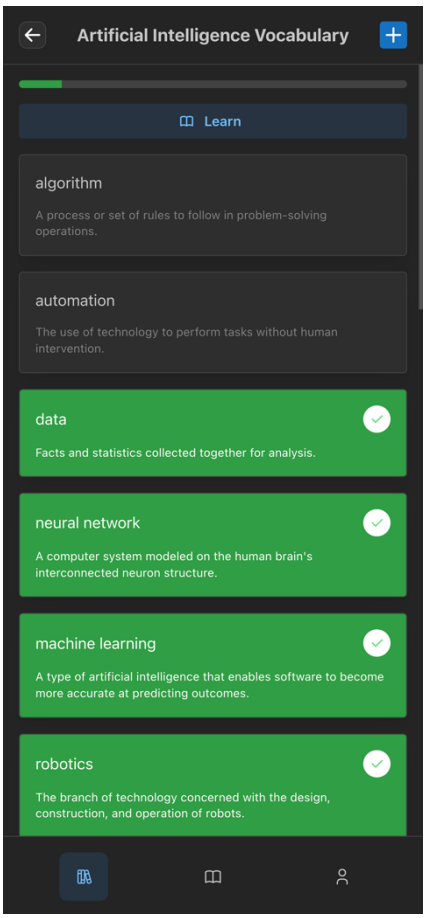

Рисунок 3.4 – Екран перегляду словника

Екран перегляду певного слова призначений для глибшого ознайомлення з лексичною одиницею, яка входить до складу словникової колекції. Він містить ключові поля, які допомагають повністю розкрити зміст слова: пояснення англійською, переклад українською, синоніми та приклади вживання у реченнях. Така структура сприяє кращому запам’ятовуванню та засвоєнню в контексті.

Кожен блок інформації відокремлений візуально і має відповідний підпис, що дозволяє користувачеві легко орієнтуватися в матеріалі. Окрема увага приділяється прикладам – вони подані у вигляді впорядкованого списку, що демонструє практичне використання слова у реальних висловах. Це особливо важливо для користувачів, які прагнуть покращити не лише словниковий запас, а й мовленнєву компетентність.

Далі можна побачити візуальне представлення такого інтерфейсу, де показано інформацію про слово natural language processing, включно з його перекладом, синонімами та прикладами (рисунок 3.5). Завдяки чіткому оформленню та послідовній структурі користувач має змогу зосередитися на ключових аспектах кожного слова.

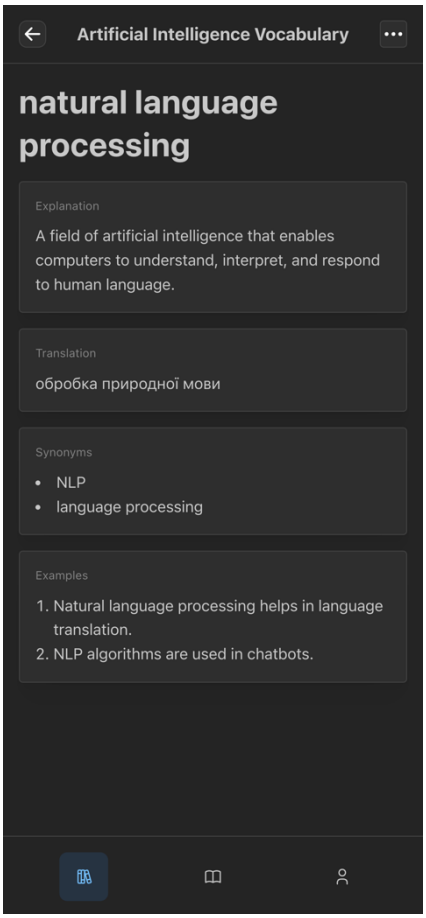

Рисунок 3.5 – Екран перегляду певного слова

3.3.5 Екран вивчення слів

Екран вивчення слів надає користувачеві можливість обрати один із кількох режимів взаємодії з колекціями, зокрема вивчення, повторення або самостійний вибір. Така організація дозволяє адаптувати процес навчання до індивідуального темпу та потреб користувача. Кожен із режимів

відображається у вигляді окремої кнопки, розміщеної у верхній частині інтерфейсу, що забезпечує швидкий доступ до відповідного функціоналу.

Нижче розміщені списки колекцій, згруповані за контекстом використання – окремо для вивчення та повторення. Кожна колекція супроводжується прогрес-баром і кількістю слів, що дає змогу швидко оцінити рівень опрацювання матеріалу. Далі можна побачити візуалізацію цього інтерфейсу з прикладом двох колекцій, розділених на блоки «To learn» і «To revise» (рисунок 3.6).

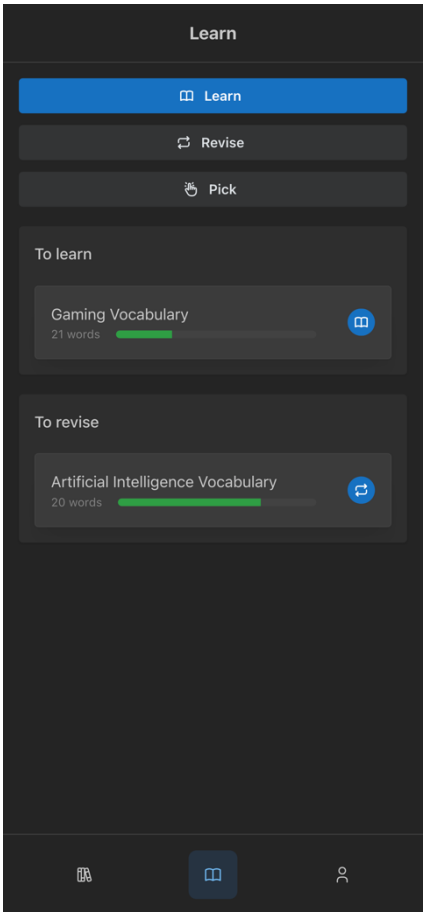

Рисунок 3.6 – Екран вивчення слів

Екран вибору конкретного словника для вивчення дозволяє користувачеві самостійно обрати будь-яку зі своїх словникових колекцій для подальшої роботи. Кожна колекція відображається у вигляді окремого елементу списку з назвою, кількістю слів і графічним індикатором прогресу.

Такий інтерфейс забезпечує інтуїтивну навігацію та дає змогу швидко переключитися між різними тематиками. Далі можна побачити приклад реалізації цього функціоналу з двома наявними колекціями (рисунок 3.7).

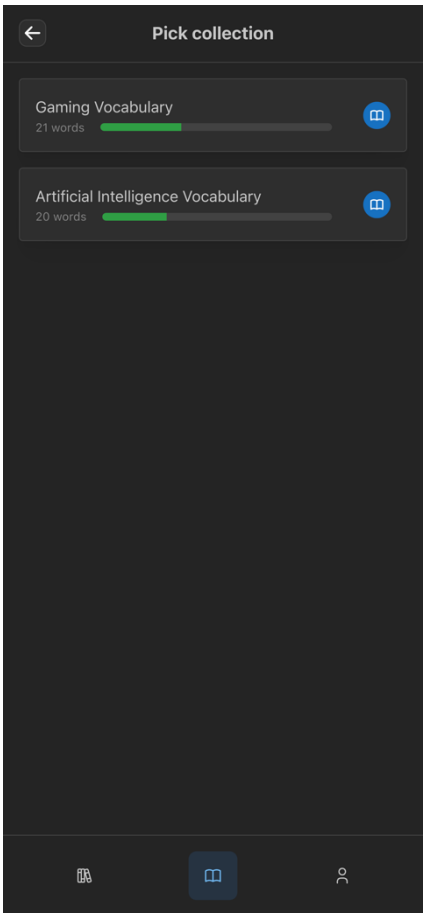

Рисунок 3.7 – Екран вибору конкретного словника для вивчення

Екран вивчення слова є основним етапом ознайомлення користувача з новою лексичною одиницею. У центрі екрана розміщується слово, яке слід засвоїти, без додаткової інформації – це дозволяє сфокусувати увагу лише на написанні та вимові. Прогресивна шкала у верхній частині візуально відображає динаміку навчання в межах поточного набору.

У нижній частині інтерфейсу доступні три основні дії: переглянути деталі слова, перейти до наступного, або розпочати практику. Такий підхід дозволяє користувачеві самостійно обирати стратегію навчання – або пасивне запам’ятовування, або активне використання в контексті. Особливо

корисною є кнопка Show, яка відкриває всю супровідну інформацію про слово.

Далі можна побачити приклад такого інтерфейсу з вивченням терміна neural network (рисунок 3.8). Завдяки простому дизайну та логічній структурі цей екран забезпечує ефективну взаємодію користувача із контентом та поступове занурення в матеріал.

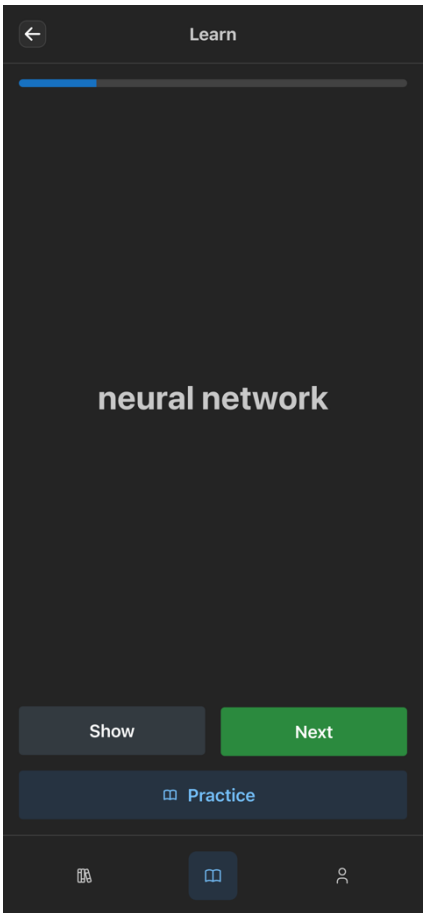

Рисунок 3.8 – Екран вивчення слова

Екран перегляду пояснення, перекладу та прикладів слова у процесі навчання розкриває повну інформацію про поточне слово, дозволяючи користувачеві закріпити його значення в різних контекстах. Тут послідовно подано блоки з поясненням англійською мовою, перекладом українською, синонімами та прикладами використання. Такий формат сприяє глибшому розумінню не лише форми, а й функціонального значення слова в мові.

Далі можна побачити візуальну реалізацію цього інтерфейсу для слова neural network (рисунок 3.9). Розміщення інформації у вигляді окремих блоків підвищує читабельність і забезпечує легкий доступ до кожного елементу. Після ознайомлення користувач може одразу перейти до практики або продовжити навчання зі словом у наступній картці.

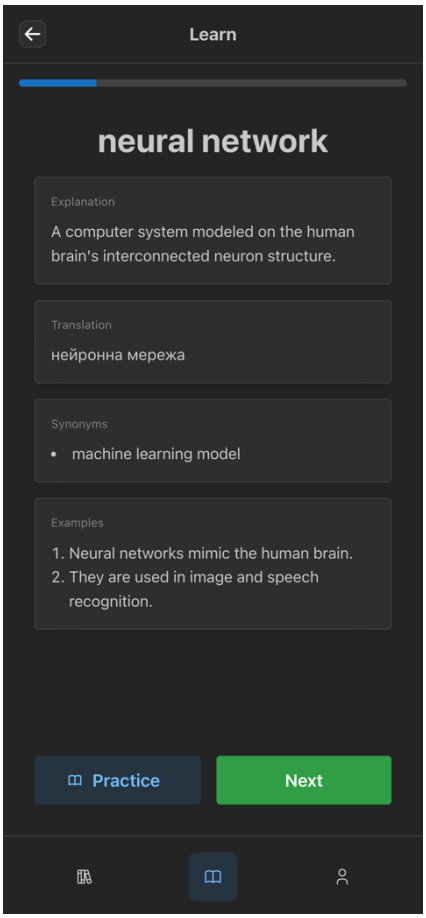

Рисунок 3.9 – Екран перегляду пояснення, перекладу та прикладів слова у процесі навчання

3.3.6 Екран практики

Екран практики дозволяє користувачеві застосувати вивчене слово в контексті, складаючи приклад речення із його використанням. Цей підхід формує навички активного мовлення та письма, що є важливими для

засвоєння лексики. Далі можна побачити, як реалізовано цей функціонал для слова neural network у застосунку (рисунок 3.10).

Інтерфейс містить поле для введення прикладу, а також кнопки перевірки, перегляду додаткової інформації та переходу до наступного слова. Верифікація виконується через OpenAI API, що дозволяє автоматично оцінити коректність речення та надати зворотний зв’язок. Такий підхід підвищує ефективність навчання та робить процес більш інтерактивним.

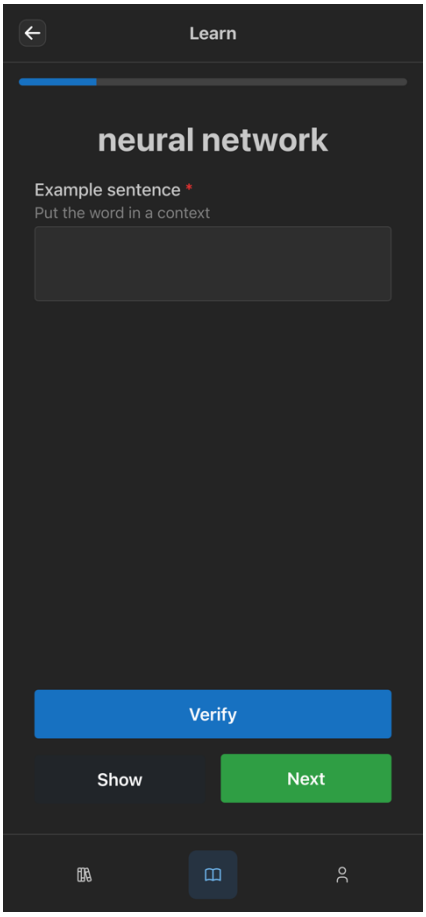

Рисунок 3.10 – Екран практики

Екран валідації у режимі практики з’являється після того, як користувач ввів приклад речення для вивченого слова і надіслав його на перевірку. У разі виявлення помилки система за допомогою штучного інтелекту надає пояснення щодо неточності, що допомагає користувачеві

скоригувати розуміння контексту. Приклад помилки для слова neural network показано на екрані верифікації (рисунок 3.11).

Використання автоматичної оцінки речень надає користувачеві змогу одразу побачити, у чому саме полягає недоліки прикладу. Система не лише фіксує факт помилки, але й надає коментар, який допомагає глибше зрозуміти значення та контекст вживання слова. Такий підхід стимулює більш усвідомлене та ефективне вивчення лексики.

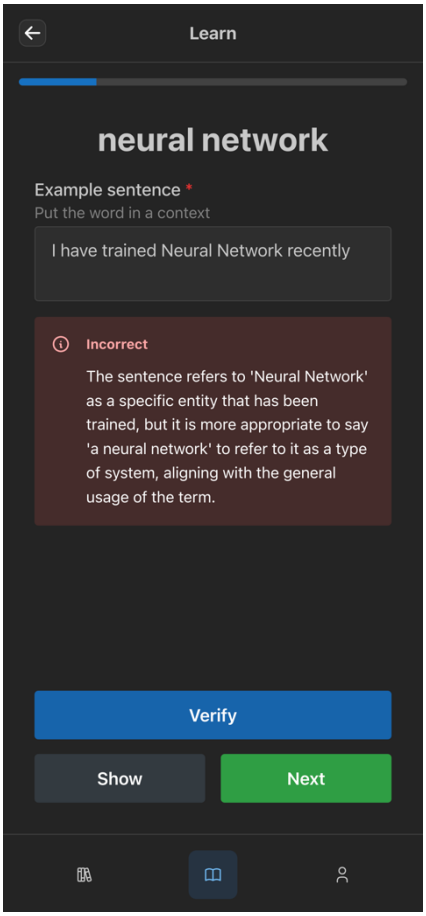

Рисунок 3.11 – Екран валідації у разі не коректного речення у режимі практики

Після введення прикладу речення користувачем і його відправлення на перевірку система аналізує його та у випадку коректного використання слова відображає позитивний результат. На екрані (рисунок 3.12)

демонструється повідомлення про правильне використання слова neural network у наданому контексті.

Такий зворотний зв’язок створює ефект підтримки та впевненості у правильності дій користувача. Завдяки цьому процес навчання стає більш інтуїтивним і мотиваційно підкріпленим, адже користувач бачить результат своїх зусиль у реальному часі.

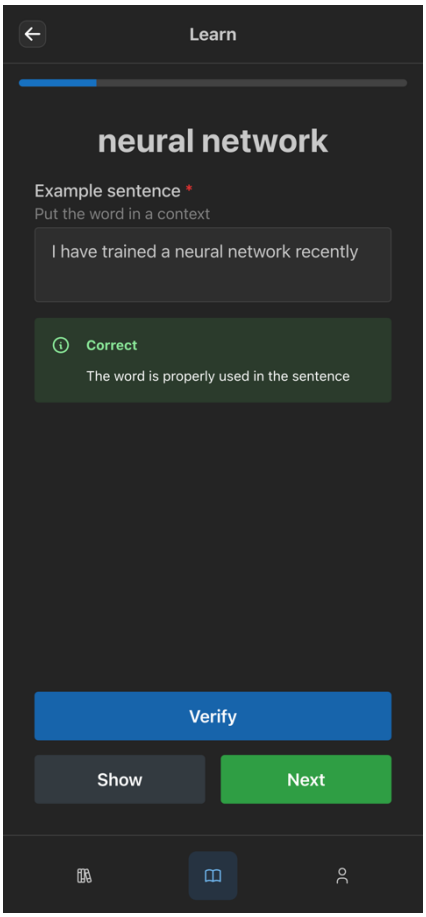

Рисунок 3.12 – Екран валідації у разі коректного речення у режимі практики

3.3.7 Екран профілю користувача

Інтерфейс сторінки профілю надає користувачеві можливість переглянути персональні дані, включаючи ім’я, електронну пошту та аватарку, а також перейти до налаштувань або вийти з облікового запису.

Далі можна побачити візуалізацію цього функціоналу (рисунок 3.13), де чітко структуровані елементи управління обліковим записом.

Реалізація даного екрану спрямована на забезпечення зручності доступу до персональної інформації та налаштувань, а також на швидкий вихід із системи. Мінімалістичний дизайн і логічна ієрархія елементів сприяють простому орієнтуванню навіть для недосвідчених користувачів.

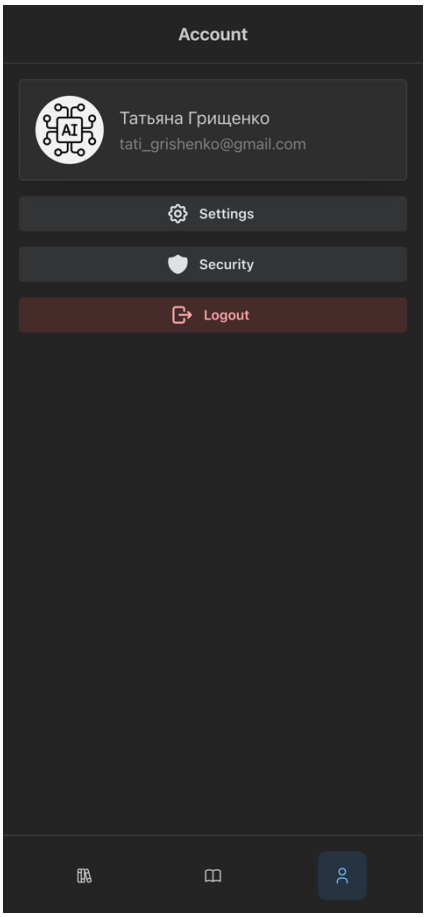

Рисунок 3.13 – Екран профілю користувача

На екрані налаштувань реалізовано функціональність персоналізації режиму навчання, зокрема можливість увімкнення або вимкнення параметра «розширена практика». Далі можна побачити відображення цього функціоналу, включаючи перемикач для активації складного режиму тренування (рисунок 3.14).

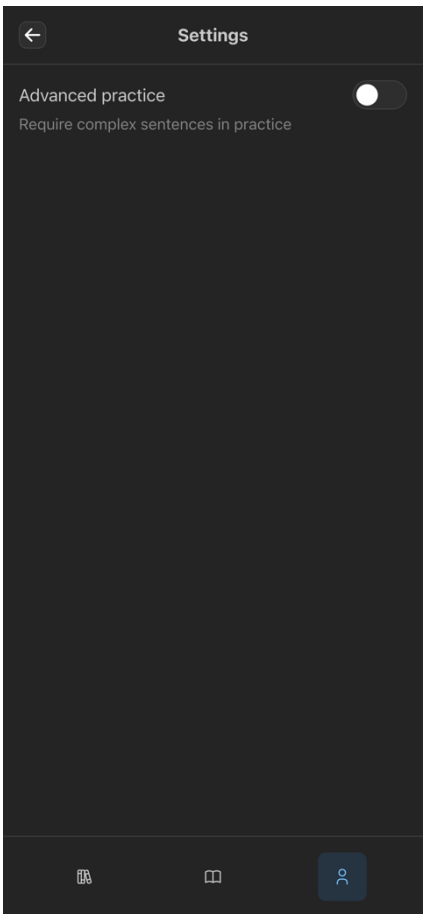

Рисунок 3.14 – Екран налаштувань користувача

Цей параметр дозволяє користувачеві визначити рівень складності під час виконання вправ на вживання слів у контексті. Якщо опція активована, система буде вимагати більш складних речень і надавати суворішу оцінку введених прикладів, що підвищує ефективність засвоєння.

Інтерфейс реалізовано таким чином, щоб забезпечити швидкий доступ до ключових налаштувань без перевантаження зайвими опціями. Збереження мінімалізму дозволяє користувачам зосередитися на головній меті застосунку, а саме – якості навчання та зручності керування особистим процесом.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи повністю реалізовано поставлену мету – розроблено мобільний застосунок для вивчення слів іноземною мовою з використанням генеративного штучного інтелекту. Було проведено повний цикл реалізації проєкту – від аналізу предметної області та формування вимог до застосунку, до створення архітектури, реалізації клієнтської та серверної частин, інтеграції з зовнішніми сервісами та проведення тестування. У додатку забезпечено функціональність генерації словникових наборів відповідно до заданих параметрів користувача, перегляду інформації про слова, інтерактивного навчання з перевіркою речень у контексті, ведення прогресу та персоналізованих налаштувань режиму практики. Якісними показниками завершення проєкту є повна відповідність функціоналу сформульованим вимогам, стабільність роботи інтерфейсів, відсутність критичних помилок та позитивна оцінка користувацького досвіду у тестовому середовищі.

Порівняно з існуючими аналогами, такими як Duolingo, Memrise або WordUp, розроблений застосунок має низку переваг, що визначають його конкурентоспроможність. По-перше, ключова відмінність полягає в застосуванні генеративної моделі штучного інтелекту OpenAI, що дозволяє створювати унікальні словникові набори за запитом користувача, а не покладатися на статичні бази даних. По-друге, перевірка речень із використанням контекстного аналізу також виконується через OpenAI, що забезпечує більш точну та адаптивну оцінку, ніж традиційні жорстко закодовані правила у багатьох конкурентних рішеннях. На відміну від програм, орієнтованих переважно на гейміфікацію, розроблена система робить акцент на контекстуальне використання та смислову глибину вивчення лексики, що наближає її до реального мовного досвіду. У порівняльному аналізі також варто відзначити ефективну інтеграцію з Firebase, яка забезпечує швидке зберігання та обробку даних користувачів,

а також Auth0 для безпечної автентифікації, що відповідає сучасним вимогам до безпеки.

У ході реалізації проєкту були також виявлені потенційні напрями для подальшого розвитку системи. По-перше, можливо розширити функціональність генерації словникових наборів з урахуванням стилістичних параметрів, таких як ділова, технічна або художня мова, що дозволить ще точніше адаптувати контент до потреб користувача. По-друге, варто впровадити систему оцінювання прогресу з елементами машинного навчання, яка буде не лише фіксувати виконані дії, але й прогнозувати слабкі сторони користувача та пропонувати персоналізовані повторення. По-третє, перспективним є впровадження офлайн-режиму навчання з локальним кешуванням словників та історії дій, що забезпечить доступність платформи у середовищах з обмеженим доступом до мережі.

Окрему увагу доцільно приділити розширенню мовної підтримки та адаптації інтерфейсу під користувачів з різним рівнем цифрової грамотності. Це може включати як розробку версій для інших мов інтерфейсу, так і впровадження голосової взаємодії та простих інструкцій для новачків. Крім того, варто розглянути створення вебверсії застосунку з синхронізацією прогресу між мобільним та десктопним середовищем. Усе це дозволить підвищити доступність та зручність системи для ширшого кола користувачів.

Таким чином, розроблений застосунок продемонстрував ефективність обраної архітектури, успішність інтеграції з передовими AI-рішеннями та потенціал до подальшої еволюції в повноцінну адаптивну платформу для вивчення мовної лексики. З огляду на динаміку розвитку технологій та інтерес до персоналізованого навчання, подальше розширення можливостей цієї системи видається доцільним і перспективним як у дослідницькому, так і у комерційному аспекті.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Д. В. Штучний інтелект у мовній освіті: досвід інтеграції генеративних моделей. *Journal of AI in Education*. 2023. № 4. С. 33–41.

2. Documentation N. NestJS – A progressive Node.js framework. URL: https://docs.nestjs.com/ (date of access: 30.04.2025).

3. EdTechReview. Мовні моделі та їх використання у навчанні. URL: https://edtechreview.in/news/ai-language-learning-tools (дата звернення: 21.04.2025).

4. Firebase G. Firebase Documentation. URL: https://firebase.google.com/docs (date of access: 12.04.2025).

5. Inc. D. Duolingo – Language learning app. URL: https://www.duolingo.com/ (date of access: 25.03.2025).

6. Inc. Q. Quizlet – Learning tools \& flashcards. URL: https://quizlet.com/ (date of access: 10.04.2025).

7. Ltd. M. Memrise – Learn a language. Meet the world. URL: https://www.memrise.com/ (date of access: 26.04.2025).

8. Okta A. b. Auth0 – Modern authentication and authorization platform. URL: https://auth0.com/docs (date of access: 27.04.2025).

9. OpenAI. GPT-4 Technical Report. URL: https://openai.com/research/gpt-4 (date of access: 14.04.2025).

10. Press C. U. Cambridge Dictionary – The most popular online dictionary and thesaurus for learners of English. URL: https://dictionary.cambridge.org/ (date of access: 29.03.2025).

11. Standardization I. O. f. ISO/IEC 29110:2011 Systems and software engineering – Lifecycle profiles for Very Small Entities (VSEs). URL: https://www.iso.org/standard/62767.html (date of access: 31.03.2025).

12. Mantine – React components and hooks library. URL: https://mantine.dev/ (date of access: 17.04.2025).