НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету	**Завідувач кафедри комп'ютерних наук**
_____ **Ігор БОЛБОТ**	_____ **Белла ГОЛУБ**
(підпис)	(підпис)
«01» _____червня_____ 2026 р.	«01» _____червня_____ 2026 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Система супроводу корейськомовним словником»

Спеціальність «122 Комп'ютерні науки»

Освітньо-професійна програма «Комп'ютерні науки»

Гарант освітньої програми
д.е.н., професор **Роман РУДЕНСЬКИЙ**

Керівник бакалаврської кваліфікаційної роботи **Ярослав ГОРДІЙ**

Консультант бакалаврської кваліфікаційної роботи
к.т.н., доцент **Белла ГОЛУБ**

Виконав **Анастасія ХАРЧЕНКО**

Київ-2026

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук

к.т.н., доцент _____ Белла ГОЛУБ
(підпис)

«06» січня 2026 р.

ЗАВДАННЯ
ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
ЗДОБУВАЧУ

Харченко Анастасії Олександрівні

Спеціальність «122 Комп'ютерні науки»

Освітньо-професійна програма «Комп'ютерні науки»

Тема бакалаврської кваліфікаційної роботи

«Система супроводу корейськомовним словником»

затверджена наказом від 06.01.2026 р. № 46 «С»
Термін подання завершеної роботи на кафедру 25.05.2026 р.
Вихідні дані до бакалаврської кваліфікаційної роботи (проєкту): веб-застосунок для вивчення корейської лексики з статистикою та прогресом користувача.
Перелік питань, що підлягають дослідженню: аналіз предметної області, проєктування системи, розробка системи, впровадження системи та тестування.
Дата видачі завдання 06.01.2026 р.

Керівник бакалаврської кваліфікаційної роботи **Ярослав ГОРДІЙ**

Консультант бакалаврської кваліфікаційної роботи **Белла ГОЛУБ**
к.т.н., доцент

Завдання взяв до виконання **Анастасія ХАРЧЕНКО**

ЗМІСТ

СПИСОК УМОВНИХ СКОРОЧЕНЬ

ССКС - Система супроводу корейськомовним словником - веб-застосунок, розроблений у межах цієї роботи, що поєднує словниковий пошук, тематичне навчання, флеш-картки, тестування та персональний прогрес користувача в одному середовищі.

БД - база даних - організоване сховище структурованої інформації, у якому зберігаються всі дані системи: словниковий матеріал, дані користувачів, результати тестів і статистика навчання.

СУБД - система керування базами даних - програмне забезпечення для створення, зберігання, оновлення та керування даними в базі даних. У цьому проєкті використовується PostgreSQL.

ФК - флеш-картки - інтерактивний режим навчання, у якому користувач послідовно переглядає слова, намагається пригадати їх значення та перевіряє себе через перевертання картки.

RLS - Row Level Security (безпека на рівні рядків) - механізм PostgreSQL, який обмежує доступ до даних на рівні окремих записів таблиці залежно від того, хто виконує запит. У ССКС застосовується для того, щоб користувач бачив лише власні дані - обране, результати тестів і прогрес.

JWT - JSON Web Token (токен автентифікації) - стандарт передачі інформації між клієнтом і сервером у вигляді підписаного токена. Використовується в ССКС для підтримки сесії авторизованого користувача після входу в систему.

API - Application Programming Interface (інтерфейс програмування застосунків) - набір правил і протоколів, за допомогою яких різні компоненти системи обмінюються даними. У проєкті клієнтська частина взаємодіє з Supabase саме через API.

SDK - Software Development Kit (набір інструментів розробника) - бібліотека або набір інструментів, що спрощує роботу з певною платформою чи сервісом. У ССКС використовується Supabase JavaScript SDK для виконання запитів до бази даних і механізмів авторизації безпосередньо з клієнтського коду.

GIN - Generalized Inverted Index (узагальнений інвертований індекс) - тип індексу в PostgreSQL, оптимізований для повнотекстового пошуку. У ССКС застосовується для прискорення пошуку слів одночасно за корейським написанням і українським перекладом.

PK - Primary Key (первинний ключ) - унікальний ідентифікатор кожного запису в таблиці бази даних. Гарантує, що жоден рядок не повториться двічі.

FK - Foreign Key (зовнішній ключ) - поле в таблиці, яке посилається на первинний ключ іншої таблиці. Забезпечує зв'язки між таблицями та цілісність даних у реляційній БД.

3НФ - третя нормальна форма - рівень нормалізації реляційної бази даних, за якого кожен не ключовий атрибуft залежить виключно від первинного ключа і не має транзитивних залежностей від інших не ключових атрибутів. Дотримання 3НФ усуває дублювання даних і спрощує їх оновлення.

RPC -Remote Procedure Call (виклик віддаленої процедури) - механізм, за допомогою якого клієнтська частина може викликати функцію, що виконується на сервері або в базі даних. У ССКС через RPC викликаються серверні SQL-функції Supabase — наприклад, пошук слів або додавання до обраного.

SQL - Structured Query Language (мова структурованих запитів) - стандартна мова для роботи з реляційними базами даних. Використовується в ССКС для створення таблиць, написання запитів, функцій і тригерів у PostgreSQL.

HTTPS - HyperText Transfer Protocol Secure (захищений протокол передачі даних) - протокол обміну даними між клієнтом і сервером із шифруванням через TLS. Забезпечує безпечну передачу всіх запитів між браузером користувача та платформою Supabase.

ВСТУП

Останніми роками спостерігається значний зріст інтересу до вивчення корейської мови серед українських користувачів. Це умовлено популяризацією корейської культури, музики, освітніх програм,корейських дорам, акторів, айдолів та міжнародних обмінів. Багато користувачів вивчають мову самостійно, використовуючи електронні словники, навчальні платформи та мобільні додатки.

Актуальність теми полягає в тому, що більшість доступних україномовних ресурсів для вивчення корейської мови забезпечують лише частину функціоналу: пошук слів, переклад, тематичні добірки чи тестування. Через це користувачеві доводиться використовувати кілька різних сервісів, що ускладнює процес навчання. Ось чому є потреба у створенні єдиної системи, яка б поєднувала весь цей функціонал.

Метою дипломної роботи є розробка ССКС є, веб-система яка забезпечує пошук, тематичне вивчення корейської лексики, перегляд прикладів вживання, навчання за допомогою ФК, перевірку тестуванням рівня знань та збереження індивідуального користувацького прогресу. Розробка такої системи дозволяє об'єднати основні етапи роботи з лексичним матеріалом у єдиному веб-середовищі.

Апробація результатів роботи. Основні результати дипломної роботи апробовано в тезах доповіді на III Міжнародній науково-практичній конференції «Актуальні питання розвитку науки та техніки в умовах глобалізації». За матеріалами дослідження підготовлено тези на тему «Система супроводу корейськомовним словником». Науковий керівник - Голуб Б. Л.

Структура пояснювальної записки. Робота складається із вступу, чотирьох розділів, висновків, списку використаних джерел та додатків,сторінок 71, додатків 5, джерел 33. Структура:

- У першому розділі розглядається предметна область ССКС: описуються основні процеси роботи системи, проводиться моделювання предметної області за допомогою UML-діаграм,

аналізуються існуючі аналоги та їхні переваги і недоліки, а також формулюються функціональні та нефункціональні вимоги до системи.

- У другому розділі представлено результати проєктування системи: спроєктовано логічну модель даних, обґрунтовано вибір СУБД і хмарного середовища, а також реалізовано структуру БД із таблицями, індексами, серверними функціями та тригерами.
- У третьому розділі описано практичну розробку програмного забезпечення: обрано засоби та технології реалізації, спроєктовано інтерфейс користувача, побудовано бізнес-логіку основних підсистем , також описано взаємодію БД з системою та загальну архітектуру системи.
- У четвертому розділі розглядається впровадження системи: наведено діаграму розгортання, описано програмні та апаратні вимоги, інсталяційний пакет, а також представлено результати тестування основних модулів системи.

1 АНАЛІЗ СИСТЕМИ СУПРОВОДУ КОРЕЙСЬКОМОВНИМ СЛОВНИКОМ

1.1 Опис процесів роботи ССКС

На практиці звичайного електронного словника для вивчення корейської мови зазвичай недостатньо. Побачити лише слово та його переклад недостатньо, адже під час вивчення корейської важливо одночасно бачити його транскрипцію та приклади використання. Саме тому потрібно не просто місце лише для пошуку слів, а й зручне середовища де можна їх вивчати, повторювати й перевіряти себе.

В сучасному світі більшість користувачів навчається за допомогою цифрових ресурсів: словники, навчальні сайти, мобільні застосунки. Однак для корейської мови цього часто недостатньо, бо навчання - це не один крок, а кілька: знайти слово, зрозуміти його значення, побачити в реченні, повторити й перевірити, чи добре запам'яталось.

У межах цієї роботи розглядається електронне середовище, побудоване навколо словникової системи. Його основне призначення полягає в тому, щоб допомогти користувачеві працювати з корейською лексикою не уривчасто, а послідовно. Тобто словник - основа всього навчального процесу.

Ключові процеси системи можна переглянути нижче.

- **Пошук слова.** Користувач має можливість ввести запит корейською або українською, та одразу отримує результат: саме слово, його переклад, транскрипцію та приклади використання.
- **Перегляд словникового запасу.** Для корейської мови особливо важливо не обмежуватися коротким перекладом, а бачити слово в ширшому контексті. Тому запис має містити додаткову інформацію - як слово звучить у реальних ситуаціях, де і як воно вживається.
- **Вивчення лексики за темами**. Під час навчання слова легша запам’ятовуються коли вони згруповані, наприклад: привітання, числа,

їжа, сім’я, кольори, професії, транспорт або погода. Такий підхід краще упорядковує матеріал і робить навчання більш послідовним. Крім того, поділ на категорії є зручним для подальшої роботи з ФК та тестами.

- **Повторення матеріалу**. Щоб слово залишалося у пам’яті не на короткий проміжок часу, до нього треба повертатися, саме тому було взято метод інтервальних повторень в вигляді карток. Тому доцільно використовувати ФК у яких користувач бачить слово, намагається самостійно пригадати його значення, а потім перевіряє себе. Такий формат робить навчання активним а не пасивним.
- **Тестування.** Після повторення слів, важливо зрозуміти наскільки добре вони засвоєні. На основі словникових даних формуються питання, варіанти відповідей і підсумковий результат. Такий підхід дає змогу бачити власний рівень підготовки та вчасно звертати на свої слабкі місця.
- **Збереження обраних слів.** У процесі навчання користувач може відзначати слова які йому здаються складними, та з якими хоче більше попрацювати. Формування власного списку робить більш індивідуальною та зручнішою - можна повертатись саме до того, що потребує додаткової уваги.
- **Накопиченням результатів**. Усі дані в ССКС фіксується: тестування, робота з ФК, збереженні слова. На основі цього система показує: статистику, прогрес і формує персоналізовану історію навчання. Це робить взаємодію зі словником не разовою, а довготривалою і змістовною.

Таки чином ССКС об'єднує кілька ключових процесів: пошук слова, перегляд словникового запису, вивчення слів за темами, повторення, тестування, збереження обраних слів і накопичення результатів. Саме з цих процесів складається робота користувача з корейською лексикою, тому вони мають бути зручно поєднані в одному електронному середовищі. У такому форматі словник перестає бути просто інструментом перекладу, а й стає середовищем, у якому

можна послідовно вчитися повертатись до матеріалу й відстежувати власний розвиток.

1.2 Моделювання предметної області

Для наочного подання предметної області та основних сценаріїв роботи користувача із системою було використано засоби UML-моделювання. Воно дає змогу формалізувати функції системи, показати взаємодію користувача з вебсайтом та описати логіку процесів.

1.2.1 Діаграма прецедентів. Ця діаграма є основною моделлю предметної області. Вона показує, що саме може робити користувач у системі та які функції для цього передбачені.

У розроблюваній системі такими акторами є гість і авторизований користувач. Основні можливості актора можна переглянути нижче:

a. Гість має доступ до базових функцій: пошуку слів, перегляду категорій, прикладів вживання, роботи з ФК, а також реєстрації або входу .
b. Авторизований користувач, окрім зазначено усього цього, може зберігати слова в обране, керувати ними та переглядати власний прогрес.

Основні процеси системи та їх короткий опис подано в табл. 1.

Таблиця 1

Основні прецеденти системи

Прецедент	**Опис процесів**
Пошук слів у словнику	Знайти слово за запитом українською або корейською
Перегляд слів за категоріями	Вивчати лексику за тематичними групами
Перегляд прикладів використання	Бачити слово в мовному контексті
Навчання за допомогою ФК	Повторювати лексику в інтервальному формі

Прецедент	Опис процесів
Реєстрація / Вхід	Отримати доступ до персоналізованих функцій
Додавання слів до обраного	Зберігати потрібні слова
Керування обраними словами	Переглядати й використовувати збережену лексику
Перегляд прогресу	Відстежувати результати навчання
Вихід із системи	Завершити роботу з особистим профілем

Діаграма прецедентів показує не лише перелік функцій, а й загальну логіку системи з погляду користувача, діаграму наведено в Додатку А рис.1.

1.2.2 Діаграма послідовності. Якщо діаграма прецедентів відповідає на питання *що*, то діаграми послідовності відповідають на питання *як*. Її призначення полягає у відображенні порядку взаємодії між користувачем та основними елементами системи в часі. Вони показують порядок взаємодії між користувачем та елементами системи, як саме виконується той чи інший сценарій крок за кроком. У межах цієї роботи діаграми побудовано для трьох сценаріїв: пошуку слова наведений в Додатку А рис.2, проходження тестування наведено в Додатку А рис.3 та роботи з ФК наведено в Додатку А рис.4. Кожна з них фіксує обмін діями між користувачем, інтерфейсом і рівнем даних - це дає змогу описати поведінку системи в конкретній ситуації.

1.2.3 Діаграма активності. Діаграма розкриває внутрішню логіку виконання процесів: послідовність кроків, умови переходу та можливі розгалуження. На відміну від діаграм послідовності, які акцентують увагу на взаємодії між учасниками, діаграми активності зосереджені на тому, що відбувається всередині самого процесу. У межах даної роботи вони дозволяють показати, як виконується пошук слова, діаграма представлена в Додатку А рис.5, як проходить тестування, діаграма представлена в Додатку А рис.6,або яким чином організовано роботу з ФК, діаграма представлена в Додатку А рис.7. Це

дає змогу побачити, як саме функціонує система зсередини, і краще зрозуміти логіку її реалізації.

1.3 Аналіз існуючих аналогів, їх переваг і недоліків

Перед тим як формувати концепцію ССКС, було розглянуто низку сучасних електронних ресурсів для вивчення іноземних мов, зокрема корейської. Серед них - онлайн-словники, перекладачі, навчальні платформи та сервіси ФК. Такий аналіз дав змогу визначити корисні функціональні рішення, які варто врахувати під час розроблення в власній розробці, а також побачити ті обмеження, які не дають змоги повністю задовольнити потреби користувача в межах одного ресурсу, використання підходів існуючих сервісів наведено в табл.2.

Електронні словники та перекладачі. Такі як **NAVER Dictionary** наведено на рис. 1**, Papago** наведено на рис.2 **та Google Translate**. Їх головна сильна сторона - швидкий пошук і простий інтерфейс.

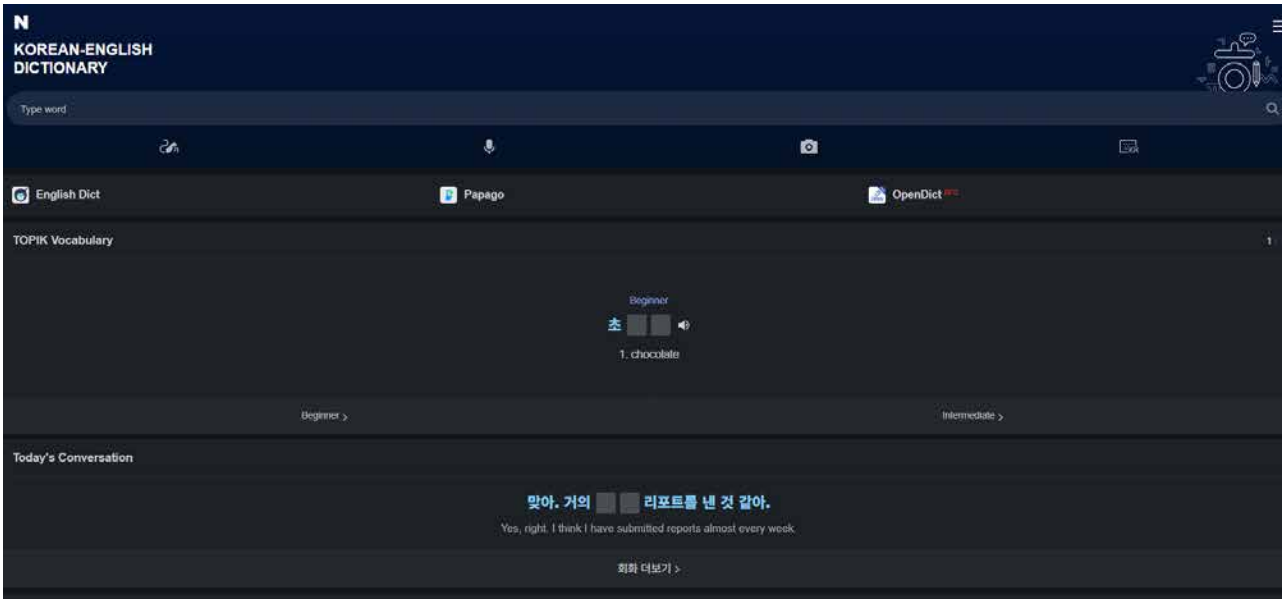

Рис.1 NAVER Dictionary

Для користувача це зручно в ситуаціях, коли потрібно швидко знайти значення слова, уточнити написання або переглянути приклад використання.

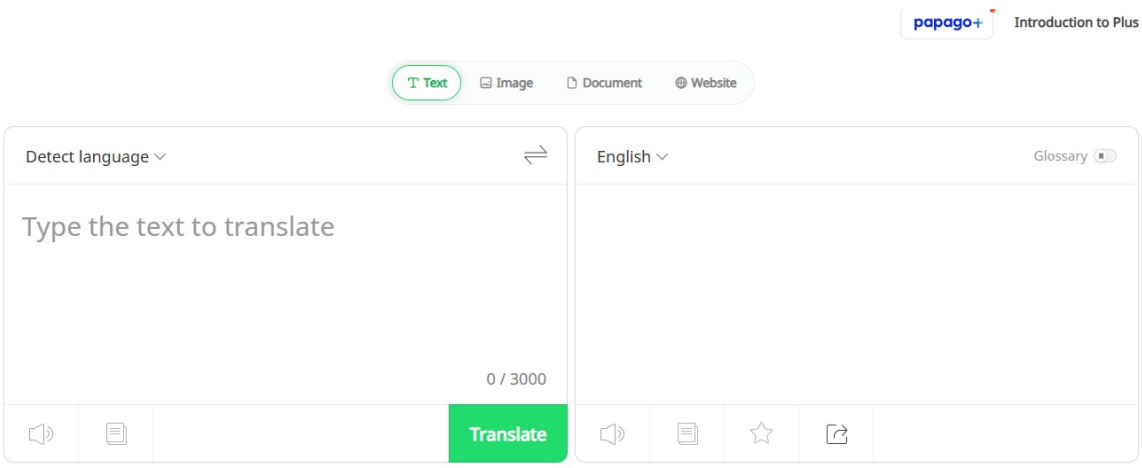

Рис.2 Papago

Переваги: швидкий доступ до результату, зрозумілий інтерфейс, наявність прикладів і транскрипції.

Недоліки: виконують лише довідкову функцію. Після перегляду перекладу користувач змушений самостійно шукати інші інструменти - для повторення, збереження або перевірки. Навчального середовища як такого немає.

Що запозичено для ССКС: принцип швидкого пошуку слова та лаконічне подання результату - переклад, транскрипція, приклади вживання.

Власне доповнення: на відміну від звичайного перекладача, у ССКС слово не існує окремо - воно пов'язане з тематичними категоріями, ФК, обраним і статистикою користувача. Додано також корейсько-українську орієнтацію, якої немає в жодному з перелічених сервісів у зручному форматі.

Навчальна платформа Duolingo. Duolingo наведено на рис.3 - один із найвідоміших прикладів того, як організувати регулярне навчання. Короткі сесії, поступове просування, нагадування, відображення прогресу - все це підтримує звичку вчитися.

Рис.3 Duolingo

Переваги: чітка структура занять, регулярна практика, наочний прогрес, орієнтація на довготривале навчання.

Недоліки: користувач рухається за наперед заданим маршрутом і не має гнучкості. Словниковий підхід тут слабкий - немає можливості вільно шукати окреме слово, переглядати його контекст або формувати власну добірку лексики.

Що запозичено для ССКС: логіка побудови тестових сценаріїв, підхід до повторення матеріалу та відображення навчального прогресу.

Власне доповнення: у ССКС тестування побудоване на основі власної словникової бази та лексичних категорій користувача - тобто людина перевіряє саме те, що вивчала, а не проходить загальний курс.

Сервіс флеш-карток Quizlet. Один із найвідоміших прикладів того, як організувати регулярне навчання, наведено на рис.4. Короткі сесії, поступове просування, нагадування, відображення прогресу - все це підтримує звичку вчитися.

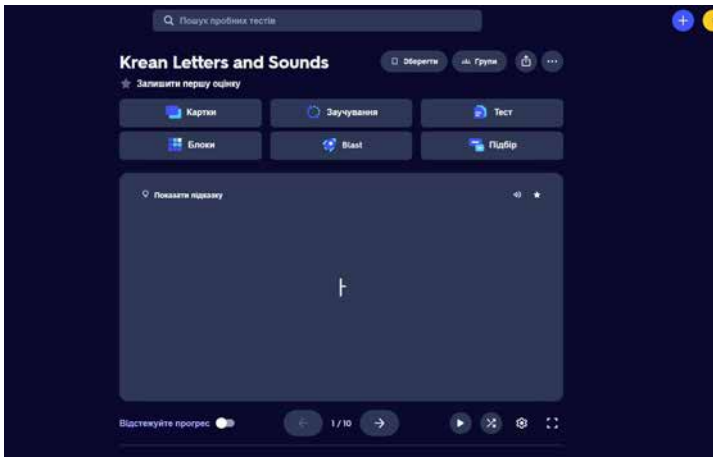

Рис.4 Quizlet

Переваги: інтерактивний формат, зручне повторення, швидкі навчальні цикли, гнучкість у створенні наборів карток.

Недоліки: сервіс зосереджений переважно на повторенні. Пошук слова, перегляд контексту, зв'язок із категоріями, детальна статистика - все це або обмежено, або винесено за межі основного сценарію.

Що запозичено в ССКС: формат ФК як основний інструмент повторення лексики.

Власне доповнення: у ССКС ФК не існують окремо - вони інтегровані зі словником, категоріями, обраним і прогресом користувача. Тобто картки відображають саме ту лексику, з якою людина вже працювала.

Таблиця 2

Використання підходів існуючих сервісів у ССКС

Аналог	**Переваги**	**Недоліки**	**Що запозичено**	**Власне доповнення**
NAVER / Papago / Google Translate	Швидкий пошук, простий інтерфейс, приклади	Лише довідкова функція, немає навчального середовища	Пошук слова, відображення перекладу та транскрипції	Зв'язок слова з категоріями, ФК, обраним і прогресом
Duolingo	Структура занять, прогрес, регулярна практика	Жорсткий маршрут, слабкий словниковий підхід	Логіка тестування та повторення	Тести на основі власної лексики користувача
Quizlet	Зручні ФК, швидке повторення	Немає пошуку, слабка статистика	Формат ФК для повторення слів	Інтеграція ФК зі словником, категоріями та статистикою

Проведений аналіз показав: кожен із розглянутих сервісів добре вирішує якесь одне завдання. Одні зручні для швидкого пошуку, інші - для регулярного навчання, треті - для повторення. Але жоден не поєднує все це в одному місці.

Саме тому головною ідеєю ССКС стала інтеграція: пошук слова, тематичне структурування, приклади вживання, ФК, тестування, обране та

персоналізований прогрес - усе це доступне в межах одного електронного середовища.

1.4 Постановка задачі та формування вимог до системи

1.4.1 Постановка задачі. Проведений аналіз предметної області та існуючих аналогів показав, що під час вивчення корейської мови користувачеві недостатньо лише окремого електронного словника або окремої навчальної платформи. На практиці виникає потреба в єдиному середовищі, у якому можна знайти слово, переглянути його значення, побачити приклади використання, вивчати лексику за темами, повторювати її та перевіряти рівень засвоєння. Через відсутність такого цілісного підходу користувачеві доводиться звертатися до кількох різних ресурсів, що ускладнює навчання та робить його менш послідовним.

Тому задача полягає в розробленні ССКС - веб-застосунку, який забезпечить цілісну роботу з корейською лексикою та підтримає повний навчальний цикл: від пошуку слова до його закріплення й перевірки знань. Система має поєднати три функції в одному місці - довідкову, навчальну та контрольну.

1.4.2 Функціональні вимоги. Функціональні вимоги визначають, які саме можливості повинна мати система для забезпечення зручної взаємодії користувача з навчальним матеріалом представлені нижче.

a) **Робота зі словником:**
   - шукати слова українською або корейською мовою;
   - переглядати словниковий запис - переклад, транскрипцію та приклади вживання;
   - переглядати слова за тематичними категоріями;
   - переходити до детальної сторінки окремого слова.

b) **Навчання та повторення:**
   - працювати з ФК для повторення та закріплення лексики;
   - проходити тести й перевіряти рівень засвоєння матеріалу;

- зберігати результати тестів і сесій повторення для подальшого аналізу.

c) **Особистий простір:**

- реєструватися та входити в систему;
- додавати слова до обраного та керувати цим списком;
- переглядати власний прогрес і статистику навчання.

d) **Навігація:** вільно переміщатися між основними розділами - словник, ФК, тести, акаунт.

1.4.3 Нефункціональні вимоги. Нефункціональні вимоги визначають якість роботи системи та зручність її використання.

До основних нефункціональних вимог належать.

- **Зрозумілість інтерфейсу.** Система повинна бути простою й зручною у використанні для користувачів різного рівня підготовки.
- **Адаптивність.** Інтерфейс повинен коректно відображатися на комп’ютерах, планшетах і мобільних пристроях.
- **Швидкодія.** Пошук слів, відкриття категорій, флеш-карток і тестів не повинні супроводжуватися відчутними затримками.
- **Надійність збереження даних.** Дані про обране, результати тестів і прогрес користувача мають зберігатися без втрати.
- **Безпека.** Доступ до персональних даних користувача повинен бути захищеним і доступним лише після авторизації.
- **Масштабованість.** Система повинна допускати подальше розширення словникової бази, категорій, режимів тестування та інших функцій.
- **Підтримуваність.** Структура системи має дозволяти оновлення, виправлення та доповнення без повної перебудови застосунку.
- **Доступність через браузер.** Користувач повинен мати можливість працювати із системою без встановлення окремого програмного забезпечення.

Такі вимоги є важливими, оскільки система орієнтована не лише на зберігання словникових даних, а й на регулярне повсякденне використання в навчальних цілях.

2 ПРОЄКТУВАННЯ СИСТЕМИ СУПРОВОДУ КОРЕЙСЬКОМОВНИМ СЛОВНИКОМ

2.1 Проєктування логічної моделі даних

Логічна модель даних - це формальний опис структури інформації, яку система зберігає та обробляє. Вона визначає, які сутності існують у системі, які атрибути вони мають і як пов'язані між собою. На відміну від фізичної моделі, логічна не залежить від конкретної СУБД чи способу зберігання - вона описує дані з погляду предметної області, а не технічної реалізації, і є проміжною ланкою між концептуальним розумінням системи та її реальною структурою в БД.

Одним із ключових етапів розроблення ССКС є проєктування саме такої моделі. На цьому етапі визначається, які сутності повинні зберігатися в системі, які зв'язки існують між ними та яким чином буде організовано роботу з навчальним і користувацьким контентом. Логічну модель БД ССКС наведено на рис. 5.

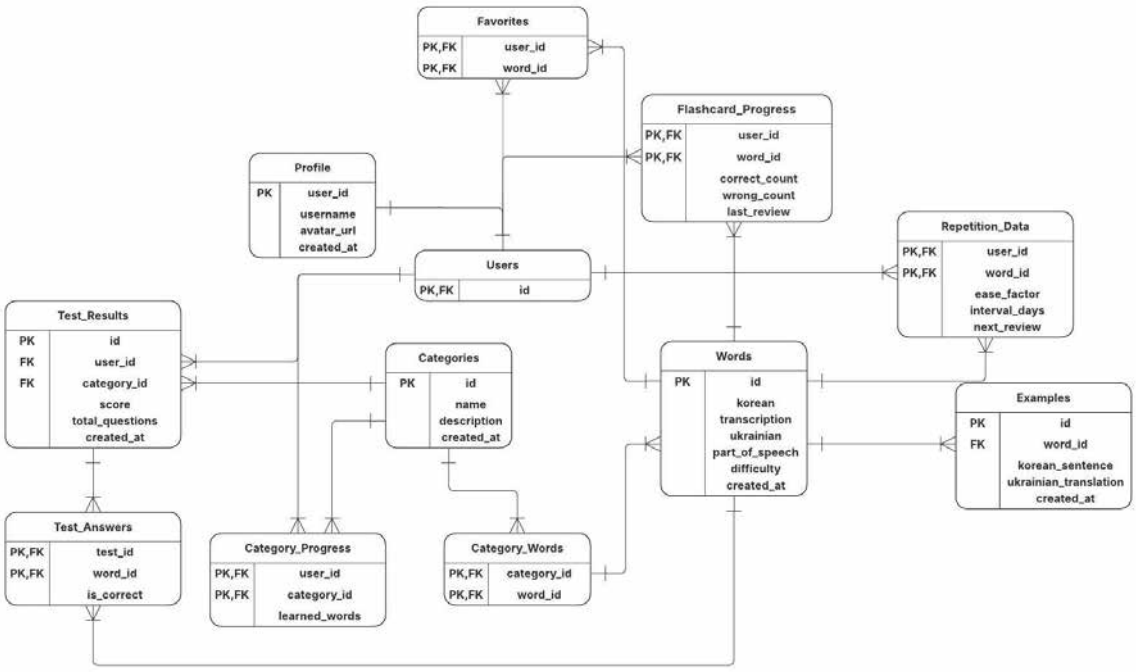

Рис. 5 Логічна модель бази даних

Центральні сутності. В основі структури - три головні сутності: Words, Users / Profile та Categories. Усі інші таблиці є допоміжними і будуються навколо них. Головні сутності наведено нижче:

- **Words.** Центральна таблиця всієї системи. Саме слово є базовою одиницею, навколо якої вибудовується вся навчальна логіка. Таблиця містить: корейське слово, транскрипцію, український переклад, частину мови, рівень складності та дату створення запису.
- **Users + Profile.** Дві пов'язані таблиці для роботи з користувачем. Users відповідає за ідентифікацію та автентифікацію, Profile - за додаткові дані: ім'я користувача, аватар і дату створення. Вони розділені, щоб не змішувати технічні й профільні дані. Зв'язок між ними: 1:1.
- **Categories.** Таблиця тематичних категорій, кожна з яких має назву, опис. Зв'язок між словами та категоріями реалізовано через проміжну таблицю Category_Words, що дозволяє одному слову належати до кількох тем, а одній категорії - містити багато слів. Це формує зв'язок M:M між Categories і Words.

Допоміжні таблиці. Навколо трьох центральних сутностей розташовані таблиці, що відповідають за конкретні функції системи:

- **Examples.** Приклади вживання слова в реченнях з українським перекладом. Одне слово може мати багато прикладів. Зв'язок: Words → Examples (1:M).
- **Favorites.** Слова, збережені користувачем до обраного. Через цю таблицю реалізується зв'язок M:M між Users і Words: один користувач може зберегти багато слів, одне слово - бути збереженим багатьма користувачами.
- **Flashcard_Progress.** Статистика роботи з ФК: кількість правильних і неправильних відповідей, дата останнього перегляду. Зв'язки: Users → Flashcard_Progress (1:M) та Words → Flashcard_Progress (1:M).

- **Repetition_Data.** Дані для організації повторення слів: коефіцієнт складності, інтервал у днях і дата наступного перегляду. Зв'язки аналогічні до Flashcard_Progress: 1:M від Users і Words.
- **Category_Progress.** Прогрес користувача в межах окремих категорій: скільки слів вивчено в кожній темі. Зв'язки: Users → Category_Progress (1:M) та Categories → Category_Progress (1:M).
- **Test_Results.** Загальна інформація про проходження тесту: користувач, категорія, кількість правильних відповідей, загальна кількість питань і дата. Зв'язки: Users → Test_Results (1:M) та Categories → Test_Results (1:M).
- **Test_Answers.** Детальні відповіді в межах конкретного тесту: яке слово було в питанні та чи правильно користувач відповів. Зв'язки: Test_Results → Test_Answers (1:M) та Words → Test_Answers (1:M).

Нормалізація структури. Побудована модель відповідає третій нормальній формі (3НФ). Кожна таблиця містить лише ті атрибути, які безпосередньо залежать від її первинного ключа - жодних транзитивних залежностей між не ключовими полями немає. Наприклад, дані профілю винесено окремо від автентифікаційних даних користувача, приклади речень - окремо від самого слова, а результати окремих відповідей тесту - окремо від загального результату тестування. Такий підхід дозволяє уникнути дублювання інформації та забезпечує цілісність даних при будь-яких змінах або розширенні системи.

2.2 Вибір системи керування БД та середовища її розміщення

Після побудови логічної моделі постає практичне питання: де і як зберігатимуться всі ці дані. Від цього вибору залежить надійність роботи системи, швидкість доступу до інформації та можливість подальшого розвитку веб-застосунку.

Реляційна СУБД. Структура даних у ССКС чітко поділяється на взаємопов'язані сутності: слова, категорії, приклади, користувачі, результати тестів, дані повторення, обране та прогрес. Між ними існують чіткі зв'язки типу 1:M та M:M, які природно виражаються засобами реляційної моделі.

Реляційна СУБД у цьому контексті забезпечує:

a. **Декларативну цілісність даних** - через обмеження NOT NULL, UNIQUE, CHECK та зовнішні ключі (FOREIGN KEY) на рівні самої БД, а не лише на рівні застосунку;
b. **Транзакційність** - операції виконуються повністю або не виконуються взагалі, що критично, наприклад, під час одночасного збереження результату тесту та оновлення прогресу;
c. **Нормалізацію** - дані зберігаються без надлишкового дублювання, а оновлення в одному місці автоматично відображається скрізь, де є відповідний зв'язок.

СУБД PostgreSQL. Як конкретну СУБД обрано PostgreSQL - одну з найпотужніших сучасних реляційних СУБД із відкритим вихідним кодом. Її вибір зумовлений кількома технічними характеристиками, важливими для цього проєкту.

a. **Підтримка складних запитів** - JOIN між кількома таблицями, підзапити, агрегатні функції (COUNT, SUM) використовуються для підрахунку прогресу, статистики тестів і формування ФК;
b. **Зовнішні ключі та каскадні операції** - при видаленні користувача можна автоматично видалити всі пов'язані записи в Favorites, Flashcard_Progress, Test_Results тощо;
c. **Індексація** - прискорює пошук слів за полями korean та ukrainian, що впливає на швидкість роботи словника;
d. **Тригери та функції** - дозволяють автоматизувати певні операції на рівні БД, наприклад оновлення лічильника вивчених слів у Category_Progress після проходження тесту;
e. **Підтримка типів даних** - наприклад, у таблиці Repetition_Data тип даних TIMESTAMP - він зберігає точну дату і час наступного повторення слова,

що дає змогу системі визначати момент, коли це слово потрібно знову подати користувачеві. У таблиці Test_Answers для поля is_correct застосовано тип BOOLEAN, оскільки він дозволяє фіксувати лише два стани відповіді: правильна або неправильна. Для поля ease_factor у таблиці Repetition_Data використано тип NUMERIC, який дає змогу зберігати числове значення коефіцієнта складності слова та використовувати його під час розрахунку інтервалу наступного повторення.

Розташування БД. PostgreSQL у цьому проєкті не розгортається як окремий локальний сервер - він працює у складі хмарної платформи Supabase, представлено на рис.6. З технічної точки зору Supabase є надбудовою над PostgreSQL, яка надає готову інфраструктуру та набір інструментів для роботи з БД у веб-застосунку.

Фізично БД розміщена на серверах Supabase, а взаємодія з нею з боку клієнтської частини відбувається через:

a) **Supabase Client SDK** - бібліотека для JavaScript, яка дозволяє виконувати запити до БД безпосередньо з клієнтського коду;
b) **REST API** - автоматично генерується для кожної таблиці на основі її структури;
c) **Row Level Security (RLS)** - механізм PostgreSQL, який Supabase активно використовує для розмежування доступу на рівні рядків таблиці: наприклад, користувач може бачити лише свої записи в Favorites або Test_Results, навіть якщо запит технічно звертається до всієї таблиці.

Supabase у цьому проєкті виступає не просто як місце зберігання даних, а як комплексна хмарна платформа, що в одному середовищі поєднує:

- **БД PostgreSQL** - зберігання всіх структурованих даних системи;
- **Auth** - вбудована автентифікація через email/пароль, JWT-токени та управління сесіями користувачів; після входу ідентифікатор користувача автоматично доступний у всіх запитах до БД;
- **Storage** - файлове сховище для збереження аватарів користувачів із публічними URL для відображення в інтерфейсі.

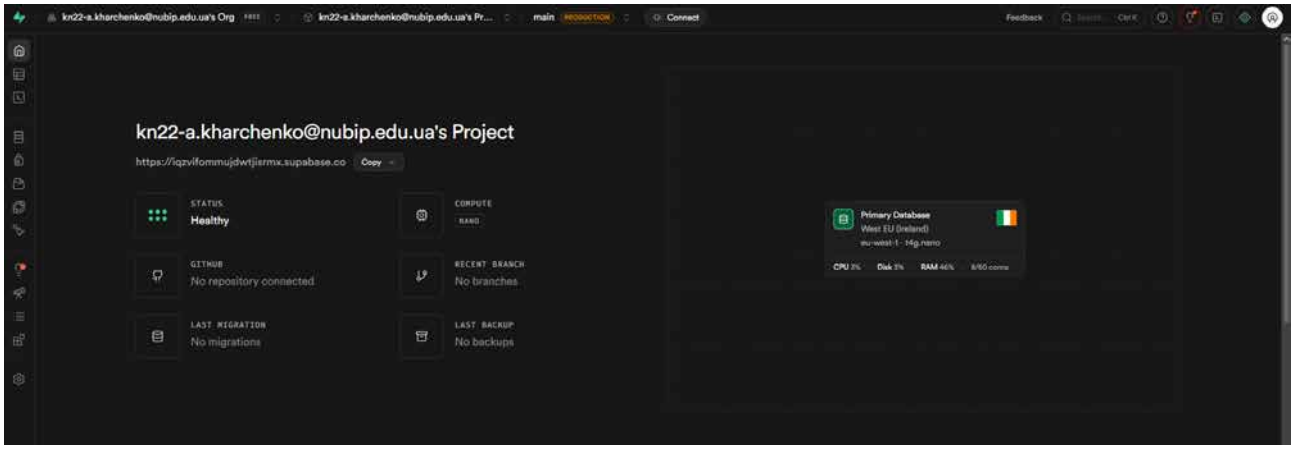

Рис.6 Хмарна платформа Supabase

На практиці обране рішення забезпечує ССКС такі переваги:

a. **Централізоване зберігання.** Словниковий матеріал, персональні дані та результати навчання знаходяться в одній БД із чіткими зв'язками між ними;
b. **Безпеку на рівні даних.** RLS гарантує, що користувач не отримає доступу до чужих записів навіть при прямому зверненні до API;
c. **Швидку взаємодію.** Supabase Client SDK мінімізує кількість коду для виконання типових операцій: вибірки, фільтрації, вставки та оновлення записів;
d. **Масштабованість.** Збільшення словникової бази, кількості категорій або користувачів не потребує змін у структурі зберігання; реляційна модель зберігає системність при будь-якому обсязі даних:
e. **Зручний супровід.** Хмарне розміщення звільняє від необхідності адмініструвати сервер БД вручну.

Таким чином, для ССКС обрано PostgreSQL як СУБД та Supabase як хмарне середовище розміщення. Це рішення є технічно обґрунтованим: реляційна модель відповідає структурі зв'язків між сутностями системи, PostgreSQL забезпечує необхідний набір інструментів для роботи з даними, а Supabase об'єднує БД, автентифікацію, файлове сховище та API в одному середовищі.

2.3 Реалізація структури БД

На основі розробленої логічної моделі бази даних була побудована фізична модель БД, яка описує конкретну реалізацію структури таблиць у системі управління базами даних PostgreSQL.

Фізична модель визначає:

- таблиці бази даних;
- поля таблиць;
- типи даних;
- первинні ключі;
- зовнішні ключі;
- індекси для оптимізації пошуку.

Розроблена БД призначена для зберігання інформації про слова корейської мови, їх категорії, приклади використання, а також для збереження даних користувачів і їхнього прогресу навчання, фізична БД наведена на рис.7.

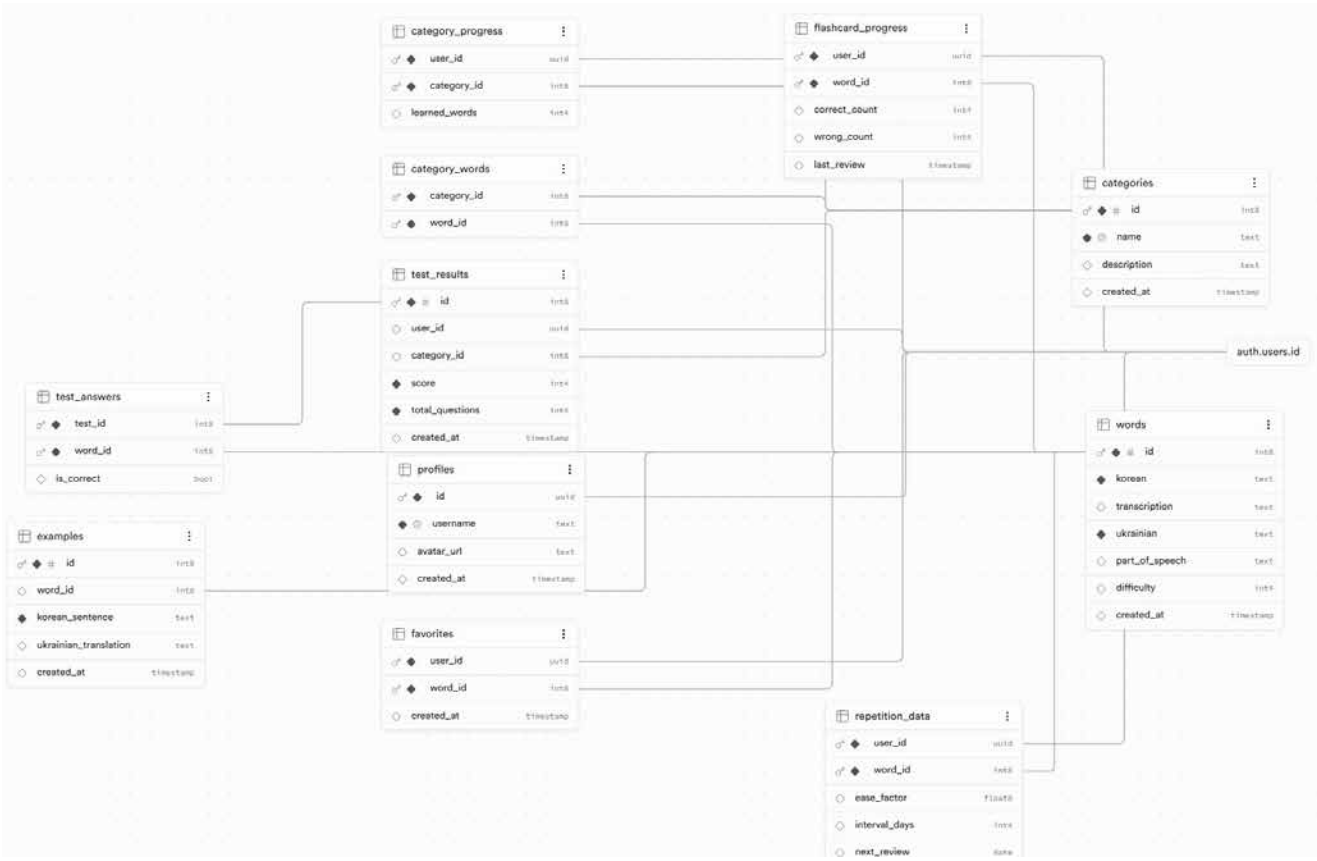

Рис.7 Фізична модель БД

Таблиці. Усі таблиці поділяються на три логічні групи залежно від того, які дані вони зберігають.

А. Словниковий матеріал:

- **Таблиця words** - центральна таблиця всієї системи.

  Містить такі поля:

  - id (PK, SERIAL) - унікальний ідентифікатор запису;
  - korean (TEXT, NOT NULL) - корейське слово;
  - ukrainian (TEXT, NOT NULL) - український переклад;
  - transcription (TEXT) - транскрипція вимови;
  - part_of_speech (TEXT) - частина мови;
  - difficulty (INTEGER) - рівень складності слова;
  - created_at (TIMESTAMP) - дата створення запису.

Ця таблиця є точкою входу для більшості запитів у системі - вона використовується словниковим модулем, підсистемою ФК, тестуванням і пошуком. Усі інші таблиці так чи інакше посилаються на неї через зовнішній ключ word_id.

- **Таблиця categories** - відповідає за тематичні категорії лексики. Містить назву категорії (name), опис (description) і дату створення (created_at). Кожна категорія - це окрема тематична група, наприклад «їжа», «числа» або «привітання».
- **Таблиця category_words** - проміжна таблиця між words і categories зі складеним PK (category_id + word_id). Реалізує зв'язок M:M: одне слово може належати до кількох категорій, одна категорія може містити скільки завгодно слів. Ця таблиця активно використовується при відображенні тематичних добірок, фільтрації лексики та формуванні наборів для тестування і ФК.
- **Таблиця examples** - приклади вживання слів у реченнях. Містить:
  - word_id (FK → words.id) - посилання на слово;
  - korean_sentence (TEXT) - речення корейською;
  - ukrainian_translation (TEXT) - переклад речення українською;
  - created_at (TIMESTAMP) - дата створення.

Зв'язок з words - 1:M: одне слово може мати кілька прикладів. Наявність цієї таблиці дозволяє користувачеві бачити слово не ізольовано, а в реальному мовному контексті.

B. Дані користувача:

- **Таблиця profiles** - зберігає додаткову інформацію про користувачів системи. Кожен запис у цій таблиці відповідає одному користувачу системи. Містить:
    - user_id (PK, FK → auth.users.id) - прив'язка до механізму автентифікації Supabase;
    - username (TEXT) - ім'я користувача в інтерфейсі;
    - avatar_url (TEXT) - посилання на аватар у файловому сховищі Supabase Storage;
    - created_at (TIMESTAMP) - дата реєстрації.

Зв'язок з auth.users - 1:1. Технічні дані для входу (email, хешований пароль, токени) зберігаються в auth.users і керуються Supabase Auth, тоді як profiles містить лише ті дані, що відображаються в інтерфейсі. Такий поділ дозволяє змінювати профільну інформацію незалежно від автентифікаційних даних.

- **Таблиця favorites** - список обраних слів користувача. Складений PK (user_id + word_id) гарантує унікальність пари: одне слово не може бути додане до обраного двічі одним користувачем. Додатково зберігається created_at - дата додавання, яка може використовуватися для сортування списку за часом.

C. Навчальна активність:

- **Таблиця flashcard_progress** - статистика роботи з ФК для кожної пари користувач + слово. Складений PK (user_id + word_id). Містить:
    - correct_count (INTEGER) - кількість правильних відповідей;
    - wrong_count (INTEGER) - кількість помилкових відповідей;
    - last_review (TIMESTAMP) - дата останнього перегляду.

Ці дані дозволяють системі визначати, які слова викликають труднощі, і відповідно формувати пріоритети для повторення.

- **Таблиця repetition_data** - таблиця для підтримки алгоритму інтервального повторення. Також має складений PK (user_id + word_id). Ключові поля:
    - ease_factor (NUMERIC) - коефіцієнт складності слова, який змінюється залежно від успішності відповідей;
    - interval_days (INTEGER) - поточний інтервал між повтореннями в днях;
    - next_review (TIMESTAMP) - дата, коли слово слід показати знову.

На основі цих трьох полів система визначає черговість показу слів у режимі повторення. Чим частіше користувач помиляється, тим менший інтервал і тим швидше слово з'являється знову.

- **Таблиця test_results** - загальний результат кожної тестової спроби. Містить:
    - id (PK, SERIAL) - унікальний ідентифікатор тесту;
    - user_id (FK → auth.users.id) - хто проходив тест;
    - category_id (FK → categories.id) - за якою категорією;
    - score (INTEGER) - кількість правильних відповідей;
    - total_questions (INTEGER) - загальна кількість питань;
    - created_at (TIMESTAMP) - дата проходження.

Кожна спроба проходження тесту зберігається як окремий запис - завдяки цьому можна бачити, як змінюються результати з часом.

- **Таблиця test_answers** - деталізація тесту на рівні окремих слів. Має складений PK (test_id + word_id). Поле is_correct (BOOLEAN) фіксує , чи правильно користувач відповів на конкретне слово.Таким чином, test_results зберігає загальний підсумок, а test_answers - повну картинку: які саме слова були в тесті та де допущено помилки.
- **Таблиця category_progress** - прогрес користувача в межах окремих тематичних категорій. Складений PK (user_id + category_id). Поле

learned_words (INTEGER) показує, скільки слів у цій категорії вже опрацьовано. Використовується для відображення загального прогресу навчання за темами.

Індекси. Для прискорення роботи БД реалізовано два індекси на таблиці words:

- **unique_korean_word** - унікальний індекс на полі korean. Гарантує, що в словнику не може з'явитися два однакові корейські слова. Це обмеження діє на рівні самої БД незалежно від перевірок на стороні застосунку.
- **words_search_idx** — GIN-індекс (Generalized Inverted Index).

GIN-індекс є стандартним інструментом PostgreSQL для повнотекстового пошуку. Функція to_tsvector перетворює текст на набір лексем, а конкатенація korean || ' ' || ukrainian дозволяє шукати одночасно за обома полями. Завдяки цьому пошуковий запит обробляється значно швидше, ніж при використанні оператора LIKE або повного перебору рядків таблиці.

Серверні функції. Частину логіки винесено на рівень БД у вигляді серверних функцій (stored functions). Це дозволяє обробляти складніші запити безпосередньо в PostgreSQL, не перекладаючи всю логіку на клієнтську частину, що зменшує кількість звернень до БД і спрощує код застосунку.

А. Робота зі словником:

- **Функція search_words(query text)** відповідає за пошук слів у БД. Вона використовує механізм повнотекстового пошуку PostgreSQL, що дозволяє знаходити слова як за корейським написанням, так і за українським перекладом. Результат обмежується 20 записами, що забезпечує швидке виконання запиту та зручність використання.
- **Функція get_random_words(limit_count int)** використовується для отримання випадкового набору слів. Вона застосовується у режимах тренування, тестування та ФК, де необхідно забезпечити випадковість вибірки. Кількість слів визначається параметром функції.

- **Функція get_word_of_the_day()** реалізує механізм вибору “слова дня”. Вона визначає слово на основі поточної дати, використовуючи порядковий номер дня місяця.
- **Функція get_examples(word_id_input int)** повертає приклади використання слова у реченнях. Вона отримує всі записи з таблиці прикладів, що відповідають заданому слову, що сприяє кращому розумінню контексту використання лексики.

B. Робота з обраним:

- **Функція add_to_favorites(user_uuid, word_id_input)** додає слово до списку обраних користувача. Використання конструкції ON CONFLICT DO NOTHING дозволяє уникнути дублювання записів і забезпечує коректність даних.
- **Функція get_favorites(user_uuid)** повертає список обраних слів користувача. Вона використовує об’єднання таблиць слів і обраного, що дозволяє отримати повну інформацію про кожне слово.

C. **Тестування та статистика:**

- **Функція save_test_result(...)** призначена для збереження результатів тестування користувача. Вона записує інформацію про користувача, категорію, кількість правильних відповідей і загальну кількість питань у таблицю результатів.
- **Функція get_user_stats(user_uuid)** зумовлює отримання статистики користувача. Вона обчислює загальну кількість пройдених тестів і середній бал, що дозволяє оцінити прогрес у навчанні.

Тригери. У структурі БД реалізовано тригерну функцію normalize_text(), яка автоматично приводить значення поля ukrainian до нижнього регістру за допомогою функції LOWER() перед збереженням запису в таблиці words. Відповідний тригер **trigger_normalize** спрацьовує на подію BEFORE INSERT -тобто нормалізація відбувається до того, як запис потрапляє в таблицю.

Такий підхід має два практичні наслідки: по-перше, забезпечується єдиний формат зберігання текстових даних незалежно від того, в якому регістрі

дані були введені; по-друге, пошук за українським перекладом стає більш стійким, оскільки порівняння відбувається між рядками в однаковому регістрі.

Таким чином, структура БД ССКС організована на трьох рівнях: таблиці зберігають усі дані системи з чіткими зв'язками між ними, індекси та тригери забезпечують цілісність, уніфікацію та швидкодію, серверні функції обробляють частину логіки безпосередньо в PostgreSQL. Разом вони формують технічну основу, на якій працюють усі модулі веб-застосунку.

3 ПРОЄКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір засобів та технологій реалізації

Перед початком розроблення ССКС постало питання: які технології використати, щоб система була зручною у розробці, функціонально повною та придатною до подальшого розвитку? Оскільки ССКС є веб-застосунком, основна увага під час вибору приділялась простоті інтеграції компонентів, ефективності клієнтської частини та надійності зберігання даних. Обраний технологічний стек поділяється на дві частини: фронтенд - те, що бачить користувач, та бекенд - те, що забезпечує збереження даних і серверну логіку. Усі використані засоби та технології зведено в таблиці 3.

Для побудови фронтенду обрано базовий стек веб-розробки: **HTML**, **CSS** і **JavaScript**:

- HTML відповідає за структуру сторінок - навігацію, пошукові форми, словникові блоки, екрани ФК, тестові форми та особистий кабінет користувача.
- CSS забезпечує зовнішній вигляд системи: кольорову схему, типографіку, оформлення карток, кнопок і форм, а також адаптивність інтерфейсу на різних пристроях - від настільних комп'ютерів до мобільних телефонів.
- JavaScript реалізує всю клієнтську логіку: обробку подій, динамічну взаємодію з інтерфейсом, виконання запитів до БД, перемикання між режимами роботи та функціонування всіх модулів системи - пошуку, ФК, тестування та прогресу.

Вибір саме цього стеку є обґрунтованим з кількох причин: по-перше, ці технології є стандартом сучасної фронтенд-розробки та не потребують складного налаштування середовища; по-друге, вони підтримуються всіма сучасними браузерами без додаткової інсталяції на стороні користувача; по-третє, використання чистого JavaScript без надлишкових фреймворків дозволило

зберегти повний контроль над логікою застосунку та зменшити кількість зовнішніх залежностей у проєкті.

Таблиця 3

Засоби та технології реалізації ССКС

Засіб / технологія	**Призначення в системі**
HTML	Формування структури сторінок вебзастосунку
CSS	Оформлення інтерфейсу та адаптивність
JavaScript	Реалізація клієнтської логіки та взаємодії з користуваче

3.2 Проєктування інтерфейсу

Інтерфейс - це те, з чим користувач взаємодіє щодня. Від того, наскільки він зрозумілий і зручний, залежить не лише навігація по системі, а й ефективність роботи з навчальним матеріалом та загальне сприйняття ССКС як цілісного ресурсу. Саме тому проєктування інтерфейсу є одним із ключових етапів розроблення - поряд із проєктуванням БД та вибором технологій.

В основу покладено принцип структурованого мінімалізму: основні функції залишаються легко доступними, а допоміжні елементи не відволікають від головного навчального сценарію. Система поділена на окремі сторінки, кожна з яких відповідає за певний функціональний блок. Такий підхід дозволяє чітко розмежувати сценарії використання та водночас зберегти єдність візуального стилю й логіки навігації по всьому застосунку.

Сторінки системи. Головна сторінка є початковою точкою входу до системи, продемонстрована на рис.8. Її завдання - одразу дати користувачеві розуміння того, для чого призначений ресурс, і забезпечити швидкий доступ до базових можливостей. На сторінці розміщено навігаційне меню, поле швидкого пошуку, короткі інформаційні блоки про можливості системи та блок «слово дня». Слово дня відображає корейське слово разом із транскрипцією та

перекладом і оновлюється щодня на основі поточної дати. Цей елемент виконує не лише інформаційну функцію - він мотивує користувача повертатися до системи регулярно та формує звичку щоденної взаємодії з матеріалом. Для неавторизованого користувача на головній сторінці також відображаються посилання на реєстрацію та вхід, що спрощує перший крок до персоналізованої роботи з системою.

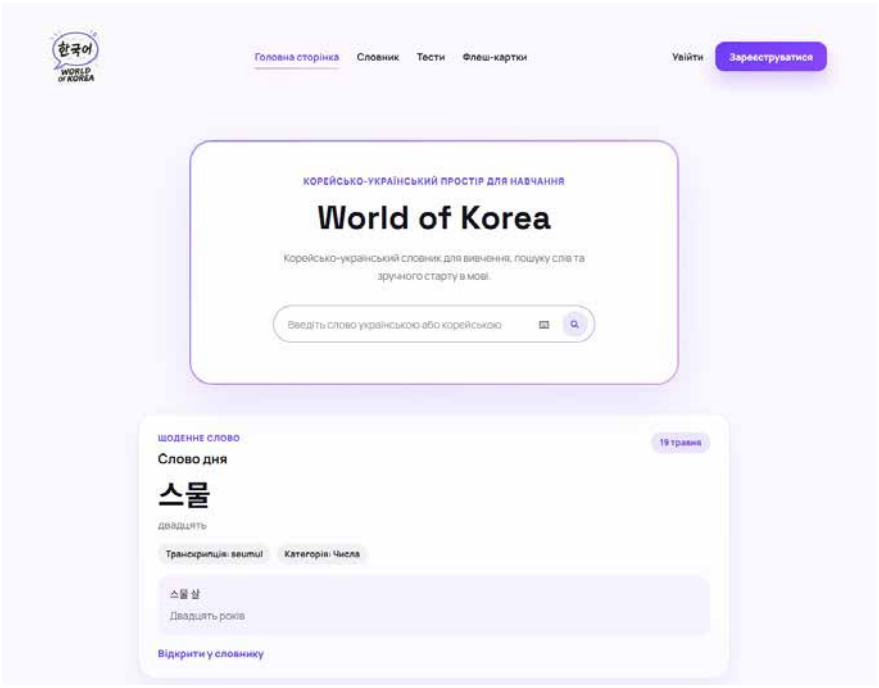

Рис.8 Головна сторінка

Сторінка словника є центральною з погляду роботи з навчальним контентом, продемонстрована на рис.9. Вона підтримує два основні режими взаємодії: пошук за введеним запитом і перегляд слів за тематичними категоріями. Пошуковий блок розміщено у верхній частині сторінки так, щоб він був одразу помітним і зручним як на великих екранах, так і на мобільних пристроях.

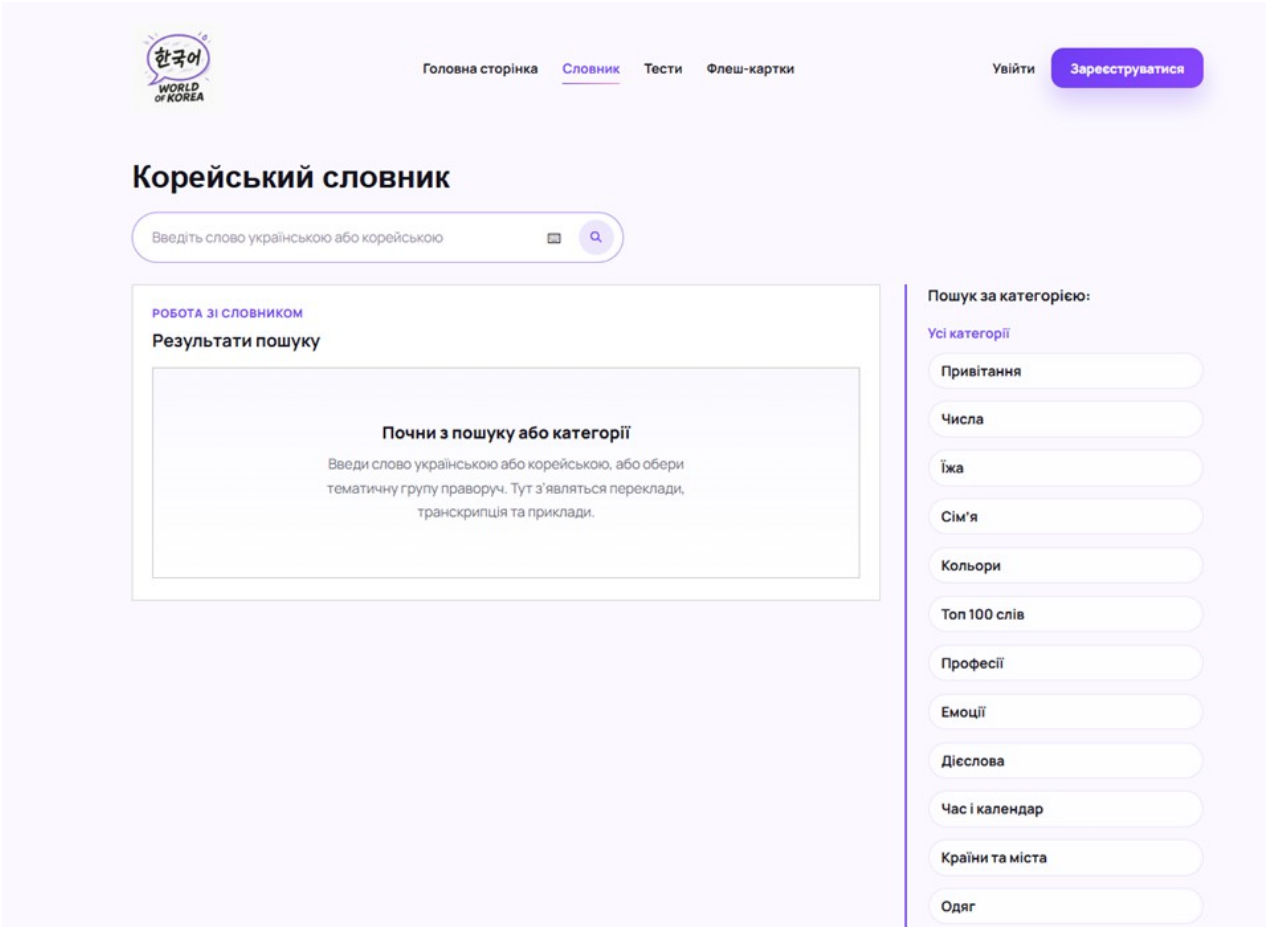

Рис.9 Сторінка словника

Поруч із полем введення передбачено міні-клавіатуру для введення корейських та українських символів - важлива деталь з огляду на те, що більшість користувачів не мають корейської розкладки на своєму пристрої, продемонстровано на рис.10.

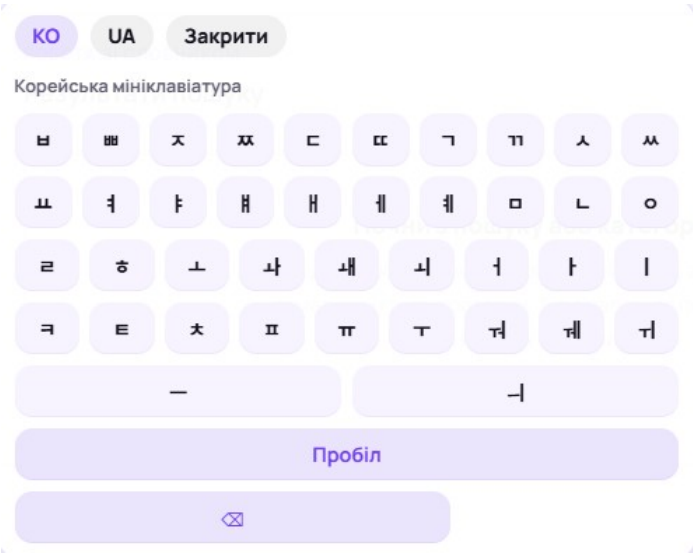

Рис.10 Міні клавіатура корейсько-українська

Нижче розміщено блок тематичних категорій у вигляді кнопок, натискання на які фільтрує лексику за обраною темою. Область результатів є динамічним блоком: залежно від дій користувача в ній відображаються результати пошуку, слова обраної категорії або початковий інформаційний стан із підказкою щодо подальших дій. Кожен результат пошуку або слово категорії відображається у вигляді картки з коротким відображенням основної інформації - корейського слова, транскрипції та перекладу - і є посиланням на детальну сторінку цього слова, продемонстровано на рис.11.

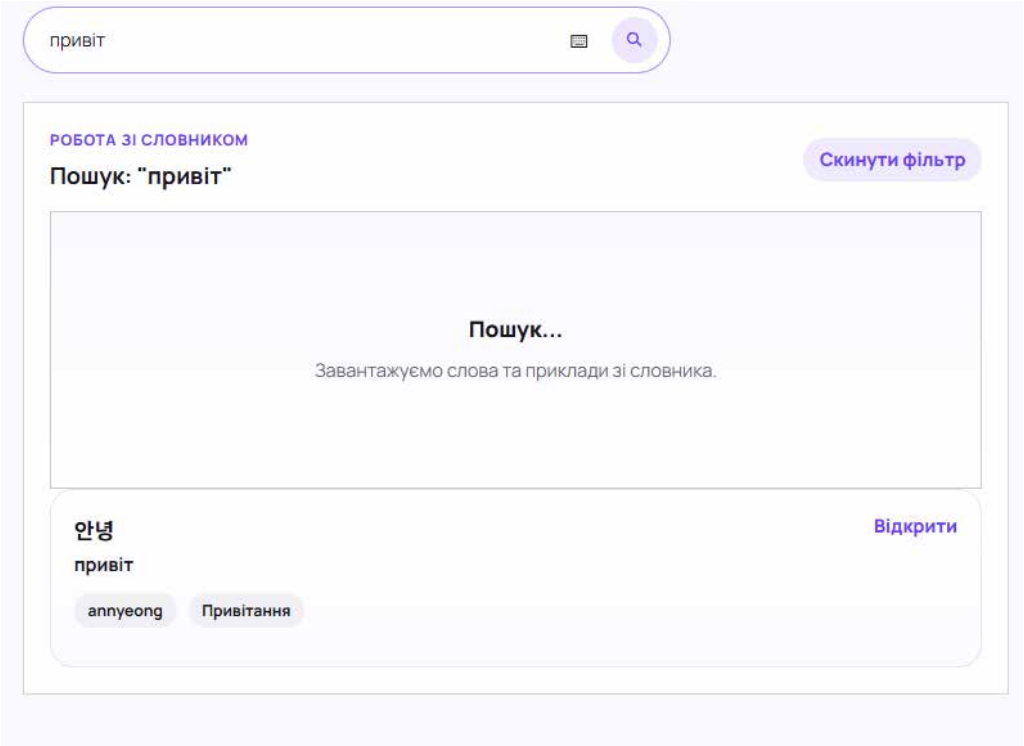

Рис.11 Виведення слова та посилання на детальну сторінку

Сторінка окремого слова. На ній відображається повна інформація про слово. Окремим блоком виведено приклади вживання слова в реченнях разом з українським перекладом - це дозволяє користувачеві бачити слово не ізольовано, а в реальному мовному контексті, можна переглянути на рис.12. На сторінці також передбачено кнопку додавання слова до обраного для авторизованого уористувача, а також навігаційні елементи для швидкого повернення до результатів пошуку.

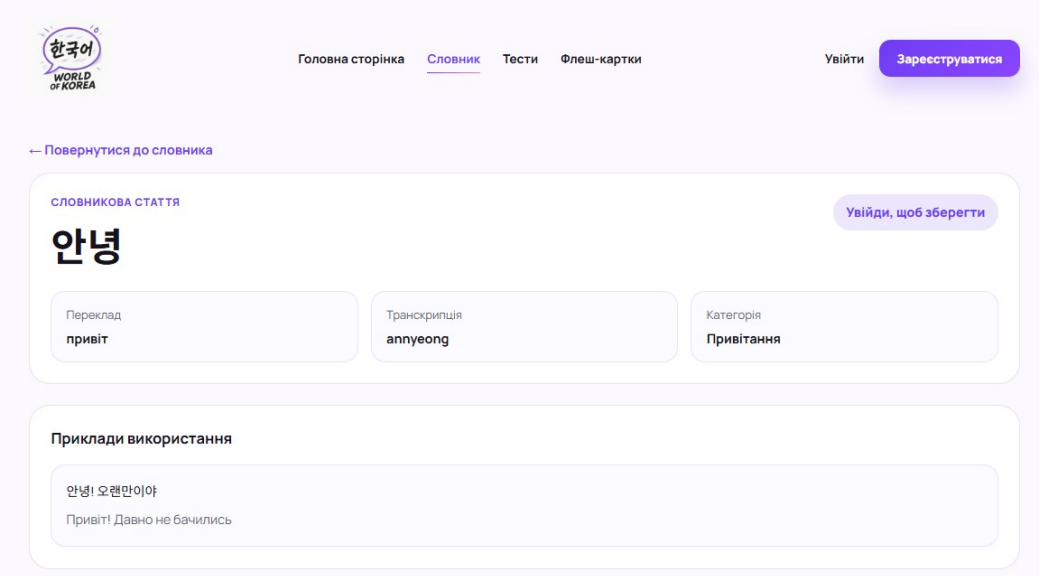

Рис.12 Окрема сторінка для слова

Сторінка ФК спроєктована під короткі та зосереджені навчальні сесії. На початку сесії користувач має змогу налаштувати режим роботи: обрати джерело слів - усі слова словника, слова певної категорії або власне обране - а також задати кількість карток у сесії. Після запуску центральне місце на сторінці займає сама картка: на лицьовому боці відображається корейське слово, а після натискання картка перевертається і показує відповідь, можна переглянути на рис.13.

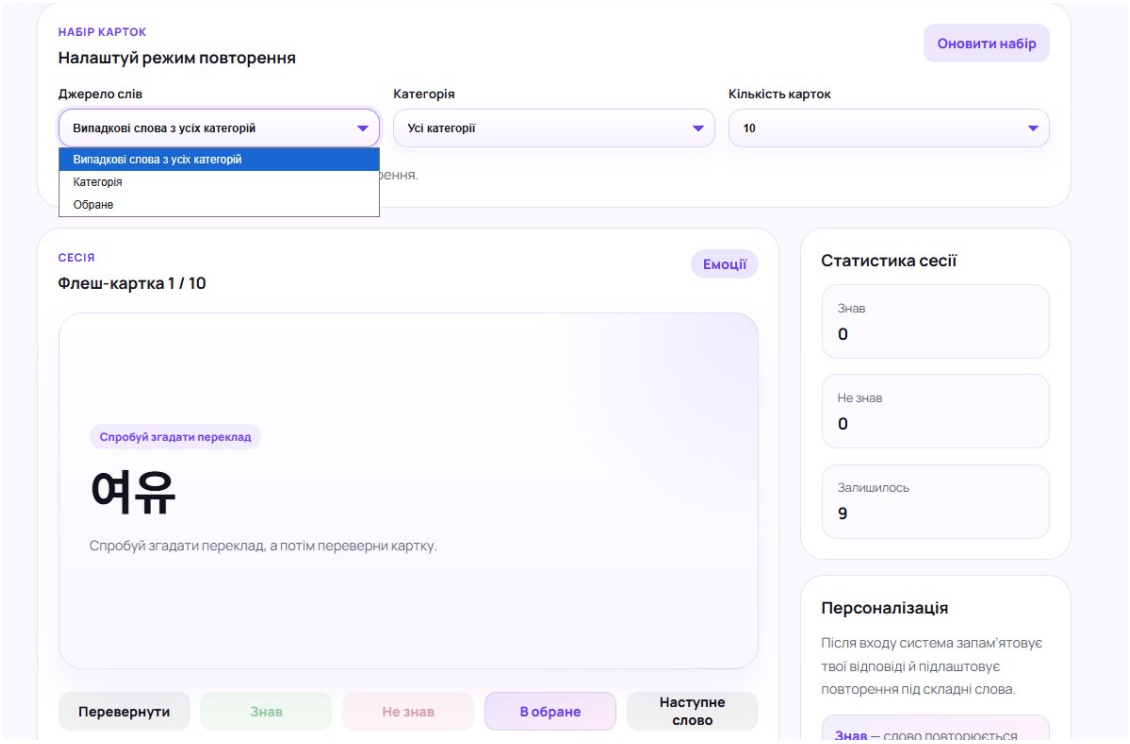

Рис.13 Сторінка ФК

Користувач самостійно оцінює свою відповідь як правильну або неправильну, і на основі цього оновлюється статистика сесії. Додаткові елементи - поточний номер картки, кількість правильних відповідей, кнопка додавання слова до обраного та назва категорії - інтегровані в інтерфейс так, щоб бути доступними, але не відволікати від основного процесу повторення. Після завершення сесії користувач бачить підсумковий екран із результатами.

Сторінка тестування організовує контрольний сценарій у чіткій та послідовній формі, можна переглянути на рис.14. Інтерфейс проходження тесту побудований покроково: спочатку користувач обирає режим, потім запускає тест і послідовно відповідає на кожне питання у форматі вибору однієї правильної відповіді з кількох варіантів. Для кожного питання відображається слово або переклад, а варіанти відповідей подаються у вигляді окремих кнопок. Після вибору відповіді система одразу показує, чи була вона правильною, і дозволяє перейти до наступного питання. Після завершення тесту відображається підсумковий екран із кількістю правильних відповідей, загальним результатом у відсотках. Такий підхід дозволяє користувачеві не лише дізнатися свій результат, а й зрозуміти, на яких словах варто зосередитися під час подальшого повторення.

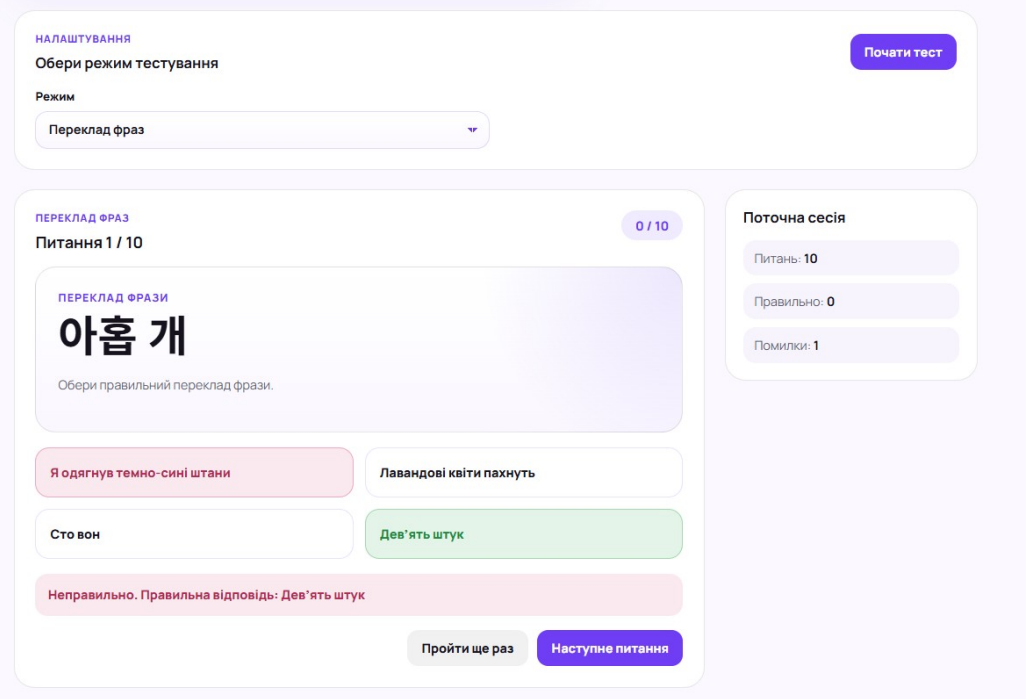

Рис.14 Сторінка тестування

Особистий кабінет є персональним простором авторизованого користувача у системі, його можна переглянути на рис.15. Він поєднує кілька інформаційних блоків, кожен з яких відображає певний аспект взаємодії користувача з матеріалом. У верхній частині розміщено дані профілю: аватар, ім'я користувача з можливістю редагування. Збоку відображаються статистичні показники: загальна кількість пройдених тестів, середній результат, кількість слів у обраному. Окремим блоком виведено список обраних слів із можливістю переходу до детальної сторінки кожного слова або видалення зі списку. Така організація особистого кабінету дозволяє користувачеві в одному місці бачити весь свій прогрес і швидко переходити до потрібного навчального режиму.

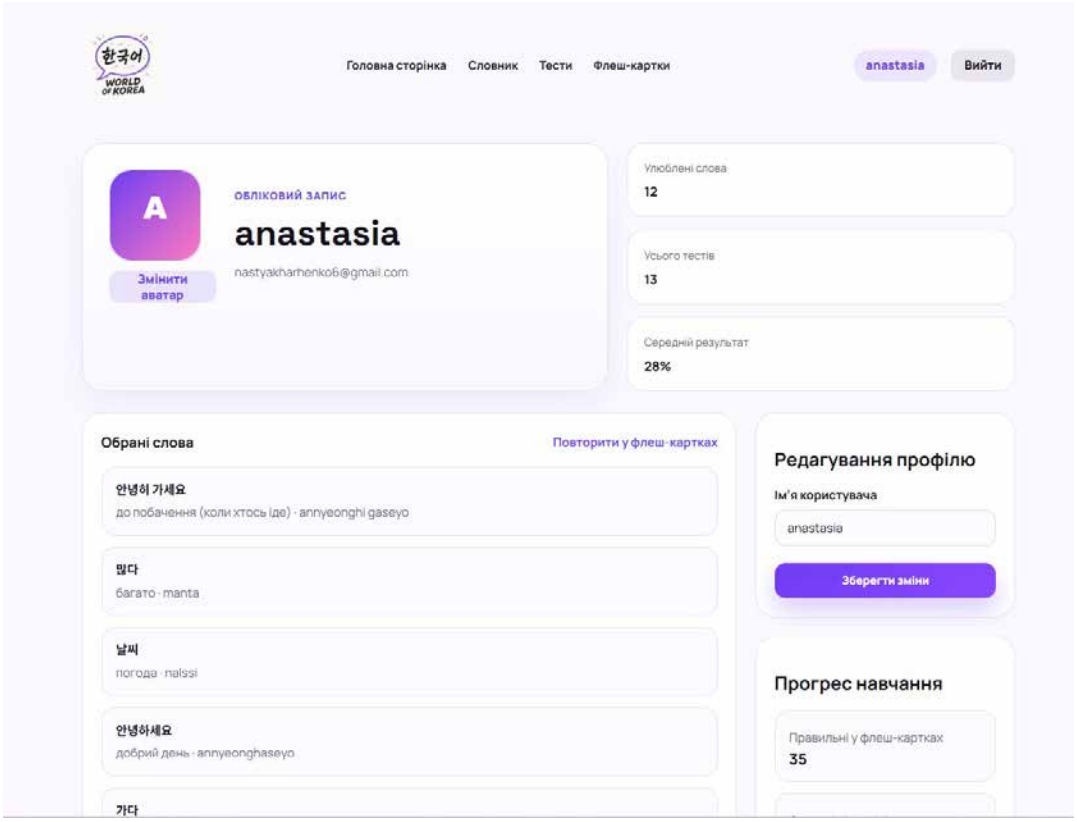

Рис.15 Сторінка особистого кабінету

Сторінки входу та реєстрації спроєктовані як мінімалістичні форми з чітким фокусом на дії, можна переглянути на рис.16. Сторінка реєстрації містить поля для введення email і пароля, а сторінка входу - аналогічну форму з посиланням на реєстрацію для нових користувачів. Після успішної авторизації користувач автоматично перенаправляється на головну сторінку з доступом до всіх персоналізованих функцій системи.

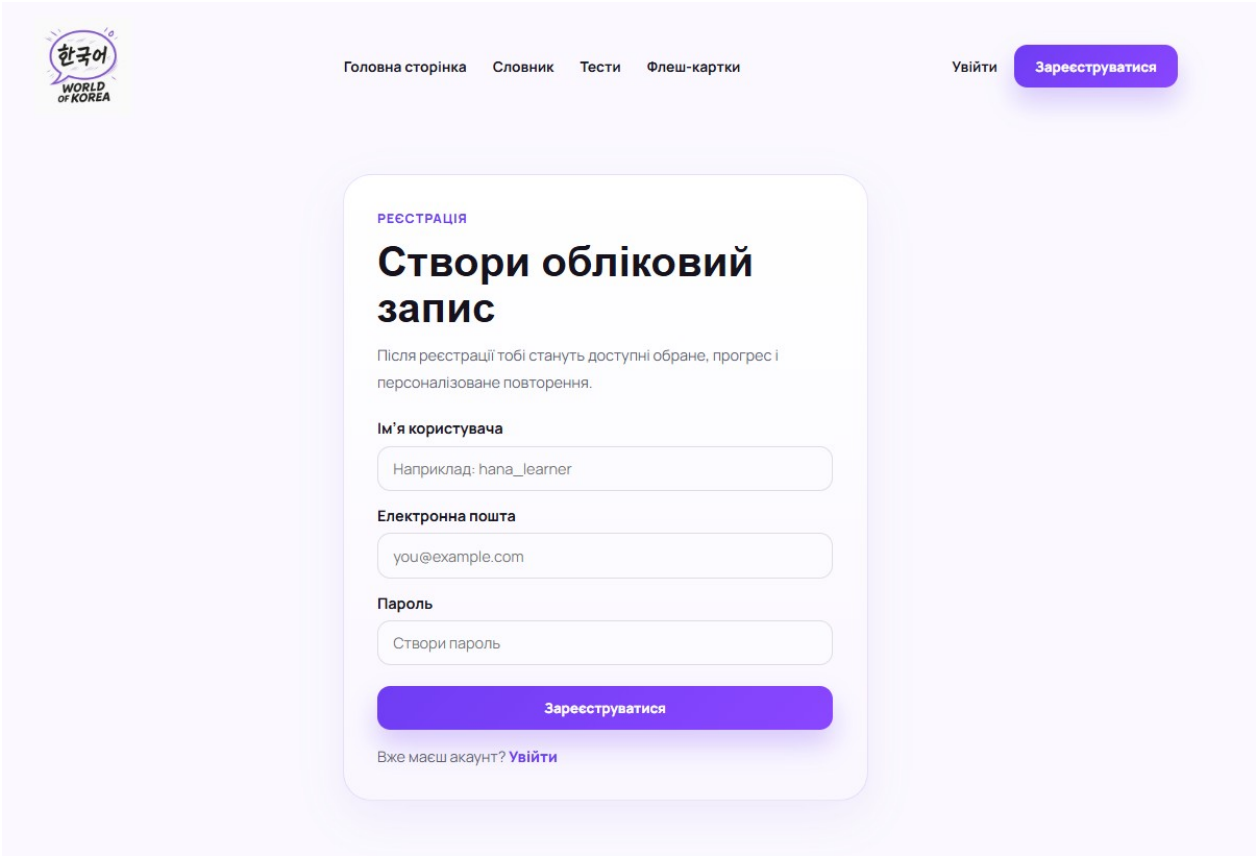

Рис.16 Сторінка входу та реєстрації

Навігація. Верхнє меню присутнє на всіх сторінках веб-застосунку та забезпечує швидкий перехід між всіма основними розділами. Вміст системи залежить від стану авторизації: гість бачить посилання на словник, ФК, тести та кнопки входу і реєстрації, тоді як авторизовані користувачі додатково отримують доступ до особистого кабінету та кнопки виходу із системи. Меню реалізоване динамічно на основі поточного стану сесії користувача і не потребує перезавантаження сторінки.

Адаптивність. Оскільки система має коректно працювати як на комп'ютерах, так і на планшетах та мобільних телефонах, інтерфейс спроєктовано з урахуванням різних розмірів екранів. Для цього використано гнучке розташування елементів на основі CSS: на широких екранах контент розміщується у кількох колонках, тоді як на вузьких - перебудовується в одну. У мобільній версії оптимізовано розміщення логотипа та навігації, пошукового блоку, кнопок категорій, карток слів, екрана ФК і статистичних панелей особистого кабінету. Завдяки цьому система зберігає зручність використання та

функціональну повноту незалежно від пристрою, мобільну версію можна переглянути переглянути в Додатку Б .

Візуальний стиль. Для створення впізнаваного та цілісного образу системи використано єдину кольорову гаму на всіх сторінках: світлий фон як основа, акцентні фіолетові елементи для кнопок, заголовків і активних станів, а також контрастна типографіка для зручного читання. Картки слів, кнопки категорій, елементи ФК і тестові блоки витримані в єдиному стилі, що формує відчуття цілісного продукту. Такий підхід полегшує сприйняття інформації, зменшує когнітивне навантаження під час навчання та підкреслює навчальний характер ресурсу.

3.3 Проєктування бізнес-логіки

3.3.1 Словникова підсистема. Вона є базовою частиною ССКС - саме через неї користувач отримує доступ до основного навчального матеріалу. Її логіка побудована так, щоб забезпечити послідовну роботу зі словом: від пошуку до перегляду детальної інформації та переходу до подальших навчальних дій.

Пошук слова. Користувач вводить запит українською або корейською мовою. Система обробляє його та виконує повнотекстовий пошук у словниковій базі. Результатом є перелік слів, що відповідають запиту, - для кожного відображаються корейське написання, переклад і транскрипція. Якщо відповідних записів не знайдено, система повідомляє про відсутність результатів.

Перегляд за категоріями. Альтернативний спосіб роботи зі словником - без введення запиту. Користувач обирає тематичну категорію, після чого система формує перелік усіх слів, що до неї належать. Такий підхід зручний для послідовного вивчення лексики в межах певної теми.

Словниковий запис. Після вибору слова користувач переходить до його детальної сторінки. Тут система відображає повну інформацію: слово, переклад, транскрипцію, тематичну належність та приклади вживання в реченнях разом із перекладом. Приклади дозволяють бачити слово не ізольовано, а в реальному мовному контексті - що особливо важливо під час вивчення корейської.

Додавання до обраного. Якщо користувач авторизований, він може зберегти слово до персонального списку. Система перевіряє наявність облікового запису та створює зв'язок між користувачем і словом. Якщо слово вже було додано раніше - повторний запис не створюється завдяки обмеженню ON CONFLICT DO NOTHING на рівні БД.

Зв'язок з іншими підсистемами. Слова, знайдені через пошук або категорії, можуть надалі використовуватися у ФК, тестуванні та персоналізованому повторенні. Таким чином словникова підсистема виконує не лише довідкову функцію, а є відправною точкою для всієї подальшої навчальної взаємодії.

3.3.2 Підсистема ФК. Ця підсистема відповідає за закріплення лексичного матеріалу в режимі активного повторення. Її логіка побудована так, щоб користувач не лише переглядав слова, а й самостійно пригадував їх значення, перевіряв себе та поступово формував персональний прогрес.

Формування набору слів. На початку сесії система формує набір карток одним із трьох способів: випадковим добором зі всієї словникової бази, вибіркою за певною категорією або на основі списку обраних слів користувача. Завдяки цьому можна працювати як із загальним набором лексики, так і з конкретною темою чи власноруч відібраними словами.

Цикл роботи з карткою. Картки подаються послідовно. Спочатку користувач бачить лише корейське слово й намагається самостійно пригадати його значення. Після цього він перевертає картку та бачить переклад, транскрипцію і, за наявності, приклад вживання. Такий сценарій забезпечує активну взаємодію з матеріалом і підтримує процес запам'ятовування.

Оцінювання відповіді. Після перегляду правильної відповіді користувач позначає, чи знав слово. На основі цього система фіксує результат: збільшує лічильник правильних відповідей або помилок у таблиці flashcard_progress. Так накопичуються дані про рівень засвоєння окремих слів.

Збереження прогресу та інтервальне повторення. Після кожної взаємодії система оновлює параметри в таблиці repetition_data: коефіцієнт складності (ease_factor), інтервал до наступного показу (interval_days) і дату

наступного перегляду (next_review). Чим частіше користувач помиляється на конкретному слові, тим менший інтервал - і тим швидше слово з'являється знову. Завдяки цьому звичайний режим ФК наближається до формату інтервального навчання.

Додавання до обраного під час сесії. Якщо під час повторення користувач вважає слово важливим або складним, він може одразу зберегти його до персонального списку - без виходу з режиму ФК. Це пов'язує підсистему повторення з функцією обраного та спрощує подальшу роботу з лексикою.

3.3.3 Підсистема тестування. Ця підсистема виконує контрольну функцію в ССКС - вона призначена для перевірки рівня засвоєння матеріалу. Її логіка побудована так, щоб користувач міг проходити перевірку знань у різних форматах, отримувати підсумковий результат і зберігати його в особистому кабінеті.

Вибір режиму та формування запитань. У системі передбачено кілька сценаріїв тестування. Швидкий тест дня формується автоматично з випадкової вибірки слів і призначений для короткої щоденної перевірки. Тестування за категорією добирає слова на основі обраної користувачем теми. Режим перекладу фраз використовує приклади вживання з таблиці examples і формує завдання, у яких потрібно визначити правильний переклад речення. Завдяки цьому тестування охоплює не лише окремі слова, а й роботу з лексикою в контексті.

Проходження тесту. Питання відображаються послідовно. Для кожного з них система показує слово або фразу та кілька варіантів відповіді у вигляді кнопок. Після вибору відповіді система одразу перевіряє її правильність, візуально позначає результат і дозволяє перейти до наступного питання. Такий покроковий сценарій дозволяє чітко розділити окремі кроки проходження та підтримує концентрацію користувача.

Підрахунок результату. Після завершення всіх питань система обчислює кількість правильних відповідей, загальну кількість завдань і формує підсумковий показник у відсотках. На фінальному екрані відображається

результат, а також перелік слів, у яких були допущені помилки - це допомагає зрозуміти, на чому варто зосередитися далі.

Збереження результатів. Для авторизованого користувача система зберігає як загальний результат у таблиці test_results, так і детальні відповіді на рівні окремих слів у таблиці test_answers. Збереження відбувається в межах однієї транзакції - це гарантує, що загальний результат і деталі завжди зберігаються разом. Надалі ці дані використовуються в особистому кабінеті для відображення статистики та аналізу прогресу.

Зв'язок з іншими модулями. Для формування запитань використовуються словникові записи, категорії та приклади вживання. Результати тестів інтегруються з блоком статистики та впливають на загальну картину прогресу. Таким чином тестування є логічним продовженням словникової та тренувальної роботи, а не ізольованим модулем.

3.3.4 Авторизація та особистий кабінет. Ця підсистема забезпечує перехід від загального навчального ресурсу до персоналізованого середовища. Базові функції - перегляд словника, ФК і тестування - доступні без облікового запису, але збереження результатів, робота з обраними словами та відображення прогресу потребують ідентифікації користувача.

Реєстрація та авторизація. Під час реєстрації користувач вводить email та пароль - система через Supabase Auth створює обліковий запис і одночасно формує відповідний запис у таблиці profiles. Під час входу система перевіряє облікові дані та встановлює авторизований стан через JWT-токен. Саме цей стан визначає доступ до особистого кабінету, збереження результатів і персоналізованих функцій. Стан авторизації перевіряється динамічно - меню та доступні дії оновлюються без перезавантаження сторінки.

Керування профілем. Після входу система отримує дані з таблиці profiles і відображає ім'я користувача та аватар в інтерфейсі. Користувач може змінити ім'я або завантажити нове зображення профілю - файл зберігається в Supabase Storage, а посилання на нього оновлюється в profiles. Зміни одразу відображаються в інтерфейсі без перезавантаження.

Обране. Слова, додані до обраного через словник або ФК, зберігаються в таблиці favorites і відображаються в особистому кабінеті окремим блоком. Звідси користувач може перейти до детальної сторінки будь-якого слова або видалити його зі списку. Цей список також доступний як окреме джерело слів у режимі ФК.

Роль у загальній логіці системи. Особистий кабінет не є ізольованим модулем - він безпосередньо пов'язаний зі словником, ФК, тестуванням і обраним. Дані, що формуються в цих підсистемах, надходять до кабінету та дозволяють оцінювати навчальну динаміку в довшій перспективі. Саме тому авторизація та особистий кабінет є завершальним елементом загальної логіки ССКС, у якому поєднуються результати всіх основних навчальних процесів.

3.3.5 Формування звітно-статистичної інформації. Звітно-статистична інформація в системі формується на основі дій користувача під час роботи з ФК, тестуванням і механізмом повторення. Після виконання відповідної дії система створює новий запис або оновлює наявні дані в таблицях flashcard_progress, test_results, test_answers і repetition_data. Усі записи пов’язуються з конкретним користувачем через поле user_id. Під час відкриття особистого кабінету клієнтська частина надсилає запити до БД через Supabase та отримує потрібні дані для подальшої обробки. На основі цих даних система обчислює статистичні показники, зокрема прогрес у ФК, середній результат тестування та кількість слів, які потрібно повторити. Після цього отримані результати відображаються в інтерфейсі особистого кабінету у вигляді звітно-статистичної інформації. Формулювання статистичних показників, наведено нижче.

Формування статистики прогресу у ФК. Статистика прогресу у ФК призначена для відображення результатів роботи користувача з режимом повторення. Вона показує, скільки разів користувач позначав слово як відоме або невідоме під час проходження навчальних сесій.

Джерелом даних для формування цього звіту є таблиця flashcard_progress. У ній зберігаються відомості про взаємодію користувача з окремими словами в режимі ФК. Оцінювання знання слова в даному випадку виконується не

автоматично, а на основі самооцінки користувача. Після перегляду перекладу користувач обирає один із двох варіантів: «знав» або «не знав». Система інтерпретує ці дії як умовно правильну або умовно неправильну відповідь і відповідно оновлює статистичні показники.

Алгоритм формування статистики є таким:

- Користувач відкриває свій особистий кабінет.
- Система визначає user_id поточного користувача.
- Виконується запит до таблиці flashcard_progress.
- Відбираються лише записи, що належать поточному користувачу.
- Обчислюється сума всіх значень correct_count та сума всіх wrong_count.
- Отримані результати відображаються в особистому кабінеті.

Формується значення зміни correct_count, як сума всіх правильних відповідей, а значення зміни wrong_count, як сума всіх неправильних відповідей.

Наприклад, якщо в системі зафіксовано 42 правильні та 15 неправильних відповідей, то ці значення відображаються як поточний результат роботи користувача у ФК. На рис. 17 відображено реалізацію, в Додатку Г наведена блок-схема формування статистики.

Рис. 17 Статистика прогресу у ФК

Такий звіт дозволяє оцінити активність користувача в режимі повторення та рівень засвоєння лексичного матеріалу.

Формування середнього результату тестування. Середній результат тестування використовується для визначення загального рівня успішності користувача за всіма пройденими тестами. Цей показник дає змогу оцінити,

наскільки ефективно користувач засвоює навчальний матеріал у межах контрольного режиму.

Джерелом даних для формування звіту є таблиця test_results, у якій зберігаються результати проходження тестів.

Алгоритм формування показника є таким:

- Користувач відкриває свій особистий кабінет.
- Система визначає user_id поточного користувача.
- Виконується запит до таблиці test_results.
- Відбираються всі записи, що належать поточному користувачу.
- Для кожного тесту обчислюється відсоток правильних відповідей пройденого тесту.
- Обчислюється середнє значення всіх отриманих відсотків.
- Середній результат відображається в особистому кабінеті.

Для розрахунку результату одного тесту у відсотках , використовується вираз: *score / total_questions * 100.*

А для середнього результату використовується: *сума відсотків усіх тестів / кількість тестів.*

Приклад результату звіту:

- Тест 1: 8/10 = 80%
- Тест 2: 7/10 = 70%
- Тест 3: 9/10 = 90%

Таким чином середній результат розраховується:

(80+70+90)/3=80%(80 + 70 + 90) / 3 = 80\%(80+70+90)/3=80%

Такий звіт використовується для оцінювання загального прогресу користувача в тестуванні. Приклад реалізації наведено на рис. 18.

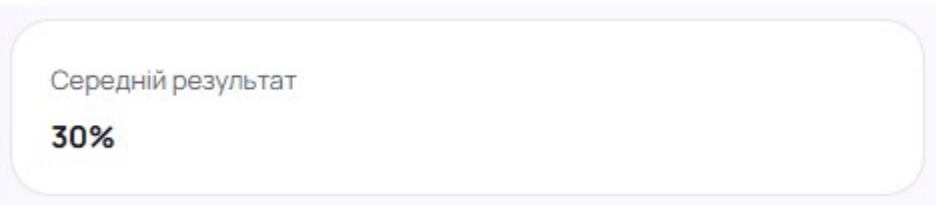

Рис.18 Середній результат тесту

Блок-схема формування статистики тестування наведена в Додатку Г.

Формування списку слів, які потрібно повторити. Ще одним важливим елементом статистичної інформації є визначення кількості слів, дата повторення яких уже настала. Джерелом даних для формування такого звіту є таблиця repetition_data.

Алгоритм формування звіту є таким:

- Користувач відкриває свій особистий кабінет.
- Система визначає user_id поточного користувача.
- Визначається поточна дата.
- Виконується запит до таблиці repetition_data.
- Відбираються записи, що належать поточному користувачу.
- Додатково застосовується умова next_review <= поточна дата, тобто враховуються лише ті слова, час повторення яких вже настав.
- Підраховується кількість отриманих записів.
- Отримане значення відображається як кількість слів, що підлягають повторенню.

Умова відбору, визначається виразом: next_review≤поточна дата

Наприклад, якщо за результатами вибірки отримано 12 записів, то в особистому кабінеті відображається значення: «Слів для повторення: 12».

Логіка роботи цього звіту базується на принципі інтервального повторення. Для кожного слова в системі зберігається дата наступного перегляду, яка оновлюється після взаємодії користувача з ФК. Якщо користувач позначає слово як відоме, інтервал повторення збільшується, а якщо як невідоме - зменшується. Таким чином система автоматично визначає слова, які потребують повторення, і формує відповідний звіт для користувача. Такий показник є важливим елементом персоналізованого навчання, оскільки дозволяє організувати повторення відповідно до рівня засвоєння окремих слів. Приклад реалізації наведено на рис. 20.

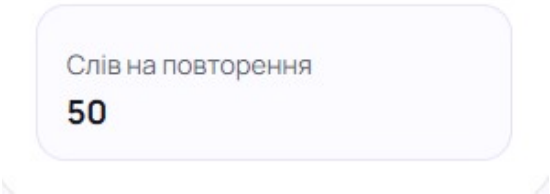

Рис.20 Формування списку слів для повторення

Блок-схему формування статистики повторення слова продемонстровано в Додатку Г.

3.4 Проєктування взаємодії з БД

Щоб усі модулі ССКС працювали злагоджено, клієнтська частина системи повинна мати зручний і надійний спосіб звертатися до БД. У цьому проєкті така взаємодія реалізована через JavaScript та клієнтську бібліотеку Supabase Client, яка забезпечує доступ до таблиць БД, виклик серверних SQL-функцій, виконання операцій select, insert, update, delete, а також роботу з механізмами авторизації.

Підключення до БД. Для централізованого доступу у файлі supabase.js створено спільний клієнт, який використовується в усіх JS-модулях системи, код підключення продемонстровано на рис.21

```
import { createClient } from "https://esm.sh/@supabase/supabase-js@2"
export const supabase = createClient(supabaseUrl, supabaseKey)
```

Рис.21 Підключення до БД

Такий підхід дозволяє не дублювати налаштування підключення в кожному модулі, а використовувати єдиний об'єкт доступу до БД скрізь.

- **Словникова підсистема.**

Пошук слів. Для пошуку використовується SQL-функція search_words, яка викликається через RPC. На рис.22 продемонстрована блок-схема, взаємодії з БД. Перед відправленням запиту введений текст проходить попередню обробку - очищується від зайвих пробілів і нормалізується. Це зменшує кількість помилок і робить результати стабільнішими. . Функція шукає збіги в таблиці words за полями korean та ukrainian, після чого система додатково звертається до category_words, щоб визначити тематичні категорії кожного знайденого слова.

Рис.22 Блок-схема взаємодія пошуку та БД

Категорії. Список тематичних груп завантажується прямим запитом до таблиці categories.

Приклади вживання. Приклади для конкретного слова підвантажуються окремим запитом до таблиці examples за word_id.

Такий підхід дозволяє завантажувати лише потрібні дані, не перевантажуючи сторінку.

Обране. Для додавання слова до обраного використовується SQL-функція add_to_favorites, яка викликається через RPC. На рис.23 продемонстрована блок-схема, взаємодії з БД.

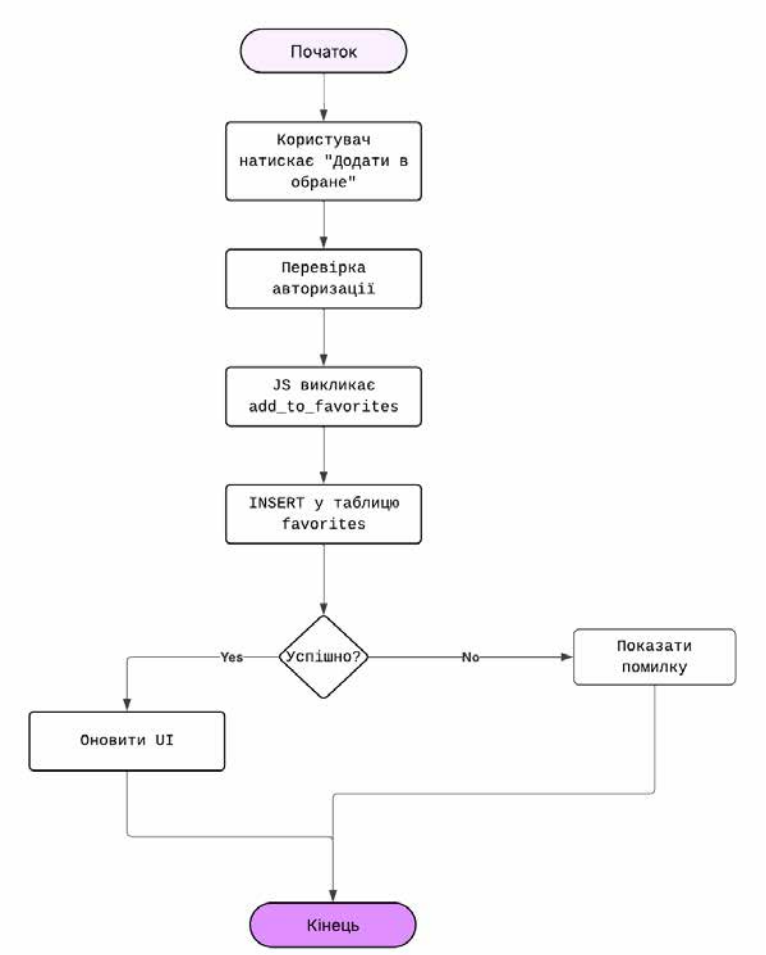

Рис.23 Блок-схема взаємодія додавання в обране з БД

Ідентифікатор користувача визначається на рівні сервера через механізм авторизації Supabase - клієнт передає лише word_id. Видалення слова з обраного виконується через пряму операцію delete.

- **Підсистема ФК.**

Формування набору. Для отримання випадкової вибірки слів використовується функція get_random_words. На рис.24 продемонстрована блок-схема, взаємодії з БД.

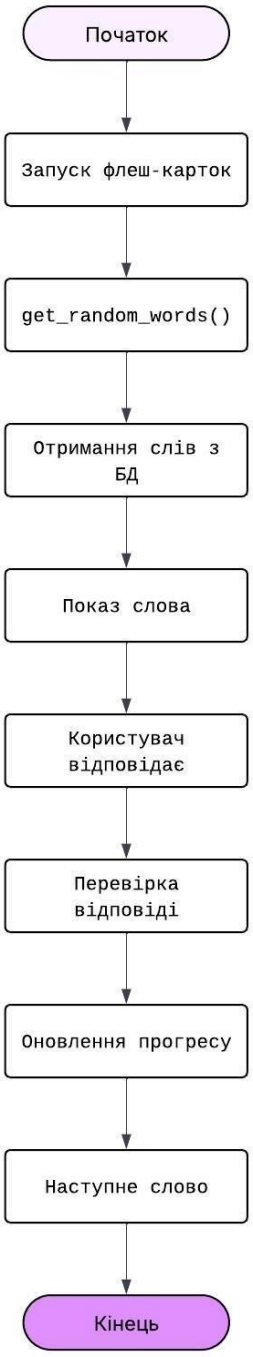

Рис.24 Блок-схема взаємодія формування набору ФК з БД

*Збереження прогресу***.** Після того як користувач оцінює свою відповідь, система оновлює запис у таблиці flashcard_progress.

Якщо запису для цієї пари user_id + word_id ще не існує - виконується insert. Так для кожного користувача накопичується інформація про правильні та неправильні відповіді й дату останнього повторення.

Інтервальне повторення. Окремо оновлюються параметри в таблиці repetition_data - коефіцієнт складності, інтервал і дата наступного перегляду. Залежно від того, чи існує вже запис для конкретного слова, виконується update або insert.

- **Підсистема тестування.**

Формування тесту. Для різних режимів тестування використовуються SQL-функції get_word_of_the_day і get_random_words, а також прямі запити до таблиць categories, category_words, words і examples.

Збереження результатів. Після завершення тесту спочатку зберігається загальний результат у test_results.

Після отримання id нового запису система зберігає деталі відповідей у test_answers.

Обидві операції виконуються в межах однієї транзакції - загальний підсумок і деталі відповідей завжди зберігаються разом.

- **Авторизація та особистий кабінет.**

Реєстрація. Під час реєстрації виконується виклик Supabase Auth.

Після створення облікового запису система формує або оновлює запис у таблиці profiles через upsert.

Персоналізовані дані. В особистому кабінеті кожен блок завантажується окремим цільовим запитом: список обраних слів - з favorites з приєднанням words, статистика тестів - з test_results, прогрес ФК - з flashcard_progress, слова на повторення - з repetition_data. Такий підхід дозволяє завантажувати лише потрібні дані для кожного блоку без зайвих запитів.

Усі основні операції взаємодії з БД зведено в таблиці 4.

Таблиця 4

Основні запити та функції взаємодії з БД

Операція	**Засіб реалізації**	**Призначення**
Пошук слів	rpc("search_words")	Пошук у словниковій базі
Випадкові слова	rpc("get_random_words")	Формування наборів для флеш-карток і тестів
Слово дня	rpc("get_word_of_the_day")	Відображення щоденного слова
Категорії	from("categories").select(...)	Завантаження тематичних груп
Приклади	from("examples").select(...)	Відображення прикладів використання
Додавання в обране	rpc("add_to_favorites")	Збереження слова для користувача
Видалення з обраного	from("favorites").delete(...)	Керування обраними словами
Збереження результату тесту	from("test_results").insert(...)	Фіксація результатів тестування
Збереження відповідей	from("test_answers").insert(...)	Деталізація проходження тесту
Оновлення прогресу карток	from("flashcard_progress").update(...)	Накопичення даних повторення
Оновлення інтервального повторення	from("repetition_data").insert/update(...)	Формування наступних повторень
Реєстрація	supabase.auth.signUp(...)	Створення облікового запису
Профіль	from("profiles").upsert(...)	Створення або оновлення профілю

Таким чином, у проєкті взаємодія з базою даних реалізована через клієнтську бібліотеку Supabase і охоплює всі основні операції: select, insert, update, delete, виклик SQL-функцій через rpc, а також механізми авторизації через supabase.auth. Така модель дозволяє поєднати клієнтську логіку вебзастосунку з централізованим збереженням словникових і персоналізованих даних. Дані словника використовуються в пошуку, категоріях, флеш-картках і тестах, а дані користувача пов’язуються через user_id, завдяки чому обране, результати тестування, прогрес повторення та статистика зберігаються окремо для кожного акаунта.

3.5 Загальна архітектура системи

Загальна архітектура ССКС побудована за модульним принципом і відображає поділ програмного рішення на клієнтську частину, backend-рівень, підсистему авторизації та рівень зберігання даних. Такий підхід забезпечує чітке розмежування відповідальності між основними компонентами системи, спрощує її супровід і створює можливість подальшого розширення функціональних можливостей.

Архітектурно система реалізована як веб-застосунок, у якому користувач взаємодіє з інтерфейсом через браузер. Клієнтська частина охоплює сторінки застосунку, елементи інтерфейсу та клієнтську логіку, реалізовану засобами JavaScript. Саме цей рівень відповідає за відображення словникових записів, результатів пошуку, сторінок ФК, тестів, профілю користувача та інших елементів взаємодії.

Backend-рівень у проєкті реалізовано на основі Supabase. Він забезпечує доступ до даних через API, виконання SQL-функцій і підтримку серверної логіки, необхідної для роботи словника, тестування, флеш-карток та персоналізованих функцій користувача. Окремо в архітектурі виділено підсистему авторизації, яка відповідає за реєстрацію, вхід до системи, керування сесією користувача та його ідентифікацію.

Рівень зберігання даних представлений реляційною базою даних, у якій зосереджено всі основні інформаційні об’єкти системи. Така організація дозволяє підтримувати цілісність даних і забезпечує спільну основу для роботи всіх функціональних модулів.

Діаграму пакетів загальної архітектури системи наведено на рис.25. Вона показує взаємозв’язок між клієнтською частиною, backend-рівнем, підсистемою авторизації та базою даних, а також відображає логіку передавання даних між цими складовими.

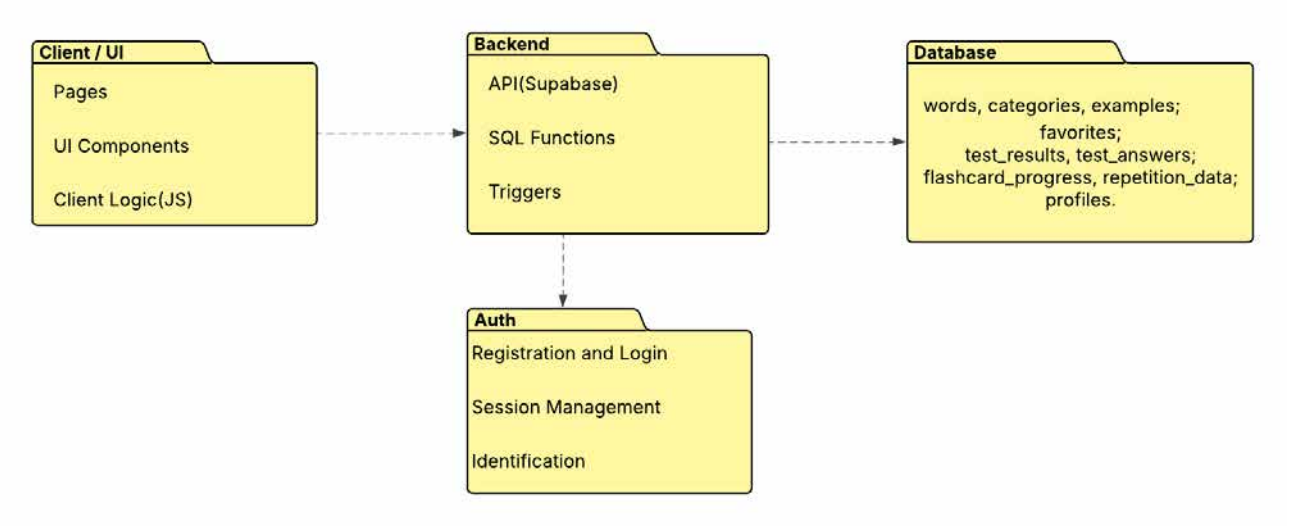

Рис.25 Діаграма пакетів

Таким чином, загальна архітектура системи поєднує інтерфейсний, прикладний та інформаційний рівні в межах єдиного веб-середовища. Це забезпечує узгоджену роботу всіх підсистем, централізоване зберігання даних і підтримку основних сценаріїв взаємодії користувача з навчальним матеріалом.

4 ВПРОВАДЖЕННЯ СИСТЕМИ

4.1 Діаграма розгортання

Діаграма розгортання призначена для подання фізичної архітектури програмної системи та відображення розміщення її основних компонентів на відповідних вузлах, а також канали зв’язку між ними. Загалом діаграма розгортання дає уявлення про технічне середовище, у якому функціонує програмний продукт.

Для ССКС діаграма розгортання наведена на рис.26. Вона відображає трирівневу структуру системи, яка складається з клієнтського вузла, серверного вузла та вузла зберігання даних.

На клієнтському вузлі розміщується користувацьке середовище, представлене клієнтським пристроєм і веб-браузером. У межах браузера виконується клієнтський застосунок World of Korea, реалізований засобами HTML, CSS і JavaScript. Саме цей компонент забезпечує формування інтерфейсу, обробку подій користувача, підготовку запитів до серверної частини та відображення отриманих результатів.

На серверному вузлі розміщується backend-рівень, реалізований у хмарному середовищі Supabase. У межах цього середовища використовуються такі компоненти:

- **REST API / RPC** - забезпечує доступ до таблиць БД і виклик серверних SQL-функцій;
- **Supabase Auth** - реалізує механізми реєстрації, авторизації та ідентифікації користувачів;
- **Supabase Storage** - використовується для збереження файлів;
- **серверні SQL-функції -** виконують окремі операції на рівні БД, наприклад пошук, формування вибірок або роботу з персоналізованими даними.

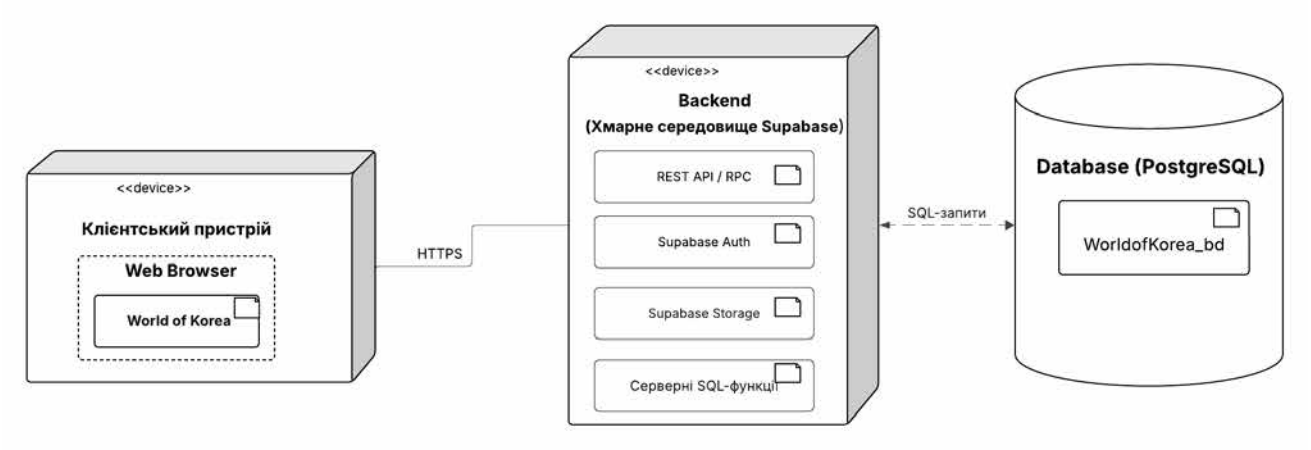

Рис.26 Діаграма розгортання ССКС

Взаємодія між клієнтським вузлом і серверним середовищем здійснюється через протокол HTTPS, що забезпечує передавання запитів і відповідей між браузером та сервісами Supabase.

Окремим вузлом на діаграмі подано **базу даних PostgreSQL**, у якій зберігаються всі основні дані системи. До них належать словникові записи, тематичні категорії, приклади використання, профілі користувачів, результати тестування, прогрес ФК, параметри повторення та списки обраних слів. Взаємодія між Supabase і PostgreSQL здійснюється через SQL-запити, у межах яких виконуються операції вибірки, додавання, оновлення й видалення записів.

4.2 Програмні та апаратні вимоги до системи

Для коректної роботи ССКС необхідно визначити окремо програмні та апаратні вимоги. Оскільки система реалізована як веб-застосунок, основні вимоги пов’язані з наявністю сучасного браузера, доступу до мережі Інтернет і пристрою, здатного забезпечити стабільну роботу клієнтської частини.

Програмні вимоги. З програмної точки зору система не потребує встановлення окремого локального програмного забезпечення, оскільки доступ до неї здійснюється через веб-браузер. Для повноцінної роботи необхідний сучасний браузер, який підтримує виконання JavaScript-коду, обробку HTTPS-запитів, динамічне оновлення сторінок та взаємодію з хмарними сервісами. Програмні вимоги до системи можна переглянути в табл.5.

Клієнтська частина системи побудована на основі HTML, CSS і JavaScript, тому пристрій користувача повинен підтримувати коректне відображення веб-сторінок та виконання сучасних веб-технологій. Це особливо важливо для функцій, пов’язаних із пошуком слів, проходженням тестів, роботою з ФК, авторизацією та особистим кабінетом.

Окремою обов’язковою умовою є наявність доступу до мережі Інтернет, оскільки система постійно взаємодіє з хмарним серверним середовищем Supabase та базою даних PostgreSQL. Без мережевого з’єднання неможливі авторизація користувача, пошук слів, завантаження категорій та інші функції.

На серверному рівні система використовує платформу Supabase, яка забезпечує роботу з базою даних PostgreSQL, механізми автентифікації, файлове сховище та виконання серверних SQL-функцій. У зв’язку з цим окремого локального налаштування серверної частини для користувача не потрібно.

Таблиця 5

Програмні вимоги до системи

Параметр	**Вимога**
Тип застосунку	Веборієнтований
Операційна система	Windows, macOS, Linux, Android, iOS
Браузер	Google Chrome, Mozilla Firefox, Microsoft Edge, Safari та інші браузери
Підтримка технологій	HTML, CSS, JavaScript, HTTPS
Доступ до Інтернету	Обов’язковий
Серверне середовище	Supabase
Система керування БД	PostgreSQL

Апаратні вимоги. З апаратної точки зору система може використовуватися на персональних комп’ютерах, ноутбуках, планшетах і смартфонах. Основною вимогою є здатність пристрою запускати веб-браузер і підтримувати стабільне підключення до Інтернету. Апаратні вимоги до системи, можна переглянути в табл.6.

Для комфортної роботи на персональному комп’ютері або ноутбуці доцільними є такі мінімальні характеристики:

- процесор середнього рівня продуктивності;
- оперативна пам’ять від 4 ГБ;
- екран із роздільною здатністю, достатньою для коректного відображення веб-інтерфейсу.

Для мобільних пристроїв важливою умовою є підтримка актуальної версії браузера та достатній розмір екрана для зручної взаємодії з елементами інтерфейсу. Оскільки система має адаптивний дизайн, її функціональні можливості доступні як на великих екранах, так і на смартфонах, однак якість роботи безпосередньо залежить від стабільності мережевого підключення та продуктивності пристрою.

Таблиця 6

Апаратні вимоги до системи

Параметр	**Вимога**
Тип пристрою	ПК, ноутбук, планшет або смартфон
Процесор	середнього рівня продуктивності
Оперативна пам’ять	від 4 ГБ
Екран	достатній для коректного відображення вебінтерфейсу
Мережеве підключення	стабільний доступ до Інтернету

Таким чином, апаратні та програмні вимоги до системи є помірними й не потребують спеціалізованого обладнання. Це робить систему доступною для широкого кола користувачів і дозволяє працювати з нею на різних типах пристроїв без складного попереднього налаштування.

4.3Інсталяційний пакет

Інсталяційний пакет ССКС містить набір файлів і компонентів, необхідних для запуску клієнтської частини веб-застосунку, підключення до

хмарного backend-середовища та підготовки бази даних. Оскільки система реалізована як веб-застосунок, процес інсталяції полягає не у встановленні виконуваного файлу, а в розгортанні структури проєкту, налаштуванні доступу до Supabase та створенні бази даних.

Склад інсталяційного пакета. До складу інсталяційного пакета входять:

a. Файли клієнтської частини.

- index.html - головна сторінка системи;
- dictionary.html - сторінка словника;
- word.html - сторінка окремого слова;
- flashcards.html - сторінка ФК;
- tests.html - сторінка тестування;
- login.html, register.html - сторінки авторизації;
- account.html - особистий кабінет користувача.

b. Файли стилізації.

- styles.css - стиль інтерфейсу.

c. JavaScript-модулі.

- script.js - логіка словника, пошуку, категорій та слова дня;
- word.js - логіка сторінки окремого слова;
- flashcards.js - логіка роботи флеш-карток;
- tests.js - логіка тестування;
- account.js - логіка особистого кабінету;
- auth.js, auth-ui.js, auth-shared.js - логіка реєстрації, входу та роботи з профілем;
- supabase.js - підключення до Supabase.

d. SQL-файли.

- schema.sql або інший SQL-скрипт створення БД - таблиці, індекси, функції, тригери, код створення можна переглянути в Додатку В.

e. Графічні ресурси.

- логотипи;
- допоміжні зображення інтерфейсу;
- аватари.

Порядок розгортання системи. Розгортання системи доцільно виконувати у такій послідовності:

- Створити проєкт у середовищі Supabase.
- Отримати параметри підключення: supabaseUrl; supabaseKey.
- Створити базу даних PostgreSQL у межах проєкту Supabase.
- Виконати SQL-скрипти створення структури бази даних.
- Завантажити початкові словникові дані.
- Налаштувати файл supabase.js у клієнтській частині.
- Розмістити клієнтські файли на вебсервері або в локальному каталозі розгортання.
- Перевірити працездатність основних модулів системи.

Розгортання бази даних. Розгортання бази даних виконується у SQL Editor середовища Supabase. На цьому етапі послідовно виконуються такі дії:

a. **Створення основних таблиць:** profiles; categories; words; category_words; favorites; flashcard_progress; category_progress; test_results; test_answers; examples; repetition_data.
b. **Створення обмежень цілісності:**
   - первинних ключів;
   - зовнішніх ключів;
   - обмежень унікальності.
c. **Створення індексів:**
   - унікального індексу для корейських слів;

- GIN-індексу для повнотекстового пошуку.

d. **Створення серверних SQL-функцій:**
   - search_words(...);
   - get_random_words(...);
   - get_word_of_the_day();
   - add_to_favorites(...);
   - інших функцій, передбачених структурою бази даних.

e. **Створення тригерів і тригерних функцій**, якщо вони використовуються для нормалізації або автоматичної обробки даних.

Після виконання цих кроків БД набуває завершеної структури та готова до наповнення.

4.4 Тестування системи

Тестування ССКС проводилося з метою перевірки коректності роботи її основних функціональних модулів, виявлення можливих помилок і підтвердження працездатності веб-застосунку в цілому. У процесі тестування особливу увагу було приділено правильності взаємодії клієнтської частини з базою даних, роботі механізмів авторизації, коректності відображення словникового матеріалу та збереженню персоналізованих даних користувача.

У межах тестування перевірялися основні сценарії використання системи, які охоплюють словниковий пошук, перегляд слів за категоріями, роботу з ФК, проходження тестів, реєстрацію та авторизацію користувача, додавання слів до обраного, а також формування особистої статистики.

Тестування словникової підсистеми. Під час тестування словникової підсистеми перевірялася коректність пошуку слів українською та корейською, відображення результатів пошуку, завантаження словникових категорій і відкриття сторінки окремого слова. Було встановлено, що система коректно обробляє введений запит, виконує пошук у словниковій базі та відображає знайдені слова разом із перекладом, транскрипцією та додатковими відомостями.

Окремо перевірялося відображення прикладів використання слова та його належності до певної категорії. Результати тестування показали, що словникова підсистема правильно взаємодіє з таблицями words, examples, categories і category_words, забезпечуючи повне відображення словникового запису.

Тестування підсистеми ФК. Під час перевірки підсистеми ФК тестувалися основні сценарії: формування набору слів, перевертання картки, оцінювання відповіді та збереження результатів сесії. Перевірка підтвердила, що система коректно формує вибірки в різних режимах, відображає слова у форматі карток і дозволяє послідовно проходити навчальну сесію без збоїв.

Окремо перевірялося оновлення даних у таблицях flashcard_progress і repetition_data. Після кожної взаємодії з карткою система зберігає кількість правильних і неправильних відповідей, дату останнього повторення, а також параметри наступного перегляду слова - все це фіксується коректно та відповідає очікуваній логіці.

Тестування підсистеми тестування. Для підсистеми тестування перевірялися вибір режиму тесту, формування запитань, перевірка відповідей і підрахунок підсумкового результату. Тестування показало, що система коректно формує набір завдань, відображає питання з варіантами відповідей і правильно визначає результат після завершення.

Окремо перевірялося збереження результатів у БД. Підтверджено, що загальний підсумок проходження записується в таблицю test_results, а детальні відповіді на рівні окремих слів - у test_answers. Завдяки цьому вся історія тестування доступна користувачеві в особистому кабінеті у вигляді статистики.

Тестування авторизації та особистого кабінету. Під час тестування підсистеми авторизації перевірялися сценарії реєстрації нового користувача, входу до системи, виходу з облікового запису та доступу до персоналізованих функцій. Результати показали, що механізм авторизації через Supabase Auth працює коректно, а після успішного входу користувач отримує доступ до особистого кабінету.

В особистому кабінеті додатково перевірялося відображення профілю, обраних слів, результатів тестування, статистики ФК і даних прогресу. Було

встановлено, що система правильно завантажує персоналізовану інформацію з БД і відображає її у відповідних блоках інтерфейсу.

Тестування роботи з БД. Окремим напрямом тестування була перевірка взаємодії системи з базою даних. Для цього контролювалися операції select, insert, update, delete, а також виклики SQL-функцій через RPC. Перевірка показала, що клієнтська частина правильно взаємодіють з БД PostgreSQL через середовище Supabase, а дані зберігаються й оновлюються відповідно до дій які виконували користувачі.

Також було перевірено цілісність збережених даних після реєстрації користувача, проходження тестів, роботи з ФК та додавання слів до обраного. У результаті підтверджено, що всі основні модулі системи використовують спільну БД узгоджено та без порушення логіки зберігання інформації.

Загальні результати тестування. Проведене тестування показало, що ССКС коректно реалізує основні функціональні можливості, передбачені під час проєктування. Усі ключові модулі системи працюють узгоджено, забезпечують правильне відображення, обробку та збереження даних, а також підтримують основні сценарії взаємодії користувача з навчальним матеріалом, можна переглянути на табл.7.

Таблиця 7

Основні сценарії тестування системи

№	**Сценарій тестування**	**Очікуваний результат**	**Фактичний результат**
1	Введення запиту українською мовою в поле пошуку	Відображаються відповідні словникові записи з перекладом, транскрипцією та додатковими відомостями	Відповідає очікуваному результату
2	Введення запиту корейською мовою в поле пошуку	Відображаються відповідні словникові записи з перекладом і супровідною інформацією	Відповідає очікуваному результату
3	Вибір тематичної	Формується та відображається	Відповідає

№	Сценарій тестування	Очікуваний результат	Фактичний результат
	категорії	список слів, що належать до вибраної категорії	очікуваному результату
4	Перехід до сторінки окремого слова	Відображається повний словниковий запис із прикладами використання	Відповідає очікуваному результату
5	Запуск режиму флеш-карток	Формується навчальна вибірка слів і відображається перша картка	Відповідає очікуваному результату
6	Перегортання флеш-картки	Відображається правильна відповідь, транскрипція та додаткові дані слова	Відповідає очікуваному результату
7	Позначення результату відповіді у флеш-картках	Результат зберігається, після чого система переходить до наступної картки	Відповідає очікуваному результату
8	Запуск тестування	Формується набір тестових завдань і відображається перше питання	Відповідає очікуваному результату
9	Вибір відповіді під час проходження тесту	Система виконує перевірку відповіді та переходить до наступного завдання	Відповідає очікуваному результату
10	Завершення тестування	Обчислюється та відображається підсумковий результат проходження тесту	Відповідає очікуваному результату
11	Реєстрація нового користувача	Створюється обліковий запис і формується профіль користувача	Відповідає очікуваному результату
12	Авторизація	Користувач отримує доступ до	Відповідає

№	Сценарій тестування	Очікуваний результат	Фактичний результат
	користувача в системі	персоналізованих функцій системи	очікуваному результату
13	Додавання слова до списку обраного	Слово зберігається у персональному списку користувача	Відповідає очікуваному результату
14	Відкриття особистого кабінету	Відображаються дані профілю, обрані слова, статистика та прогрес користувача	Відповідає очікуваному результату

Таким чином, результати тестування підтверджують працездатність розробленого веб-застосунку та його придатність до використання як інструмента для вивчення корейської мови, повторення словникового матеріалу, проходження тестування та відстеження персонального прогресу.

ВИСНОВКИ

У результаті виконання дипломної роботи розроблено ССКС - веб-застосунок, орієнтований на підтримку самостійного вивчення корейської мови в межах єдиного середовища. Головна ідея роботи полягала в тому, щоб поєднати всі основні функції в одному місці - без необхідності перемикатися між різними сервісами.

Аналіз предметної області та існуючих аналогів показав, що сучасні ресурси здебільшого добре реалізують щось одне: переклад, повторення або тестування.

У ході роботи було сформульовано функціональні та нефункціональні вимоги, спроєктовано логічну модель БД, обґрунтовано вибір PostgreSQL і Supabase, реалізовано структуру БД для зберігання словникових записів, категорій, прикладів, результатів тестів і персоналізованих даних користувача. Також спроєктовано інтерфейс веб-застосунку, побудовано бізнес-логіку основних підсистем і реалізовано всі ключові модулі.

Створена система забезпечує: пошук слів українською та корейською мовами; перегляд лексики за тематичними категоріями; роботу з прикладами вживання; повторення матеріалу через ФК; проходження тестів; додавання слів до обраного; перегляд статистики та персонального прогресу.

Проведене тестування підтвердило, що всі основні модулі функціонують коректно, взаємодія з БД виконується стабільно, а результати роботи користувача зберігаються та відображаються відповідно до передбаченої логіки.

Таким чином, мету дипломної роботи досягнуто, а всі визначені завдання виконано. ССКС є працездатним веб-застосунком навчального призначення, який може використовуватися як інструмент для вивчення корейської мови, повторення словникового матеріалу, перевірки знань і відстеження власного прогресу. Практичне значення роботи полягає у створенні програмного рішення, яке може стати основою для подальшого розвитку подібних мовних навчальних систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. **Duolingo.** About Duolingo [Електронний ресурс]. – Режим доступу: https://about.duolingo.com/
2. **Duolingo.** Duolingo [Електронний ресурс]. – Режим доступу: https://en.duolingo.com/
3. **Google.** Google Translate [Електронний ресурс]. – Режим доступу: https://translate.google.com/
4. **Google.** Google Translate Help [Електронний ресурс]. – Режим доступу: https://support.google.com/translate/
5. **Happy Monday.** Українську мову вивчають у 107 країнах світу: що показав звіт про мовні тренди 2025 року [Електронний ресурс]. – Режим доступу: https://happymonday.ua/vyvchennia-mov-u-2025-preply
6. **MDN Web Docs.** CSS: Cascading Style Sheets [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/CSS (дата звернення: 20.05.2026).
7. **MDN Web Docs.** Fetch API [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
8. **MDN Web Docs.** HTML: HyperText Markup Language [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/HTML (дата звернення: 20.05.2026).
9. **MDN Web Docs.** JavaScript [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/JavaScript (дата звернення: 20.05.2026).
10. **Ministry of Culture, Sports and Tourism of the Republic of Korea.** 2024 Overseas Hallyu Survey Revealed 70% Korean Wave Experiencers View K-Content Positively [Електронний ресурс]. – Режим доступу: https://www.mcst.go.kr/english/policy/pressView.jsp?pSeq=383
11. **NAVER.** NAVER Dictionary [Електронний ресурс]. – Режим доступу: https://dict.naver.com/
12. **NAVER.** Papago [Електронний ресурс]. – Режим доступу: https://papago.naver.com/

13. **Object Management Group**. UML – Unified Modeling Language [Електронний ресурс]. – Режим доступу: https://www.omg.org/uml/
14. **Object Management Group.** What is UML? [Електронний ресурс]. – Режим доступу: https://www.omg.org/uml/what-is-uml.htm
15. **Освіта.ua.** Мовні тренди в Україні 2025: що вивчають українці і скільки за це платять [Електронний ресурс]. – Режим доступу: https://osvita.ua/vnz/94777/
16. **PostgreSQL Global Development Group.** CREATE FUNCTION [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/current/sql-createfunction.html
17. **PostgreSQL Global Development Group.** CREATE TABLE [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/current/sql-createtable.html
18. **PostgreSQL Global Development Group.** PostgreSQL Documentation [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/
19. **Preply.** Most learned languages in the world, America, and Europe: Global language learning statistics and trends by Preply [Електронний ресурс]. – Режим доступу: https://preply.com/en/blog/global-language-learning-report/
20. **Quizlet.** About Quizlet [Електронний ресурс]. – Режим доступу: https://quizlet.com/mission
21. **Study in Korea.** Study in Korea | Run by Korean Government [Електронний ресурс]. – Режим доступу: https://www.studyinkorea.go.kr/
22. **Supabase.** Auth [Електронний ресурс]. – Режим доступу: https://supabase.com/docs/guides/auth
23. **Supabase.** Database [Електронний ресурс]. – Режим доступу: https://supabase.com/docs/guides/database/overview
24. **Supabase.** JavaScript: Introduction [Електронний ресурс]. – Режим доступу: https://supabase.com/docs/reference/javascript/introduction
25. **Supabase.** Storage [Електронний ресурс]. – Режим доступу: https://supabase.com/docs/guides/storage
26. **TOPIK.** TOPIK 한국어능력시험 [Електронний ресурс]. – Режим доступу: https://www.topik.go.kr/

27. **Голос України.** Новини [Електронний ресурс]. – Режим доступу: https://www.golos.com.ua/news/9516
28. **БФК.** [Електронний ресурс]. – Режим доступу: http://bkeipr.com/
29. **Logical Data Model** [Електронний ресурс]. – Режим доступу: https://www.1keydata.com/datawarehousing/logical-data-model.html
30. **Data normalization**: the Third Normal Form (3NF) [Електронний ресурс]. – Режим доступу: https://uk.itpedia.nl/2025/05/02/datanormalisatie-de-derde-normaalvorm-3nf-is-essentieel-voor-je-database/
31. **СУБД.** Які бувають та як вибрати [Електронний ресурс]. – Режим доступу: https://highload.tech/uk/subd-yaki-buvayut-yak-vibrati/
32. Схеми бази даних [Електронний ресурс]. – Режим доступу: https://foxminded.ua/skhemy-bazy-danyh/
33. **Інтерфейс.**Що таке UI [Електронний ресурс]. – Режим доступу: https://wezom.com.ua/ua/blog/chto-takoe-ui-i-kak-polzovatelskij-interfejs-vliyaet-na-prodazhi-internet-magazina

ДОДАТКИ

ДОДАТОК А
UML-діаграми

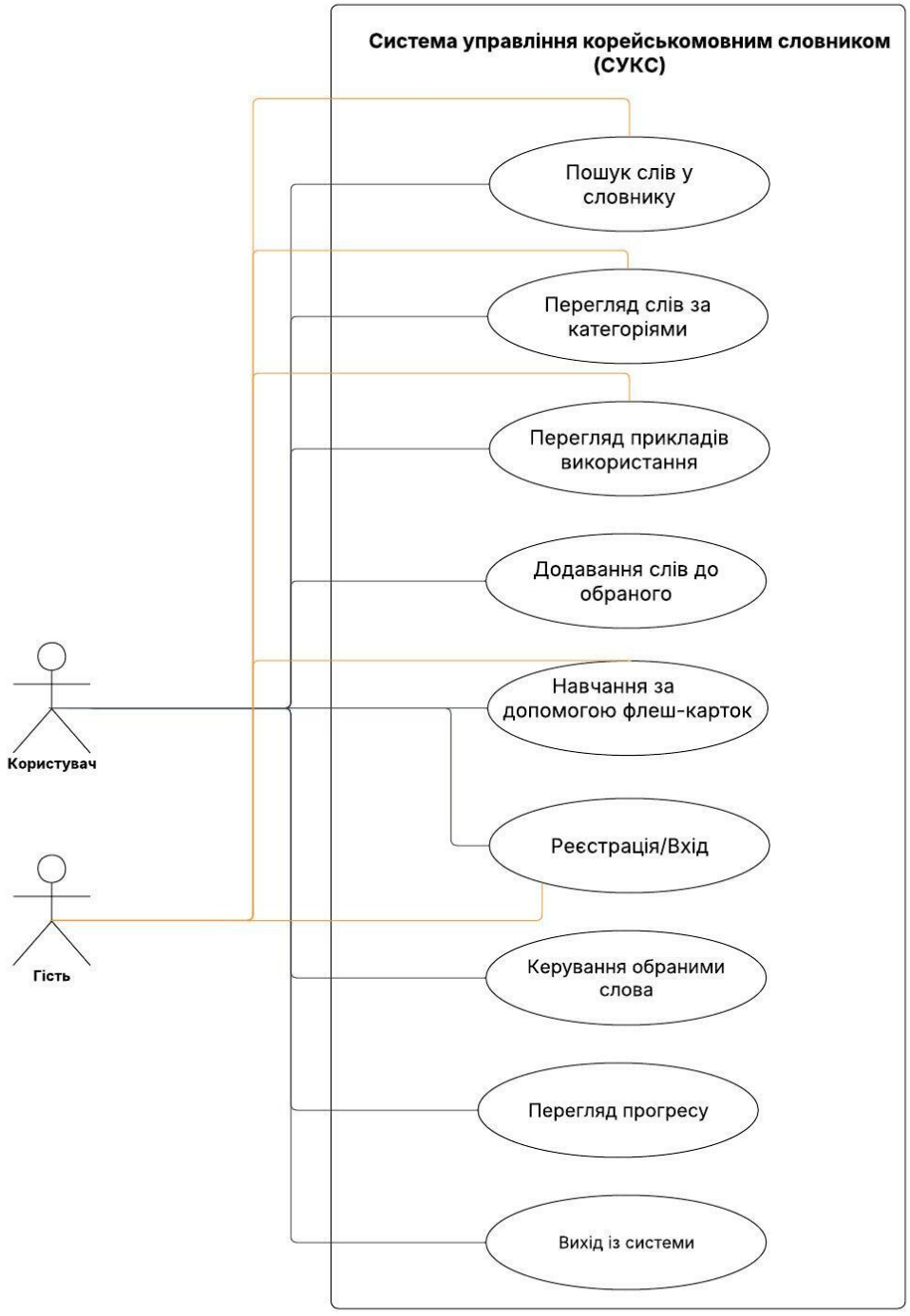

Рис.1 Діаграма прецедентів

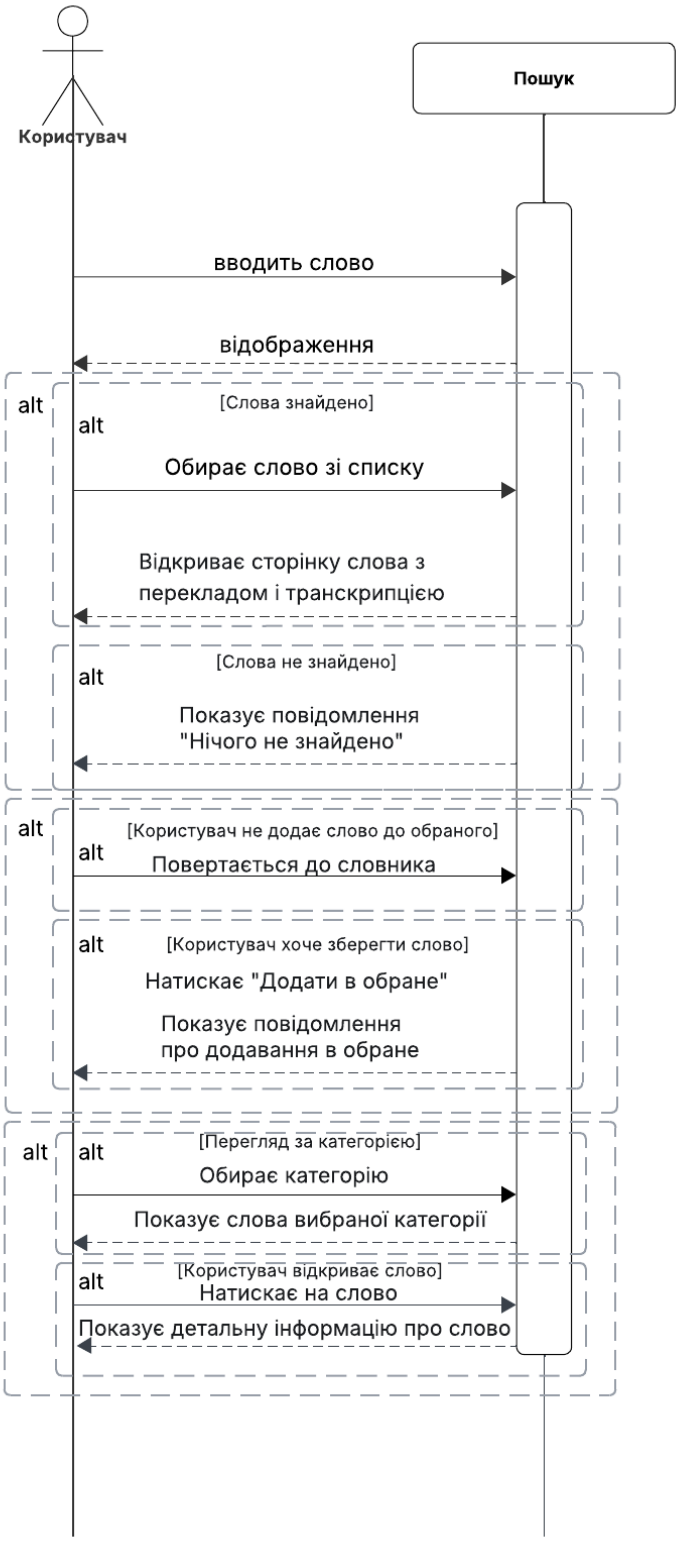

Рис.2 Діаграма послідовності Пошук слова

Рис.3 Діаграма послідовності Тестування

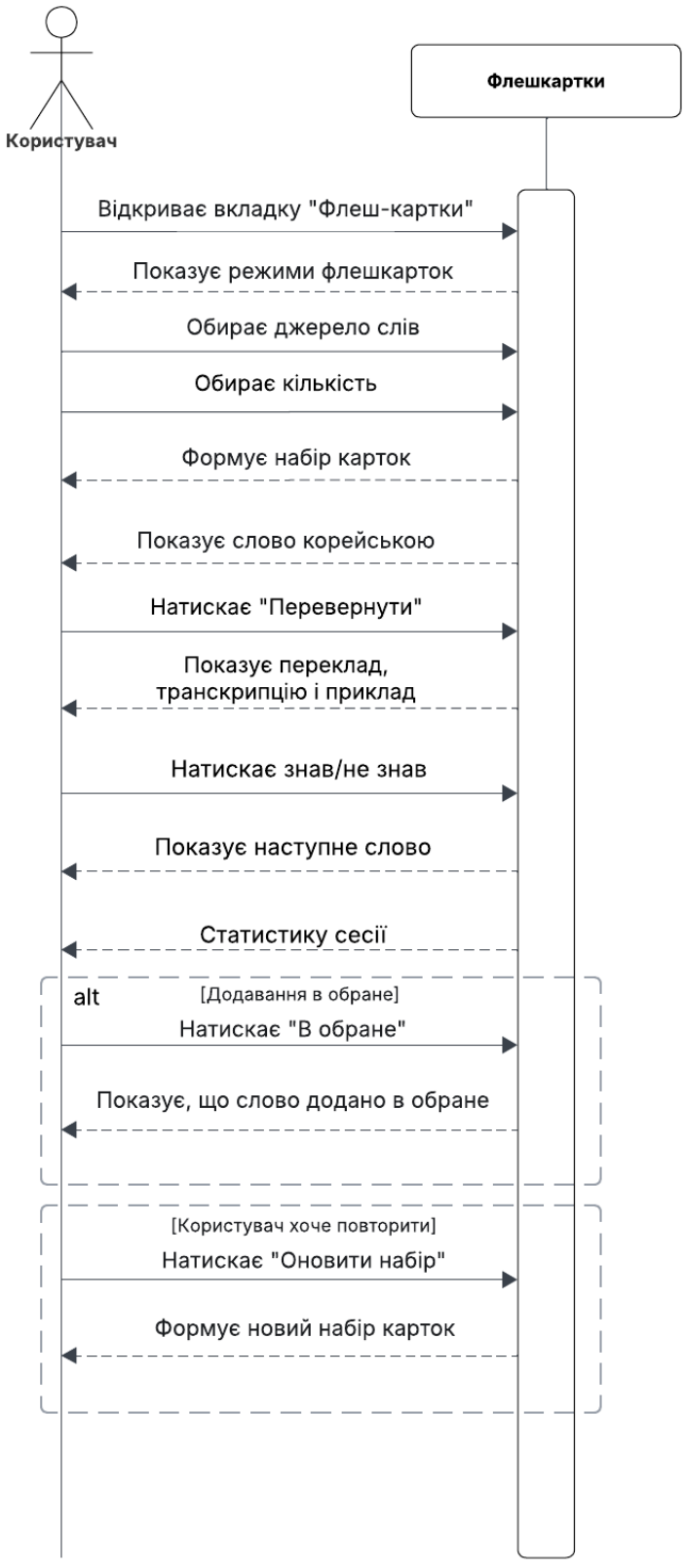

Рис.4 Діаграма послідовності Робота з флеш-картками

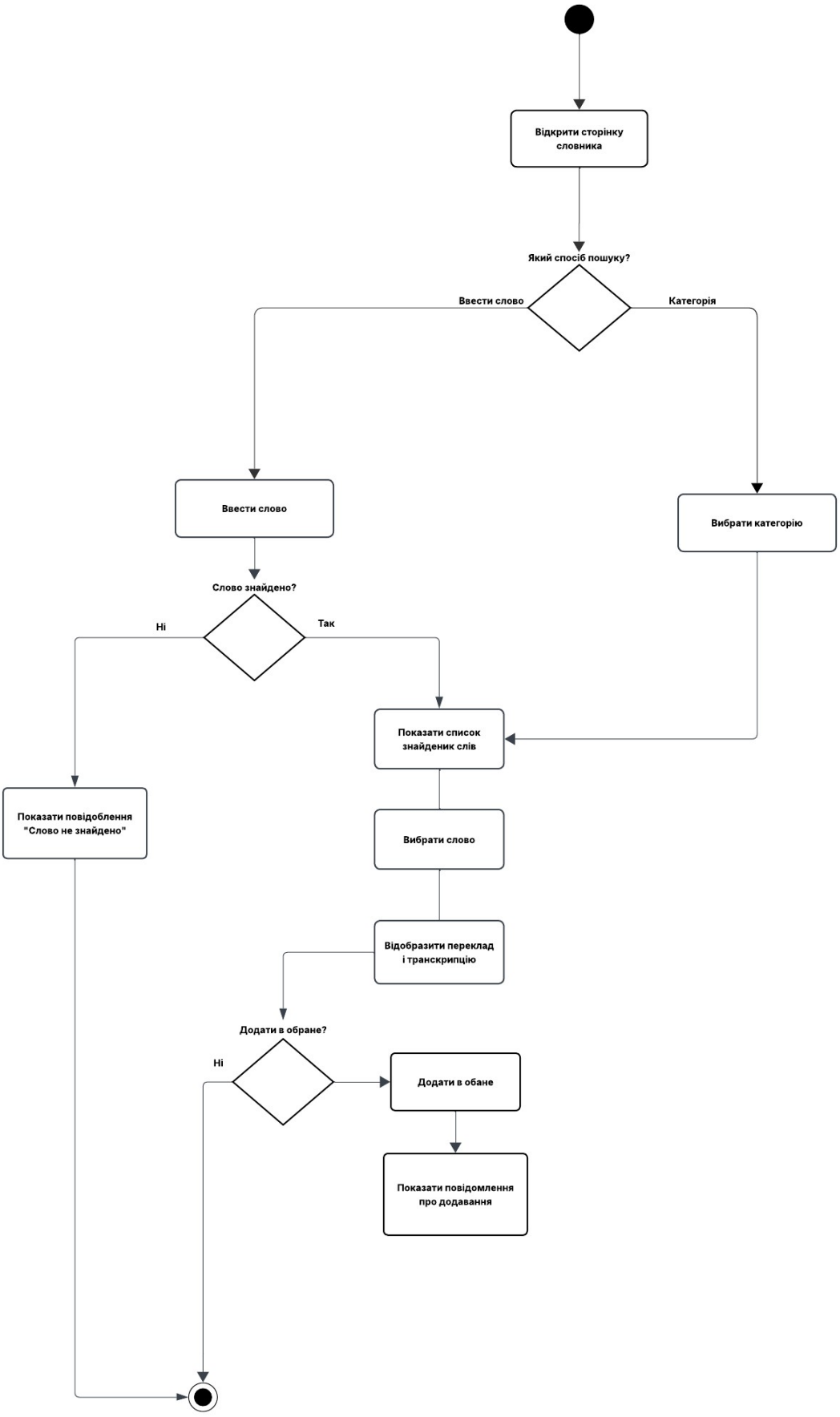

Рис.5 Діаграма активності Пошук слова

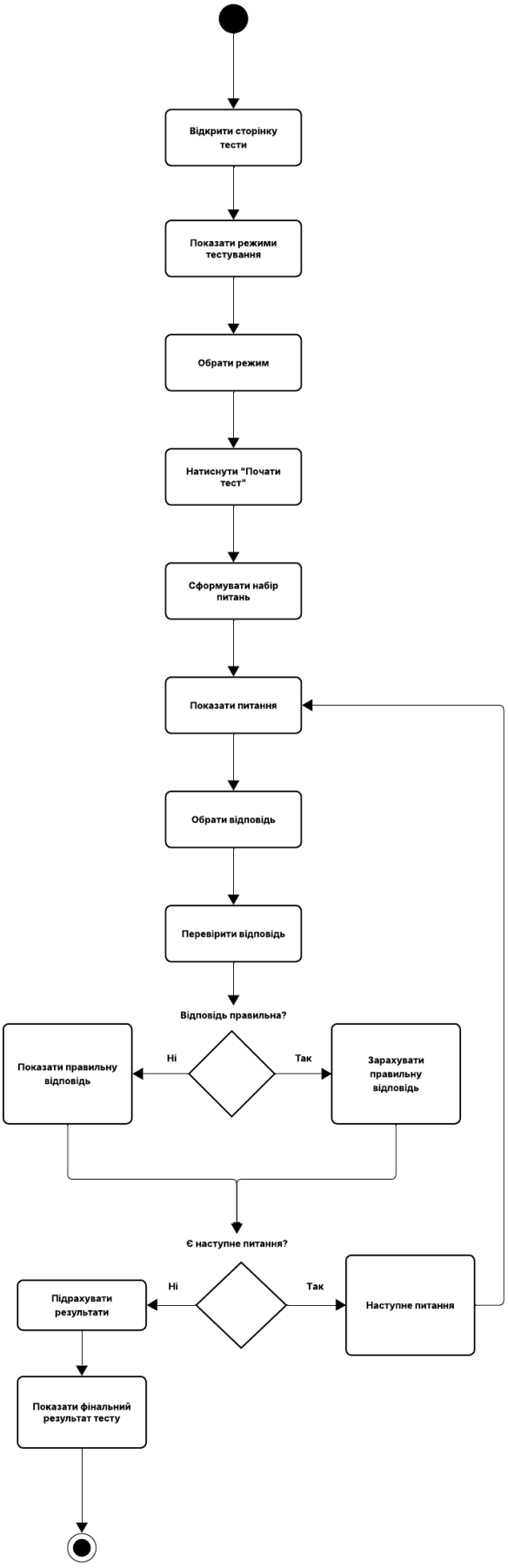

Рис.6 Діаграма активності Тестування

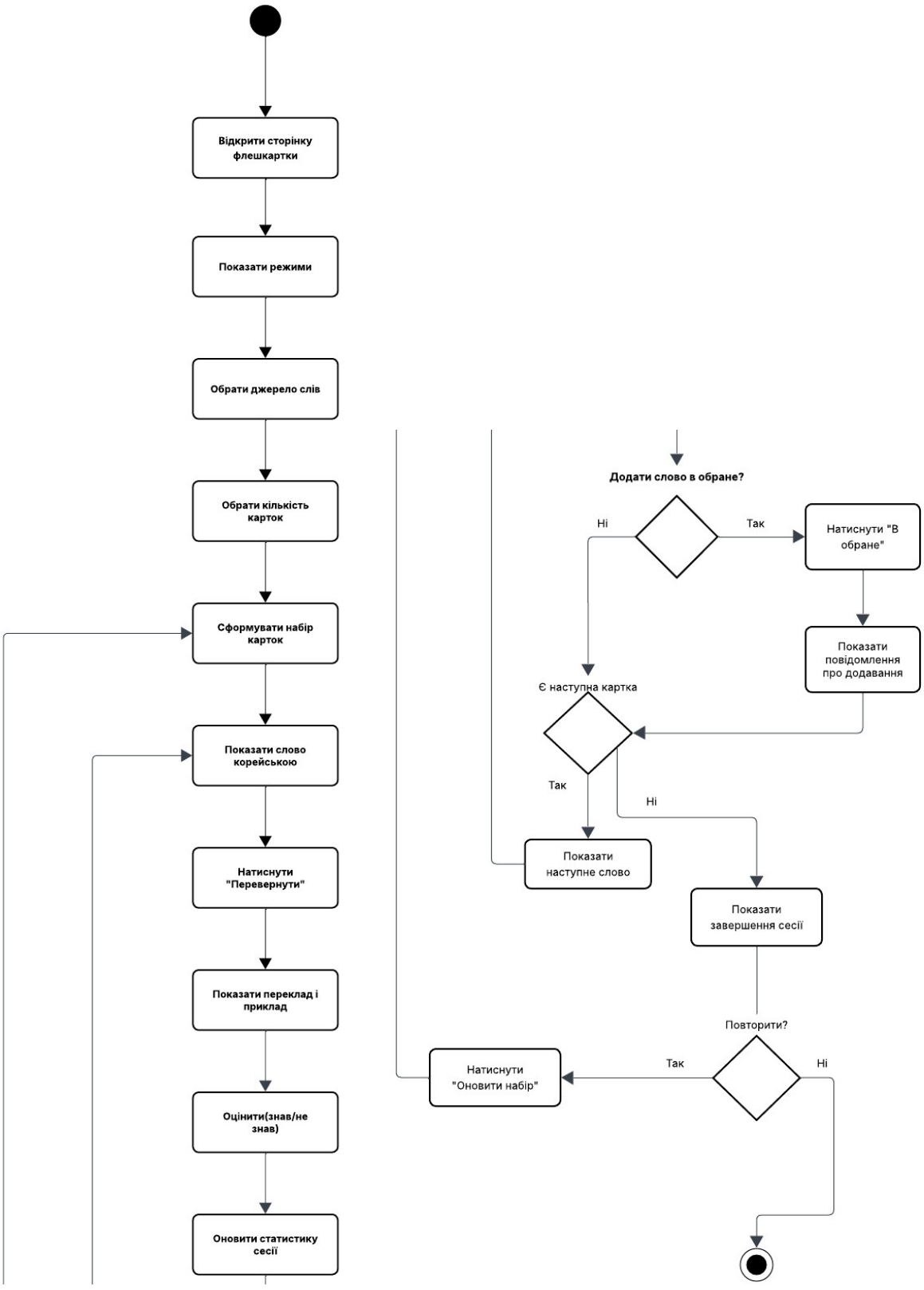

Рис.7 Діаграма активності ФК

ДОДАТОК Б

Мобільна версія веб-додатку

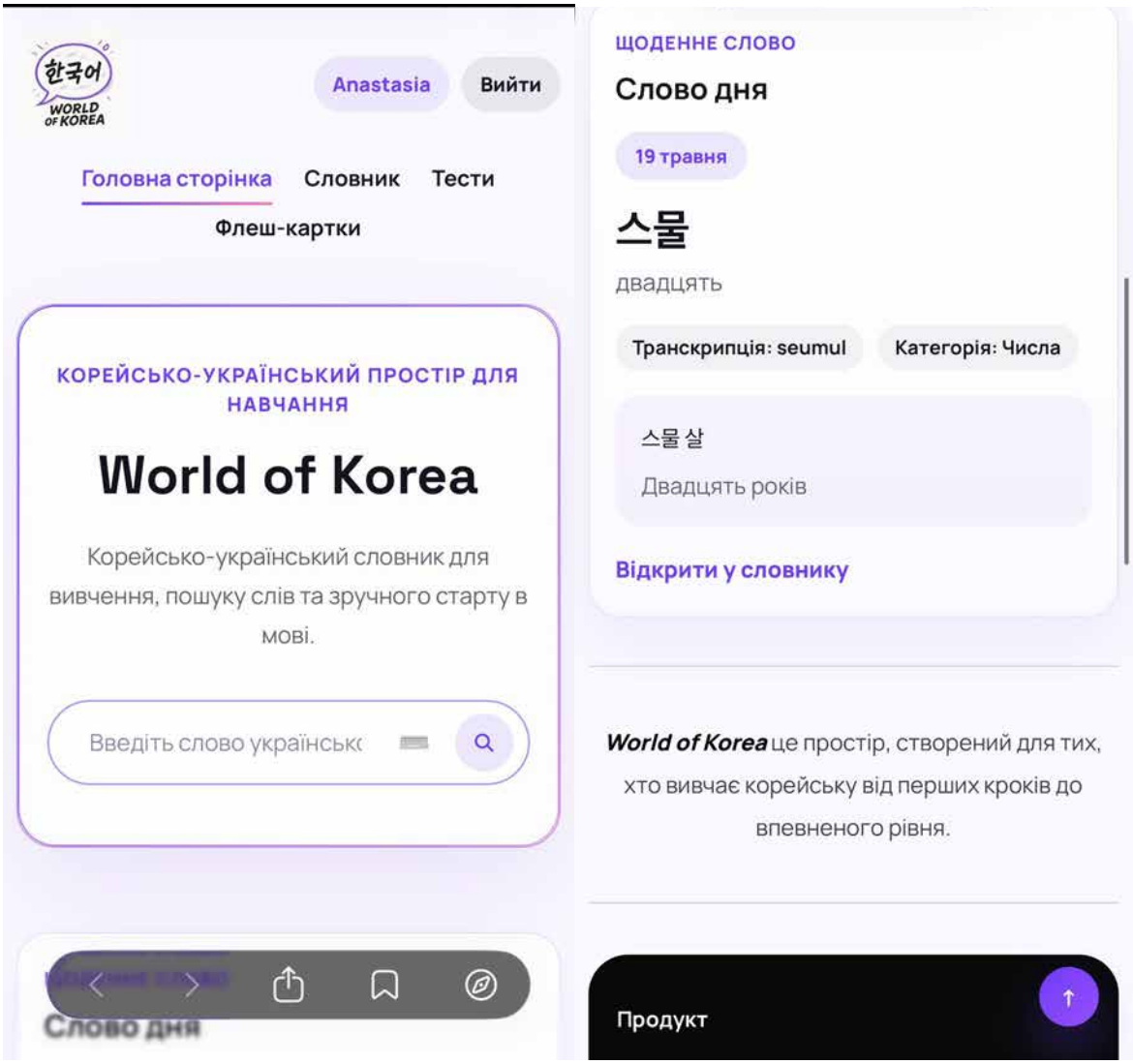

Рис.8 Головна сторінка

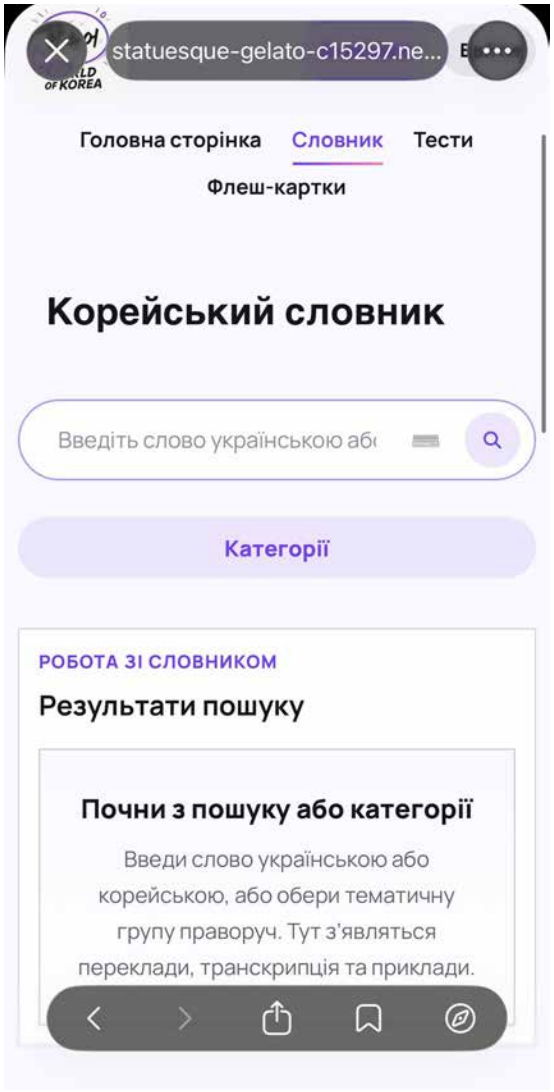

Рис.9 Сторінка словника

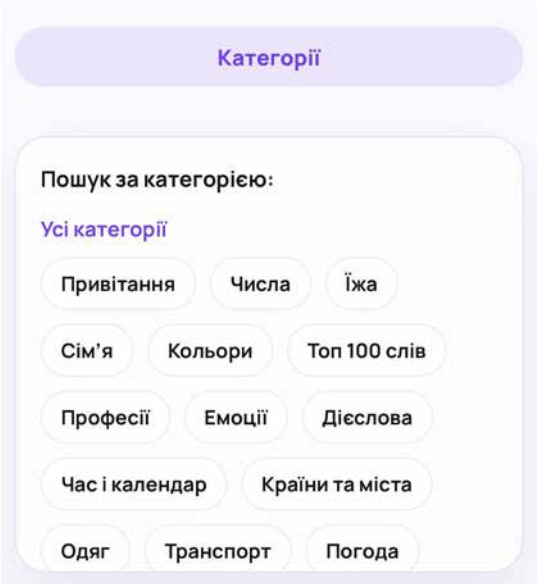

Рис.10 Категорії в словнику

Рис.11 Пошук

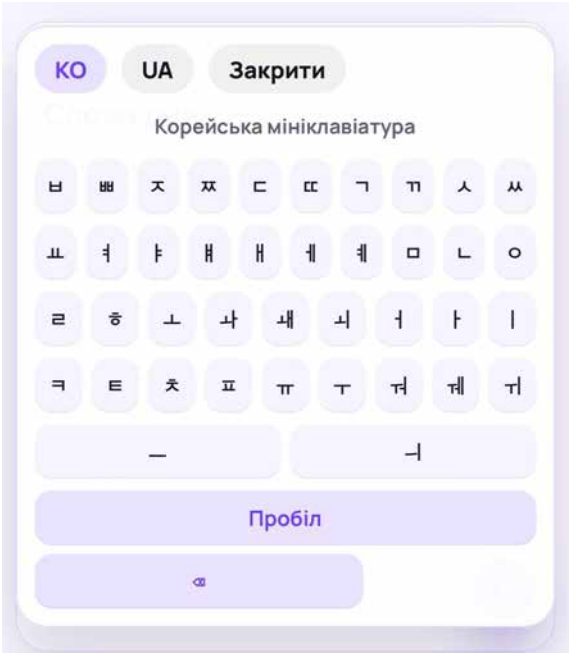

Рис.12 Клавіатура

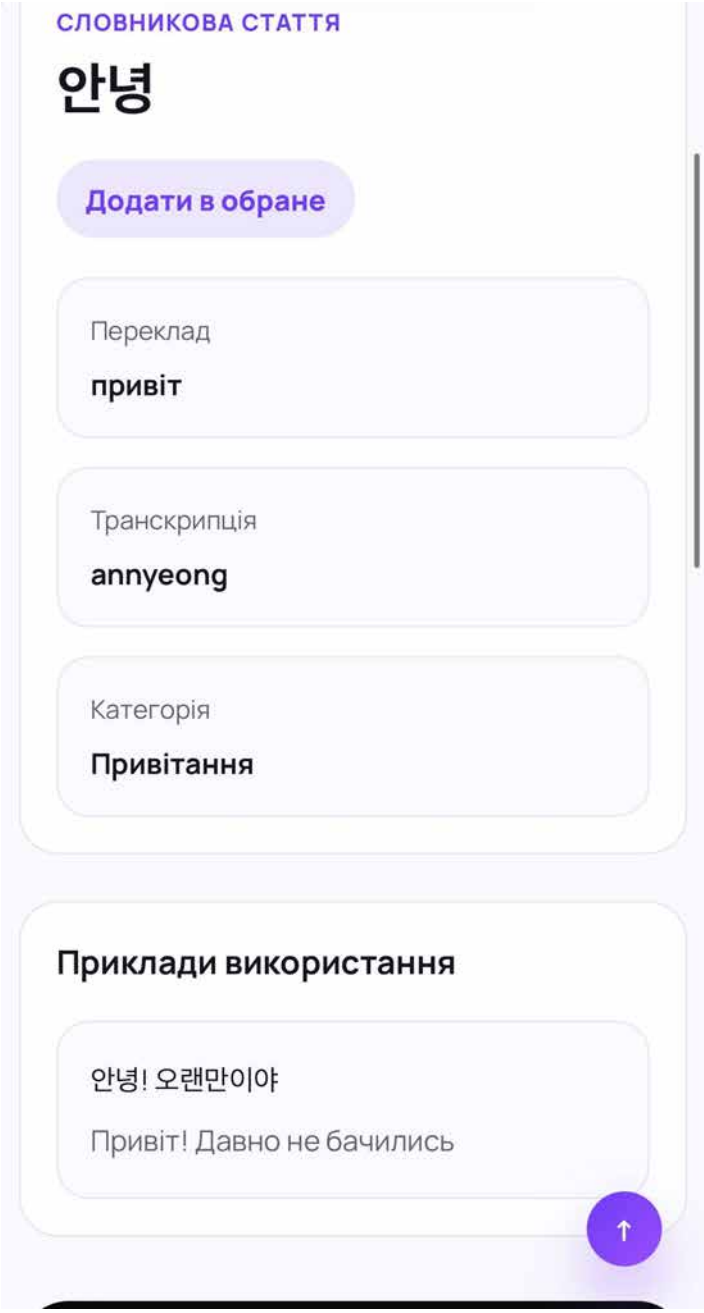

Рис.13 Сторінка слова

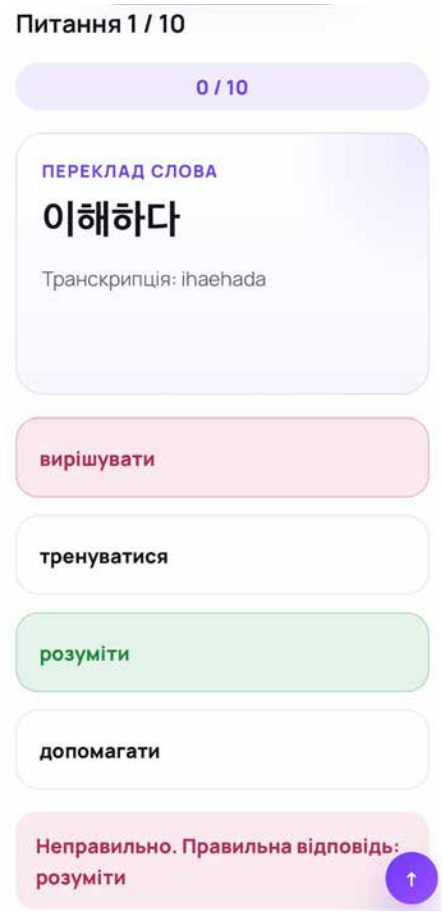

Рис.14 Сторінка тестування

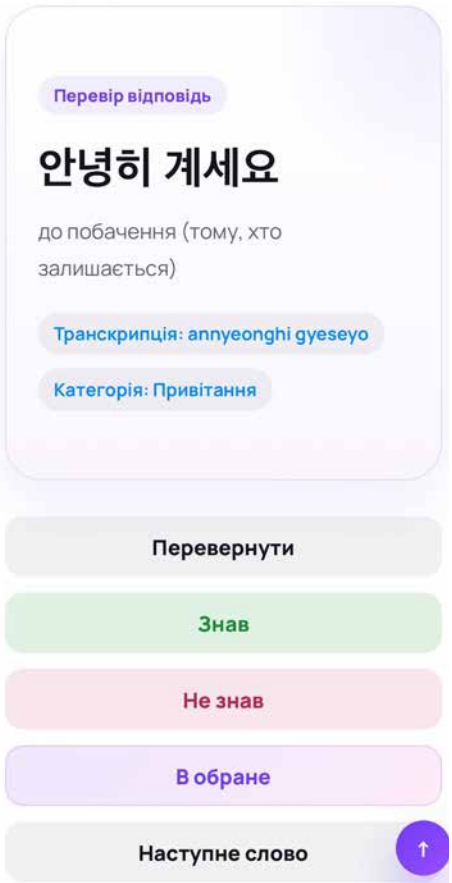

Рис.15 Сторінка флеш-карток

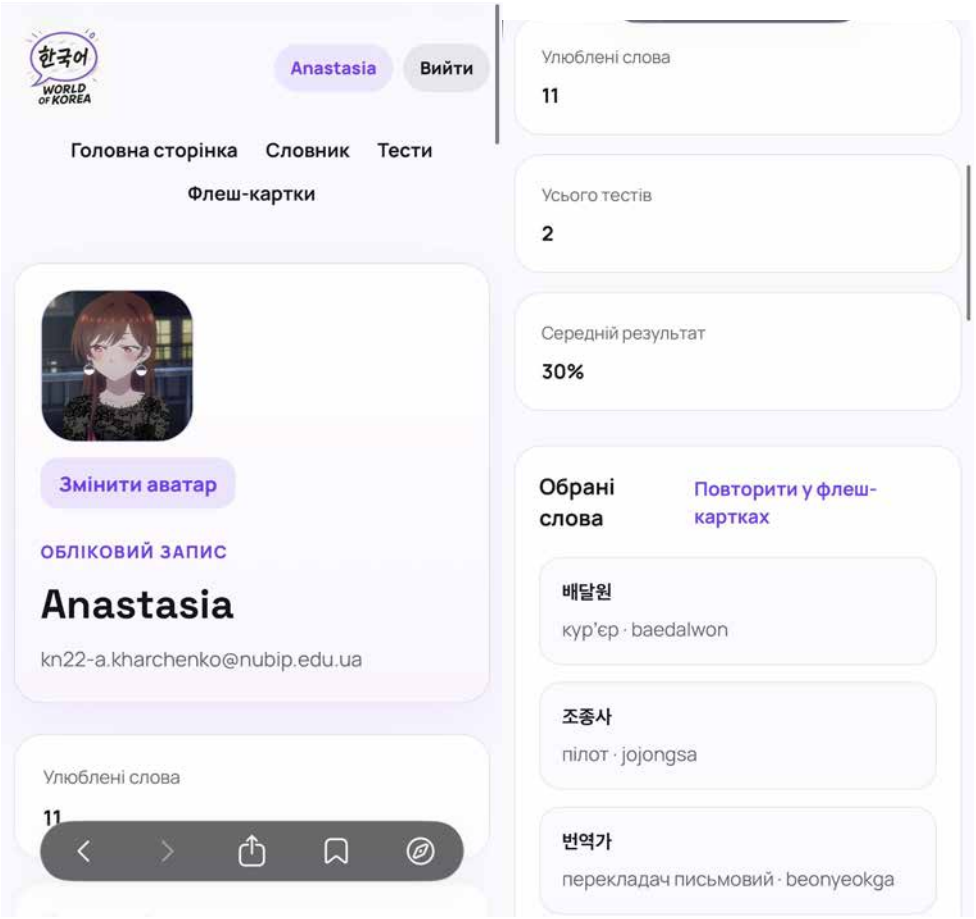

Рис.16 Обліковий запис користувача

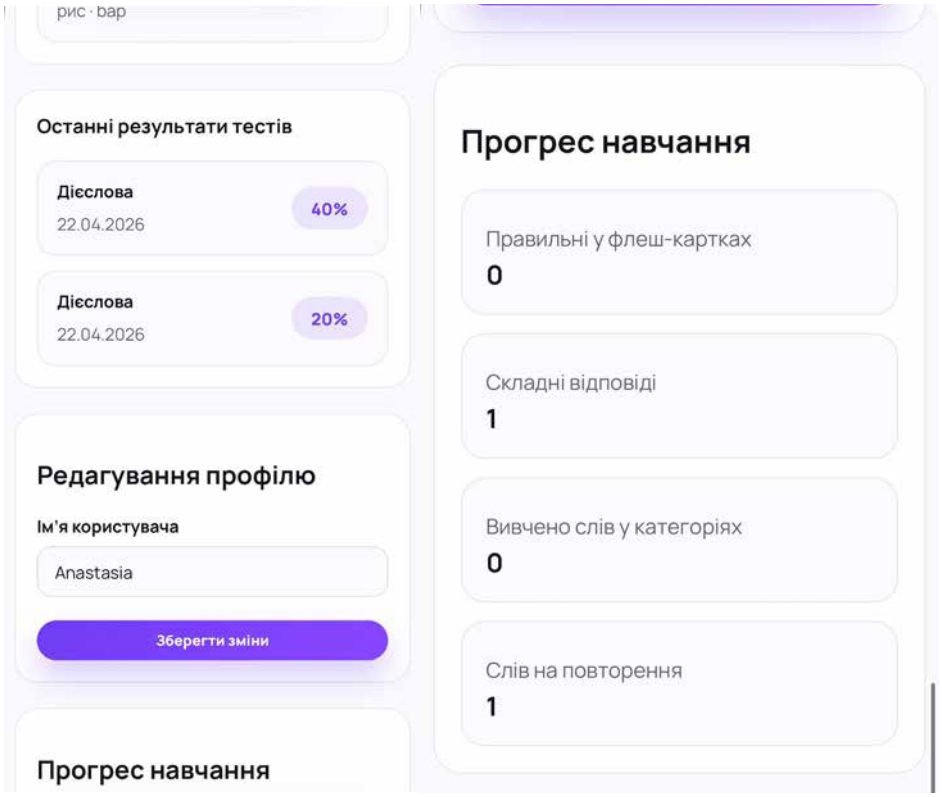

Рис.17 Обліковий запис користувача продовження

ДОДАТОК В

Інсталяційний код БД

```
-- ============================
-- TABLES CREATION
-- ============================
create table profiles (
    id uuid primary key references auth.users(id) on delete cascade,
    username text unique not null,
    avatar_url text,
    created_at timestamp default now()
);
create table categories (
    id bigint generated always as identity primary key,
    name text not null unique,
    description text,
    created_at timestamp default now()
);
create table words (
    id bigint generated always as identity primary key,
    korean text not null,
    transcription text,
    ukrainian text not null,
    part_of_speech text,
    difficulty int default 1,
    created_at timestamp default now()
);
create table category_words (
    category_id bigint references categories(id) on delete cascade,
    word_id bigint references words(id) on delete cascade,
    primary key (category_id, word_id)
);
create table favorites (
    user_id uuid references auth.users(id) on delete cascade,
    word_id bigint references words(id) on delete cascade,
    created_at timestamp default now(),
    primary key (user_id, word_id)
);

create table flashcard_progress (
    user_id uuid references auth.users(id) on delete cascade,
```

```
    word_id bigint references words(id) on delete cascade,
    correct_count int default 0,
    wrong_count int default 0,
    last_review timestamp,
    primary key (user_id, word_id)
);
create table category_progress (
    user_id uuid references auth.users(id) on delete cascade,
    category_id bigint references categories(id) on delete cascade,
    learned_words int default 0,
    primary key (user_id, category_id)
);
create table test_results (
    id bigint generated always as identity primary key,
    user_id uuid references auth.users(id) on delete cascade,
    category_id bigint references categories(id) on delete cascade,
    score int not null,
    total_questions int not null,
    created_at timestamp default now()
);
create table test_answers (
    test_id bigint references test_results(id) on delete cascade,
    word_id bigint references words(id),
    is_correct boolean,
    primary key (test_id, word_id)
);
create table examples (
    id bigint generated always as identity primary key,
    word_id bigint references words(id) on delete cascade,
    korean_sentence text not null,
    ukrainian_translation text,
    created_at timestamp default now()
);
create table repetition_data (
    user_id uuid references auth.users(id) on delete cascade,
    word_id bigint references words(id) on delete cascade,
    ease_factor float default 2.5,
    interval_days int default 1,
    next_review date,
    primary key (user_id, word_id)
```

```
);
-- ============================
-- INDEXES & CONSTRAINTS
-- ============================
-- Унікальність слова
create unique index unique_korean_word
on words (korean);

-- Пошук (full-text)
create index words_search_idx
on words using gin (
    to_tsvector('simple', korean || ' ' || ukrainian)
);

insert into categories (name, description)
values
('Привітання','Базові слова для привітання'),
('Числа','Корейські числа'),
('Їжа','Назви їжі'),
('Сім’я','Члени сім’ї'),
('Кольори','Назви кольорів'),
('Топ 100 слів','Найчастіше вживані корейські слова'),
('Професії','Назви професій корейською'),
('Емоції','Емоції та почуття'),
('Прикметники','Описові слова'),
('Погода','Слова про погоду та клімат'),
('Транспорт','Транспортні засоби'),
('Одяг','Предмети одягу'),
('Країни та міста','Різні країни та міста'),
('Час і календар','Дні тижня та місяці'),
('Дієслова','Базові корейські дієслова');

insert into words (korean, ukrainian, transcription)
values
('하나','один','hana'),
('둘','два','dul'),
('셋','три','set'),
('넷','чотири','net'),
('다섯','п’ять','daseot'),
```

('여섯','шість','yeoseot'),
('일곱','сім','ilgop'),
('여덟','вісім','yeodeol'),
('아홉','дев’ять','ahop'),
('열','десять','yeol'),
('열하나','одинадцять','yeolhana'),
('열둘','дванадцять','yeoldul'),
('열셋','тринадцять','yeolset'),
('열넷','чотирнадцять','yeolnet'),
('열다섯','п’ятнадцять','yeoldaseot'),
('열여섯','шістнадцять','yeolyeoseot'),
('열일곱','сімнадцять','yeolilgop'),
('열여덟','вісімнадцять','yeolyeodeol'),
('열아홉','дев’ятнадцять','yeolahop'),
('스물','двадцять','seumul'),
('서른','тридцять','seoreun'),
('마흔','сорок','maheun'),
('쉰','п’ятдесят','swin'),
('예순','шістдесят','yesun'),
('일흔','сімдесят','ilheun'),
('여든','вісімдесят','yeodeun'),
('아흔','дев’яносто','aheun'),
('백','сто','baek'),
('일','один','il'),
('이','два','i'),
('삼','три','sam'),
('사','чотири','sa'),
('오','п’ять','o'),
('육','шість','yuk'),
('칠','сім','chil'),
('팔','вісім','pal'),
('구','дев’ять','gu'),
('십','десять','sip'),

```
('천','тисяча','cheon'),
('만','десять тисяч','man')
on conflict (korean) do nothing;

insert into category_words (category_id, word_id)
select c.id, w.id
from categories c
join words w on true
where c.name = 'Числа'
and w.korean in (
'하나','둘','셋','넷','다섯','여섯','일곱','여덟','아홉','열',
'열하나','열둘','열셋','열넷','열다섯','열여섯','열일곱','열여덟','열아홉',
'스물','서른','마흔','쉰','예순','일흔','여든','아흔',
'백','일','이','삼','사','오','육','칠','팔','구','십','천','만'
)
on conflict do nothing;

insert into examples (word_id, korean_sentence, ukrainian_translation)
select w.id, e.kor, e.ua
from words w
join (
values
('하나','하나 주세요','Дайте один'),
('둘','둘 있어요','Є два'),
('셋','셋 입니다','Це три'),
('넷','넷 명','Чотири людини'),
('다섯','다섯 개','П’ять штук'),
('여섯','여섯 시','Шоста година'),
('일곱','일곱 명','Сім людей'),
('여덟','여덟 개','Вісім штук'),
('아홉','아홉 개','Дев’ять штук'),
('열','열 개','Десять штук'),
('스물','스물 살','Двадцять років'),
('서른','서른 명','Тридцять людей'),
('마흔','마흔 개','Сорок штук'),
('쉰','쉰 살','П’ятдесят років'),
```

```
('예순','예순 명','Шістдесят людей'),
('일흔','일흔 개','Сімдесят штук'),
('여든','여든 살','Вісімдесят років'),
('아흔','아흔 명','Дев’яносто людей'),
('백','백 원','Сто вон'),
('천','천 원','Тисяча вон'),
('만','만 원','Десять тисяч вон')
) as e(korean, kor, ua)
on w.korean = e.korean
on conflict do nothing;
-- ============================
-- FUNCTIONS
-- ============================
-- Пошук слів
create or replace function search_words(query text)
returns setof words
as $$
select *
from words
where to_tsvector('simple', korean || ' ' || ukrainian)
@@ plainto_tsquery('simple', query)
limit 20;
$$ language sql;
-- Випадкові слова (для тестів/флешкарток)
create or replace function get_random_words(limit_count int)
returns setof words
as $$
select *
from words
order by random()
limit limit_count;
$$ language sql;
-- Слово дня
create or replace function get_word_of_the_day()
returns words
as $$
select *
from words
order by id
```

```
limit 1 offset (
   extract(day from now())::int % (select count(*) from words)
);
$$ language sql;
-- Приклади для слова
create or replace function get_examples(word_id_input int)
returns setof examples
as $$
select *
from examples
where word_id = word_id_input;
$$ language sql;
-- Додати в обране
create or replace function add_to_favorites(user_uuid uuid, word_id_input int)
returns void
as $$
insert into favorites (user_id, word_id)
values (user_uuid, word_id_input)
on conflict do nothing;
$$ language sql;
-- Отримати обране
create or replace function get_favorites(user_uuid uuid)
returns setof words
as $$
select w.*
from words w
join favorites f on w.id = f.word_id
where f.user_id = user_uuid;
$$ language sql;
-- Зберегти результат тесту
create or replace function save_test_result(
   user_uuid uuid,
   category_id_input bigint,
   score_input int,
   total_input int
)
returns void
as $$
insert into test_results (user_id, category_id, score, total_questions)
values (user_uuid, category_id_input, score_input, total_input);
```

```
$$ language sql;
-- Статистика користувача
create or replace function get_user_stats(user_uuid uuid)
returns table (
    total_tests int,
    avg_score numeric
)
as $$
select
    count(*) as total_tests,
    avg(score) as avg_score
from test_results
where user_id = user_uuid;
$$ language sql;
-- ============================
-- TRIGGERS
-- ============================
-- Нормалізація тексту (нижній регістр)
create or replace function normalize_text()
returns trigger as $$
begin
    if new.ukrainian is not null then
        new.ukrainian := lower(new.ukrainian);
    end if;
    return new;
end;
$$ language plpgsql;
create trigger trigger_normalize
before insert on words
for each row
execute function normalize_text();
```

ДОДАТОК Д
Блок-схеми формування статистики

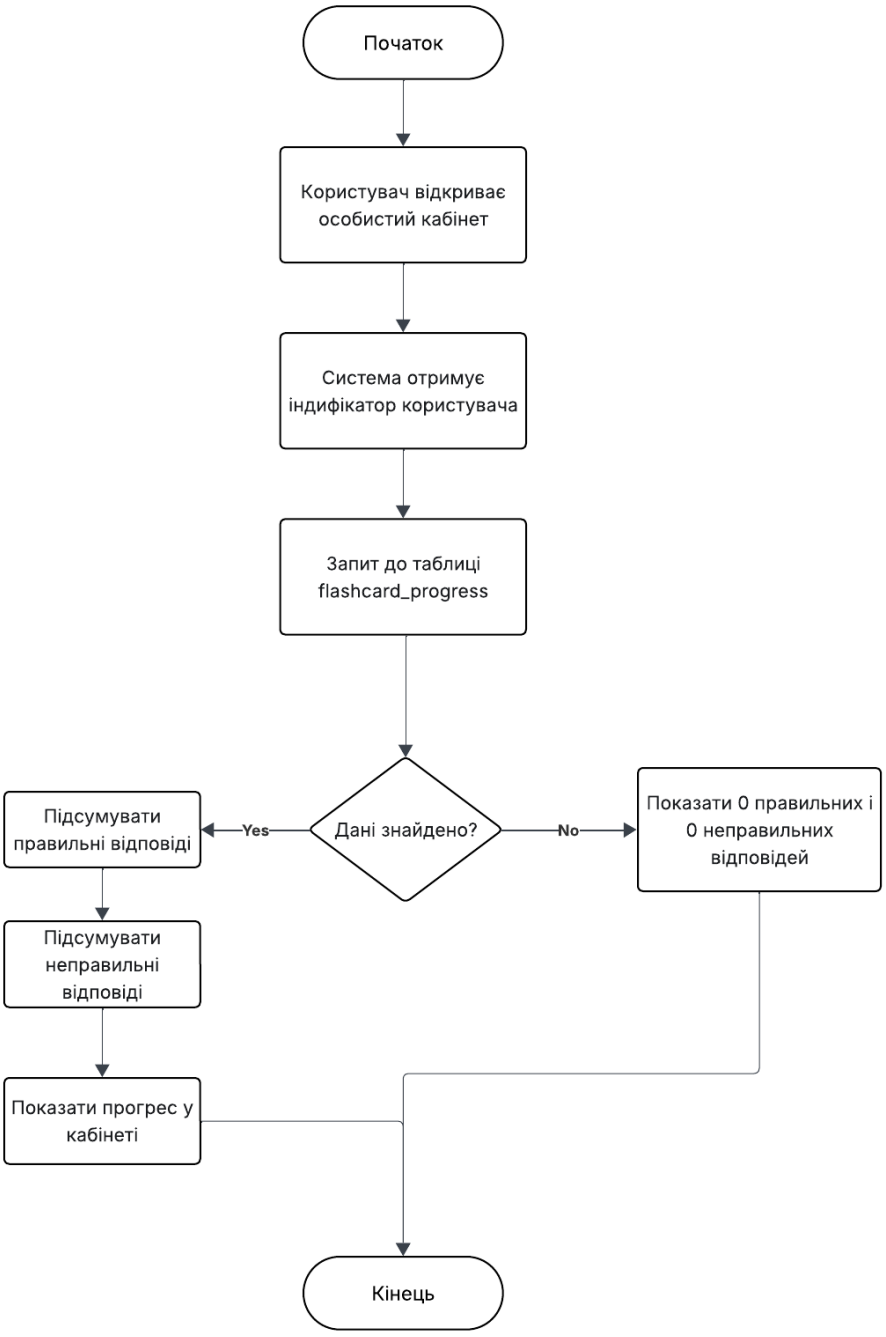

Рис. 18 Блок-схема формування статистики ФК

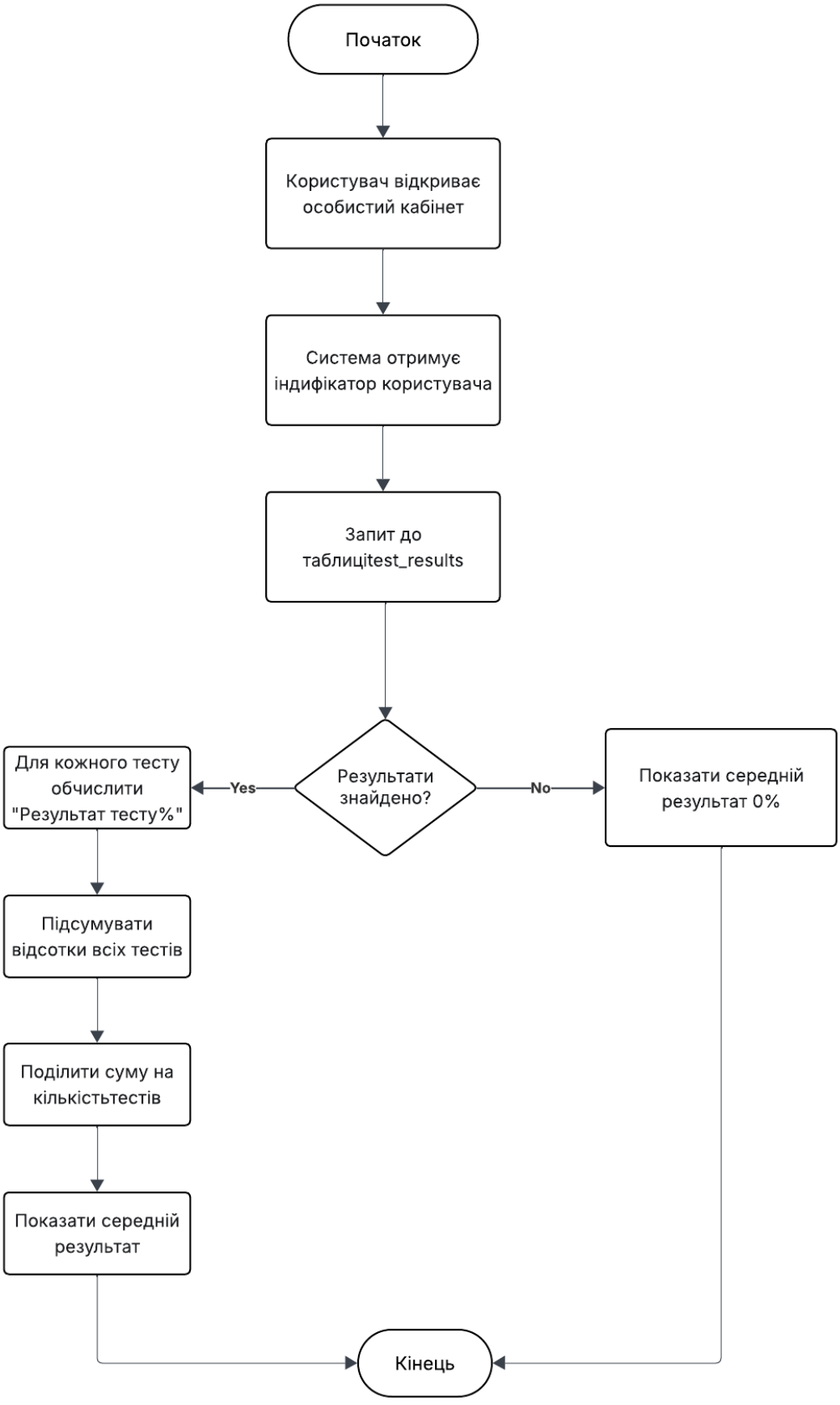

Рис.19 Блок-схема формування статистики тесту

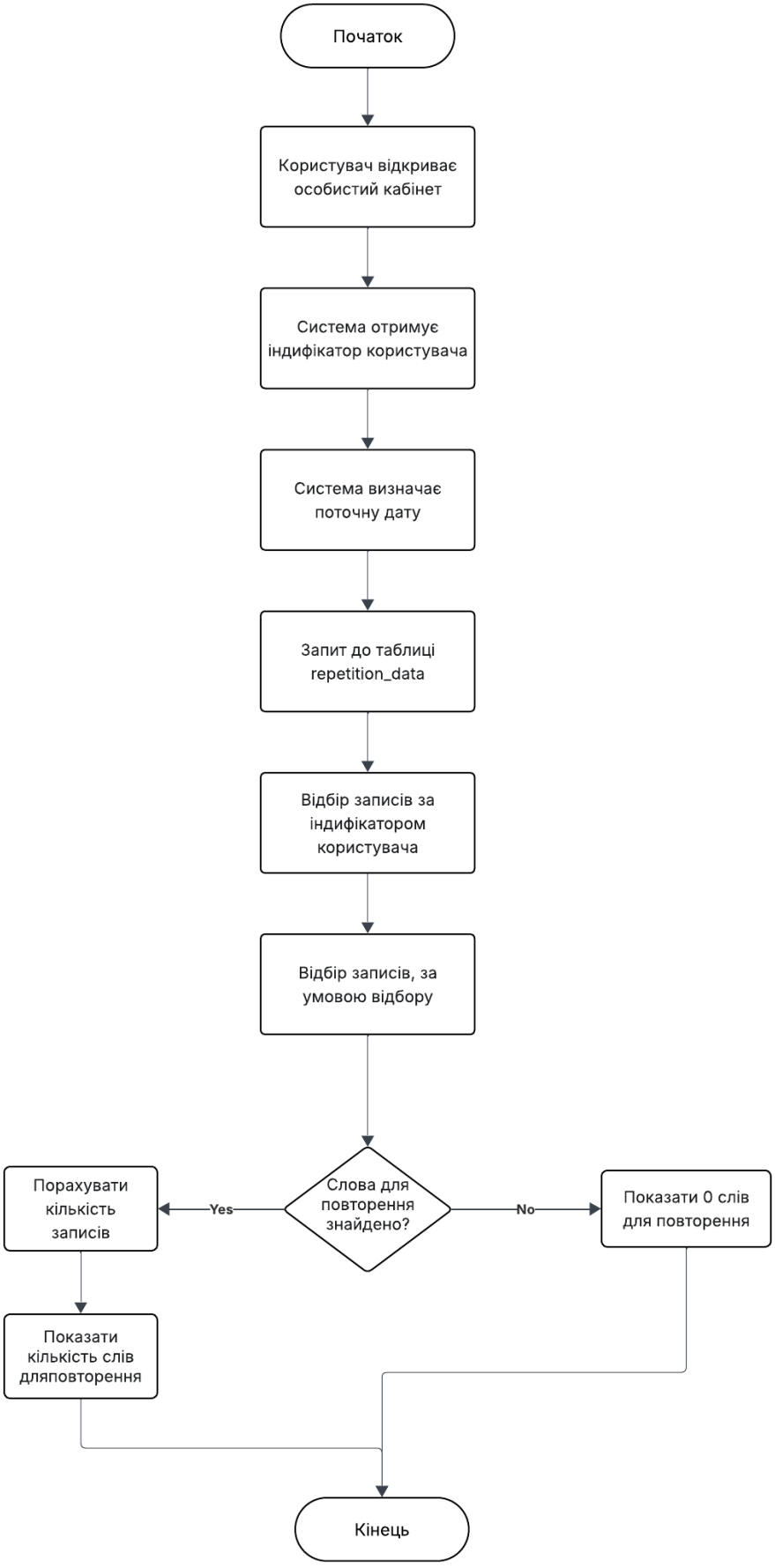

Рис.20 Формування статистики для повторення слів

ДОДАТОК Е
ДЕКЛАРАЦІЯ ЗДОБУВАЧА

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

Декларація про академічну доброчесність та використання ШІ
ДЕКЛАРАЦІЯ ЗДОБУВАЧА

Я, Харченко Анастасія Олександрівна, здобувачка НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

1. Моя бакалаврська кваліфікаційна робота (проєкт) на тему «Система супроводу корейськомовного словника» є результатом моєї особистої праці.
2. Усі запозичення з текстів інших авторів мають відповідні посилання згідно з чинними стандартами.
3. Щодо використання інструментів ШІ зазначаю, що (обрати необхідне):
    a. [*] інструменти генеративного ШІ не використовувалися.
    b. [*] інструменти ШІ ChatGPT та Claude були використані для:
        i. [*] перекладу іншомовних джерел;
        ii. [*] стилістичного редагування та перевірки граматики;
        iii. [ ] допомоги у написанні/відлагодженні програмного коду;
        iv. [ ] інше: _____.

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

«___» _____ 2026р. _____ Анастасія Харченко