**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**Сумський державний університет**
Центр заочної, дистанційної та вечірньої форм навчання
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
Ігор ШЕЛЕХОВ
_____
(підпис)
« » травня 2024 р.

# КВАЛІФІКАЦІЙНА РОБОТА

**на здобуття освітнього ступеня бакалавр**

зі спеціальності 122 - Комп'ютерних наук,
освітньо-професійної програми «Інформатика»
на тему: «Мобільний застосунок для вивчення іноземних слів за допомогою карток»
здобувача групи ІНз-01с Шепелюка Владислава Володимировича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Владислав ШЕПЕЛЮК
(підпис)

Керівник,
старший викладач кафедри комп'ютерних наук, к.т.н., доцент
Борис КУЗІКОВ _____
(підпис)

Суми – 2024

**Сумський державний університет**
Центр заочної, дистанційної та вечірньої форм навчання
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
Ігор ШЕЛЕХОВ
_____ (підпис)

# ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

**на здобуття освітнього ступеня бакалавра**

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»
здобувача групи ІНз-01с Шепелюка Владислава Володимировича

1. Тема роботи: «Мобільний застосунок для вивчення іноземних слів за допомогою карток» затверджую наказом по СумДУ від *«26» квітня 2024 р. № 0438-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до «05» червня 2024 року*
3. Вхідні дані до кваліфікаційної роботи _
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Аналіз видів та методів вивчення іноземної мови, постановка й формування завдань досліджень 2) Проектування застосунку та опис обраних технологій 3) Розробка мобільного застосунку для вивчення іноземних слів за допомогою карток..*
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

| Розділ | Консультант | Підпис, дата | |
|---|---|---|---|
| | | Завдання видав | Завдання прийняв |
| | | | |

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ (підпис) Керівни к _____ (підпис)

# КАЛЕНДАРНИЙ ПЛАН

| № п/п | Назва етапів кваліфікаційної роботи | Термін виконання | Примітка |
|---|---|---|---|
| *1* | *Аналіз видів та методів вивчення іноземної мови, постановка та формування завдань досліджень* | | |
| *2* | *Проектування застосунку та опис обраних технологій* | | |
| *3* | *Розробка мобільного застосунку для вивчення іноземних слів за допомогою карток.* | | |
| *4* | *Оформлення пояснювальної записки до кваліфікаційної роботи* | | |

Здобувач вищої освіти _____ Керівни к _____

(підпис) (підпис)

## АНОТАЦІЯ

**Записка**: 63 ст., 1 додаток, 31 літературних джерела.

**Обґрунтування актуальності теми роботи** - Тема розробки мобільного застосунку для вивчення іноземних слів через картки актуальна через зростання популярності мобільних пристроїв та потребу в ефективному навчанні мов. Такий застосунок надасть користувачам можливість навчатися мові в будь-який час та в будь-якому місці, зробивши процес навчання більш гнучким і доступним.

**Об’єкт дослідження** - процес вивчення іноземної мови.

**Мета роботи** - розробка мобільного застосунку для вивчення іноземних слів за допомогою карток.

**Методи дослідження** – технології та алгоритми створення мобільних застосунків.

**Результати** - розроблено мобільний застосунок для вивчення іноземних слів за допомогою методу карток. Виконано планування майбутніх оновлень на покращення гнучкості додатку під більшість користувачів.

МОБІЛЬНИЙ ЗАСТОСУНОК, ВИВЧЕННЯ МОВИ, КАРТКИ, DART, FLUTTER, SQLITE, ANDROID

# ЗМІСТ

**ВСТУП** **5**
**1 АНАЛІТИЧНИЙ ОГЛЯД** **7**
1.1 Аналіз видів та методів вивчення іноземної мови 7
1.2 Дослідження видів та методів вивчення іноземних слів 8
1.3 Вивчення слів за допомогою методу карток та методу інтервального повторення 10
1.4 Детальний аналіз та пошук додатків для вивчення слів іноземної мови 11
1.4.1 Застосунок Flashcards World 11
1.4.2 Застосунок DouCards 12
1.4.3 Застосунок Words 13
1.5 Постановка задачі дослідження 14
**2 ПРОЕКТУВАННЯ ЗАСТОСУНКУ ТА ОПИС ТЕХНОЛОГІЙ** **15**
2.1 Проектування функціональних можливостей застосунку 15
2.1.1 Опис функціональних можливостей сторінки "Навчання" 16
2.1.2 Опис функціональних можливостей сторінки "Теми" 17
2.1.3 Опис функціональних можливостей сторінки "Словник" 17
2.2 Проектування макету застосунку відповідно до функціональних можливостей 18
2.2.1 Макет сторінки "Навчання" 18
2.2.2 Макет сторінки "Теми” 22
2.2.3 Макет сторінки "Словник" 25
2.2.4 Вибір дизайну , загального стилю застосунку 26
2.3 Проектування баз даних 27
2.4 Стек технологій на якому буде створено додаток 27
2.4.1 Мова програмування 27
2.4.2 База даних 28
**3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ** **30**
3.1 Створення порожнього проекту та тестовий запуск 30
3.2 Програмна реалізація особливостей застосунку відповідно до функціональних особливостей та стилю 32
3.3 Планування майбутніх оновлень для покращення застосунку 39
**ВИСНОВКИ** **41**
СПИСОК ЛІТЕРАТУРИ 42
**ДОДАТОК А** **46**

# ВСТУП

**Актуальність.** Мова завжди відігравала ключову роль у спілкуванні та розвитку в сучасному світі. Вивчення іноземної мови – це подорож у світ нових можливостей та розширення світогляду. Різноманітні методи, від глибокого вивчення граматики до практики в реальних бесідах, допомагають крок за кроком занурюватися у новий мовний простір. Зі своєю унікальністю та викликами вивчення іноземної мови відкриває перед нами не лише нові слова, але й розуміння інших культур, традицій та поглядів. Це сприяє не лише комунікації, а й глибшому взаєморозумінню у світі, де границі стають все менш помітними.

Нещодавно я почав переглядати вакансії інженерів і все більше зустрічав — що всюди потрібно знати англійську мову мінімум рівня B2. Вивчення англійської мови завжди для мене було викликом. Через короткий проміжок часу я втрачав мотивацію та жагу до нових знань. Причиною було довгий час навчання та неможливість приділяти цьому лічені хвилини на добу в будь якому місці.

Тому, вивчення мови є дуже важливим напрямком для кожного, особливо в такі важкі часи. І задіяти в цьому напрямку технології, які можуть бути використані в будь який момент є перспективною задачею.

Враховуючи те, що щодня люди проводять години в смартфонах, дозволить їм виділяти зовсім трохи часу щодня, для вивчення нових, корисних і цікавих слів іншою мовою. Це поглибить словниковий запас користувача і зробить процес навчання більш гнучким і доступним.

**Об’єкт дослідження.** Процес вивчення іноземної мови.

**Предмет дослідження.** Розробка мобільного застосунку для вивчення іноземних слів за допомогою карток.

**Гіпотеза.** Використання мобільного застосунку для вивчення лексики іноземної мови через картки буде сприяти:

- мотивації вивчення слів;

- персоналізації вивчення і користування додатком;
- зручності(гнучкості) вивчення, оскільки не будеш прив’язаний до часу та місця;
- частота вивчення буде більшою, ніж при звичайному вивченні.

**Новизна.** Описане в цій роботі програмне забезпечення дозволить вивчати слова іноземною мовою за допомогою карток не будучи прив’язаним до інтернету, а також простого дизайну, який є зручним та зрозумілим будь кому. Завдяки простому дизайну користувач більше приділяти уваги словам для вивчення. Окрім цього, є можливість самому створювати теми зі словами, редагувати їх та самому додавати слова, які б хотів би вивчити та не забувати.

**Структура.** Дана робота складається зі вступу, аналітичного огляду, постановки задачі, вибір методу розв’язання поставленої задачі, опису програмного забезпечення інформаційної системи, висновків, списку використаних джерел та додатків.

# 1 АНАЛІТИЧНИЙ ОГЛЯД

## 1.1 Аналіз видів та методів вивчення іноземної мови

Вивчення іноземної мови - це складний, але цікавий процес, що включає в себе різноманітні методи та стратегії. Кожен процес має свою унікальну структуру та звичайно залежить від вчителів та методологій вивчення. Основні методи вивчення іноземної можна класифікувати на кілька ключових напрямків:

- аудіювання та читання:
    1) Аудіо та відео матеріали. Слухання аудіо та перегляд відео на англійській мові допомагають розвивати навички аудіювання та звільняють відповідність вимови з реальними ситуаціями.
    2) Читання текстів. Читання книг, статей, новин та інших текстів сприяє розширенню словникового запасу та вдосконаленню граматики.
- розмовна практика:
    1) Комунікація з носіями мови. Обговорення та спілкування з носіями мови розвиває навички спілкування, адаптує до реальних обставин та поліпшує вимову.
    2) Участь у групових заняттях. Заняття в групі надають можливість обговорювати теми, виправляти помилки та збагачувати словниковий запас через взаємодію з іншими студентами.
- вивчення граматики:
    1) Граматичні вправи. Систематичне вирішення граматичних вправ допомагає закріплювати правила та забезпечує правильне використання граматичних конструкцій.
    2) Граматичні підручники та онлайн-ресурси. Використання підручників та інтернет-ресурсів для самостійного вивчення граматики.
- вивчення слів та фраз:

1) Метод карток. Використання карток для вивчення нових слів, їх перекладу та вживання у реченнях.
2) Читання та вивчення сталих висловів. Запам'ятовування сталих висловів та вживання їх у конкретних ситуаціях сприяє природному використанню мови.

Проаналізувавши основні види і методи вивчення іноземної мови, я дійшов висновку, що важливим етапом в цьому процесі є вивчення слів і сталих висловів. Це ключовий компонент успішного освоєння мови, оскільки він не лише розширює словниковий запас, але й допомагає у власноручному використанні мовних конструкцій у практичних ситуаціях. Вивчення слів та сталих висловів стає фундаментом для розвитку всіх аспектів мови, сприяючи більш ефективному спілкуванню та розумінню іноземної мови.

### 1.2 Дослідження видів та методів вивчення іноземних слів

Вивчення слів іноземної мови - ключовий елемент мовного навчання, і наявні різні види та методи, які допомагають в цьому процесі. Ось деякі з них:

- Метод Карточок (Flashcards) - цей метод передбачає написання слова або вислову на одній стороні картки та його переклад чи визначення на іншій.
- Вивчення слів у контексті. В даному методі слова вивчаються в конкретних ситуаціях або реченнях, щоб зрозуміти їхній справжній вжиток.
- Використання Мультимедійних Ресурсів. Залучення аудіо- та відеоматеріалів для вивчення вимови та використання слів у реальних ситуаціях.
- Вивчення слів за допомогою ігор. Використання ігор, кросвордів, словникових гравців для навчання нових слів.
- Вивчення слів за допомогою ведення словнику та формування власного словникового запасу.
- Вивчення слів за допомогою асоціативних схем.

Проаналізувавши недоліки і переваги кожного виду вивчення слів, я прийшов до висновку, що для себе оптимальними буде «Метод Карточок

(Flashcards)». Використання карток дозволить мені швидко ознайомитися з новими словами, мобільно використовувати їх, змінювати, редагувати, швидко заміняти та зберігати у великій кількості.

Основною проблемою при вивчення великого запасу слів — це пам’ятати ці слова. Під час аналізу та пошуку в інтернеті корисної інформації, я зіштовхнувся з рецензію на цікаве дослідження, яке допомагало мені в пошуку ідеальної формули для вивчення та запам’ятовування великої кількості слів. Крива забування Еббінгауза є концепцією, розробленою психологом Германом Еббінгаузом в кінці XIX століття. Ця крива відображає процес забування інформації з часом як це показано на рис. 1.1 . Згідно з дослідженнями Еббінгауза, найбільша частина забування відбувається найшвидше після вивчення матеріалу, а потім темп забування зменшується.

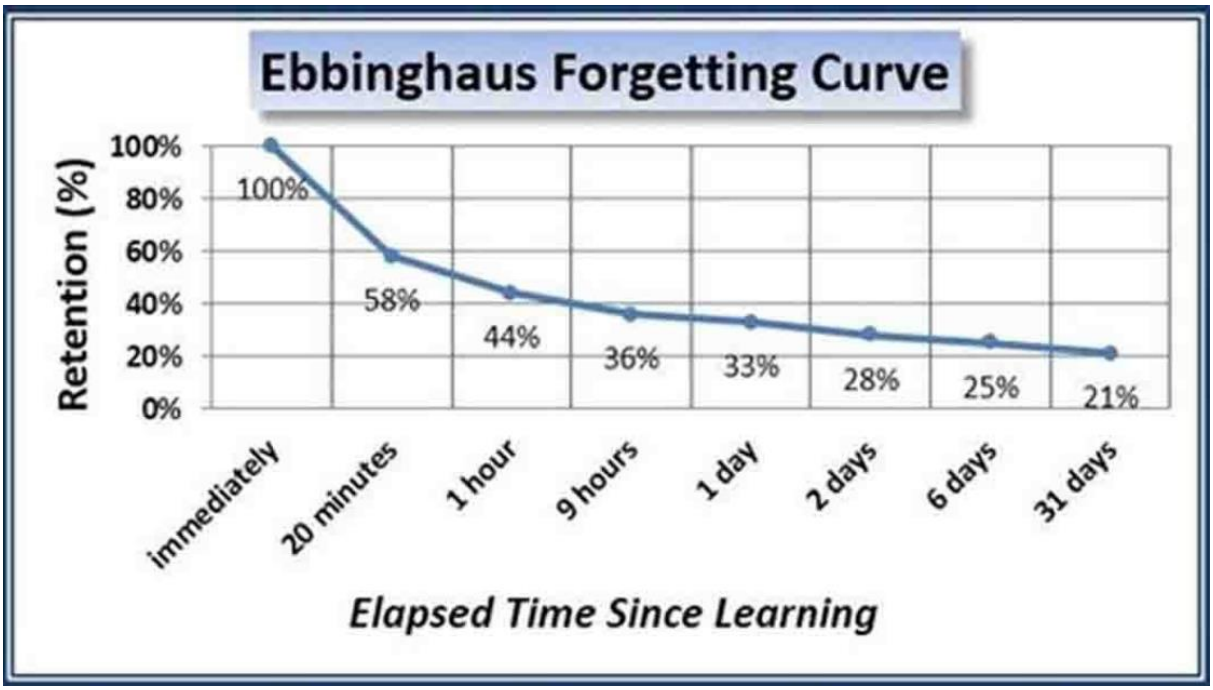



Рисунок 1.1 — Крива забування Германа Еббінгауза [25]

Зазвичай крива забування представляє собою експоненційне зменшення пам'яті з часом. Спочатку втрата інформації велика, але потім спадає до стабільного рівня. Важливою властивістю кривої забування є те, що період найінтенсивнішого забування припадає на короткий проміжок часу після

вивчення, після чого зменшується темп забування. Ця концепція є основою для розуміння, як ефективно використовувати повторення і перегляд для покращення запам'ятовування. За використанням оптимальних інтервалів між повтореннями можна підтримувати стабільний рівень пам'яті та зменшити ефект кривої забування.

Метод Інтервального Повторення: заснований на ідеї систематичного повторення слова чи вислову з регулярними інтервалами і базується на основі концепції Германа Еббінгауза. Основні стадії повторення вивченого матеріалу:

- після отримання нової інформації;
- через 20-30 хвилин після першого повторення;
- через 1 день після другого повторення;
- через 2-3 тижні після третього повторення;
- через 2-3 місяці після четвертого повторення.

Отже підбиваючи підсумки, ідеальною формулою для вивчення великої кількості інформації — це «метод карток» та «метод інтервального повторення».

### 1.3 Вивчення слів за допомогою методу карток та методу інтервального повторення

Мій вибір зосередився на використанні карток "Eng Student", показано на рис.1.2. Проте, роздумуючи, виявилася проблема — неможливість самостійного обирати слова та теми для вивчення, що вимагало б вручну створення багатьох карток. Цей недолік підштовхнув мене до висновку, що ефективним варіантом було б знаходження додатку, який не лише надає можливість вивчати слова за допомогою карток, але й дозволяє створювати самостійно слова та теми для вивчення. Такий підхід забезпечив би більш гнучкий та персоналізований підхід до вивчення мови. Таким чином, в пошуку оптимального методу вивчення іноземної мови, я визначив важливість знаходження додатку, що враховує мої індивідуальні потреби та забезпечує максимальний контроль над процесом вивчення.


Рисунок 1.2 - Набір карток "Eng Student".

### 1.4 Детальний аналіз та пошук додатків для вивчення слів іноземної мови

Для кращого аналізу додатків я вирішив сформувати плюси та мінуси кожного з додатків. Після аналізу порівняти функціональність та недоліки додатків і обрати оптимальний для використання.

#### 1.4.1 Застосунок Flashcards World

Основні негативні та позитивні відмінності цього додатку під час мого використання:

- Недоліки:
    1) інтерфейс додатка доступний лише англійською мовою, і його неможливо змінити на іншу, як це показано на рис 1.3 . Це може відвернути користувачів, чиї знання англійської мови є обмеженими.
- Переваги:
    1) простота використання: Додаток "Flashcards" відзначається своєю легкістю використання, що робить процес вивчення нових слів зручним та ефективним;

2) персоналізація навчання: Додаток надає можливість користувачам створювати власні картки із словами, що вони хочуть вивчати, забезпечуючи індивідуальний підхід до навчання;
3) легкість обміну картками для парного вивчення.

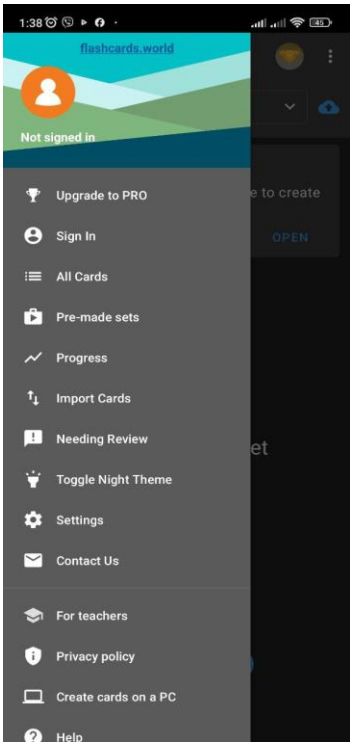



Рисунок 1.3 — Бокове меню додатку Flashcards World

### 1.4.2 Застосунок DouCards

Основні негативні та позитивні відмінності цього додатку під час мого використання:

- Недоліки:
  1) інтерфейс додатку може виявитися не досить інтуїтивним для новачків;
  2) залежність від доступу до Інтернету: використання відеоматеріалів може вимагати стійкого підключення до Інтернету, що може бути не завжди доступним для користувачів у різних ситуаціях.
- Переваги:

1) можливість вивчення нових слів за допомогою відео, що сприяє кращому запам'ятовуванню та розумінню контексту як це показано на рис. 1.4;
2) велика кількість вже створених наборів картинок як це показано на рис. 1.4;
3) можливість вивчати нові слова за допомогою створених курсів як це показано на рис. 1.4.

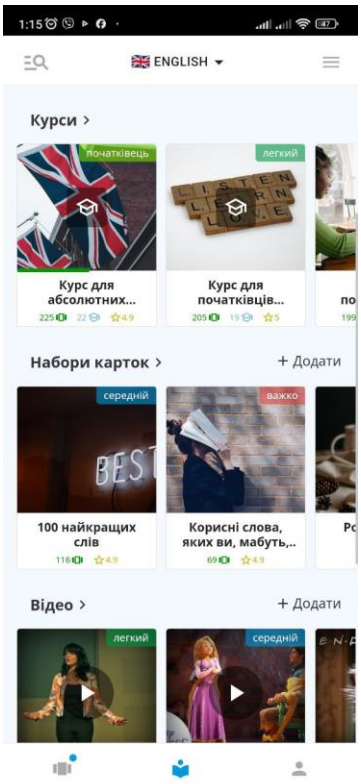



Рисунок 1.4 — Головний екран додатку DouCards

### 1.4.3 Застосунок Words

Основні негативні та позитивні відмінності цього додатку під час мого використання:

- Недоліки:
  1) обмеження на безкоштовне додавання слів, дозволяючи додавати лише 5 нових слів щодня. Для можливості додавання більшої кількості слів користувачам необхідно придбати підписку;

2) інтерфейс додатка доступний лише англійською мовою як це показано на рис.1.5, і його неможливо змінити на іншу. Це може відвернути користувачів, чиї знання англійської мови є обмеженими.

- Переваги:
  1) інтуїтивний і простий у використанні інтерфейс, що робить його зручним для всіх користувачів незалежно від рівня технічних навичок;
  2) великий набір функціональних можливостей.

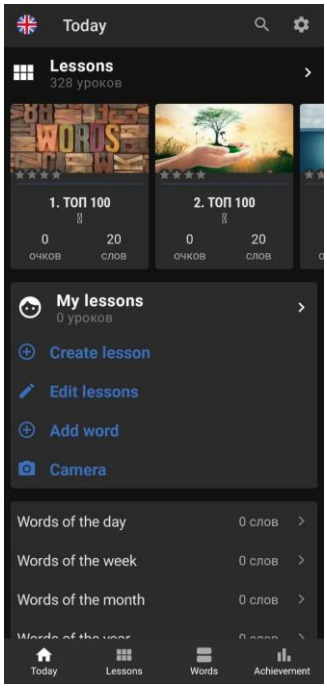



Рисунок 1.5 — Головний екран застосунку Words

### 1.5 Постановка задачі дослідження

Метою роботи є розробка мобільного застосунку для вивчення іноземних слів за допомогою методу карток. Дане дослідження виконується в рамках в рамках кваліфікаційної роботи магістра за спеціальністю 122 «Комп'ютерні науки».

Розроблений програмний продукт має задовольняти наступні вимоги:

- можливість зберігання великої кількості карток;
- можливість додавання та видалення карток;
- функціонал вивчення та повторення карток;

- мати зручний для користувача інтерфейс.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

1) спроектувати застосунок
2) обрати стек технологій на якому буде розроблено застосунок;
3) реалізувати мобільний застосунок.
4) проаналізувати ряд майбутніх покращень.

Предметом дослідження є методологія вивчення лексики іноземної мови за допомогою методу карток та інтервального повторення.

Наукова новизна полягає в тому, що на відміну від існуючих аналогів мобільних застосунків, описане у даній роботі програмне рішення дозволить зосередитись користувачу тільки на вивченні слів, представити їх у вигляді карток, додавати та видаляти.

Практичною значимістю буде те, що застосунок дозволить вивчати слова іноземною мовою за допомогою карток не будучи прив’язаним до інтернету, а також простого дизайну, який є зручним та зрозумілим будь кому. Завдяки простому дизайну користувач буде більше приділяти уваги словам для вивчення. Окрім цього, є можливість самому створювати теми зі словами, редагувати їх та самому додавати слова, які б хотів би вивчити та не забувати.

# 2 ПРОЕКТУВАННЯ ЗАСТОСУНКУ ТА ОПИС ТЕХНОЛОГІЙ

## 2.1 Проектування функціональних можливостей застосунку

Оцінивши свій досвід після використання, тестування та аналізу вже існуючих додатків, я, для себе, спланував необхідно-мінімальний функціонал який хотів би бачити у додатку. Додаток повинен мати низку корисних можливостей, які будуть розподілені на три головних розділи:

- "Навчання";
- "Теми";
- "Словник".

До кожного розділу можливо перейти в будь який момент користування додатком. Розглянемо більш детальніше кожен з цих розділів.

### 2.1.1 Опис функціональних можливостей сторінки "Навчання"

Під час вмикання застосунку, це перший розділ який бачить користувач. У цьому розділі користувач повинен обрати тип вивчення слів за яким буде виконуватись або сортування слів для вивчення або можливість повторювати слова та підтвердження вибору відповідною кнопкою. Із цього опису виникають такі пункти:

- випадковий вибір слів для вивчення;
- вивчення слів з обраної користувачем теми;
- повторення слів.

З вибору відповідної категорії відкривається додатковий, за можливості, вибір. Так, із вибором першої категорії - "Випадкові слова", з’явиться кнопка в нижній частині екрану "Вчити", що запустить процес переходу до вивчення слів у випадковому порядку із тих, що маються у базі даних.

Із вибором наступної пункту - "Слова по темі", користувачу буде запропоновано випадний список з усіх тем, що є в наявності із розділу "Теми". Після вибору конкретної теми стане активною кнопка "Вчити", що є ідентичною до тієї, що при виборі першого категорії із списку вивчення слів. Далі запуститься процес вивчення слів, які поєднані в певну, обрану тему.

При виборі останньої категорії - "Повторення слів", користувач, схожий на перший варіант, з’явиться кнопка "Повторювати", що буде показувати для слова, які уже помічені, як вивчені, з метою повторення та кращого запам’ятовування.

Після вибори однієї з описаних категорій вивчення слів з’являються картки зі словами, що потрібно вчити. На цих картках маємо наступні поля:

- слово;
- тема, до якої належить це слово;
- словосполучення із цим словом.

На звороті картці, яку можна побачити при натисканні на картку, є ті самі поля та переклад слова, що вивчається. Щоб відмітити, що слово вже вивчено потрібно здійснити перегортання картки в праву сторону. Якщо ж користувач хоче вивчити слово пізніше, або ще не достатньо вивчив, він може здійснити перегортання картки в ліву сторону.

### 2.1.2 Опис функціональних можливостей сторінки "Теми"

У цьому розділі користувачу демонструється весь перелік тем які є в додатку. Для зручного користування застосунком є можливість використовувати пошук, для швидкого знаходження потрібної теми. Функціонально є можливість додати нову тему та переглянутим та редагувати існуючу. Детально про форми:

- форма "Додати тему" включає поле для введення назви теми та відповідно кнопки додавання та відміни додавання;
- сторінка “Перегляд існуючої” включає список слів в цій темі. Швидкий пошук, для зручного використання застосунку. Можливість редагувати або видаляти.

### 2.1.3 Опис функціональних можливостей сторінки "Словник"

У цьому розділі користувачі можуть переглядати всі слова в алфавітному порядку. Це надає зручну можливість швидко знайти потрібне слово та перевірити його значення. Також для більш швидкого знаходження слова є пошукова компонента. На цій сторінці присутній такий функціонал як форма "Додати слово". Який включає в себе сторінку, де додається слово із наступними даними:

- тема, з усіх існуючих;
- слово;
- переклад;
- короткий опис чи використання слова у словосполученні.

На сторінці "Словник" користувач має можливість переглядати вміст вибраного слова за допомогою форми "Перегляд слова". Вона включає список властивостей слова, що додаються при створенні слова, тобто всю доступну

інформацію про слово, що мається. Також видалити його чи редагувати(слово, переклад, тему чи опис).

## 2.2 Проектування макету застосунку відповідно до функціональних можливостей

Враховуючи досвід користування додатками із вивчення мов і слів(розділ 1) можна зробити висновок, що вони є нагромаджені багатьма елементами, що ускладнюють орієнтування та вивчення. Тому було вирішено зробити додаток із мінімальною кількістю ефектів та зайвих елементів, що буде інтуїтивно зрозуміло те, що потрібно.

Так, для того, щоб головні розділи були завжди доступні у будь-який момент було вирішено їх винести як меню у нижню частину екрану, як показано на рис. 2.1.

Варто не забувати, що у подальших підпунктах будуть показані макети додатка у відповідності до функціональних вимог, що не є дизайном самого додатку.


Рисунок 2.1 – Меню вибору розділу

### 2.2.1 Макет сторінки "Навчання"

Відповідно до функціональних можливостей, що описані у пункті 2.1.1, враховуючи всі вимоги і побажання, перша сторінка із вибором методу навчання слів на рис. 2.2.

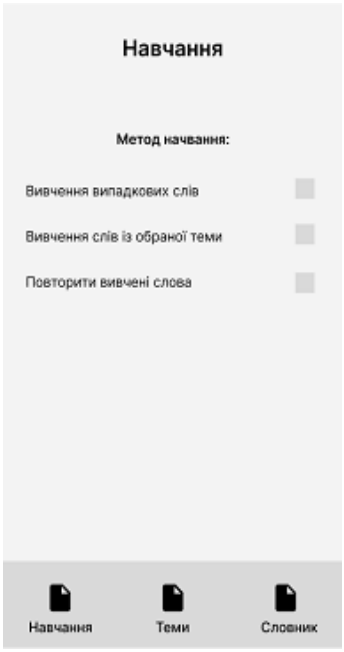
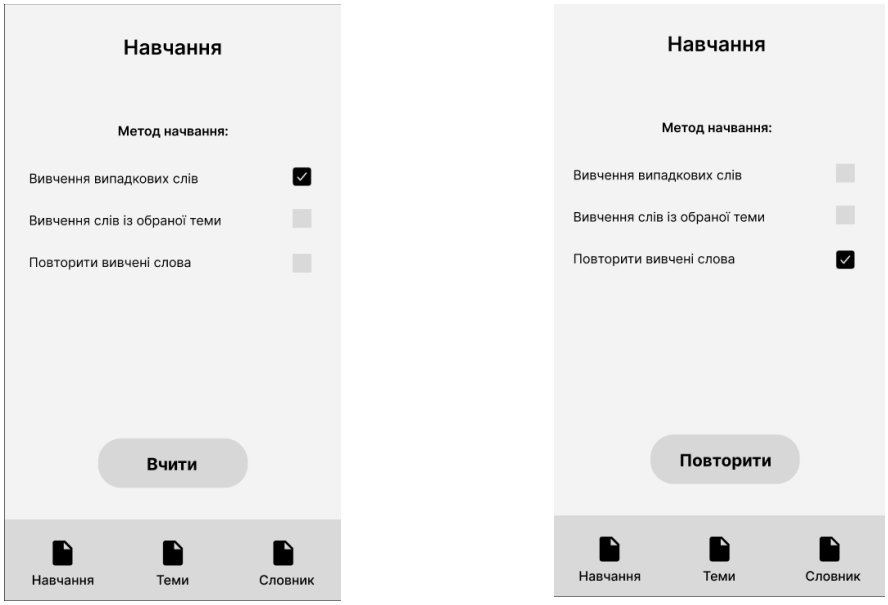



Рисунок 2.2 – Початкова сторінка із розділу "Навчання"

В залежності від вибору потрібного пункту відкриється нова деталь, що веде далі до вивчення. В цьому, за вимогами, варіанти "Вивчення випадкових слів" і " Повторити вивчені слова" мають майже ідентичне продовження, тобто з’явиться кнопка, що веде діла для вивчення.

Як показано на рис.2.3 різницею між цими двома варіантами є текст в кнопці, що з’являється і дає підказку, що ви будете вчити слова чи повторювати уже вивчені слова.




а) б)

Рисунок 2.3 Сторінка розділу "Навчання" із вибором: а) "Вивчення випадкових слів" ; б) "Повторити вивчені слова"

У випадку із вибором пункту "Вивчення слів із обраної теми" користувачу буде запропоновано вибрати тему, як показано на рис.2.4.

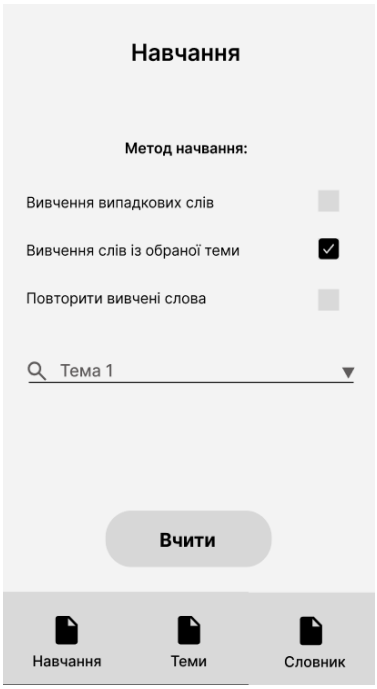



Рисунок 2.4 - Сторінка розділу "Навчання" із вибором пункту "Вивчення слів із обраної теми"

Кнопка "Вчити" у цьому випадку буде неактивною, допоки користувач не вибере тему з випадного списку, і тільки тоді він перейде до вивчення слів тільки певної теми.

Всі варіанти переходу до вивчення слів чи їх повтор будуть вести на сторінку із картками, як показано на рис.2.5. Як і описано у вимогах, картка має слово, тему і застосування(опис) і номер картки. Переклад слова відображається після натиску на картку і мінімальній анімації повороту цієї ж картки. Для того, щоб позначити, що ти вивчив це слово потрібно легким рухом перенести картку в лівий бік, та замість неї з'явилася інша картка з іншим слово. Якщо ж користувач хоче вивчити слово пізніше, або ще не достатньо вивчив, він може здійснити перегортання картки в ліву сторону, та замість неї з'явилася інша картка з іншим слово. А картка зі слово яке перегорнули залишиться в черзі карток для вивчення.

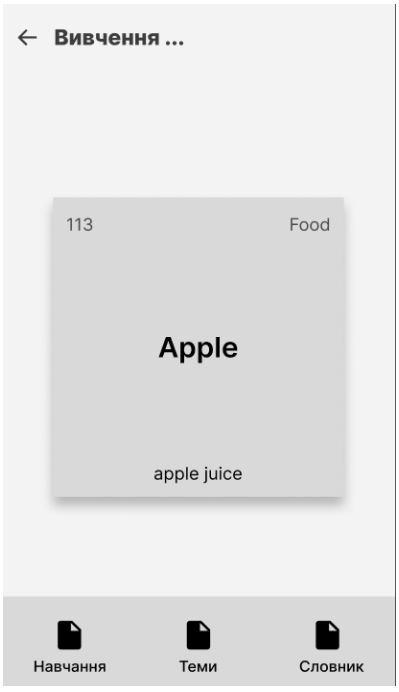



Рисунок 2.5 - Сторінка розділу "Навчання" з картками для вивчення

Згідно з вимогами, які описані в пункті 2.1.1 все реалізовано на макетах, як того і вимагалося. Перейдемо до наступного розділу.

### 2.2.2 Макет сторінки "Теми”

Перейти на розділ "Теми" можна по меню в нижній частині додатку. Перейшовши на сторінку показано: список тем, рядок пошуку і кнопка додавання нової теми, як показано на рис.2.6.

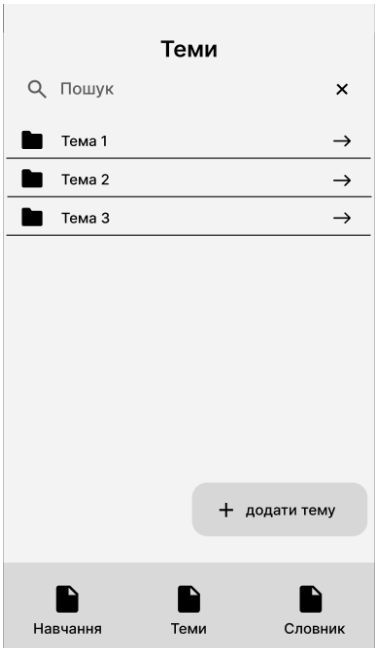



Рис.2.6 - Сторінка розділу "Теми"

У відповідності до вимог можна здійснити не тільки пошук тем, ще і додати у модальному вікні нову тему. Вибравши певну тему та перейшовши на її, відкривається список слів, що є в цій темі, а також функціонал, який дозволяє редагувати назву теми і видалити її (рис.2.7-2.8).

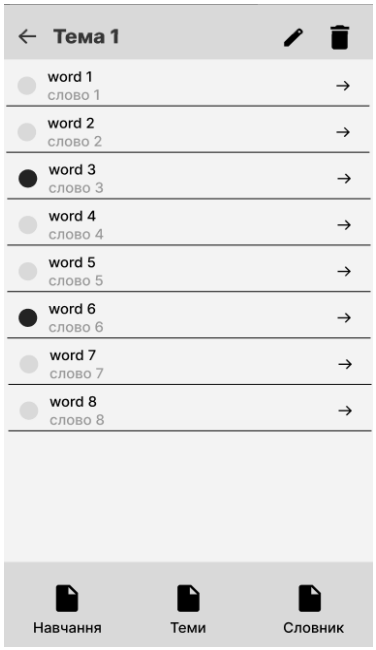
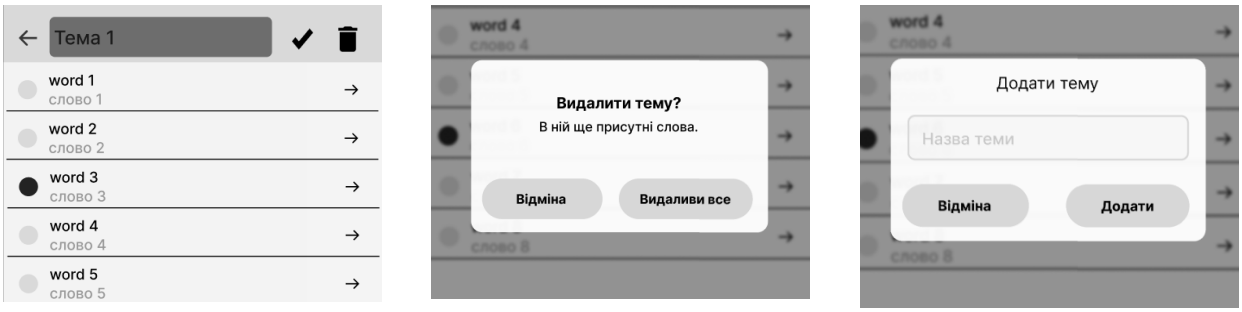



Рисунок 2.7 - Сторінка розділу "Теми" із вибраною темою

Як показано на рис.2.8, редагування назви відбувається в тому самому рядку, що і назва, не створюючи нових компонентів, а ось видалення та додавання тем неможливе без такого функціоналу, і тому при цих діях викликається модальне вікно із відповідним змістом.




а) б) в)

Рисунок 2.8 – Дії з темами: а) редагування назви; б) видалення теми; в) додавання теми

Всі дії, не тільки з темами, а зі словами(додавання, редагування та видалення) супроводжуються додатковим повідомленням про успішність того чи іншого процесу (рис.2.9). Таке повідомлення з’являється над меню на декілька

секунд із відповідним текстом, після того, як додаток перейде на головну сторінку того розділу, на якому ти знаходишся.

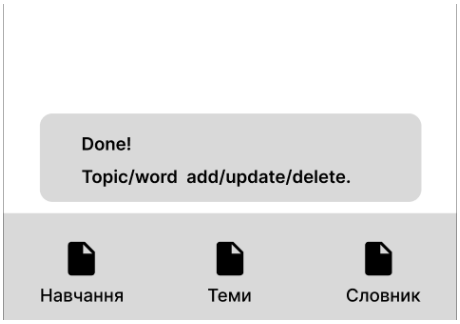



Рисунок 2.9 – Повідомлення про успішність однієї з подій над темою чи словом(або додавання або редагування або видалення)

Також на сторінці однієї теми є умовні позначення – кружечки різних кольорів. Це позначення слів, які були вивчені, а які ще не вивчені. В залежності від цього маркування і будуть видавати різні слова на сторінці розділу "Навчання".

Окрім цього, також як і раніше на сторінці перегляду слів із певної теми, є маленька підказка у вигляді стрілка, що дозволяє перейти на сторінку перегляду слова (рис.2.10).

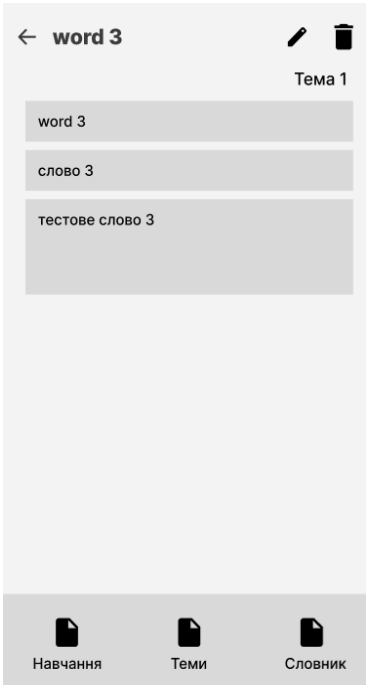



Рисунок 2.10 – Сторінка перегляду інформації про слово

Функціонал цієї сторінки схожий із попередньою в тому, що під час перегляду деталі про слово(саме слово, переклад, та застосування) можливо редагувати інформацію про слово, в тому числі і тему. Макет цієї сторінки є ідентичний до сторінки додавання слова (рис.2.11), що пропонує заповнити поля, і напівпрозорий текст є підказкою. При редагуванні ж використовують уже наявні дані, і тому підказки відсутні, хоча все зрозуміло інтуїтивно, яке поле за що відповідає.

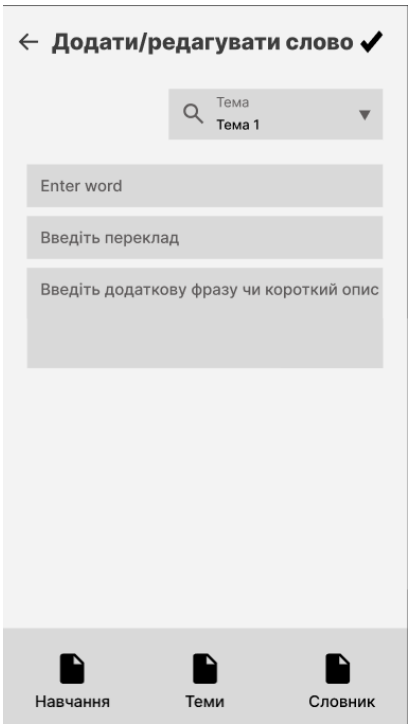



Рисунок 2.11 - Сторінка із формою додавання чи редагування інформації про слово

Згідно з вимогами, які описані в пункті 2.1.2 все реалізовано на макетах, як того і вимагалося. Перейдемо до наступного розділу.

### 2.2.3 Макет сторінки "Словник"

Маємо останній розділ із трьох головних в меню - "Словник". Він, як і решта знаходиться внизу. Під час переходу на нього відкривається список із всіх слів в алфавітному порядку, рядок пошуку слів та кнопка, в якої буде функціонал додавати нові слова (рис.2.12) .

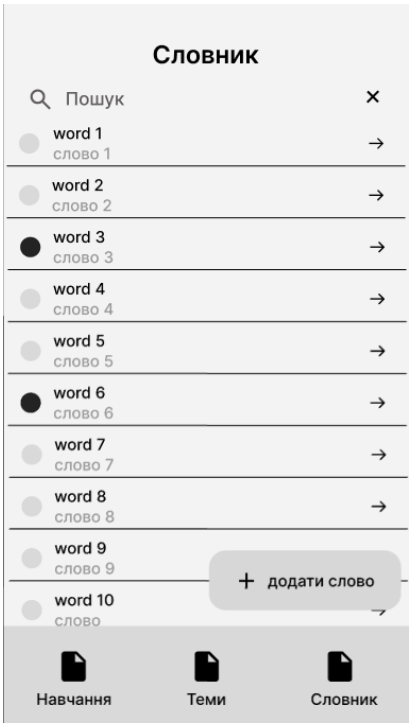



Рисунок 2.12 - Сторінка розділу "Словник"

Як бачимо по макету, можна перейти до певного слова за детальною інформацією, що показується на картці в розділі "Навчання". Детально про перегляд слова, його редагування та додавання описано в попередньому пункті. Тому згідно з функціональними вимогами до сторінки "Словник" вони виконані в повній мірі.

### 2.2.4 Вибір дизайну , загального стилю застосунку

Коли готовий макет до додатка, варто вирішити якусь певну стилістику для всього продукту. Оскільки було вирішено робити застосунок в мінімалістичному стилі, без зайвих деталей та елементів, що буде інтуїтивно зрозуміло відбувся пошук тих стилів і бібліотек, що можуть підійти.

Кращим і швидшим варіантом для такого став Material 3. Такі стилі мають ряд переваг:

- версія систем готових дизайнів від Google;
- буде мати підтримку ще довгий час;
- має невелику потребу у ресурсах телефона;

- має велику підтримку серед розробників сайтів та додатків на різних системах;
- має звичний для користувачів Android стиль.

Саме з такими перевагами було обрано Material 3. Оскільки було обрано, яку систему дизайнів використовувати, настав час із поміж всього вибрати кольорову гаму для застосунку.

### 2.3 Проектування баз даних

Враховуючи функціональні вимоги, які описані в розділі 2.1 щодо того, що необхідно було в додатку та макети, які створені на основі цих вимог потрібно створити базу даних для збереження цим всіх даних. На основі цього була сформована така база даних, як на рис.2.13.

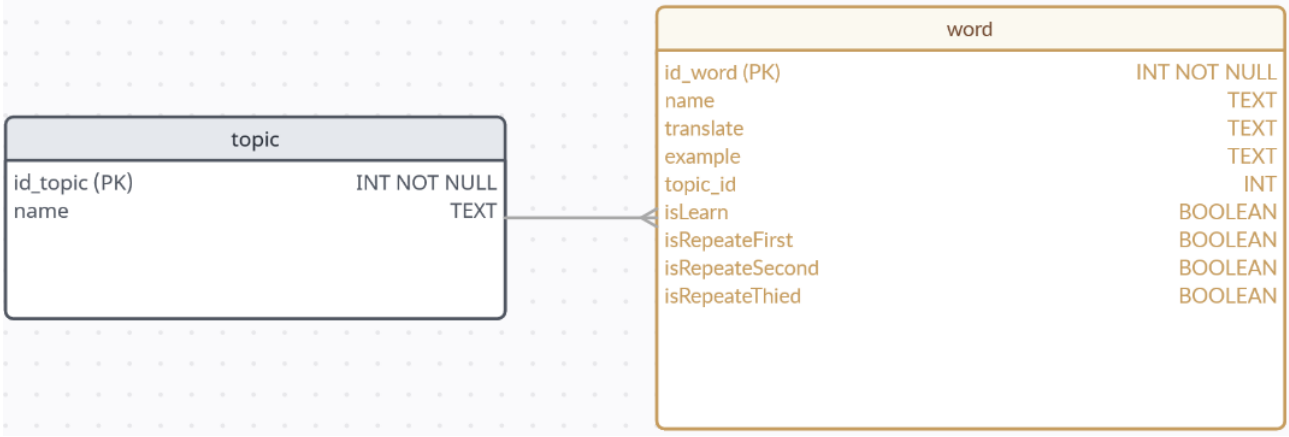



Рисунок 2.13 – База даних додатку

Маємо дві таблиці, які представляють собою теми і слова. Зв’язок у між ним один до багатьох, тобто в одній темі може бути багато слів, а одне слово матиме тільки одну тему. Зв’язком між таблицями є topic_id якому відповідає зовнішній ключ id_topic.

### 2.4 Стек технологій на якому буде створено додаток

#### 2.4.1 Мова програмування

Для вирішення поставленої проблеми я вирішив використати доволі нову за часом створення мову програмування Dart та його фреймворк Flutter.

Dart - це сучасна мова програмування, розроблена компанією Google, зорієнтована на швидкість, продуктивність та надійність. Dart має чистий синтаксис, що робить його легким для вивчення, навіть для початківців. Він поєднує в собі властивості звичайних об'єктно-орієнтованих мов зі сучасними можливостями, такими як асинхронні функції та стрілкові функції. Dart компілюється у виконуваний код, що дозволить досягти високої швидкодії виконання. Це особливо важливо для розробки мобільних додатків, де продуктивність грає ключову роль. Dart підтримує кросплатформенну розробку, що означає, що ми можемо використовувати один і той же код для розробки застосунків для різних платформ, таких як iOS, Android та веб.

Flutter - це відкритий фреймворк для розробки кросплатформенних мобільних застосунків, який також розроблений компанією Google. Одна з найбільш важливих особливостей Flutter - це можливість створення кросплатформенних застосунків з використанням єдиної кодової бази. Це означає, що ми можемо розробляти один раз і запускати застосунок як на iOS, так і на Android, що робить процес розробки ефективнішим та менш затратним. За допомогою власного рушія рендерингу, Flutter забезпечує відмінну продуктивність та привабливий вигляд для мобільних застосунків. Його швидкість робить його ідеальним вибором для розробки відчутних застосунків з відмінною анімацією та інтерактивністю. Flutter надає розробникам широкі можливості для розширення функціоналу застосунків за допомогою власних віджетів, пакетів та розширень.

Комбінація мови програмування Dart та фреймворку Flutter створює потужний інструмент для розробки кросплатформенних мобільних застосунків, що дозволяє розробникам швидко, ефективно та елегантно створювати високоякісні додатки для широкого кола платформ.

### 2.4.2 База даних

SQLite - це легка вбудовувана реляційна база даних, яка не потребує окремого сервера та може бути використана безпосередньо в програмах. Вона

широко використовується для розробки мобільних додатків, зокрема для платформи Android, через свою простоту використання та ефективність. SQLite - це бібліотека, яка легко вбудовується в мобільні додатки. Вона не потребує окремого сервера баз даних, тому всі дані зберігаються безпосередньо на пристрої користувача. SQLite має дуже невеликий розмір, що робить його ідеальним вибором для мобільних додатків, які мають обмежений обсяг. Він також є досить ефективним у використанні ресурсів пристрою, що дозволяє забезпечити високу продуктивність додатків. SQLite підтримує мову запитів SQL, що дозволяє розробникам виконувати різні операції з даними, такі як вибірка, вставка, оновлення та видалення записів. SQLite підтримує транзакції, що дозволяє виконувати групу операцій як один атомарний процес. Крім того, він дотримується властивостей ACID (Atomicity, Consistency, Isolation, Durability), що забезпечує надійність даних. Для використання SQLite в мобільних додатках для Android існує вбудована підтримка через пакет android.database.sqlite. Цей пакет надає API для створення та управління базами даних SQLite в мобільних додатках.

SQLite є потужним та ефективним інструментом для роботи з базами даних в мобільних додатках на платформі Android. Він дозволяє розробникам зберігати та управляти даними безпосередньо на пристрої користувача з мінімальними витратами ресурсів та коду.

# 3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## 3.1 Створення порожнього проекту та тестовий запуск

Після встановлення всіх інструментів варто створити тестовий проект, та перевірити роботу головних компонентів. Для цього необхідно:

- Запустити застосунок Android Studio та у вінкі натиснути на кнопку “New Flutter Project”.
- У відповідному полі “Project name” вказати назву проекту, в даному випадку це test_project та натиснути внизу вікна кнопку Create.
- В результаті чого відкриється повноцінне вікно редактору та сам проект, як показано на рис. 3.1.

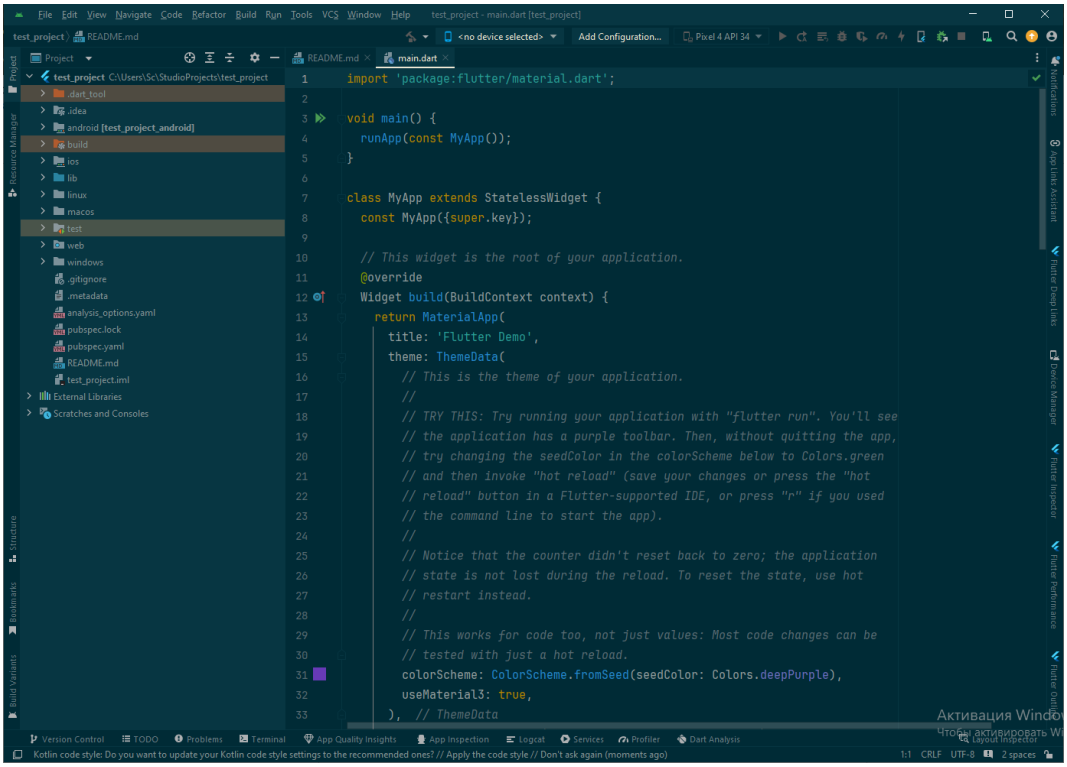



Рисунок 3.1 - Вікно редактора Android Studio

Після створення проекту потрібно виконати тестовий запуск на пристрої. На вибір у користувача є використання фізичного чи віртуального пристрою. Для запуску тестового додатку на віртуальному пристрої необхідно виконати, ряд простих дій, а саме:

- У верхньому правому куті натиснути на відповідну кнопку, як показано на рис. 3.2 та відкрити Device manager. Він містить в собі список віртуальних пристроїв, налаштованих в IDE. Пристроєм за замовчуванням є Pixel 3A.
- Натиснути на кнопку старту пристрою.
- Відкриється вікно на якому буде відображатись віртуальний пристрій.

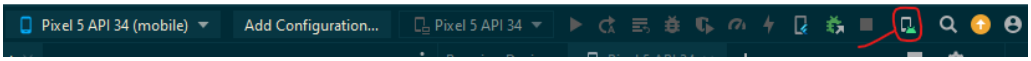



Рисунок 3.2 - Розташування кнопки Device manager на панелі запуску та дебагінгу проекту

- Після цього на панелі запуску та дебагінгу проекту натиснути відповідну кнопку запустити проект. В результаті на віртуальному пристрої встановиться наш застосунок в режимі дебагінгу , що дозволяє тестувати та проводити маніпуляції над ним,як показано на рис. 3.3.

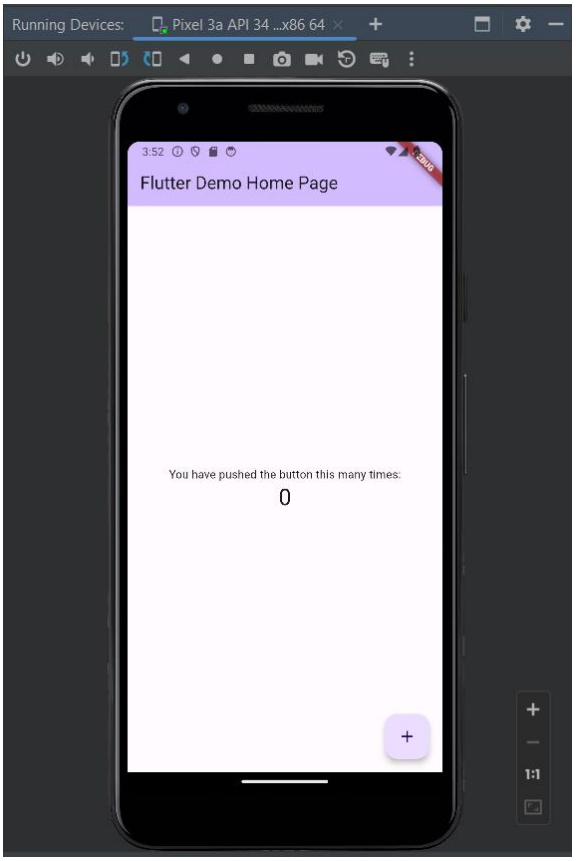



Рисунок 3.7 - Віртуальний пристрій з тестовим додатком

### 3.2 Програмна реалізація особливостей застосунку відповідно до функціональних особливостей та стилю

Розробка додатку відбувається відповідно до макету та вимог, що описані у розділі 2. Але під час реалізації застосунку, з'явились деякі деталі, які не описані в макеті чи функціональних вимогах. Зупинюсь на деяких більш детально.

Навігація - один з ключових механізмів роботи в будь якому додатку. Для реалізації навігації у своєму застосунку я скористався не типовими методами а окремою імпортованою бібліотекою auto_route. Вона дозволяє передавати аргументи зі строгим типом, легко створювати глибокі посилання та використовує генерацію коду для спрощення налаштування маршрутів. Зважаючи на це, для створення всього необхідного для навігації всередині вашої програми потрібна мінімальна кількість коду. Для реалізації цього пакунку в файл pubspec.yaml в блок dependencies імпортували 3 сторонні пакети, як показано на рис 3.4.

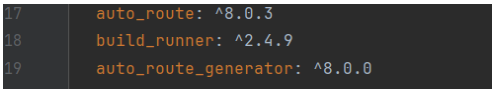



Рисунок 3.4 - Імпорт бібліотек навігації у файлі pubspec.yaml

В корені проекту створено теку з назвою router, та додано до нього файл router.dart з кодом, як показано на рис 3.5.

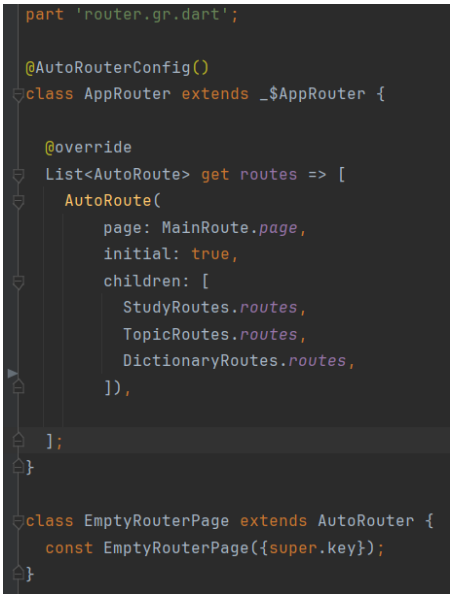



Рисунок 3.5 Файл router.dart

Цей файл конфігурує ієрархію підпорядкування сторінок один одному. В нашому випадку головною сторінкою буде сторінка MainPage, яка відповідає за реалізацію нижнього меню та переходу по ньому. Також в цьому файлі описані сторінки які відображаються від обраної навігації. Вони відображені в параметрі children[]. Для ініціалізації сторінок в конфігураційному файлі ми декларували класи сторінок за допомогою команди декларування @RoutePage(). Наприклад, сторінку MainPage , як показано на рис 3.6.

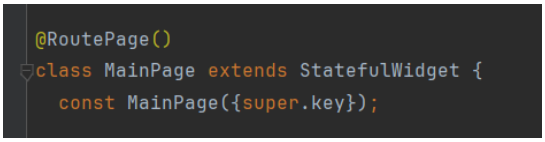



Рисунок 3.6 Декларування сторінки класу MainPage для ініціалізації

Через архітектурну особливість, застосунок на сторінці MainPage повинен відображати три головні функціональні сторінки. Для їх декларування розділили кожну сторінку на окремий функціонал додатку за допомогою створення тек, як показано на рис 3.9. Для кожної сторінки додано теку з назвою router, для налаштування навігації. Наприклад, для сторінки StudyPage в router теці створено два файли, як показано на рис. 3.9 - study_routes.dart та study_route_wrapper_screen.dart. Ці файли потрібні для корректного генерування зв’язків між сторінками.

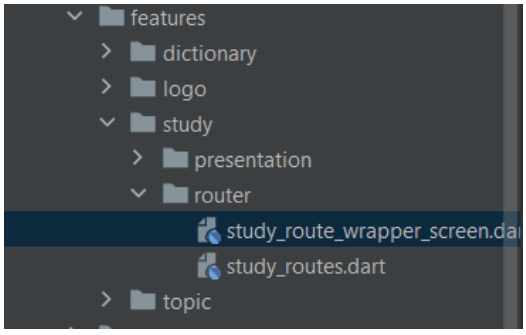



Рисунок 3.7 - Розділення сторінок на окремий функціонал

У файлі study_routes.dart описана ієрархія підпорядкування сторінок, як показано на рис 3.8.

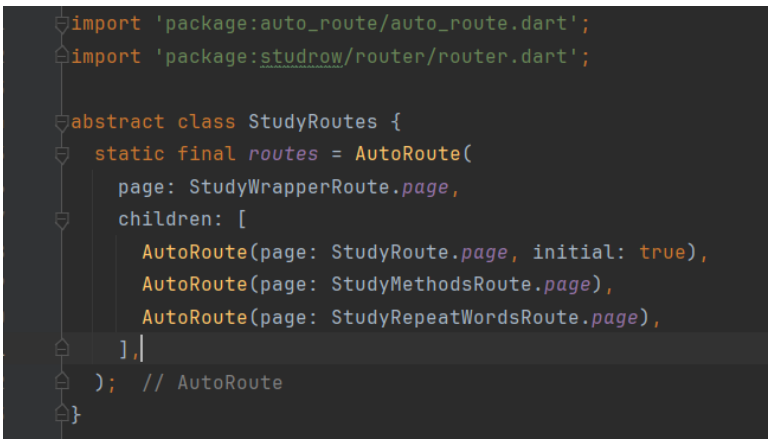



Рисунок 3.8 - Файл study_routes.dart

Для коректної роботи згенерували об’єднання всіх відповідних зв’язків за допомогою допоміжної бібліотеки, яку ми імпортували раніше build_runner, командою в терміналі проекту flutter packages pub run build_runner build. Після виконання комнади згенерувався файл router.gr.dart який є окремою частиною файлу router.dart. В цьому файлі відповідні перетворення та утворення зв’язків, як показано на рис 3.8.

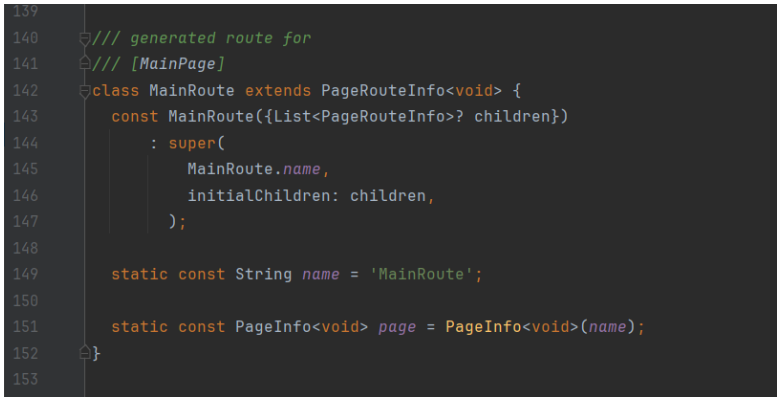



Рисунок 3.8 Згенеровані зв’язки сторінки MainPage

В результаті використання цих бібліотек для налаштування та конфігурації навігації в застосунку, маємо зручність використання у вигляді короткого коду (наприклад AutoRouter.*of*(context).push(StudyRepeatWordsRoute()); - перехід до наступної підпорядкованої сторінки), швидкий спосіб конфігурування та гнучкість налаштувань.

Дані - це ще один з ключових механізмів роботи в будь якому додатку. В застосунку використовується база даних sqlite. Для зручної роботи з базою даних використано сторонню біблотеку sqflite. Проста у використанні бібліотека, надає можливості зручного формування запитів до бази даних у вигляді запиту або у

вигляді функції з параметрами. Для використання створено у теці domain теку repository, де створено файл main_repository.dart. В цьому файлі описано підключення до бази даних та основні методи роботи з нею. Для підключення бази даних використано метод _database(), як показано на рис 3.9. Для використання методу _database() обов'язково потрібен метод _createDB(Database db) , якому в параметри передається база даних. При відсутності бази даних на пристрої метод _createDB(Database db) потрібен для створення таблиць у базі даних.

```
static const _dbName = 'learny_word.db';
static const _tableNameTopic = 'topic';
static const _tableNameWord = 'word';
static int i = 20;

static Future<Database> _database() async {
  final database = openDatabase(
    join(await getDatabasesPath(), _dbName),
    onCreate: _createDB,
    version: 1,
  );
  return database;
}

//initial
static Future _createDB(Database db, int version) async {
  await db.execute(
      'CREATE TABLE $_tableNameTopic(id_topic INTEGER PRIMARY KEY NOT NULL, name TEXT)');
  await db.execute('CREATE TABLE $_tableNameWord ('
      'id_word INTEGER PRIMARY KEY NOT NULL, '
      'name TEXT,'
      'translate TEXT,'
      'example TEXT,'
      'topic_id INTEGER,'
      'isLearn BOOLEAN,'
      'isRepeatFirst BOOLEAN,'
      'isRepeatSecond BOOLEAN,'
      'isRepeatThird BOOLEAN,'
      'FOREIGN KEY(topic_id) REFERENCES $_tableNameTopic(id_topic))');
```

Рисунок 3.9 Підключення та створення таблиць в базі даних

В результаті створення застосунку та реалізації функціональних вимог було використано різні способи запитів до бази даних. Так було використано метод _incertWord(Word word). Цей метод приймає в себе параметр клас Word та записує нове слово до бази даних. В цьому прикладі використано запит до бази даних у вигляді функції,як показано на рис 3.10, без написання власноруч запиту.

```
static Future<void> insertWord(Word word) async {
  final db = await _database();
  await db.insert(
    _tableNameWord,
    word.toMap(),
    conflictAlgorithm: ConflictAlgorithm.fail,
  );
}
```

Рисунок 3.10 - Метод _insertWord

Також було використано запит до бази даних тільки за допомогою запиту. Метод getCountRepetition створює запит до бази даних та отримує кількість слів які користувачу потрібно повторити. Цей метод побудований за допомогою запиту та параметрів.

```
static Future<int?> getCountRepetition(int isLearn, int isRepeatFirst,
    int isRepeatSecond, int isRepeatThird) async {
  final db = await _database();
  int? count = Sqflite.firstIntValue(await db.rawQuery(
      'SELECT COUNT(*) FROM $_tableNameWord '
      'WHERE isLearn = ? AND isRepeatFirst = ? AND isRepeatSecond = ? AND isRepeatThird = ?;',
      [isLearn, isRepeatFirst, isRepeatSecond, isRepeatThird]));
  await db.close();
  return count;
}
```

Рисунок 3.11 - Метод getCountRepetition

Теми - це один із компонентів дизайну та стилю. Для створення додатку в якому всі частини швидко замінні, використано механізм генерації тем та кольорів material theme builder. Цей сервіс допомагає обрати загальну кольорову гамму, шрифт та згенерувати файли, як показано на рис 3.12.

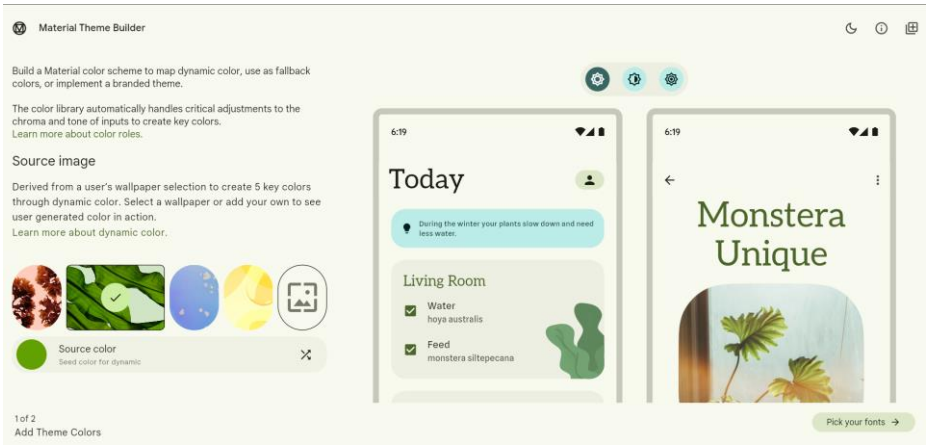



Рисунок 3.12  Material theme builder

Ці файли містять  функції з кольорами які приймають участь в цій кольоровій гамі та шрифтами. Створено відповідно теку в дереві файлів проекту, як показано на рис 3.13.

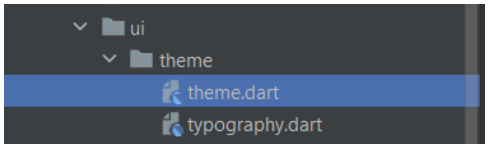



Рисунок 3.13 - Ієрархія тек та файлів для тем

В такому методі реалізації до кольорів теми є один з головних позитивних аспектів це те, якщо з’явиться бажання змінити кольори додатку, буде достатньо сформувати набір кольорової гами, згенерувати файли та замінити відовідні файли в проекті. В результаті чого весь проект може змінитись до невпізнаваності за допомогою пару нискань миші.

Ініціалізація даних - це механізм для швидкого ознайомлення користувача з застосунком. При першому запуску застосунку та створенні бази даних, програмно створюється набір слів та тем. Ініціалізацію даних виконують два методи. За ініціалізацію тем відповідає метод insertInitializationTopics(Database db), якому в параметри передаємо базу даних. Для додавання однієї теми в базу даних в методі використано цей код “await db.execute("INSERT INTO $_tableNameTopic(id_topic,name) VALUES (?, ?);", [null, "Eating"]);”. За ініціалізацію слів відповідає метод insertInitializationWords(Database db), якому в параметри передаємо базу даних. Для додавання одного слова в базу даних в методі використано цей код “await db.execute("INSERT INTO $_tableNameWord(id_word,name, translate, example, topic_id, isLearn, isRepeatFirst, isRepeatSecond, isRepeatThird) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?);", [null, 'cake (a)', 'торт', 'Do you want some cake?', 1, 0, 0, 0, 0]);” .

Під час першого запуску тепер користувач бачe ініціалізований набір тем та слів, як показано на рис 3.7.

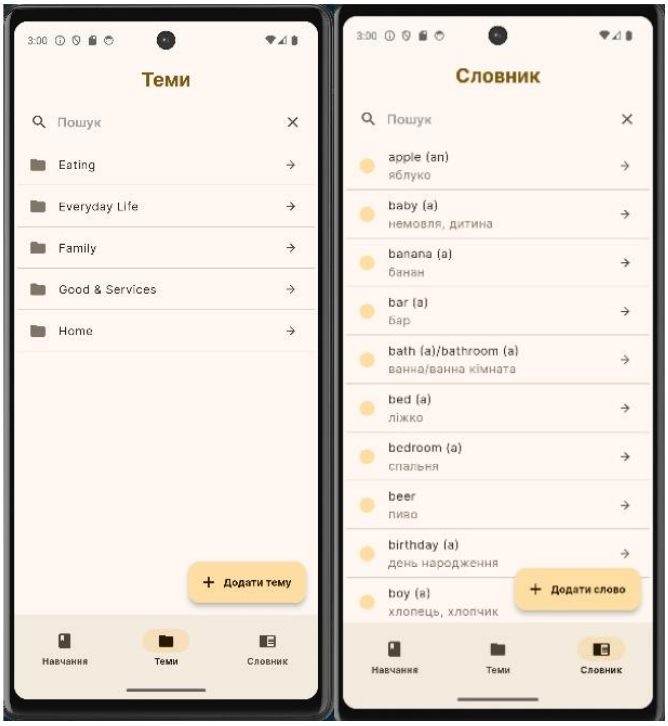



Рисунок 3.7 - Ініціалізований набір даних

Написи - це один з компонентів дизайну та стилю. Для реалізації швидкої змінюваності додатку, вирішено було використовувати всі написи у застосунку як константи та зберігати в одному файлі, який називається costants.dart. За допомогою цього методу,якщо є бажання змінити назву кнопки, то більше не потрібно шукати де ця кнопка реалізована чи запрограмована. Для цього необхідно тільки змінити значення змінної яка відповідає за цю кнопку в файлі constants.dart. В результаті чого зручно змінювати назви та повідомлення системні та в майбутніх оновленнях можливо з меншими модифікаціями архітектури додатку додати локалізацію.

В результаті кропіткої робити, проект додатку має вигляд, як показано на рисунку 3.4, а застосунок відповідає заявленому макету та функціональним вимогам.

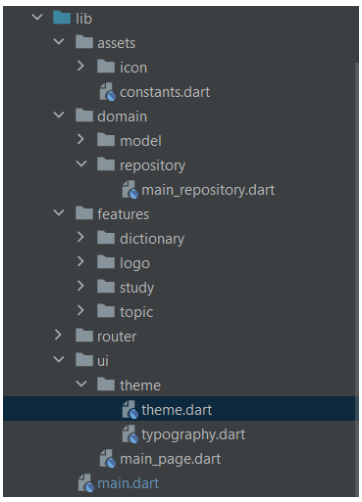



Рисунок 3.4 – Ієрархічна структура проекту додатку

### 3.3 Планування майбутніх оновлень для покращення застосунку

Після тестового використання застосунку особисто я сформував певні функціональні особливості, які хотів би покращити чи додати в наступних оновленнях:

- Авторизація та аутентифікація. В застосунку потрібно реалізувати збереження даних користувача в його особистому акаунті. Дані про акаунт повинні зберігатись у хмарі та мати віддалений доступ через мережу інтернет. Додати сторінки авторизації та аутентифікації користувача. За допомогою цієї функціональної можливості, можливо реалізувати доступ до списків слів з інших пристроїв.
- Імпорт/Експорт. В застосунку наступною важливою функцією потрібно додати імпорт та експорт слів. Для зручного обміну словами між друзями, чи обміну словами між викладачем та учнями.
- Додавання зображень. Деякі люди, для кращого запам’ятовування слів асоціюють їх з зображеннями. Для реалізації цього методу, слід додати можливість за бажанням додавати до слова зображення.
- Локалізація. В застосунку слід додати локалізацію, для використання застосунку в інших країнах.
- Налаштування. Реалізувати можливість налаштувань додатку під потреби коистувача.

- Вивчення декількох мов. Додати можливість створювати словники для різних мов, таким чином користувач може вивчати декілька мов одночасно.
- Система мотивації. В застосунку для користувачів систему мотивації, яка буде полягати в тому що за вивчення слів користувач буде отримувати накопичувальні бали. За кількістю цих балів, можна буде переглянути кращих учнів в твоєму класі, місті.

# ВИСНОВКИ

У процесі дослідження та аналізу існуючих додатків для вивчення слів за допомогою карток було виявлено відсутність повноцінного та ефективного рішення для швидкого та зручного навчання. Для заповнення цієї прогалини було сплановано розробку власного мобільного застосунку, який має на меті забезпечити користувачам простий та зручний інструмент для вивчення нових слів з використанням карток. Застосунок пропонує широкий спектр можливостей, таких як система організації за темами, можливість додавання власних слів та персоналізований словник для зручного вивчення. Завдяки цим функціям цей додаток має потенціал стати незамінним інструментом для всіх, хто прагне активно вдосконалювати свої мовні навички та розширювати словниковий запас. В результаті тестування та використання застосунку мною було сформовано список функціональних можливостей, які я хотів би реалізувати в наступних оновлення додатку.

## СПИСОК ЛІТЕРАТУРИ


1. Barat K. Proceedings 1 st International Conference on Literacy and Education Exploring innovations, opportunities, and challenges in Literacy and Education in the era of industrial revolution 4.0 Rectorate of IKIP-PGRI Pontianak Editor: Aunurrahman STUDY PROGRAM OF ENGLISH LANGUAGE EDUCATION FACULTY OF LANGUAGE AND ARTS EDUCATION IKIP-PGRI PONTIANAK. ISBN 978-623-93430-0-2.
2. Lai C. H., Jong B. S., Hsia Y. T. et al. Integrating flash cards with narratives for mobile learning of english vocabulary. International Journal of Interactive Mobile Technologies. Vol. 14, Issue 4. P. 4–16. DOI:10.3991/IJIM.V14I04.11723.
3. Nguyen-Thi H., Hwang W. Y., Pham X. L. et al. Understanding the participant behaviors in the online english learning mobile App: A case study among 106,912 learners via google play app store. Proceedings - IEEE 16th International Conference on Advanced Learning Technologies, ICALT 2016. 28.11.2016. P. 104–108. DOI:10.1109/ICALT.2016.137.
4. Safarova F. I. Mobile Applications As A Modern Means of Learning English. International Conference on Information Science and Communications Technologies: Applications, Trends and Opportunities, ICISCT 2019. 01.11.2019. DOI:10.1109/ICISCT47635.2019.9011897.
5. Scott K., Sengsri S., Tyson R. et al. Android Apps for Education: An International Collaboration. Journal of Community Development Research (Humanities and Social Sciences). Vol. 8, Issue 1. P. 1–15. DOI:10.14456/JCDR.V8I1.924.
6. Wu Q. Designing a smartphone app to teach English (L2) vocabulary. Computers & Education. Vol. 85, 01.07.2015. P. 170–179. DOI:10.1016/J.COMPEDU.2015.02.013.
7. Ефективне вивчення англійської мови: методи й підходи [Електронний ресурс] – Режим доступу до ресурсу:

https://p12.com.ua/journal/view/effektivnoe-izuchenie-anglijskogo-metody-i-podhody

8. П'ять порад для початку вивчення іноземної мови. [Електронний ресурс] – Режим доступу до ресурсус: https://abc-lviv.com.ua/blog/piat-porad-dlia-pochatku-vyvchennia-inozemnoi-movy
9. Магазин з картками для вивчення мов Eng Student. [Електронний ресурс] – Режим доступу до ресурсу: https://eng-student.com/ua
10. 12 порад, як швидко вивчити англійську і як це зробити онлайн. [Електронний ресурс] – Режим доступу до ресурсу: https://tvoemisto.tv/news/12_porad_yak_shvydko_vyvchyty_angliysku_66614.html
11. Як вчити англійські слова швидко та ефективно. [Електронний ресурс] – Режим доступу до ресурсу: https://www.oratorica.ua/ua/blog/yak-vivchati-angl-ysk-slova-schodnya-poradi-l-ngv-st-v
12. Як вчити англійську за допомогою ігор, караоке та фільмів. [Електронний ресурс] – Режим доступу до ресурсу: https://laba.ua/blog/3536-yak-vchyty-angliysku-za-dopomogoyu-igor-karaoke-ta-filmiv
13. Секрети вивчення іноземних мов. [Електронний ресурс] – Режим доступу до ресурсу: https://buki.com.ua/blogs/sekrety-vyvchennya-inozemnykh-mov/
14. 5 Найбільш ефективних методів вивчення іноземних мов. [Електронний ресурс] – Режим доступу до ресурсу: https://osvitanova.com.ua/posts/502-5-naybilsh-efektyvnykh-metodiv-vyvchennya-inozemnoyi-movy
15. 4 способи збільшити свій словниковий запас англійської мови. [Електронний ресурс] – Режим доступу до ресурсу: https://enguide.ua/magazine/5-sposobov-uvelichit-slovarnyy-zapas-angliyskogo-yazyka
16. 8 ефективний способів вдосконалення словникового запасу англійської мови. [Електронний ресурс] – Режим доступу до ресурсу: https://belsmalta.com/ua

17. 12 легких способів швидко вивчити англійські слова. [Електронний ресурс] – Режим доступу до ресурсу: https://redford.ua/ua/12-legkix-sposobiv-shvidko-vivchiti-anglijski-slova/
18. Ваш власний гайд: як поповнити словниковий запас англійської мови? [Електронний ресурс] – Режим доступу до ресурсу: https://cererra.com/blog/slovarnyj-zapas
19. Як збільшити словниковий запас англійської: різні способи. [Електронний ресурс] – Режим доступу до ресурсу: https://bigbro.com.ua/yak-zbilshiti-slovnikovij-zapas-anglijskoyi-rizni-sposobi/
20. Не говори чи не говоримо? Як айтішніку вивчити англійську мову. [Електронний ресурс] – Режим доступу до ресурсу: https://javarush.com/ua/groups/posts/uk.285.ne-govori-chi-ne-govorimo-jak-aytshniku-vivchiti-anglysjhku
21. Вивчення іноземної мови: топ лайфхаків. [Електронний ресурс] – Режим доступу до ресурсу: https://sky-lingua.com/ua/vivchennya-inozemnoi-movi-top-layfhakiv/
22. Застосунок Flashcard World [Електронний ресурс] – Режим доступу до ресурсу: https://play.google.com/store/apps/flashcards.words
23. Застосунок DouCards [Електронний ресурс] – Режим доступу до ресурсу: https://duocards.com/en/
24. Застосунок Words [Електронний ресурс] – Режим доступу до ресурсу: https://play.google.com/store/apps/details?id=words
25. Секрети пам’яті або як ефективно вивчати і запам’ятовувати іноземні слова? [ Електронний ресурс] – Режим доступу до ресурсу: https://greencountry.com.ua/journal/read/sekreti-pamyati-abo-yak-efektivno-vivchati-i-zapamyatovuvati-inozemni-slova
26. Завантаження Android Studio [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/studio

27. Завантаження та налаштування Flutter [Електронний ресурс] – Режим доступу до ресурсу: https://docs.flutter.dev/get-started/install/windows/mobile?tab=download
28. Завантаження та налаштування Git [Електронний ресурс] – Режим доступу до ресурсу: https://git-scm.com/downloads
29. Завантаження та налаштування Sqlite [Електронний ресурс] – Режим доступу до ресурсу: https://www.sqlite.org/download.html
30. Material Design 3 [Електронний ресурс] – Режим доступу до ресурсу: https://m3.material.io/develop/flutter
31. Material Theme Builder [Електронний ресурс] – Режим доступу до ресурсу:https://material-foundation.github.io/material-theme-builder/

# ДОДАТОК А

study_page.dart - цей файл відповідає за створення сторінки Навчання

```
import 'package:auto_route/auto_route.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:flutter_spinkit/flutter_spinkit.dart';
import 'package:provider/provider.dart';
import 'package:studrow/domain/model/type_learn.dart';
import
'package:studrow/features/study/presentation/widgets/snap_message/snap_message.dart';
import 'package:studrow/router/router.dart';
import '../../../../domain/model/topic.dart';
import '../../../topic/presentation/provider/topic_provider.dart';
import '../provider/study_provider.dart';
import 'package:studrow/assets/constants.dart' as Const;

@RoutePage()
class StudyPage extends StatefulWidget {
 const StudyPage({super.key});

 @override
 State<StudyPage> createState() => _StudyPageState();
}

class _StudyPageState extends State<StudyPage> {
 final TextEditingController _topic = TextEditingController();
 bool isLoading = true;
```

```
Topic? selectedTopic;
bool dontEdit = true;
//checkbox
bool isLearnRandom = true;
bool isLearnByTopics = false;
bool isRepeatOldWord = false;
//
//logic repeat and study
String titleButton = Const.STUDY_MAIN_BUTTON_LEARN;
bool mustRepeatWord = false;

@override
void initState() {
  Future.delayed(Duration(seconds: 1), (){
    setState(() {
      isLoading = false;
    });
  });
  super.initState();
}

@override
Widget build(BuildContext context) {
  final provider = Provider.of<TopicProvider>(context);
  if (!provider.isLoadingTopicPage && !isLoading)
  {
    return Scaffold(
      appBar: AppBar(
        scrolledUnderElevation: 0,
        title: Center(
```

```
    child: Text(
      Const.STUDY_TITLE,
      style: TextStyle(
          fontSize: 26, color: Theme.of(context).colorScheme.primary),
    ),
  ),
),
body: Center(
  child: Padding(
    padding: const EdgeInsets.all(20.0),
    child: Container(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.spaceAround,
        children: [
          Container(
            child: Column(
              children: [
                getMethodStudy(),
                getChooseTopic(),
              ],
            ),
          ),
          SizedBox(
            width: 200,
            height: 60,
            child: getButton(titleButton),
          ),
        ],
      ),
    ),
```

```
      ),
    ),
  );
} else {
  return Scaffold(
    appBar: AppBar(
      title: Center(
        child: Text(
          Const.STUDY_TITLE,
          style: TextStyle(
              fontSize: 26, color: Theme.of(context).colorScheme.primary),
        ),
      ),
    ),
    body: Center(
      child: Column(
        mainAxisSize: MainAxisSize.min,
        children: [
          SpinKitRotatingPlain(
            color: Theme.of(context).colorScheme.secondary,
            size: 80,
          ),
          Text(
            Const.SPINKIT_LOADING,
            style: TextStyle(
                fontSize: 20,
                color: Theme.of(context).colorScheme.secondary),
          ),
        ],
      ),
```

```
      ),
    );
  }
}

Widget getMethodStudy() {
  return Container(
    child: Column(
      children: [
        const Padding(
          padding: EdgeInsets.symmetric(vertical: 4),
          child: Text(
            Const.STUDY_CHOOSE_METOD_LEARN_WORD_TITLE,
            style: TextStyle(fontSize: 20),
          ),
        ),
        Padding(
          padding: const EdgeInsets.symmetric(vertical: 10, horizontal: 8),
          child: Column(
            children: <Widget>[
              CheckboxListTile(
                value: isLearnRandom,
                onChanged: (bool? value) {
                  setState(() {
                    if (value! == isLearnByTopics &&
                        value! == isRepeatOldWord) {
                      isLearnRandom = value!;
                    } else {
                      isLearnRandom = value!;
                      isLearnByTopics = !value;
```

```
        isRepeatOldWord = !value;
        titleButton = Const.STUDY_MAIN_BUTTON_LEARN;
      }
    });
  },
  title: const Text(
    Const.STUDY_CHECKBOX_FIRST,
    style: TextStyle(fontSize: 16),
  ),
),
CheckboxListTile(
  value: isLearnByTopics,
  onChanged: (bool? value) {
    setState(() {
      Provider.of<StudyProvider>(context, listen: false).getTopics();
      selectedTopic = null;
      _topic.clear();
      if (value! == isLearnRandom &&
          value! == isRepeatOldWord) {
        isLearnByTopics = value!;
      } else {
        isLearnByTopics = value!;
        isLearnRandom = !value;
        isRepeatOldWord = !value;
        titleButton = Const.STUDY_MAIN_BUTTON_LEARN;
      }
    });
  },
  title: const Text(
    Const.STUDY_CHECKBOX_SECOND,
```

```
                style: TextStyle(fontSize: 16),
              ),
            ),
            CheckboxListTile(
              value: isRepeatOldWord,
              onChanged: (bool? value) {
                setState(() {
                  if (value! == isLearnByTopics &&
                      value! == isLearnRandom) {
                    isRepeatOldWord = value!;
                  } else {
                    isRepeatOldWord = value!;
                    isLearnByTopics = !value;
                    isLearnRandom = !value;
                    titleButton = Const.STUDY_MAIN_BUTTON_REPEAT;
                  }
                });
              },
              title: const Text(
                Const.STUDY_CHECKBOX_THIRD,
                style: TextStyle(fontSize: 16),
              ),
            ),
          ],
        ),
      ),
    ],
  ),
 );
}
```

```
Widget getChooseTopic() {
  if (isLearnByTopics) {
    return Container(
      child: Column(
        children: [
          const Divider(
            height: 0,
          ),
          const Padding(
            padding: EdgeInsets.symmetric(vertical: 10),
            child: Text(
              Const.STUDY_CHOOSE_TOPIC_TITLE,
              style: TextStyle(fontSize: 20),
            ),
          ),
          SizedBox(
            width: double.infinity,
            child: Padding(
              padding:
                  const EdgeInsets.symmetric(horizontal: 8),
              child: Consumer<StudyProvider>(
                builder: (context, provider, child) {
                  return provider.topics.isEmpty
                      ? const Center(child: Text(Const.LIST_EMPTY))
                      : DropdownMenu<Topic>(
                          width: (MediaQuery.of(context).size.width - 36),
                          menuHeight: 200,
                          controller: _topic,
                          enableFilter: true,
```

```
                label: Text(Const.STUDY_SEARCH_TOPIC_LABLE),
                requestFocusOnTap: true,
                leadingIcon: const Icon(Icons.search),
                inputDecorationTheme: const InputDecorationTheme(
                  filled: false,
                  contentPadding:
                      EdgeInsets.symmetric(vertical: 5.0),
                ),
                onSelected: (Topic? topic) {
                  setState(() {
                    selectedTopic = topic;

                  });
                },
                dropdownMenuEntries:
                    provider.topics.map<DropdownMenuEntry<Topic>>(
                  (Topic topic) {
                    return DropdownMenuEntry<Topic>(
                        value: topic,
                        label: topic.name,
                        );
                  },
                ).toList(),
              );
            },
          ),
        ),
      ),
    ],
  ),
```

```
    );
  } else {
    return Container();
  }
}

Widget getButton(String name){
  return ElevatedButton(
      onPressed: pressButtonLearn,
      child: Text(
        name,
        style: TextStyle(fontSize: 20),
      ));
}

pressButtonLearn() async {
  //logic main

  //logic check must repeat study
  final providerRepeat = Provider.of<StudyProvider>(context, listen: false);
  providerRepeat.getRepeatFilled();
  if(providerRepeat.countFirstRepetition != null &&
      providerRepeat.countSecondRepetition != null &&
      providerRepeat.countThirdRepetition !=null) {
    if(providerRepeat.countFirstRepetition! > 3 ||
        providerRepeat.countSecondRepetition! > 7 ||
        providerRepeat.countThirdRepetition! > 12) {
      mustRepeatWord = true;
    }
  }
```

```
          //random
          if (isLearnRandom) {
            if(!mustRepeatWord){
              AutoRouter.of(context).push(StudyMethodsRoute(typeLearn:
TypeLearn.random));
            }else{
              FlashMessage(
                messageShort: Const.STUDY_MESSAGE_SHORT_MUST_REPEAT,
                messageLong: Const.STUDY_MESSAGE_LONG_MUST_REPEAT,
                colorMessage: Theme.of(context).colorScheme.primaryContainer)
                .getScaffoldMessage(context);
            }
          }
          //
          //topics study
          else if (isLearnByTopics) {
            if(!mustRepeatWord) {
              if (selectedTopic != null) {
                AutoRouter.of(context).push(StudyMethodsRoute(
                  typeLearn: TypeLearn.topic, topicid: selectedTopic!.id_topic));
              } else {
                FlashMessage(
                  messageShort:
Const.STUDY_MESSAGE_SHORT_ERROR_CHOOSE_TOPIC,
                  messageLong:
Const.STUDY_MESSAGE_LONG_ERROR_CHOOSE_TOPIC,
                  colorMessage: Theme
                    .of(context)
                    .colorScheme
                    .errorContainer)
```

```
            .getScaffoldMessage(context);
        }
        selectedTopic = null;
        _topic.clear();
      }else{
        FlashMessage(
            messageShort: Const.STUDY_MESSAGE_SHORT_MUST_REPEAT,
            messageLong: Const.STUDY_MESSAGE_LONG_MUST_REPEAT,
            colorMessage: Theme.of(context).colorScheme.primaryContainer)
            .getScaffoldMessage(context);
      }
    }
    //
    //repeat
    else if(isRepeatOldWord){
      if(mustRepeatWord){
        _logicGetWordsList();
      }else{
        FlashMessage(
            messageShort:
Const.STUDY_MESSAGE_SHORT_NOTHING_REPEAT,
            messageLong:
Const.STUDY_MESSAGE_LONG_NOTHING_REPEAT,
            colorMessage: Theme.of(context).colorScheme.primaryContainer)
            .getScaffoldMessage(context);
      }
    }
    else{
      FlashMessage(
```

```
            messageShort:
Const.STUDY_MESSAGE_SHORT_ERROR_CHOOSE_METHOD_LEARN,
            messageLong:
Const.STUDY_MESSAGE_LONG_ERROR_CHOOSE_METHOD_LEARN,
            colorMessage: Theme.of(context).colorScheme.primaryContainer)
            .getScaffoldMessage(context);
      }
    }

    _logicGetWordsList() async {
      //logic get list
      final providerRepeat = Provider.of<StudyProvider>(context, listen: false);
      if(providerRepeat.countFirstRepetition! > 3){
        //>40
        providerRepeat.getRepeatWordsList(1,0,0,0);
        AutoRouter.of(context).push(StudyRepeatWordsRoute());
        mustRepeatWord = false;
      }else if(providerRepeat.countSecondRepetition! > 7){
        //>120
        providerRepeat.getRepeatWordsList(1,1,0,0);
        AutoRouter.of(context).push(StudyRepeatWordsRoute());
        mustRepeatWord = false;
      }else if(providerRepeat.countThirdRepetition! > 12){
        //>250
        providerRepeat.getRepeatWordsList(1,1,1,0);
        AutoRouter.of(context).push(StudyRepeatWordsRoute());
        mustRepeatWord = false;
      }
    }
```

topic_page.dart - цей файл відповідає за створення сторінки “Теми”

```
import 'package:flutter/material.dart';
import 'package:auto_route/auto_route.dart';
import 'package:flutter_spinkit/flutter_spinkit.dart';
import 'package:provider/provider.dart';
import
'package:studrow/features/topic/presentation/pages/dictionary_topic_list.dart';
import
'package:studrow/features/topic/presentation/provider/topic_provider.dart';
import '../form/form_add_topic.dart';
import 'package:studrow/assets/constants.dart' as Const;

@RoutePage()
class TopicsPage extends StatefulWidget {
 const TopicsPage({super.key});

 @override
 State<TopicsPage> createState() => _TopicsPageState();
}

class _TopicsPageState extends State<TopicsPage> {
 bool isLoading = true;

 @override
 void initState() {
   Future.delayed(Duration(seconds: 1), (){
     setState(() {
       isLoading = false;
     });
   });
```

```
  super.initState();
}
@override
Widget build(BuildContext context) {
  final provider = Provider.of<TopicProvider>(context);
  if (!provider.isLoadingTopicPage && !isLoading) {
    return Scaffold(
      appBar: AppBar(
        scrolledUnderElevation: 0,
        title: Center(
           child: Text(
          Const.TOPICS_APP_BAR_TITLE,
          style: TextStyle(fontSize: 26,
          color: Theme.of(context).colorScheme.primary),
        )),
      ),
      body: TopicsPageBody(),
      floatingActionButton: FloatingActionButton.extended(
        onPressed: () {
          showDialog(
            context: context,
            builder: (context) {
              return Dialog(
                child: TopicFormAdd(),
              );
            },
          );
        },
        tooltip: Const.TOPICS_BOTTOM_BUTTON_TOOLTIP,
        icon: const Icon(Icons.add),
```

```
      label: const Text(
          Const.TOPICS_BOTTOM_BUTTON_TITLE
      ),
    ),
  );
} else {
  return Scaffold(
    appBar: AppBar(
      title: Center(
          child: Text(
            Const.TOPICS_APP_BAR_TITLE,
            style: TextStyle(fontSize: 26,
                color: Theme.of(context).colorScheme.primary),
          )),
    ),
    body: Center(
      child: Column(
        mainAxisSize: MainAxisSize.min,
        children: [
          SpinKitRotatingPlain(
            color: Theme.of(context).colorScheme.secondary,
            size: 80,
          ),
          Text(
            Const.SPINKIT_LOADING,
            style: TextStyle(
                fontSize: 20,
                color: Theme.of(context).colorScheme.secondary),
          ),
        ],
```

```
          ),
        ),
      );
    }
  }
}
```

dictionary_page.dart - цей файл відповідає за створення сторінки “Словник”

```
import 'package:flutter/material.dart';
import 'package:auto_route/auto_route.dart';
import
'package:studrow/features/dictionary/presentation/pages/dictionary_word_list.dart';
import 'package:studrow/router/router.dart';
import 'package:studrow/assets/constants.dart' as Const;

@RoutePage()
class DictionaryPage extends StatefulWidget {
 const DictionaryPage({super.key});

 @override
 State<DictionaryPage> createState() => _DictionaryPageState();
}

class _DictionaryPageState extends State<DictionaryPage> {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    scrolledUnderElevation: 0,
    title: Center(
```

```
        child: Text(
          Const.DICTIONARY_APP_BAR_TITLE,
          style: TextStyle(
              fontSize: 26, color: Theme.of(context).colorScheme.primary),
        ),
      ),
    ),
    body: WordsListPage(),
    floatingActionButton: FloatingActionButton.extended(
      onPressed: () {
        AutoRouter.of(context).push(WordFormAddRoute());
      },
      tooltip: Const.DICTIONARY_BOTTOM_BUTTON_TOOLTIP,
      icon: const Icon(Icons.add),
      label: Text(Const.DICTIONARY_BOTTOM_BUTTON_TITLE),
    ),
  );
 }
}
```