Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки

До захисту допущено
Кафедрою комп'ютерних систем та робототехніки
протокол № __ від __ червня 2026 р.

завідувач кафедри _____ Максим ХРУСЛОВ
(підпис)
«___»_____ 2026 р.

Кваліфікаційна робота

здобувача першого (бакалаврського) рівня вищої освіти

«Комп'ютерна модель навчання жестової мови на основі технології Flutter»

Спеціальність ***151 – Автоматизація та комп'ютерно-інтегровані технології***

Освітня програма ***Автоматизація та комп'ютерно інтегровані технології***

Виконавець  **Кирило КАРПЕНКО**
(підпис)

Науковий керівник  **Олена ТОЛСТОЛУЗЬКА**
(підпис)

Харків – 2026

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи бакалавра складається зі вступу, чотирьох розділів, висновків, списку використаних джерел та чотирьох додатків. Загальний обсяг роботи складає 144 сторінки, із яких 50 сторінок основної частини з 23 рисунками, 14 найменувань у списку використаних джерел та чотирьох додатків.

Метою роботи є підвищення доступності та ефективності навчання української жестової мови для людей з порушеннями слуху шляхом використання комп'ютерної моделі на основі технології Flutter.

Об'єктом дослідження є процес навчання української жестової мови людей з порушеннями слуху із використанням веб-застосунків.

Предметом дослідження є методи, засоби та програмні реалізації комп'ютерних моделей навчання української жестової мови.

Завдання, яке вирішується в кваліфікаційній роботі, полягає у розробці комп'ютерної моделі навчання людей з порушеннями слуху української жестової мови засобами технології Flutter з метою підвищенні доступності та ефективності навчання української жестової мови для людей з порушеннями слуху шляхом використання сучасних цифрових та асистивних технологій, а також забезпечення зручного доступу до візуалізованих навчальних матеріалів і засобів перевірки знань.

Область застосування – інформаційні системи підтримки інклюзивної освіти, цифрові освітні платформи та програмні засоби навчання людей з особливими потребами, а також освітній процес за спеціальністю 151 «Автоматизація та комп'ютерно-інтегровані технології».

Ключові слова: автоматизація та комп'ютерно-інтегровані технології, комп'ютерна модель, програмний застосунок, Flutter, Dart, жестова мова, інклюзивне навчання, люди з порушеннями слуху, цифрові технології, асистивні технології.

ABSTRACT

The explanatory note to the bachelor's qualification work consists of an introduction, four sections, conclusions, a list of sources used and four appendices. The total volume of the work is 144 pages, of which 50 pages are the main part with 23 figures, 14 items in the list of sources used and four appendices.

The purpose of this work is to improve the accessibility and effectiveness of learning Ukrainian Sign Language for people with hearing impairments through the use of a computer model based on Flutter technology.

The object of the study is the process of learning Ukrainian Sign Language by people with hearing impairments using web applications.

The subject of the study is the methods, tools, and software implementations of computer-based models for teaching Ukrainian Sign Language.

The task addressed in this qualification thesis is the development of a computer model for teaching Ukrainian Sign Language to people with hearing impairments using Flutter technology. The purpose of the model is to enhance the accessibility and effectiveness of learning Ukrainian Sign Language through the application of modern digital and assistive technologies, as well as to provide convenient access to visualized educational materials and knowledge assessment tools.

Area of application – information systems supporting inclusive education, digital educational platforms, software tools for teaching people with special needs, as well as the educational process in specialty 151 "Automation and Computer-Integrated Technologies".

Keywords: automation and computer-integrated technologies, computer model, software application, Flutter, Dart, sign language, inclusive education, people with hearing impairments, digital technologies, assistive technologies.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ..... 6

ВСТУП.....7

РОЗДІЛ 1

АНАЛІЗ СУЧАСНИХ ПІДХОДІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ НАВЧАННЯ ЛЮДЕЙ З ОСОБЛИВИМИ ПОТРЕБАМИ.....9

1.1 Особливості навчання людей з інклюзивними потребами.....9

1.2 Роль цифрових технологій у навчанні людей з особливими потребами.....10

1.3 Асистивні технології як інструмент підтримки навчання..... 11

1.4 Проблеми цифрового навчання людей з особливими потребами.....13

1.5 Постановка задачі дослідження..... 15

Висновки до розділу 1.....17

РОЗДІЛ 2

ПРОЄКТУВАННЯ КОМП'ЮТЕРНОЇ МОДЕЛІ ТА АРХІТЕКТУРИ ПРОГРАМНОГО ЗАСТОСУНКУ..... 18

2.1 Аналіз технологій розробки програмного застосунку.....18

2.2 Навчальний контент та організація навчального процесу.....19

2.3 Проектування комп'ютерної моделі навчання української жестової мови..... 23

2.4 Архітектура програмного застосунку відповідно до принципів Clean Architecture.....28

Висновки до розділу 2.....30

РОЗДІЛ 3

ПРОГРАМНА РЕАЛІЗАЦІЯ МОДУЛІВ НАВЧАЛЬНОГО ЗАСТОСУНКУ... 31

3.1 Реалізація навігації застосунку..... 31

3.2 Програмна реалізація модулів теоретичного розділу навчання на

основі Clean Architecture.....32

3.3 Реалізація практичного розділу навчання та перевірки знань..... 37

РОЗДІЛ 4

ПРАКТИЧНІ РЕКОМЕНДАЦІЇ ПО ЗАСТОСУВАННЮ КОМП'ЮТЕРНОЇ МОДЕЛІ.....42

4.1 Демонстрація роботи комп'ютерної моделі.....42

Висновки до розділу 4.....49

ВИСНОВКИ..... 50

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....52

ДОДАТКИ..... 54

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

Clean Architecture	– підхід до проєктування програмного забезпечення;
Dart	– об’єктно-орієнтована мова програмування;
Flutter	– кросплатформний фреймворк з відкритим вихідним кодом;
React Native	– кросплатформний фреймворк для мобільної розробки;
УТОГ	– українське товариство глухих;
UNESCO	– United Nations Educational, Scientific and Cultural Organization
UNICEF	– United Nations Children's Fund.
Асистивні технології	– спеціалізовані програмні та апаратні засоби;
Кросплатформеність	– властивість ПЗ функціонувати на різних пристроях;
Нативна розробка	– підхід до створення програмного забезпечення;
НБУ	– Національна Бібліотека України;
ПЗ	– програмне забезпечення;
УТОГ	– українське товариство глухих;

ВСТУП

У сучасних умовах цифровізації освітнього середовища все більшої актуальності набуває питання розробки програмних засобів, що забезпечують доступність та ефективність навчання для людей з особливими освітніми потребами. Особливо це стосується осіб із порушеннями слуху, для яких основним каналом сприйняття інформації є візуальний, що вимагає спеціалізованих підходів до подання навчального матеріалу.

Актуальність роботи зумовлена недостатнім рівнем розвитку програмних засобів для вивчення української жестової мови, які б поєднували візуалізацію навчального контенту, інтерактивні засоби перевірки знань та можливість самостійного навчання користувача. Наявні рішення часто мають обмежену функціональність, недостатній рівень адаптації до потреб користувачів або складний інтерфейс, що ускладнює процес навчання.

У зв'язку з цим виникає необхідність розробки комп'ютерної моделі навчання, яка забезпечує структуроване подання навчального матеріалу у візуальній формі, підтримує інтерактивну перевірку знань та дозволяє організувати індивідуальний темп навчання користувача.

Метою роботи є підвищення доступності та ефективності навчання української жестової мови для людей з порушеннями слуху шляхом використання комп'ютерної моделі на основі технології Flutter.

Об'єкт дослідження – процес навчання української жестової мови людей з порушеннями слуху з використанням вебзастосунків.

Предмет дослідження – методи, засоби та програмні реалізації комп'ютерних моделей навчання української жестової мови.

Завдання дослідження

1. Виконати аналіз сучасних цифрових та асистивних технологій навчання людей з особливими освітніми потребами з метою визначення їхніх можливостей та обмежень.

2. Виконати аналіз наявних програмних рішень для навчання української жестової мови та визначити їхні основні функціональні недоліки.

3. Розробити комп'ютерну модель навчання української жестової мови з використанням візуалізованого навчального контенту та структурованої подачі матеріалу.

4. Розробити програмний застосунок для реалізації запропонованої моделі навчання засобами технології Flutter.

5. Реалізувати тестові завдання для перевірки знань користувачів.

6. Провести тестування програмного застосунку та оцінити його функціональні можливості, зручність використання й ефективність у навчальному процесі.

РОЗДІЛ 1
АНАЛІЗ СУЧАСНИХ ПІДХОДІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ НАВЧАННЯ ЛЮДЕЙ З ОСОБЛИВИМИ ПОТРЕБАМИ

1.1 Особливості навчання людей з інклюзивними потребами

Освіта для людей з особливими потребами є важливим напрямом сучасної педагогіки, оскільки вона забезпечує рівний доступ до знань незалежно від фізичних чи когнітивних обмежень. До цієї категорії належать особи з порушеннями слуху, зору, мовлення, опорно-рухового апарату, а також з інтелектуальними чи комплексними порушеннями.

Традиційні методи навчання часто не адаптовані до індивідуальних особливостей таких здобувачів освіти, що створює бар'єри для засвоєння ними інформації. Через це врахування специфіки розвитку кожного є критично важливим. Наприклад, для осіб із порушеннями слуху основним каналом сприйняття є зоровий, тому доцільно використовувати візуальні матеріали, як-от зображення, відео, жести та текстові пояснення.

Вагому роль відіграє також темп навчання. Люди з особливими освітніми потребами можуть потребувати більше часу для засвоєння матеріалу, тому необхідно забезпечити можливість навчатися у зручному для них темпі. Це досягається завдяки використанню адаптивних навчальних програм та індивідуальних освітніх траєкторій.

Не менш важливою є потреба у створенні сприятливого психологічного середовища, адже підтримка, відсутність дискримінації та мотивація сприяють якісному засвоєнню знань і підвищують впевненість у власних силах. Важливу роль відіграє підготовка викладачів до роботи в інклюзивному середовищі.

Завдяки сучасним технологіям можливості освітнього процесу можна значно розширити. Використання мобільних застосунків, мультимедійних ресурсів та інтерактивних платформ дозволяє зробити навчання доступнішим, наочнішим і зрозумілішим.

Отже, навчання осіб з особливими потребами має комплексний характер і вимагає поєднання педагогічних підходів, технологічних рішень та індивідуалізації освітнього процесу. Лише за таких умов можна забезпечити ефективне засвоєння знань та повноцінну інтеграцію людей у суспільство.

1.2 Роль цифрових технологій у навчанні людей з особливими потребами

З розвитком інформаційних технологій значно розширилися можливості організації освітнього процесу для людей з особливими освітніми потребами. Цифрові інструменти дозволяють адаптувати контент відповідно до індивідуальних особливостей користувачів, підвищити доступність матеріалів та забезпечити гнучкіший підхід до навчання.

Однією з найбільш поширених технологій є мультимедійні навчальні системи, які поєднують текстову інформацію, зображення, відеоматеріали та анімацію. Застосування такого контенту дозволяє значно покращити сприйняття інформації людьми з порушеннями слуху, оскільки основний акцент робиться на візуальному поданні матеріалу. Особливо ефективними є відеоматеріали з демонстрацією жестів, анімовані ілюстрації та інтерактивні зображення.

Важливу роль у сучасній інклюзивній освіті відіграють мобільні навчальні застосунки. Такі програмні засоби забезпечують постійний доступ до матеріалів незалежно від місця перебування користувача. Крім того, вони дозволяють організувати навчання у зручному темпі, що є важливим для осіб із різними особливостями сприйняття інформації.

Окрему увагу слід приділити технологіям адаптивного навчання. Такі системи здатні змінювати складність матеріалу, обсяг інформації та спосіб її подання відповідно до індивідуальних потреб користувача. Наприклад, для людей із порушеннями слуху важливим є використання графічних матеріалів та жестової мови, тоді як для користувачів із порушеннями зору можуть застосовуватися аудіосупровід та технології озвучення тексту.

Також активно використовуються хмарні технології та вебплатформи для дистанційного навчання. Такі рішення дозволяють забезпечити доступ до навчального контенту через мережу Інтернет, організовувати взаємодію між користувачами та зберігати результати навчальної діяльності.

У контексті глобальної стратегії цифровізації освіти експерти UNESCO [11] наголошують, що впровадження новітніх цифрових технологій є фундаментальним чинником забезпечення рівного доступу до знань, зокрема для осіб з інвалідністю. Згідно з їхніми дослідженнями, використання інтерактивних та мультимедійних форматів суттєво підвищує інклюзивність навчального середовища, забезпечує глибше засвоєння матеріалу та активізує внутрішню мотивацію користувачів. Для людей із порушеннями слуху це відкриває можливості безбар'єрного сприйняття інформації завдяки динамічній візуалізації контенту та адаптивним інтерфейсам, що трансформує традиційні освітні підходи у гнучкі індивідуалізовані траєкторії навчання.

Таким чином, цифрові технології є невід'ємною складовою сучасної інклюзивної освіти, оскільки забезпечують доступність навчання, підтримують індивідуалізацію освітнього процесу та створюють умови для ефективнішого засвоєння знань людьми з особливими освітніми потребами.

1.3 Асистивні технології як інструмент підтримки навчання

Асистивні технології спрямовані на подолання бар'єрів у доступі до інформації та створення рівних умов для навчання незалежно від фізичних чи когнітивних особливостей користувача. У сучасних умовах вони відіграють важливу роль у реалізації принципів інклюзивної освіти, оскільки дозволяють адаптувати навчальний процес відповідно до індивідуальних потреб здобувачів освіти та забезпечують можливість повноцінної участі в навчанні.

Однією з найбільш поширених асистивних технологій є програми екранного доступу. Такі програмні засоби дозволяють перетворювати текстову інформацію на звукову форму, що забезпечує доступ до цифрових ресурсів для

людей із порушеннями зору. Ці інструменти використовуються під час роботи з вебсайтами, електронними підручниками та навчальними платформами.

Також широко застосовуються альтернативні пристрої введення інформації, серед яких – спеціалізовані клавіатури, сенсорні екрани, адаптивні маніпулятори та системи керування поглядом. Вони забезпечують можливість повноцінної роботи з програмним забезпеченням навіть за наявності обмежень моторики.

Для людей із порушеннями слуху особливе значення мають технології візуалізації інформації. До них належать автоматичні субтитри, системи перекладу жестовою мовою, а також мультимедійні навчальні матеріали з візуальним супроводом. Наприклад, платформи YouTube та Microsoft Teams підтримують автоматичне створення субтитрів, що значно покращує доступність навчального контенту для осіб із порушеннями слуху.

Важливу роль у сучасному освітньому середовищі відіграють адаптивні навчальні платформи, які здатні автоматично змінювати спосіб подання матеріалу відповідно до потреб користувача. Такі системи можуть регулювати темп навчання, складність завдань, розмір тексту, колірну схему інтерфейсу та інші параметри. Подібні платформи підтримують інтеграцію мультимедійних та інтерактивних елементів навчання.

Використання асистивних технологій дозволяє не лише компенсувати функціональні обмеження, а й значно підвищує рівень самостійності користувачів у процесі навчання. Завдяки таким рішенням здобувачі освіти отримують можливість працювати з навчальними матеріалами без сторонньої допомоги, що позитивно впливає на їхню впевненість та мотивацію.

Крім того, асистивні технології забезпечують доступ до широкого спектра освітніх ресурсів, зокрема до електронних підручників, онлайн-курсів та інтерактивних платформ. Це створює умови для безперервного навчання та саморозвитку незалежно від місця перебування користувача.

За офіційними даними Всесвітньої організації охорони здоров'я [12], асистивні технології є фундаментальним інструментом для підтримки та покращення функціональних можливостей людини в таких сферах, як слух, комунікація, когніція та мобільність. Експерти організації наголошують, що впровадження цих засобів безпосередньо забезпечує інклюзію та залученість людей з інвалідністю в освітній процес, ринок праці та суспільне життя. Доступ до якісних асистивних продуктів дозволяє користувачам вести здорове, продуктивне й незалежне життя, що не лише мінімізує потребу у сторонньому догляді, а й позитивно впливає на загальний економічний розвиток і зниження соціальної нерівності.

Таким чином, асистивні технології є важливим інструментом підтримки навчання людей з особливими освітніми потребами, оскільки забезпечують доступність, адаптивність та ефективність освітнього процесу, сприяючи формуванню інклюзивного освітнього середовища.

1.4 Проблеми цифрового навчання людей з особливими потребами

навчання людей з особливими потребами все ще супроводжується низкою проблем. Більшість наявних цифрових рішень не повною мірою враховують індивідуальні особливості користувачів, що негативно впливає на ефективність освітнього процесу та доступність матеріалів.

Однією з ключових проблем є спосіб подання навчальної інформації. Значна частина освітніх платформ орієнтована переважно на текстове або аудіальне викладення матеріалу, що створює труднощі для людей із порушеннями слуху. Для цієї категорії користувачів основним є зоровий канал сприйняття, тому недостатня кількість графічних матеріалів, відеодемонстрацій та перекладу жестовою мовою ускладнює засвоєння інформації.

Важливою проблемою також є недостатній рівень візуалізації контенту. У багатьох цифрових системах матеріали подаються у вигляді великих текстових блоків без належної кількості ілюстрацій, схем або інтерактивних елементів. Це

негативно впливає на сприйняття інформації людьми з особливими освітніми потребами, особливо коли навчання пов'язане з візуальним запам'ятовуванням жестів, символів або зображень.

Ще однією проблемою є обмежена кількість інтерактивних навчальних матеріалів. Частина цифрових освітніх ресурсів орієнтована лише на пасивне сприйняття інформації без можливості перевірки знань, виконання вправ або взаємодії з контентом. Відсутність інтерактивності знижує залученість користувача та ускладнює закріплення отриманих знань.

Суттєвим недоліком багатьох платформ є недостатня підтримка індивідуального темпу навчання. Користувачі з особливими потребами можуть потребувати більше часу для опрацювання матеріалу або повторного перегляду інформації. Проте частина цифрових систем не дозволяє гнучко адаптувати процес навчання відповідно до потреб конкретного користувача.

Важлива також доступність ресурсів у цифровому середовищі. Не всі користувачі мають доступ до сучасних технічних засобів, стабільного інтернету або спеціалізованого софту. Крім того, деякі цифрові сервіси мають складний інтерфейс або недостатньо адаптовані для людей із порушеннями слуху, зору чи моторики.

Вагомою проблемою залишається слабка адаптація наявних цифрових рішень до специфіки сприйняття людей із порушеннями слуху. Більшість навчальних платформ не підтримують жестову мову як основний спосіб подання інформації, що обмежує ефективність навчання для цієї категорії користувачів.

Згідно зі Стратегією цифрової освіти UNICEF [10], однією з головних проблем цифровізації є неефективне використання технологій через механічне переведення традиційних матеріалів у цифровий формат без урахування й усунення глибинних бар'єрів навчання. Аналітики наголошують, що помилкова ставка лише на масову дистрибуцію пристроїв не здатна покращити освітні

результати, що призводить до неефективного розвіювання мільярдних глобальних інвестицій в освітні технології. Для людей із порушеннями слуху така фрагментарність і відсутність людиноцентрованого підходу створюють перешкоди, оскільки розробники рідко залучають профільних фахівців до проєктування інтерфейсів, що зумовлює дефіцит адаптованого візуального контенту та унеможливлює самостійне сприйняття інформації.

Отже, виявлені проблеми цифрового навчання свідчать про необхідність створення спеціалізованих програмних рішень, орієнтованих на візуалізацію навчального матеріалу, інтерактивність, підтримку індивідуального темпу навчання та забезпечення доступності освітніх ресурсів для людей з особливими потребами.

1.5 Постановка задачі дослідження

Аналіз особливостей навчання людей із порушеннями слуху, а також сучасних цифрових підходів дає змогу виділити основні аспекти, що впливають на ефективність засвоєння матеріалу:

1. спосіб подання навчальної інформації;
2. рівень візуалізації навчального контенту;
3. наявність інтерактивних навчальних матеріалів;
4. забезпечення індивідуального темпу навчання;
5. доступність навчальних ресурсів у цифровому середовищі.

Однією з ключових проблем є форма подання інформації. Для людей із порушеннями слуху основним каналом сприйняття є зоровий, тому навчальні матеріали мають бути орієнтовані на використання зображень, відео та жестової мови. Традиційні текстові або аудіальні методи навчання є малоефективними для цієї категорії користувачів.

Для підвищення ефективності освітнього процесу необхідно розробити програмний застосунок зі зручним та інтуїтивно зрозумілим інтерфейсом, який забезпечуватиме подання матеріалу у візуальній формі. Такий застосунок має

поєднувати текстову інформацію з відповідними зображеннями жестів, що сприятиме кращому запам'ятовуванню та розумінню.

Першим завданням під час реалізації програмного продукту є розробка функцій для відображення навчального контенту у вигляді зображень та їхніх текстових відповідників. Це дозволить користувачеві встановлювати зв'язок між словом та відповідним жестом.

Другим важливим завданням є реалізація інтерактивних вправ для перевірки знань. Необхідно забезпечити можливість проходження тестових завдань, у яких користувач має обрати правильну відповідь серед запропонованих варіантів. Це сприятиме закріпленню вивченого матеріалу та підвищенню рівня залученості.

Окрім цього, важливим є забезпечення індивідуального темпу навчання. Користувач повинен мати можливість самостійно обирати швидкість опрацювання матеріалу та кількість завдань, що дозволить адаптувати навчальний процес відповідно до його потреб.

Ще одним завданням є організація доступу до додаткових матеріалів, зокрема відеоуроків та корисних ресурсів. Це розширить можливості навчання та забезпечить користувача необхідною інформацією для самостійної роботи.

Також потрібно забезпечити простоту використання застосунку. Інтерфейс має бути зрозумілим для непідготовленого користувача, не перевантаженим зайвими елементами та відповідати принципам інклюзивності.

Отже, для досягнення мети дослідження необхідно реалізувати програмний застосунок, який забезпечить візуалізоване подання навчального матеріалу, інтерактивну перевірку знань, індивідуалізацію процесу та доступ до додаткових освітніх ресурсів.

Висновки до розділу 1

1. Здійснено аналіз особливостей навчання людей з особливими освітніми потребами та визначено основні чинники, що впливають на ефективність засвоєння матеріалу. Встановлено, що для осіб із порушеннями слуху ключову роль відіграє візуальний спосіб подання інформації.

2. Проаналізовано роль цифрових технологій у сфері інклюзивної освіти. Встановлено, що використання мультимедійних систем, мобільних застосунків та інтерактивних платформ дозволяє підвищити доступність навчання, забезпечити індивідуалізацію освітнього процесу та покращити якість засвоєння знань.

3. Досліджено сучасні асистивні технології, зокрема програми екранного доступу, системи розпізнавання мовлення, інструменти субтитрування та адаптивні навчальні платформи. Визначено, що використання таких рішень сприяє підвищенню рівня самостійності користувачів та забезпечує рівні можливості доступу до освітніх ресурсів.

4. Виявлено основні проблеми цифрового навчання людей з особливими потребами, серед яких — недостатній рівень візуалізації контенту, обмежена кількість інтерактивних матеріалів, слабка адаптація цифрових платформ до специфіки сприйняття користувачів із порушеннями слуху та обмежена підтримка індивідуального темпу навчання.

5. Обґрунтовано доцільність розробки програмного застосунку для вивчення української жестової мови, який забезпечуватиме візуалізоване подання навчального матеріалу, інтерактивну перевірку знань, доступ до додаткових ресурсів та підтримку індивідуального темпу навчання.

6. Сформульовано основні задачі дослідження, які полягають у розробці концептуальної моделі навчання жестової мови, реалізації інтерактивного програмного застосунку засобами Flutter та забезпеченні доступності освітнього процесу для людей з особливими потребами.

РОЗДІЛ 2
ПРОЄКТУВАННЯ КОМП'ЮТЕРНОЇ МОДЕЛІ ТА АРХІТЕКТУРИ ПРОГРАМНОГО ЗАСТОСУНКУ

2.1 Аналіз технологій розробки програмного застосунку

Для реалізації програмного застосунку з вивчення української жестової мови було обрано фреймворк Flutter. Цю технологію визначено пріоритетною завдяки її сучасним можливостям у сфері розробки інтерфейсів, високій продуктивності та підтримці створення кросплатформних застосунків.

Однією з ключових переваг використання Flutter є можливість написання єдиного програмного коду для різних платформ, що дозволяє створити застосунок, здатний функціонувати на різних операційних системах без необхідності окремої реалізації для кожної з них. Такий підхід значно скорочує час розробки, спрощує підтримку програмного продукту та забезпечує однакову поведінку інтерфейсу незалежно від платформи.

Кросплатформність є особливо важливою в контексті розробки освітніх застосунків для людей з особливими потребами. Використання фреймворку Flutter дозволяє забезпечити повноцінну підтримку ключових операційних систем, зокрема мобільних (iOS, Android) і настільних (Windows, macOS, Linux) платформ, а також вебсередовища. Такий підхід сприяє суттєвому зменшенню технологічних бар'єрів у доступі до знань та значно підвищує гнучкість інклюзивного навчання.

Мова програмування Dart забезпечує високу швидкість проєктування завдяки функції миттєвого відображення змін у користувацькому інтерфейсі після коригування коду (Hot Reload). Це значно пришвидшує процес розробки візуальної частини та тестування застосунку. Крім того, Dart гарантує високу продуктивність виконання, що є критично важливим для плавної роботи інтерфейсу.

Ще однією вагомою перевагою Flutter є використання власного рушія візуалізації. Це дозволяє створювати графічно узгоджений та передбачуваний інтерфейс, що є вкрай важливим для користувачів з особливими освітніми потребами, зокрема для осіб із порушеннями слуху, які значною мірою покладаються на візуальне сприйняття інформації.

У порівнянні з альтернативними технологіями, такими як React Native або нативна розробка для окремих платформ, Flutter забезпечує стабільнішу роботу інтерфейсу та меншу залежність від сторонніх компонентів. Нативний підхід потребує створення окремих програмних продуктів для кожної операційної системи, тому використання Flutter дозволяє суттєво заощадити час та ресурси.

Таким чином, вибір Flutter як основної технології розробки є повністю обґрунтованим, оскільки він гарантує кросплатформність, високу продуктивність, зручність проєктування та можливість створення доступного інтерфейсу, що є визначальними чинниками при побудові освітнього застосунку для людей з особливими потребами.

2.2 Навчальний контент та організація навчального процесу

Ефективність програмного застосунку для вивчення української жестової мови значною мірою залежить від якості та структури навчального матеріалу. Важливо звернути особливу увагу на формування контенту таким чином, щоб забезпечити зрозуміле, послідовне та доступне засвоєння нових знань користувачами з порушеннями слуху.

Основу навчального матеріалу складають зображення жестів української жестової мови та їхні текстові відповідники, приклади яких наведено на рисунках 2.1 та 2.2.



Рисунок 2.1 – Навчальне зображення літери “А”.

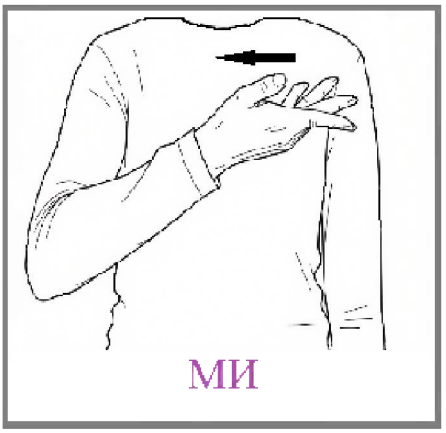


Рисунок 2.2 – Навчальне зображення займенника “Ми”.

Такий підхід дозволяє формувати асоціативний зв’язок між словом і відповідним жестом, що сприяє швидшому запам’ятовуванню навчального матеріалу. Для людей із порушеннями слуху візуальний спосіб подання інформації є найбільш ефективним, тому використання графічних матеріалів є ключовим елементом освітнього процесу.

Структура навчального контенту побудована за тематичним принципом. Весь матеріал розподілено на окремі категорії, кожна з яких відповідає певному етапу вивчення жестової мови:

- абетка;
- цифри:
  - від 0 до 20;
  - десятки;

- сотні;
- тисячі;
- мільйони;

- слова та відповідні символи:
  - займенники;
  - іменники;
  - дієслова;
  - прикметники;
  - прислівники;
- базові речення:
  - “Як вас звати?”;
  - “Мене звати ...”;
  - “Скільки вам років?”;
  - “Мені ... років”;
  - “Як справи?”;
- корисні матеріали:
  - офіційний сайт українського товариства глухих (УТОГ);
  - уроки жестової мови від Національної Бібліотеки України (НБУ) ім. Ярослава Мудрого (для початківців);
  - веб-сайт з функцією пошуку потрібного слова або речення;
  - слова за темою "Природа і природні явища" жестовою мовою від УТОГ;
  - слова за темою "Календар" жестовою мовою від УТОГ;
  - слова за темою "Фрукти й ягоди" жестовою мовою від УТОГ;
  - слова за темою "Овочі" жестовою мовою від УТОГ';
  - слова за темою "Соціум" жестовою мовою від УТОГ;
  - слова за темою "Родина" жестовою мовою від УТОГ;
  - слова за темою "Питання" жестовою мовою від УТОГ;
  - слова за темою "Займенники" жестовою мовою від УТОГ.

Для кожної категорії передбачено окремий екран навчання. Користувач має можливість послідовно переглядати матеріал у власному темпі без часових обмежень. Такий тематичний розподіл контенту спирається на методики поступового ускладнення матеріалу (від простого до складного) та враховує комунікативні потреби початківців. Кожен із представлених розділів виконує конкретну функцію у процесі засвоєння української жестової мови:

1. Розділ «Абетка» є фундаментальною базою, з якої традиційно розпочинається вивчення будь-якої мови. У жестовій мові знання дактилю є критично важливим, оскільки за допомогою літер передаються власні назви (імена, географічні назви), абревіатури, а також слова, для яких ще не сформовано або невідомо спеціального жесту.

2. Розділ «Цифри» є базовою необхідністю для побутового спілкування (називання віку, часу, цін у магазинах, дат). Поділ цього розділу на мікрогрупи (від 0 до 20, десятки, сотні, тисячі, мільйони) реалізовано з метою уникнення когнітивного перевантаження.

3. Розділ «Слова та відповідні символи» було розбито на групи за морфологічним принципом, що допомагає користувачеві структурувати словниковий запас.

4. Розділ «Базові речення» слугує містком між ізольованим вивченням окремих жестів та реальною комунікацією. Вивчення найуживаніших фраз знайомства та повсякденного спілкування дозволяє користувачу отримати швидкий практичний результат.

5. Окремої уваги заслуговує розділ «Корисні матеріали», який реалізовано у вигляді каталогу посилань на перевірені зовнішні веб-ресурси. Включення цього компонента до структури застосунку зумовлено тим, що статичних зображень не завжди достатньо для повного розуміння динаміки жестів та культурного контексту. Інтегровані інтернет-джерела органічно доповнюють базовий курс програми та надають користувачу додаткову інформацію.

Отже, структура навчального контенту програмного застосунку забезпечує послідовне засвоєння української жестової мови та підтримує самостійне навчання користувача.

2.3 Проектування комп'ютерної моделі навчання української жестової мови

Для забезпечення ефективного навчання людей з порушеннями слуху розроблена комп'ютерна модель повинна враховувати особливості сприйняття інформації користувачами та забезпечувати реалізацію основних функцій навчального процесу. Основною метою моделі є створення доступного програмного середовища для вивчення української жестової мови із використанням візуальних навчальних матеріалів та інтерактивних засобів контролю знань.

Комп'ютерна модель навчання української жестової мови має вирішувати такі завдання:

1. забезпечення відображення навчального матеріалу у вигляді зображень жестів та їхніх текстових відповідників;
2. організація інтерактивної перевірки знань за допомогою тестових завдань;
3. забезпечення індивідуального темпу навчання користувача;
4. надання доступу до додаткових освітніх ресурсів для самостійного опрацювання матеріалу;
5. забезпечення простого та зрозумілого користувацького інтерфейсу;
6. підтримка роботи застосунку на різних програмно-апаратних платформах.

З урахуванням зазначених вимог було спроєктовано функціональну модель системи навчання української жестової мови в нотації IDEF0, яка відображає основні інформаційні потоки, механізми функціонування та результати роботи програмного застосунку (рисунки 2.3–2.5).

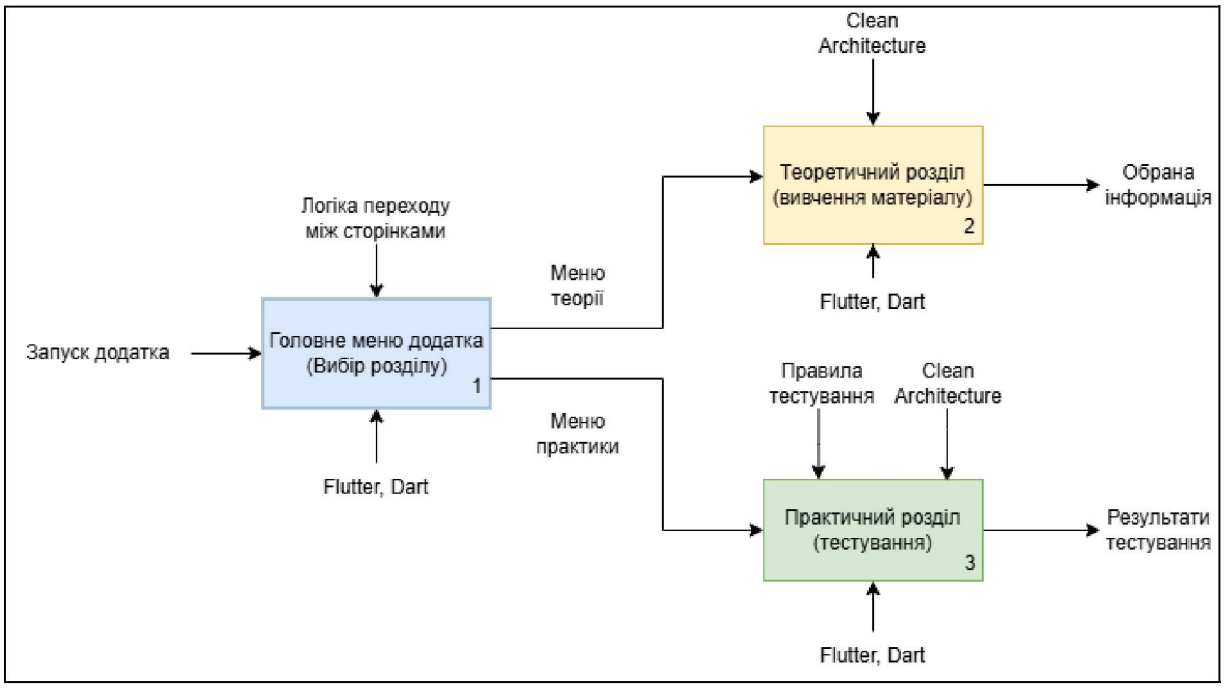


Рисунок 2.3 – Контекстна діаграма IDEF0.

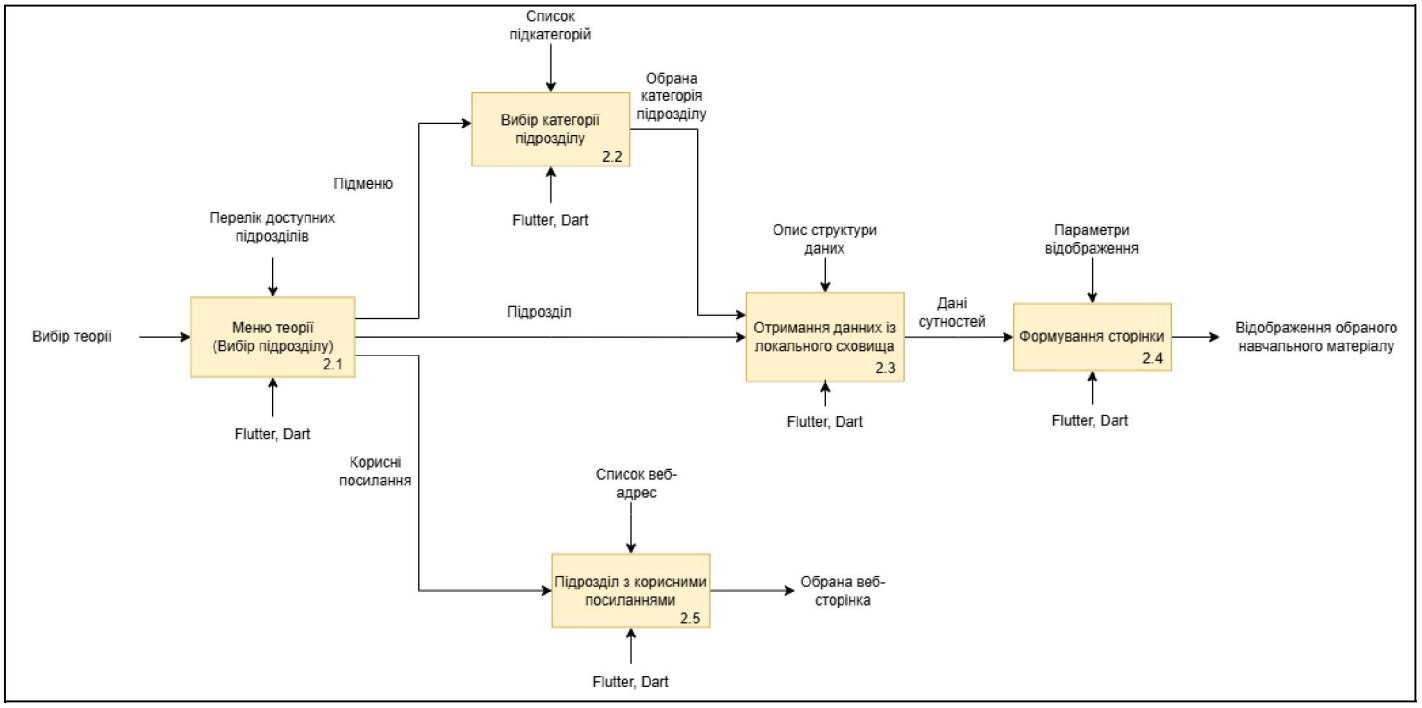


Рисунок 2.4 – Декомпозиція теоретичного розділу.

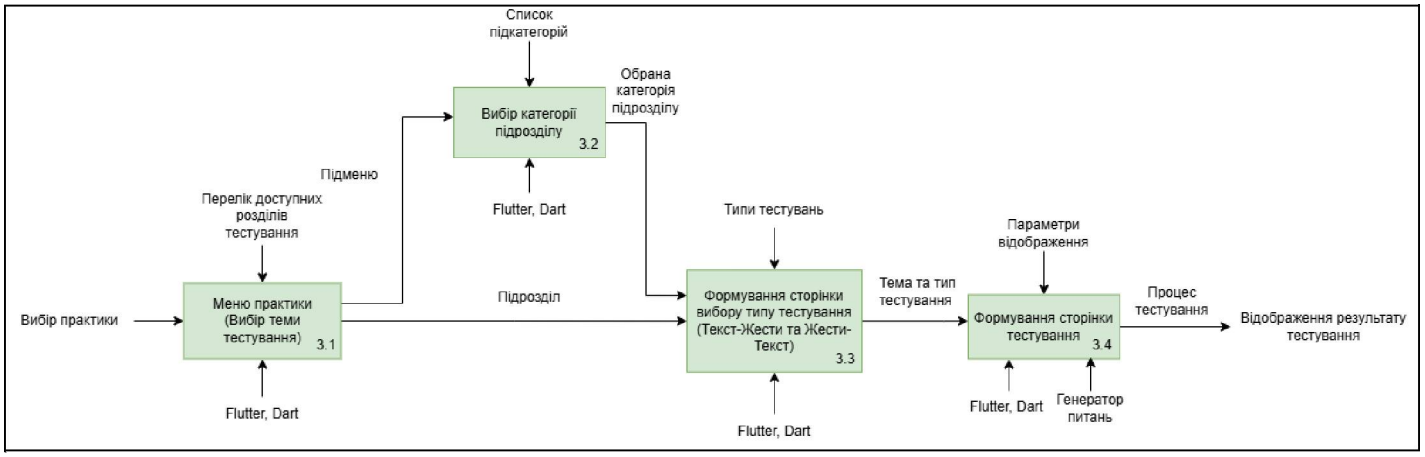


Рисунок 2.4 – Декомпозиція практичного розділу.

Основою програмної реалізації моделі є фреймворк Flutter та мова програмування Dart. Використання Flutter дозволяє створити кросплатформний застосунок із єдиною кодовою базою для мобільних та настільних платформ, що спрощує супровід програмного продукту та забезпечує однаковий користувацький досвід незалежно від типу пристрою.

Архітектура програмного забезпечення побудована відповідно до принципів Clean Architecture. Такий підхід забезпечує розділення бізнес-логіки, користувацького інтерфейсу та рівня доступу до даних, що підвищує

масштабованість, зручність супроводу та можливість подальшого розширення функціоналу застосунку.

Комп'ютерна модель складається з двох основних функціональних модулів:

- теоретичний модуль;
- практичний модуль.

Теоретичний модуль забезпечує структуроване відображення навчального матеріалу у вигляді набору жестів із текстовими відповідниками для різних тем. Користувач має можливість вибирати потрібну тему та переглядати навчальні картки, ознайомлюватися із новими жестами та повторювати раніше вивчений матеріал. Також до цього модуля відноситься розділ з додатковими освітніми ресурсами, його основною функцією є надання доступу до допоміжних матеріалів, які можуть бути використані для поглибленого вивчення української жестової мови.

Для підвищення ефективності навчання до складу моделі включено практичний модуль. Його призначенням є перевірка рівня засвоєння матеріалу після проходження теоретичного курсу у вигляді тестування, яке має два можливі типи:

- відображається зображення жесту та варіанти відповіді у текстовому вигляді
- відображається текстове питання та варіанти відповіді як зображення жесту

Після вибору типу тестування користувачу відображається згенероване завдання, де необхідно обрати правильну відповідь серед запропонованих варіантів, яка підсвічується зеленим кольором, а невірна – червоним. Результати тестування дозволяють оцінити рівень знань користувача.

З погляду інформаційної взаємодії користувач є центральним елементом моделі. Саме він ініціює процес навчання, отримує доступ до навчальних матеріалів, проходить тестування та взаємодіє з додатковими ресурсами. Результатом функціонування системи є формування та закріплення знань української жестової мови.

При проєктуванні моделі особливу увагу приділено забезпеченню простоти використання та доступності інтерфейсу, який повинен бути інтуїтивно зрозумілим, містити мінімальну кількість зайвих елементів та забезпечувати швидкий доступ до основних функцій системи. Це дозволяє зменшити когнітивне навантаження на користувача та підвищити ефективність взаємодії із програмним продуктом.

Функціонування розробленої моделі можна представити у вигляді послідовності таких етапів:

1. Запуск програмного застосунку.

2. Вибір навчального розділу.

3. Перегляд навчальних матеріалів.

4. Засвоєння відповідностей між жестами та словами.

5. Виконання тестових завдань.

6. Отримання результатів тестування.

7. Перехід до додаткових навчальних ресурсів.

8. Повторне проходження матеріалу за необхідності.

Таким чином, розроблена комп’ютерна модель забезпечує комплексний підхід до навчання української жестової мови та поєднує візуалізоване подання навчального контенту, інтерактивну перевірку знань і доступ до додаткових освітніх ресурсів. Це створює умови для ефективного навчання людей з порушеннями слуху та відповідає основним принципам інклюзивної освіти.

2.4 Архітектура програмного застосунку відповідно до принципів Clean Architecture

Для забезпечення масштабованості, підтримуваності та розширюваності програмного застосунку було обрано архітектурний підхід Clean Architecture. Цей підхід дозволяє розділити програмну систему на незалежні рівні, кожен з яких виконує окремі функції та має чітко визначену відповідальність.

Основною перевагою використання Clean Architecture є незалежність бізнес-логіки від інтерфейсу користувача, джерел даних та зовнішніх бібліотек. Завдяки цьому можлива подальша модифікація окремих компонентів системи без внесення суттєвих змін до інших частин програмного забезпечення.

Архітектура розробленого застосунку складається з трьох основних рівнів:

- Presentation Layer — рівень представлення;
- Domain Layer — рівень бізнес-логіки;
- Data Layer — рівень роботи з даними.

Структуру архітектури програмного застосунку наведено на рисунку 2.6.

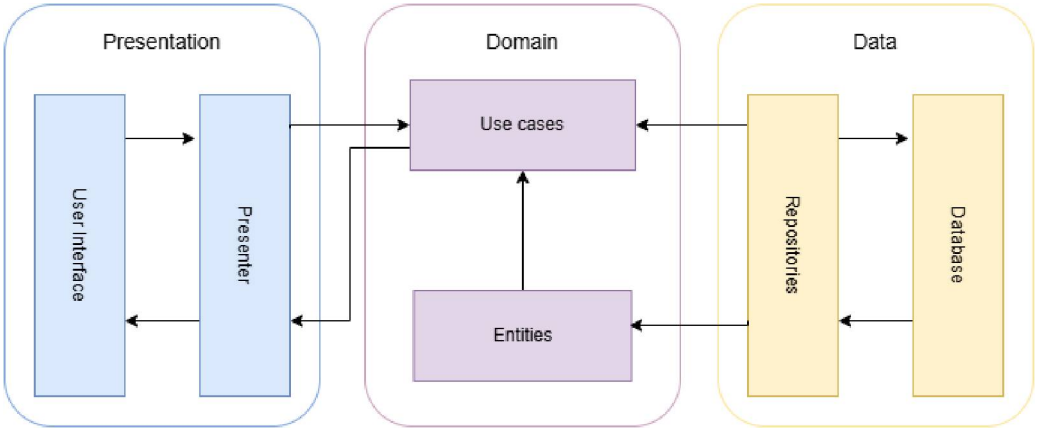


Рисунок 2.6 – Архітектура програмного застосунку відповідно до принципів Clean Architecture.

Рівень Presentation відповідає за взаємодію користувача із застосунком. До його складу входять елементи користувацького інтерфейсу (User Interface) та

компонент Presenter, який забезпечує обробку дій користувача, підготовку даних для відображення та передачу запитів до бізнес-логіки застосунку. Саме цей рівень реалізує екрани теоретичного навчання, тестування та навігацію між розділами програми.

Рівень Domain є центральною частиною архітектури та містить основну логіку роботи застосунку. У ньому розташовані Entities та Use Cases. Сутності (Entities) описують основні об'єкти предметної області, зокрема навчальні матеріали, категорії жестів та тестові завдання. Компоненти Use Cases реалізують сценарії використання системи, такі як отримання навчального контенту, завантаження категорій жестів, формування тестових завдань та перевірка відповідей користувача.

Рівень Data забезпечує роботу з джерелами даних. До його складу входять Repositories та Database. Репозиторії виступають посередниками між бізнес-логікою та джерелами даних, приховуючи особливості їхньої реалізації. База даних використовується для збереження навчальних матеріалів, інформації про жести та тестових завдань. Такий підхід дозволяє за необхідності змінювати спосіб зберігання даних без внесення змін до бізнес-логіки застосунку.

Взаємодія між рівнями здійснюється відповідно до принципу залежності від внутрішніх компонентів архітектури. Користувацький інтерфейс передає запити до рівня Domain через Presenter, після чого відповідні Use Cases звертаються до репозиторіїв для отримання або збереження необхідних даних. Отримана інформація повертається у зворотному напрямку та відображається користувачеві у зручній візуальній формі.

Використання такої архітектури має такі переваги:

- розділення відповідальності між компонентами системи;
- спрощення тестування окремих модулів;
- незалежність бізнес-логіки від інтерфейсу користувача;
- можливість подальшого розширення функціоналу;

- спрощення супроводу програмного забезпечення.

Таким чином, застосування принципів Clean Architecture дозволить побудувати гнучку структуру програмного застосунку для вивчення української жестової мови, що забезпечує зручність подальшого розвитку та підтримки системи.

Висновки до розділу 2

1. Показано, що вибір кросплатформного фреймворку Flutter та мови програмування Dart є оптимальним технологічним рішенням для реалізації застосунку, оскільки вони забезпечують високу продуктивність, створення єдиної кодової бази для різних операційних систем та стабільну роботу візуального інтерфейсу, що є критично важливим чинником для людей з порушеннями слуху.

2. Встановлено ефективність побудови навчального контенту за тематичним і морфологічним принципами (поділ на абетку, числа, частини мови та базові речення), що дозволяє реалізувати методику поступового ускладнення матеріалу, запобігає когнітивному перевантаженню та сприяє швидкому формуванню асоціативних зв'язків між словом і жестом.

3. Показано функціональну структуру розробленої комп'ютерної моделі за допомогою методології IDEF0, яка чітко визначає інформаційні потоки, послідовність етапів навчання та механізми взаємодії користувача як із теоретичним модулем, так і з практичним.

4. Встановлено, що проєктування програмного застосунку відповідно до принципів Clean Architecture з жорстким розподілом системи на три незалежні рівні — представлення (Presentation), бізнес-логіки (Domain) та роботи з даними (Data) — гарантує високу масштабованість продукту, зручність його супроводу та незалежність внутрішньої логіки від зовнішніх інтерфейсів і баз даних.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ МОДУЛІВ НАВЧАЛЬНОГО ЗАСТОСУНКУ

3.1 Реалізація навігації застосунку

Одним із ключових компонентів програмного застосунку є система навігації, яка забезпечує взаємодію користувача з основними функціональними модулями. Оскільки застосунок призначений для людей з порушеннями слуху, під час проєктування інтерфейсу особлива увага приділялася простоті використання, зрозумілій структурі меню та мінімізації кількості дій, необхідних для доступу до навчальних матеріалів.

Структура навігації реалізована засобами фреймворку Flutter із використанням механізму маршрутів (Routes). Такий підхід дозволяє забезпечити швидкий перехід між екранами та спрощує подальше розширення функціональних можливостей програмного продукту.

Для реалізації навігації у мобільному застосунку використовується вбудований механізм Navigator, який забезпечує керування стеком екранів та дозволяє здійснювати переходи між ними.

Навігаційна структура застосунку включає 6 основних екранів меню:

- MainMenuScreen;
- NumbersMenuScreen;
- PracticeMenuScreen;
- SentencesMenuScreen;
- TheoryMenuScreen;
- WordsMenuScreen.

Також було розроблено віджет MenuButton, який слугує загальним шаблоном для кнопок переходу. Винесення кнопки в окремий віджет забезпечує повторне використання коду, покращує його читабельність та спрощує подальшу підтримку й модифікацію інтерфейсу.

Структура навігаційних екранів однакова, оскільки вони слугують для систематизації та упорядкування навчального матеріалу. Кожен з них має певну описану кількість кнопок, що забезпечують перехід між основними розділами програмного застосунку, які реалізовані за допомогою віджета MenuButton, якому при створенні надаються такі параметри:

- title – текстова мітка, яка візуалізується користувачеві для розуміння призначення кнопки;
- onPressed – зворотній виклик для навігації на цільовий екран.

У лістингах Г.1-Г.7 наведено програмний код описаних вище екранів та віджета.

Результатом навігації стала побудова зрозумілої структури програмного застосунку, яка забезпечує швидкий доступ до навчального контенту, спрощує взаємодію користувача із системою та створює основу для подальшого розширення функціональних можливостей програмного продукту.

3.2 Програмна реалізація модулів теоретичного розділу навчання на основі Clean Architecture

Основним призначенням програмного застосунку є забезпечення користувача навчальними матеріалами для вивчення української жестової мови. Для спрощення структури програмного продукту всі навчальні матеріали організовані у вигляді окремих функціональних модулів, які мають схожий принцип роботи та відрізняються лише змістом навчального контенту. До них належать:

- абетка (alphabet);
- числа (numbers);
- слова:
    - прислівники (adverbs);
    - прикметники (adjective);
    - іменники (nouns);

- займенники (pronouns);
- дієслова (verbs);

- речення (sentences);
- корисні матеріали (resources).

Ці модулі можна розбити на три групи за принципом роботи:

1. Без додаткових параметрів при створенні екрана: абетка (alphabet), слова (прислівники (adverbs), прикметники (adjectives), іменники (nouns), займенники (pronouns), дієслова (verbs)).
2. З параметрами створення екрана (список обраного матеріалу (category) та назва (title)): числа (numbers), речення (sentences).
3. Модуль із функцією відкриття посилань: корисні матеріали (resources).

Розглянемо першу групу таких модулів на прикладі розділу «Абетка», що призначений для вивчення жестових позначень літер українського алфавіту. На рисунку 3.1 наведено структуру модуля alphabet.

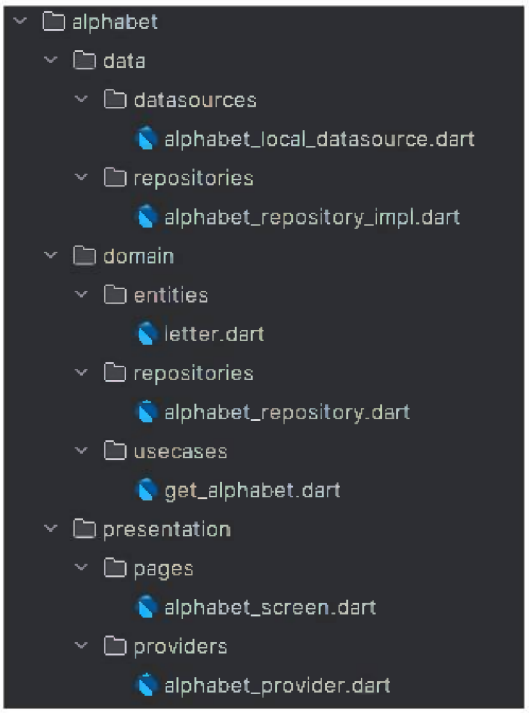


Рисунок 3.1 – Структура модуля alphabet.

З рисунку 3.1 видно, що архітектура цього модуля побудована за принципами Clean Architecture, що забезпечує чіткий розподіл відповідальності,

масштабованість та зручність у тестуванні. Структурно модуль alphabet розділений на три основні директорії:

1. У шарі Domain визначено базову сутність Letter (ідентифікатор id та шлях до зображення imagePath), абстрактний інтерфейс AlphabetRepository та варіант використання GetAlphabet.

2. Шар Data містить клас AlphabetLocalDataSourceImpl, який програмно генерує константний масив із 33 об'єктів Letter із динамічною підстановкою індексу для локальних ресурсів, а AlphabetRepositoryImpl транслює ці дані вище.

3. На рівні Presentation засіб управління станом alphabetProvider (пакет Riverpod) ініціалізує залежності й передає масив в AlphabetScreen. Візуалізація реалізована через конструктор GridView.builder (по 4 елементи в рядку) з підтримкою відкладеного завантаження (lazy loading), що мінімізує навантаження на оперативну пам'ять

Детальніше з цим модулем можна ознайомитися за допомоги лістингів Г.8-Г.14.

Далі розглянемо другу групу модулів, яка має вхідні параметри при створенні екрана, на прикладі розділу «Базові речення», що призначений для вивчення простих речень жестовою мовою. На рисунку 3.2 наведено структуру модуля sentences.

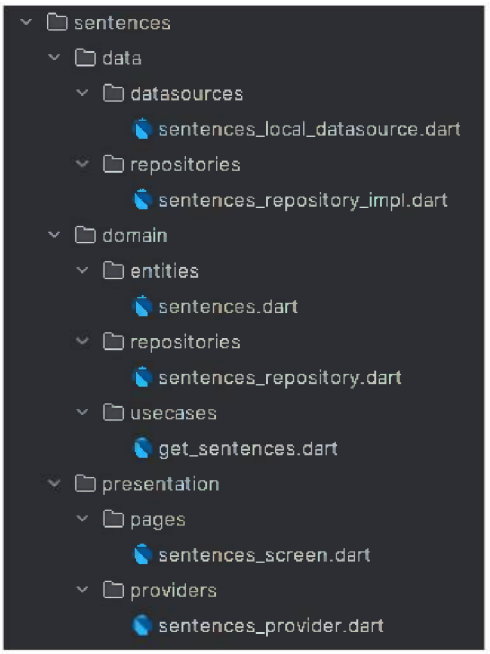


Рисунок 3.2 – Структура модуля sentences.

З рисунку 3.2 видно, що архітектура цього модуля відповідає попередньому alphabet, бо обидва побудовані за принципами Clean Architecture. Різниця між ними у тому, що sentence має виводити інформацію лише про одне речення, яке обрав користувач, а не формувати загальний список елементів.

Ця особливість диктує низку відмінностей у реалізації кожного з архітектурних шарів:

1. У шарі Domain сутність SentenceItem містить масив посилань на серію послідовних рухів (List<String> images) та текстовий опис-підказку description. Варіант використання GetSentence приймає унікальний ідентифікатор String key, щоб повернути лише необхідний об'єкт.

2. У шарі Data метод getSentence локального джерела даних здійснює вибірку об'єкта за допомогою вбудованої функції Dart firstWhere, що повертає конкретний елемент, поле key якого збігається із запитаним параметром.

3. У шарі Presentation реактивне керування станом реалізовано через модифікатор Provider.family, який дозволяє передавати параметри з інтерфейсу користувача безпосередньо в логіку провайдера (sentenceProvider). Екран SentenceScreen будується на основі вертикального ListView для забезпечення

послідовного читання жестів зверху вниз. Під масивом зображень розташовується текстовий блок опису (BoxDecoration).

Детальніше з цим модулем можна ознайомитися за допомоги лістингів Г.15-Г.21.

Остання, третя група модулів містить лише розділ «Корисні матеріали». Структурно він не відрізняється від попередніх, бо також побудований за принципами Clean Architecture, але на відміну від попередніх груп, тут на екран виводиться не заздалегідь підготовлена інформація, а кнопки (віджет ChooseButton), що розроблені спеціально для цього розділу. При натисканні на них відкривається посилання на вебсайт, що зазначено в їхніх назвах. На рисунку 3.3 наведено структуру модуля resources.

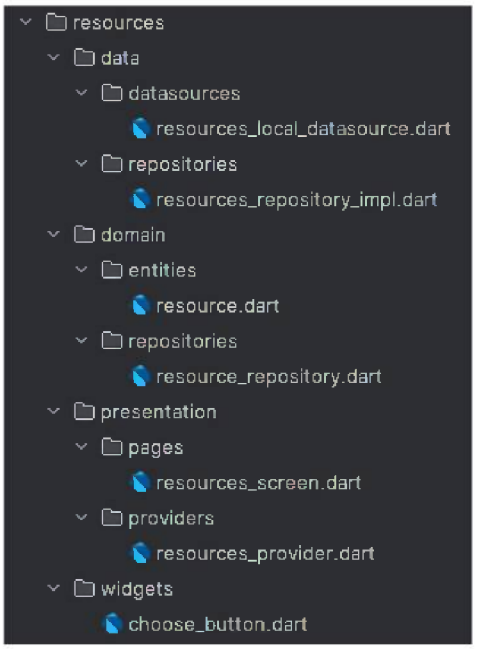


Рисунок 3.3 – Структура модуля resources.

Розберемо шари модуля resources детальніше:

1. Шар Domain містить спрощену модель Resource (title та url). На відміну від інших модулів, тут відсутній рівень варіантів використання (usecases), а взаємодія з джерелом інформації відбувається безпосередньо через інтерфейс ResourceRepository, оскільки бізнес-логіка не передбачає фільтрації чи обробки даних.

2. Шар Data у класі ResourceLocalDataSource фіксує статичний реєстр корисних адрес (УТОГ, «Spreadthesign», тематичні YouTube-плейлисти).

3. Шар Presentation використовує базовий resourceProvider та спеціалізований шаблон кнопок ChooseButton. Ключовою особливістю є асинхронний метод _openLink(String url), який за допомогою зовнішньої залежності url_launcher переформатовує рядок у Uri та викликає системну функцію launchUrl, безпечно відкриваючи посилання у вбудованому браузері пристрою.

Детальніше з модулем можна ознайомитися за допомогою лістингів Г.22-Г.28.

Такий модульний підхід дозволяє легко додавати нові категорії навчальних матеріалів, дублюючи цю структуру, редагуючи лише джерело даних, тип моделі та за потреби вносячи зміни до інших частин модуля.

3.3 Реалізація практичного розділу навчання та перевірки знань

Практичний розділ є основною складовою розробленого програмного застосунку та призначений для вивчення української жестової мови користувачами з порушеннями слуху. Головною метою цього модуля є забезпечення ефективного засвоєння навчального матеріалу завдяки тестуванню.

Нижче на рисунках 3.4–3.6 наведено структуру модуля exercises.

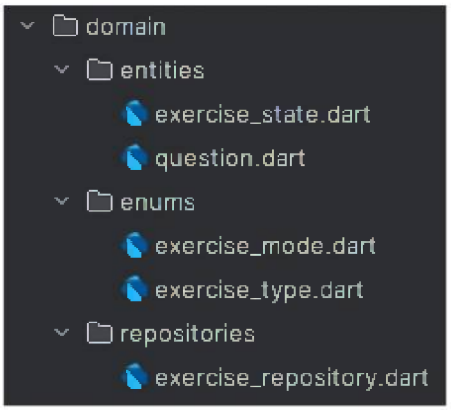


Рисунок 3.4 – структура модуля exercises (шар domain).

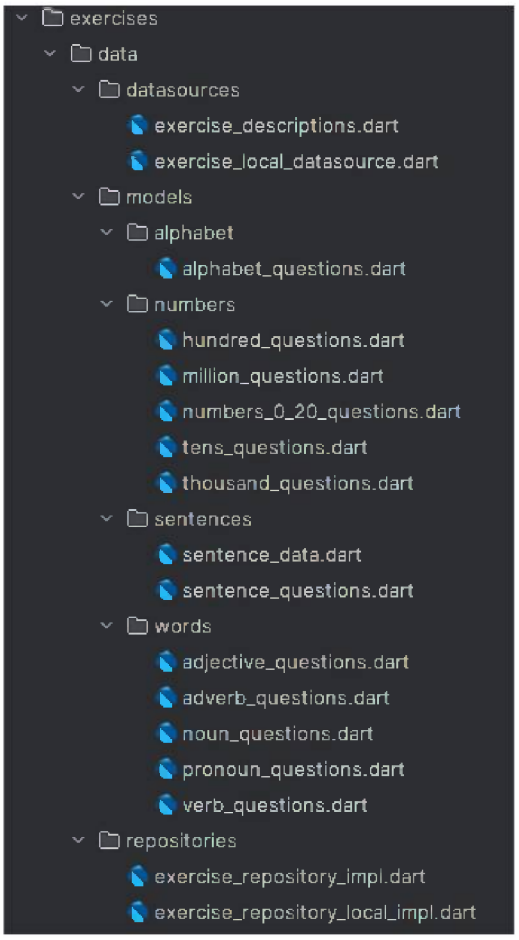


Рисунок 3.5 – структура модуля exercises (шар data).

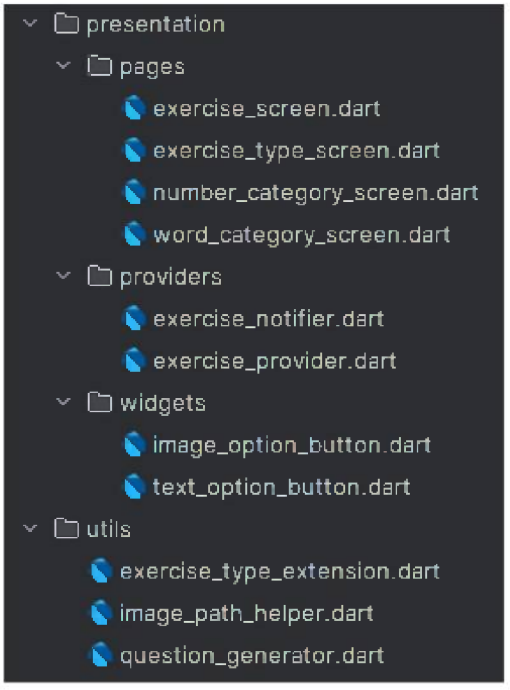


Рисунок 3.6 – структура модуля exercises (шари presentation та utils).

На відміну від теоретичного блока, де навчальний матеріал міг групуватися у відокремлені модулі, практичний розділ реалізовано у вигляді єдиного комплексного модуля exercises. Таке архітектурне рішення зумовлене високим рівнем шаблонності процесу тестування.

Незалежно від того, яка саме категорія знань перевіряється (знання дактильної абетки, числівників, окремих слів чи цілих речень), базова механіка взаємодії користувача з системою залишається незмінною: система генерує питання, пропонує варіанти відповідей, фіксує вибір та підраховує кількість правильних відповідей. Тому створення окремих модулів для кожного типу тестів призвело б до масового дублювання коду. Замість цього розроблено єдиний універсальний механізм, який динамічно адаптується під потрібний тип контенту. Як і теоретичний розділ, модуль exercises побудовано за принципами Clean Architecture, що видно з рисунків 3.4-3.6. Розглянемо детальніше реалізацію його базових шарів.

1.Шар предметної області (Domain) містить незалежне ядро бізнес-логіки системи тестування:

- Перелічення (Enums): ExerciseType визначає доступні категорії тестування (абетка, займенники, числа тощо) і слугує ключем для маршрутизації даних; ExerciseMode задає напрямок тестування: symbolToLetter (система показує візуальний жест, а користувач обирає текстове значення) та letterToSymbol (система показує текстове слово, а користувач має обрати відповідне зображення жесту).
- Сутності (Entities): клас Question інкапсулює текст питання, правильну відповідь, масив варіантів та шлях до графічного ресурсу. Клас ExerciseState зберігає поточний стан сесії (масив питань, індекс, рахунок користувача та обрану відповідь) і містить геттер isFinished для фіксації завершення тесту.
- Інтерфейси (Repositories): абстрактний клас ExerciseRepository декларує єдиний контракт getQuestions для отримання контенту за типом і режимом вправи.

2. Шар даних (Data) відповідає за підготовку та постачання тестових завдань:

- Моделі даних (Models): статичні дані словника згруповані у профільні класи (AlphabetQuestions, NounQuestions тощо), які містять константні масиви слів, літер або чисел. Замість того щоб самостійно створювати тести, ці класи передають вихідні списки слів до утилітарного класу QuestionGenerator, який автоматично перемішує їх, додає варіанти відповідей та формує готові завдання.
- Джерела даних (Datasources): конкретна реалізація фасаду ExerciseLocalDataSourceImpl аналізує переданий ExerciseType і викликає відповідний клас-модель, забезпечуючи єдину точку доступу до тестів. Клас ExerciseDescriptions виконує роль реєстру інструкцій до вправ.
- Реалізація репозиторію (Repositories): клас ExerciseRepositoryImpl пов'язує джерело даних із шаром бізнес-логіки й транслює сформовані масиви питань на вищі архітектурні рівні.

3. Допоміжні утиліти (Utils) інкапсулюють математичні й алгоритмічні функції, розвантажуючи інші шари:

- Генератор завдань (QuestionGenerator): містить методи автоматичного формування списку Question. Він випадково перемішує базові масиви за допомогою функції shuffle(), відбирає потрібну кількість елементів та викликає метод generateOptions. Останній гарантує наявність правильної відповіді у списку та заповнює решту позицій випадковими неправильними варіантами, повторно перемішуючи масив.
- Розширення типу вправи (ExerciseTypeX): реалізовано через механізм extension мови Dart. Геттер folder зіставляє тип вправи із точною назвою директорії з ресурсами в активах застосунку, не змінюючи доменний enum.
- Помічник шляхів (ImagePathHelper): функция getImagePath координує збірку фінального посилання на локальне зображення жесту, адаптуючись під специфіку контенту (для слів використовується саме слово, для речень — назва файлу з карти відповідностей).

4. Шар представлення (Presentation) відповідає за графічний інтерфейс та реактивне керування станом:

- Внутрішня маршрутизація: екрани категорій (NumberCategoryScreen, WordCategoryScreen) розташовані безпосередньо всередині модуля exercises, а не у глобальній навігації. Це нівелює жорстку зв'язність, оскільки глобальний маршрутизатор не імпортує специфічні доменні моделі вправ. Екрани побудовані на базі StatelessWidget і лише передають параметри (ExerciseType) на конфігураційний вузол — ExerciseTypeScreen.
- Екран конфігурації (ExerciseTypeScreen): динамічно підвантажує інструкцію до обраного тесту та надає користувачеві можливість обрати режим ExerciseMode, після чого ініціює перехід на головний екран тестування.
- Екран тестування (ExerciseScreen): базується на ConsumerWidget від Riverpod. Провайдер exerciseProvider.family приймає пару параметрів (тип, режим) та ізолює стан для кожного тесту. Метод build розгалужує інтерфейс користувача: у фазі завершення генерується підсумковий результат, в активній — поточне питання. Залежно від обраного режиму ExerciseMode, варіанти відповідей відображаються або графічними кнопками ImageOptionButton, або текстовими кнопоками TextOptionButton.

Детальніше модуль можна розглянути за допомоги додатків Г.29-Г.58.

РОЗДІЛ 4
ПРАКТИЧНІ РЕКОМЕНДАЦІЇ ПО ЗАСТОСУВАННЮ КОМП'ЮТЕРНОЇ МОДЕЛІ

4.1 Демонстрація роботи комп'ютерної моделі

Уявімо, що ми – користувач, який планує вивчити дактильну абетку. Розглянемо всі етапи його навчання:

Першим кроком після запуску програми користувач побачить головне меню, яке наведено на рисунку 4.1.

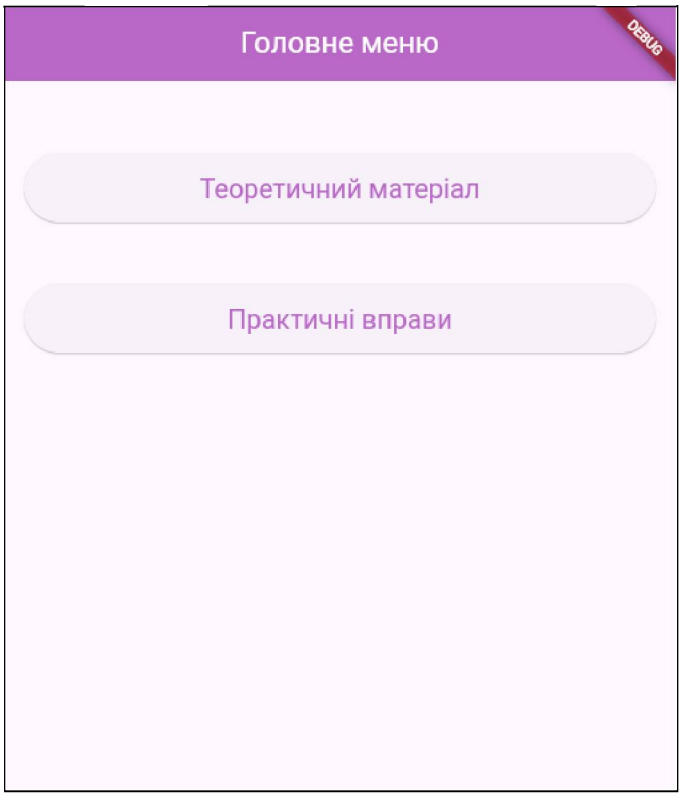


Рисунок 4.1 – Головне меню.

Наступним логічним кроком буде натискання на кнопку "Теоретичний матеріал". Результат цієї дії – відкриття екрану з темами для навчання, його можна побачити на рисунку 4.2.

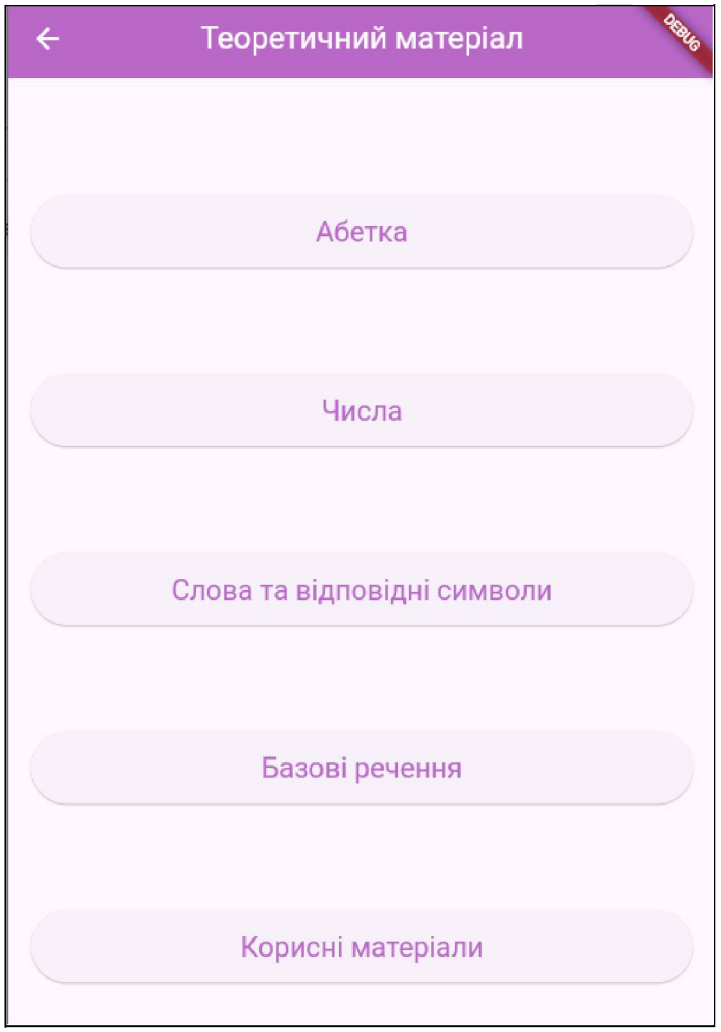


Рисунок 4.2 – Екран розділу “Теоретичний матеріал”.

Після цього користувач обирає потрібний йому розділ «Абетка» і в результаті бачить шуканий для навчання матеріал. На рисунках 4.3–4.4 наведено дві частини екрана з абеткою: перша — початковий стан після переходу до розділу, а друга — стан після прокручування сторінки вниз.



Рисунок 4.3 – Екран підрозділу “Абетка” (частина 1).



Рисунок 4.4 – Екран підрозділу “Абетка” (частина 2).

Після ознайомлення з абеткою наступним кроком буде спроба перевірити свої знання. Для цього потрібно натиснути у лівому верхньому кутку на стрілку, яка повертає до попереднього розділу (її можна було побачити на рисунках 4.1–4.4). Як наслідок, користувач потрапляє до головного меню (рис. 4.1), де обирає розділ «Практичні вправи», наведений на рис. 4.5.

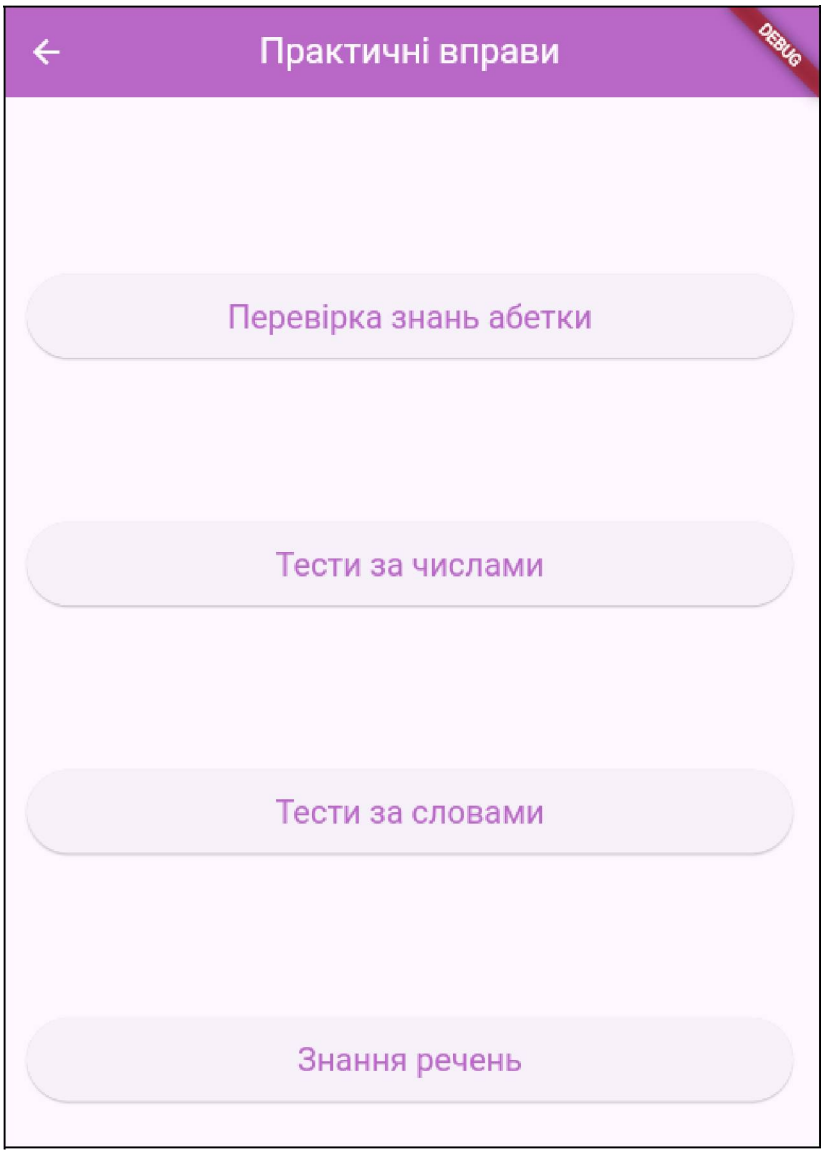


Рисунок 4.5 – Екран розділу “Практичні вправи”.

Далі логічним буде обрати підрозділ «Перевірка знань абетки» для контролю відповідних знань. На рисунку 4.6 наведено екран вибору типу тестування.

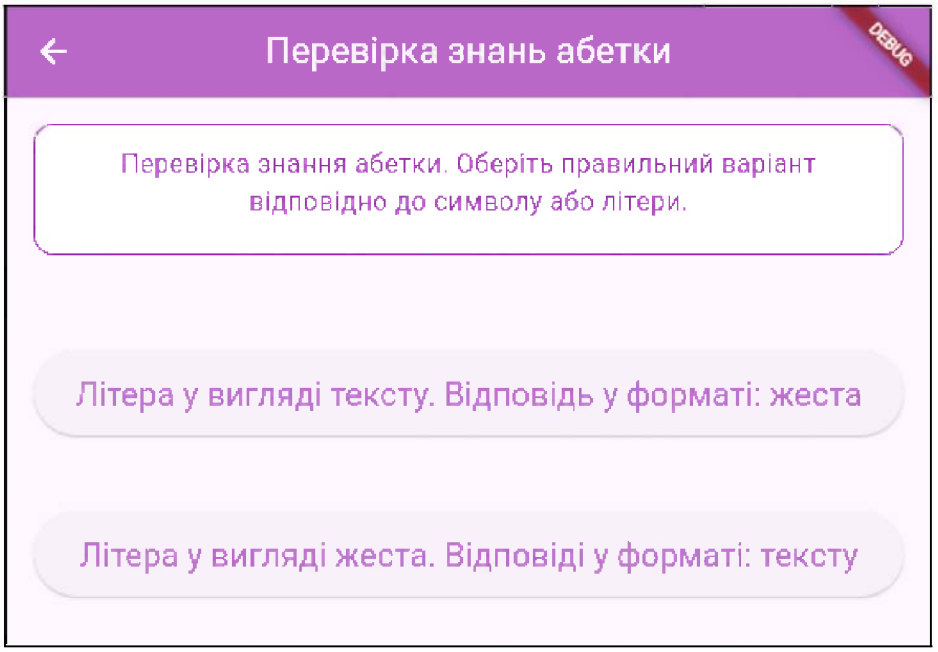


Рисунок 4.6 – Екран вибору типу тестування за темою “Абетка”.

Оберемо тип тестування: “Літера у вигляді тексту. Відповідь у форматі: жеста”, після чого починається тестування; екран із першим питанням наведено на рисунку 4.7.



Рисунок 4.7 – Екран тестування з першим питанням.

Уявімо, що користувач обрав правильний варіант. Тоді відповідь підсвічується зеленим кольором, і за секунду питання змінюється на наступне. Описані процеси наведено на рис. 4.8 та 4.9 відповідно.

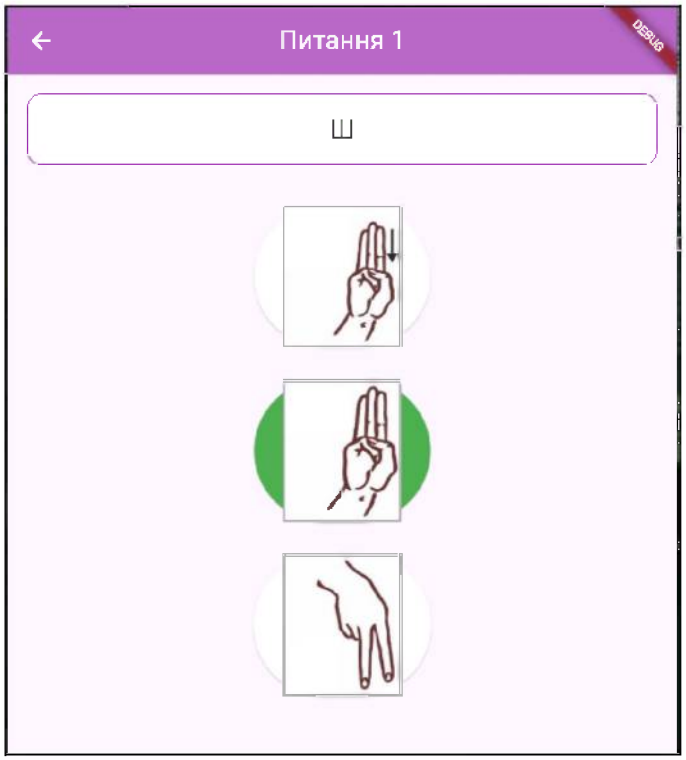


Рисунок 4.8 – Результат вибору правильної відповіді.



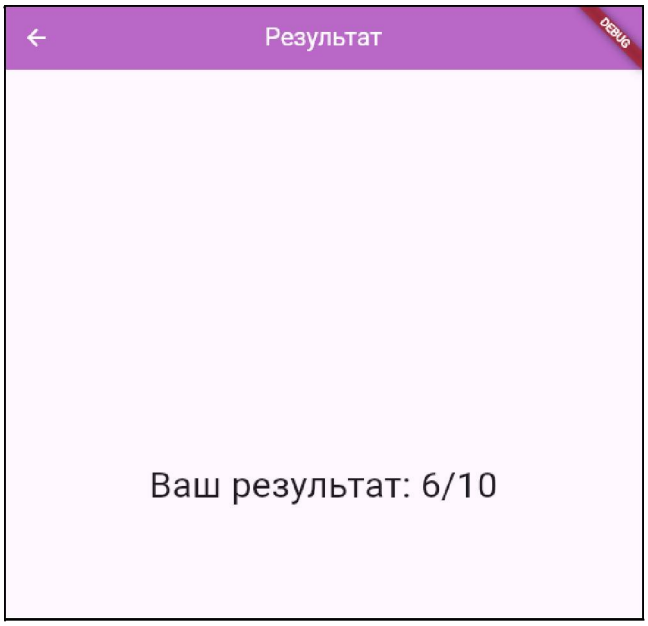
Рисунок 4.9 – Екран тестування з другим питанням.

При імітації вибору неправильної відповіді спостерігається аналогічний результат, проте на екрані одночасно підсвічуються як правильний варіант, так і неправильний (обраний). Цей процес продемонстровано на рисунку 4.10.



Рисунок 4.10 – Результат вибору неправильної відповіді.

Після проходження тесту користувач бачить результат, за яким він розуміє рівень своєї обізнаності з цієї теми. Екран із результатом тестування наведено на рисунку 4.11.



Рисунку 4.11 – Екран тестування з результатом.

Висновки до розділу 4

1. Показано повний цикл взаємодії користувача з програмним забезпеченням, що охоплює послідовні етапи від запуску головного меню та ознайомлення з теоретичним матеріалом до проходження підсумкового тестування.

2. Встановлено, що структура інтерфейсу забезпечує інтуїтивно зрозумілу навігацію, дозволяючи користувачеві легко перемикатися між вивченням дактильної абетки та режимом перевірки знань.

3. Показано високу функціональність теоретичного підрозділу «Абетка», який забезпечує зручний перегляд усіх дактилем завдяки підтримці вертикального прокручування екрана.

4. Встановлено ефективність механізму вибору типів тестування (наприклад, переклад із текстового формату в жестовий), що дозволяє адаптувати процес навчання під індивідуальні потреби користувача.

5. Показано дієвість інтерактивної системи оцінювання, яка надає миттєвий візуальний зворотний зв’язок шляхом підсвічування обраних відповідей: зеленим кольором для правильних варіантів та червоним для помилкових.

6. Встановлено доцільність реалізованої програмної затримки тривалістю в одну секунду після вибору відповіді, що забезпечує користувачеві необхідний час для фіксації результату та аналізу помилки перед переходом до наступного питання.

7. Показано завершеність процесу навчання через відображення фінального екрана з результатами, що дозволяє користувачеві об’єктивно оцінити рівень власної обізнаності за обраною темою.

ВИСНОВКИ

У кваліфікаційній роботі розв’язано актуальне науково-прикладне завдання, що полягає у розробці комп’ютерної моделі та інклюзивного програмного застосунку для вивчення української жестової мови людьми з порушеннями слуху. Усі завдання, поставлені у вступі до дослідження, були успішно та в повному обсязі виконані. На початковому етапі здійснено ґрунтовний аналіз сучасних цифрових та асистивних технологій навчання осіб з особливими освітніми потребами, а також досліджено наявні програмні рішення. Виявлено, що більшість наявних систем не забезпечують належного рівня візуалізації контенту, інтерактивності та підтримки індивідуального темпу навчання, що обґрунтувало необхідність створення концептуально нового програмного продукту. Для розв'язання цієї проблеми було розроблено функціональну комп’ютерну модель навчання в нотації IDEF0, яка базується на структурованому поданні навчального матеріалу від простого до складного та орієнтована на візуальний канал сприйняття інформації.

На основі запропонованої моделі засобами кросплатформеної технології Flutter розроблено сучасний програмний застосунок. Його програмна основа побудована за принципами Clean Architecture з чітким розподілом на шари представлення, предметної області та даних, а керування станом реалізовано за допомогою реактивного засобу Riverpod. Для виконання завдання з реалізації механізмів перевірки знань створено єдиний інтерактивний практичний модуль. Він автоматично генерує тестові завдання, підтримує два взаємозворотні режими перевірки (переклад із текстового формату в жестовий і навпаки) та забезпечує миттєвий візуальний зворотний зв'язок через кольорову індикацію правильних і неправильних відповідей, які створюються випадковим чином у вигляді дистракторів.

На завершальному етапі проведено комплексне тестування розробленого програмного застосунку шляхом практичної демонстрації та покрокового аналізу дій користувача. Результати тестування підтвердили коректність роботи

всіх компонентів системи, зручність та інклюзивність інтерфейсу, а також високу ефективність застосунку як інструменту для самостійного вивчення української жестової мови. Таким чином, розроблений цифровий продукт повністю відповідає сучасним вимогам інклюзивної освіти та компенсує функціональні недоліки наявних аналогів.

У результаті проведеного дослідження поставлену мету повністю досягнуто — завдяки використанню комп'ютерної моделі на основі технології Flutter створено умови для суттєвого підвищення доступності та ефективності навчання української жестової мови для людей з порушеннями слуху.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Жестова мова — мова для людей без слів. JIT Academy [Електронний ресурс]. – режим доступу: URL: https://www.jit.academy/movy-zhestiv-movy-dlya-lyudej-bez-sliv/ (дата звернення – 02.10.2025).

2. Жестова мова по-українськи. Український інтерес [Електронний ресурс]. – режим доступу: URL: https://uain.press/articles/zhestova-mova-po-ukrayinski-1083664/ (дата звернення – 04.10.2025).

3. Flutter documentation. Flutter dev [Електронний ресурс]. – режим доступу: URL: https://docs.flutter.dev/ (дата звернення – 11.11.2025).

4. Martin R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Prentice Hall, 2017р. 432 с.

5. Riverpod: A reactive caching and data-binding framework. Riverpod dev [Електронний ресурс]. – режим доступу: URL: https://riverpod.dev/ (дата звернення – 20.11.2025).

6. Stack Overflow: Where Developers Learn, Share, & Build Careers. Stack Overflow [Електронний ресурс]. – режим доступу: URL: https://stackoverflow.com/ (дата звернення – 01.03.2026).

7. Spreadthesign: Online dictionary for sign languages [Електронний ресурс]. – режим доступу: URL: https://spreadthesign.com/uk.ua/search/ (дата звернення – 09.01.2026).

8. Українське товариство глухих (УТОГ). Офіційний сайт [Електронний ресурс]. – режим доступу: URL: https://utog.org/uslc/ (дата звернення – 02.10.2025).

9. Yeri AI: AI-powered sign language gesture generation [Електронний ресурс]. – режим доступу: URL: https://yeri.ai/ (дата звернення – 25.02.2026).

10. UNICEF – Digital Education Strategy 2025-2030: Beyond Digital as Usual [Електронний ресурс]. – Режим доступу: URL: https://www.unicef.org/digitaleducation/digital-education-strategy-2025-2030 (дата звернення – 03.02.2026).

11. UNESCO – Digital learning and transformation of education [Електронний ресурс]. – режим доступу: URL: https://www.unesco.org/en/digital-education (дата звернення – 18.10.2025).

12. WHO – Assistive technology [Електронний ресурс]. – Режим доступу: URL: https://www.who.int/news-room/fact-sheets/detail/assistive-technology (дата звернення – 08.10.2025).

13. НБУ ім. Ярослава Мудрого. YouTube-канал [Електронний ресурс]. – режим доступу: URL: https://www.youtube.com/@mudryilibrary/featured (дата звернення – 15.10.2025).

14. Українська жестова мова. YouTube-канал [Електронний ресурс]. – режим доступу: URL: https://www.youtube.com/@Українськажестовамова (дата звернення – 10.10.2025).

ДОДАТКИ

Додаток А

Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних
систем та робототехніки
Максим ХРУСЛОВ
_____
(підпис)
«13» жовтня 2025року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувача першого (бакалаврського) рівня вищої освіти
(першого (бакалаврського) / другого (магістерського))

КАРПЕНКА КИРИЛА РОМАНОВИЧА
(прізвище, ім'я, по батькові здобувача)

Спеціальність: 151 – Автоматизація, комп'ютерно-інтегровані технології
Освітня програма: «Автоматизація, комп'ютерно-інтегровані технології»
1. Тема роботи **КОМП'ЮТЕРНА МОДЕЛЬ НАВЧАННЯ ЖЕСТОВОЇ МОВИ НА ОСНОВІ ТЕХНОЛОГІЇ FLUTTER**
керівник роботи **Толстолузька Олена Геннадіївна, доктор технічних наук, професор,**
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від ***№ 4101-5/1245 від 28.05.2026 року.***
2. Строк подання студентом роботи ***30 травня 2026 року***
3. Вихідні дані до роботи:
- сучасні підходи та цифрові технології навчання людей з особливими потребами;
- методи візуального представлення навчального матеріалу для вивчення української жестової мови;
- вимоги до доступності, зручності використання навчального контенту;
- персональний комп'ютер для розробки та тестування.

4. Перелік питань, які потрібно розробити:
1) Аналіз сучасних підходів та цифрових технологій навчання людей з особливими потребами.
2) Розробка концептуальної комп'ютерної моделі навчання мови жестів із використанням фреймворку Flutter.
3) Проведення тестування моделі з метою оцінки зручності використання, надійності та продуктивності системи.
4) Аналіз можливих обмежень і проблем розробленої системи, зокрема адаптації навчального матеріалу, індивідуальних потреб користувачів та технічних рамок.

5. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Літературний огляд мови жестів та сучасних технологій навчання людей з особливими потребами	02.10.2025 - 30.10.2025
2	Формулювання проблеми, мети та завдань дослідження, доведення актуальності	01.11.2025 – 10.11.2025
3	Аналіз технології Flutter та інструментів розробки застосунків для освітніх систем	11.11.2025 – 18.11.2025
4	Розробка концепції комп'ютерної моделі навчання мови жестів	19.11.2025 – 17.12.2025
5	Проектування архітектури застосунку та структури навчального контенту	18.12.2025 – 20.01.2026
6	Реалізація застосунку засобами Flutter, створення бази жестів	21.01.2026 – 02.03.2026
7	Тестування застосунку з метою оцінки зручності використання, ефективності навчання та продуктивності системи	03.03.2026 – 10.03.2026
8	Аналіз результатів тестування, визначення обмежень системи та утворення інструкції з розгортання	11.03.2026 – 19.03.2026
9	Формулювання рекомендацій щодо вдосконалення моделі та її пристосування до умов дійсного використання	20.03.2026 – 30.03.2026
10	Підготовка та оформлення звітних матеріалів, пояснювальної записки і додатків	01.04.2026 – 29.04.2026
11	Формування звіту про переддипломну практику	30.09.2026 – 10.05.2026
12	Завершення оформлення пояснювальної записки кваліфікаційної роботи для перевірки на антиплагіат	до 25.05.2026
13	Відгук керівника, висновки та підготовка роботи для рецензування	до 30.05.2026

Дата видачі завдання ***02 жовтня 2025 року.***

Здобувач вищої освіти _____ Кирило КАРПЕНКО
(підпис) (ім'я прізвище)

Керівник роботи _____ Олена ТОЛСТОЛУЗЬКА
(підпис) (ім'я прізвище)

ДОДАТОК Б

Затверджую

_____

«_____»_____2026 р.

Технічне завдання
на розробку програмного вироби
«Комп’ютерна модель навчання жестової мови на основі технології Flutter»

Назва розділу	Назва та зміст підрозділу
1. Вступ	1.1. Назва програмного виробу: «Комп’ютерна модель навчання української жестової мови на основі технології Flutter» (програма). 1.2. Галузь застосування: програма належить до сфери освітніх інформаційних технологій та використовується в галузі інклюзивної освіти.
2. Підстава для розробки	2.1. Навчальний план ННІ КН та ШІ за спеціальністю 151 – «Автоматизація, комп'ютерно-інтегровані технології та робототехніка» 2.2.Завдання на дипломну роботу бакалавра, затверджено наказом ХНУ імені В. Н. Каразіна № 4101-5/1245 від 28.05.2026 р.
3. Призначення розробки	3.1. Метою розробки є підвищення доступності та ефективності навчання української жестової мови для людей з порушеннями слуху шляхом використання комп’ютерної моделі на основі технології Flutter. 3.2. Призначення програмного виробу: організація процесу навчання жестової мови шляхом надання навчальних матеріалів та виконання інтерактивних вправ. 3.3. Початковими даними для розробки є перелік навчальних задач з вивчення української жестової мови.
4. Технічні вимоги до програмного виробу	4.1. Вимоги до функціональних характеристик: ● забезпечувати вибір користувачем розділів навчального матеріалу; ● забезпечувати відображення навчального контенту у текстовому та графічному вигляді; ● реалізовувати режими навчання (теоретичний та практичний); ● формувати тестові завдання на основі навчального матеріалу; ● забезпечувати вибір відповіді користувачем;

	● виконувати перевірку правильності відповіді; ● відображати правильний варіант відповіді; ● здійснювати підрахунок результатів тестування; ● відображати підсумкові результати; ● забезпечувати навігацію між екранами застосунку; ● забезпечувати час відгуку інтерфейсу не більше 1 секунди. 4.2. Вимоги до надійності: програма повинна забезпечувати стабільне функціонування без збоїв під час виконання основних функцій, коректну обробку дій користувача та відсутність аварійного завершення роботи. Результати виконання завдань повинні оброблятися коректно та зберігатися протягом сеансу роботи. 4.3. Вимоги до умов експлуатації: умови експлуатації відповідають умовам використання сучасних мобільних пристроїв та персональних комп'ютерів; програма розрахована на непрофесійного користувача; інтерфейс повинен бути простим та інтуїтивно зрозумілим; для роботи достатньо базових навичок користування пристроєм; спеціальне обслуговування не потрібне. 4.4. Вимоги до складу параметрів технічних засобів: ● мобільний пристрій або персональний комп'ютер; ● підтримка ОС: Android, iOS, Windows, Linux, macOS; ● оперативна пам'ять — не менше 2 ГБ; ● вільне місце — не менше 200 МБ; ● процесор — не нижче 1.5 ГГц; ● роздільна здатність екрана — не менше 1280×720; ● введення даних — сенсорний екран / клавіатура / миша. 4.5. Вимоги до інформаційної та програмної сумісності: ● забезпечувати коректну обробку текстових та графічних даних; ● забезпечувати сумісність на всіх підтримуваних платформах;

<table>
<tr><td></td><td>

використовувати уніфіковану структуру вхідних та вихідних даних;
реалізація з використанням мови програмування Dart та фреймворку Flutter.

4.6. Вимоги до маркування та упаковки – відсутні.
4.7. Вимоги до транспортування та зберігання – відсутні.
4.8. Спеціальні вимоги до програмного виробу – відсутні.
</td></tr>
<tr><td>5. Вимоги до програмної документації.</td><td>Програмною документацією до виробу «Комп’ютерна модель навчання української жестової мови на основі технології Flutter» вважати:
1) Справжнє Технічне завдання на розробку програмного виробу (передбачити у вигляді таблиці)
2) Програму та методику випробувань розробленого програмного виробу (представити у текстуальному вигляді)
3) Опис програмного виробу (не потрібно).
4) Текст програми (не потрібно).</td></tr>
<tr><td>6. Техніко-економічні показники</td><td>У цьому розділі можуть бути:
1) порівняльний структурно-функціональний аналіз представити в розділі 1 пояснювальної записки до кваліфікаційної роботи;
2) вимоги до розрахунку техніко-економічних показників не потрібні.</td></tr>
<tr><td>7. Стадії та етапи розробки</td><td>
<table>
<tr><th>№ з/п</th><th>Назви етапів роботи</th><th>Срок виконання етапів роботи</th></tr>
<tr><td>1</td><td>Літературний огляд мови жестів та сучасних технологій навчання людей з особливими потребами</td><td>02.10.2025 – 30.10.2025</td></tr>
<tr><td>2</td><td>Формулювання проблеми, мети та завдань дослідження, доведення актуальності</td><td>01.11.2025 – 10.11.2025</td></tr>
<tr><td>3</td><td>Аналіз технології Flutter та інструментів розробки застосунків для освітніх систем</td><td>11.11.2025 – 18.11.2025</td></tr>
</table>
</td></tr>
</table>

	4	Розробка концепції комп'ютерної моделі навчання мови жестів	19.11.2025 – 17.12.2025
	5	Проектування архітектури застосунку та структури навчального контенту	18.12.2025 – 20.01.2026
	6	Реалізація застосунку засобами Flutter, створення бази жестів	21.01.2026 – 02.03.2026
	7	Тестування застосунку з метою оцінки зручності використання, ефективності навчання та продуктивності системи	03.03.2026 – 10.03.2026
	8	Аналіз результатів тестування, визначення обмежень системи та утворення інструкції з розгортання	11.03.2026 – 19.03.2026
	9	Формулювання рекомендацій щодо вдосконалення моделі та її пристосування до умов дійсного використання	20.03.2026 – 30.03.2026
	10	Підготовка та оформлення звітних матеріалів, пояснювальної записки і додатків	01.04.2026 – 29.04.2026
	11	Формування звіту про переддипломну практику	30.09.2026 – 10.05.2026
	12	Завершення оформлення пояснювальної записки кваліфікаційної роботи для перевірки на антиплагіат	до 25.05.2026
	13	Відгук керівника, висновки та підготовка роботи для рецензування	до 30.05.2026

8. Порядок контролю та приймання	1) Перевірку ходу розроблення заданих проектних документів керівнику роботи здійснювати 1 раз на тиждень. 2) Розроблені проектні документи подати в електронному вигляді та на паперових носіях в одному примірнику.

Виконавець

студент групи КУ-41

Карпенко К.Р.



(підпис)

Замовник

доктор тех. наук, професор

Толстолузька О.Г.



(підпис)

Додаток В

Програма і методика випробувань програмного виробу

«Комп’ютерна модель навчання жестової мови на основі технології Flutter»

1. Об'єкт випробувань

1.1 Найменування випробуваного програмного виробу «Комп’ютерна модель навчання української жестової мови на основі технології Flutter».

1.2 Область його застосування навчання людей з особливими потребами.

2. Мета випробувань

Перевірка працездатності програмного застосунку «Комп’ютерна модель навчання української жестової мови на основі технології Flutter», підтвердження коректності реалізації основних функцій та відповідності вимогам технічного завдання.

3. Загальні положення

3.1 Підстави для проведення випробувань

Підставою для проведення випробувань є наказ про призначення атестаційної комісії.

3.2 Місце і тривалість випробувань

Приймальні (приймально-здавальні) випробування проводяться на базі комп'ютерного класу кафедри в період роботи атестаційної комісії.

3.3 Обсяг випробувань

Приймальні випробування програмного вироби проводяться в обсязі відповідному цієї Програми і методики випробувань.

3.4 Організації, які беруть участь у випробуваннях

Приймальні випробування проводяться атестаційною комісією напередодні засідання (або в процесі засідання) за участю Замовника, Виконавця та інших осіб, присутніх на засіданні.

4. Вимоги до програми або програмного виробу

4.1. Вимоги до функціональних характеристик:

1) представляти собою програмну реалізацію навчального застосунку для вивчення української жестової мови;

2) забезпечувати доступ до теоретичного матеріалу (абетка, числа, слова, базові речення, корисні матеріали);

3) забезпечувати відображення навчального матеріалу у вигляді зображень жестів;

4) забезпечувати вибір підрозділів навчального матеріалу (частини мови, числові групи, речення);

5) реалізовувати практичний режим із тестовими завданнями;

6) забезпечувати два типи тестування: «текст → жест» та «жест → текст»;

7) забезпечувати перевірку відповідей користувача з візуальним відображенням результату;

8) відображати підсумковий результат після завершення тестування;

9) забезпечувати навігацію між екранами програмного застосунку.

4.2. Вимоги до надійності:

Програма повинна забезпечувати стабільне функціонування під час виконання всіх передбачених функцій. Система повинна коректно обробляти дії користувача, не допускати аварійного завершення роботи та забезпечувати правильне відображення результатів тестування.

4.3. Вимоги до умов експлуатації:

Умови експлуатації програмного застосунку відповідають умовам експлуатації сучасних персональних комп'ютерів та мобільних пристроїв. Програма розрахована на непрофесійного користувача та не потребує спеціальних знань для роботи.

4.4. Вимоги до складу і параметрів технічних засобів:

Для роботи програмного застосунку необхідний персональний комп'ютер або мобільний пристрій (ноутбук, смартфон) із встановленою сучасною операційною системою (Android, iOS, Windows, Linux, macOS).

4.5. Вимоги до інформаційної та програмної сумісності:

Програмний застосунок повинен забезпечувати коректну роботу на різних програмних платформах. Для розробки використовується мова програмування Dart та фреймворк Flutter, що забезпечує кросплатформеність програмного продукту.

4.6. Вимоги до маркування та упаковки – відсутні.

4.7. Вимоги до транспортування і зберігання – відсутні.

4.8. Спеціальні вимоги – не пред'являються.

5. Вимоги до програмної документації

Склад програмної документації, що подається на випробування, включає:

1) Технічне завдання на розробку програмного виробу (представлено в Додатку Б до пояснювальної записки до дипломної роботи).

2) Ця Програма і методика випробувань розробленого програмного вироби (представлена в Додатку В до пояснювальної записки до дипломної роботи).

3) Опис програмного виробу (представлено в розділі 3 пояснювальної записки до дипломної роботи).

6. Засоби і порядок випробувань

6.1 Засоби випробувань.

Випробування проводяться на технічних засобах: персональний комп'ютер, ноутбук або мобільний пристрій. Для тестування використовується середовище розробки Android Studio та емулятори.

6.2 Порядок проведення випробувань.

6.2.1 Перевірка програмної документації.

Перевірка здійснюється за критерієм наявності повного комплекту документації відповідно до технічного завдання.

6.2.2 Перевірка якості програмної документації.

Перевірка здійснюється за критерієм відповідності вимогам єдиної системи програмної документації (ЄСПД).

6.2.3 Перевірка виконання програми.

Для проведення випробувань необхідний персональний комп'ютер або мобільний пристрій із встановленим програмним застосунком «Комп'ютерна модель навчання української жестової мови на основі технології Flutter».

Для проведення випробувань використовуються:

- персональний комп'ютер або ноутбук;
- середовище розробки Android Studio;
- Flutter SDK;
- емулятор мобільного пристрою для тестування.

6.2 Порядок проведення випробувань

Випробування програмного виробу здебільшого проводяться у два етапи:
-ознайомчий етап
-етап перевірки функціонування програмного застосунку.

Перелік перевірок, що проводяться на першому етапі випробувань, включає:

1. Перевірку комплектності програмної документації;
2. Перевірку відповідності програмної документації вимогам ЄСПД;
3. Перевірку комплектності програмних та технічних засобів;
4. Перевірку можливості запуску програмного застосунку.

Якість програмної документації перевіряється на відповідність вимогам стандартів ДСТУ та технічного завдання.

Перелік перевірок, що проводяться на другому етапі випробувань, включає:

перевірку відповідності функціональних можливостей застосунку вимогам технічного завдання;

1. Перевірку роботи теоретичного розділу;

2. Перевірку роботи практичного розділу;
3. Перевірку коректності проходження тестування;
4. Перевірку навігації між екранами програмного застосунку;
5. Перевірку коректності відображення графічних матеріалів та результатів тестування.

Програмний застосунок повинен функціонувати відповідно до умов експлуатації персонального комп'ютера або мобільного пристрою.

Для роботи програмного застосунку необхідний персональний комп'ютер, ноутбук або мобільний пристрій із підтримкою операційних систем Android, Windows, Linux або macOS.

Для проведення випробувань пропонуються тести 1-4.

Перевірити підсвічування неправильної відповіді червоним кольором та правильної — зеленим.

Завершити тестування.

Перевірити відображення підсумкового результату.

У разі коректної перевірки відповідей та правильного відображення результатів тест вважається успішно пройденим.

Тест 1: Перевірка можливості запустити програму

Перевіряється успішний запуск та відображення усіх потрібних елементів головного меню на екрані, результат наведено на рисунку В.1

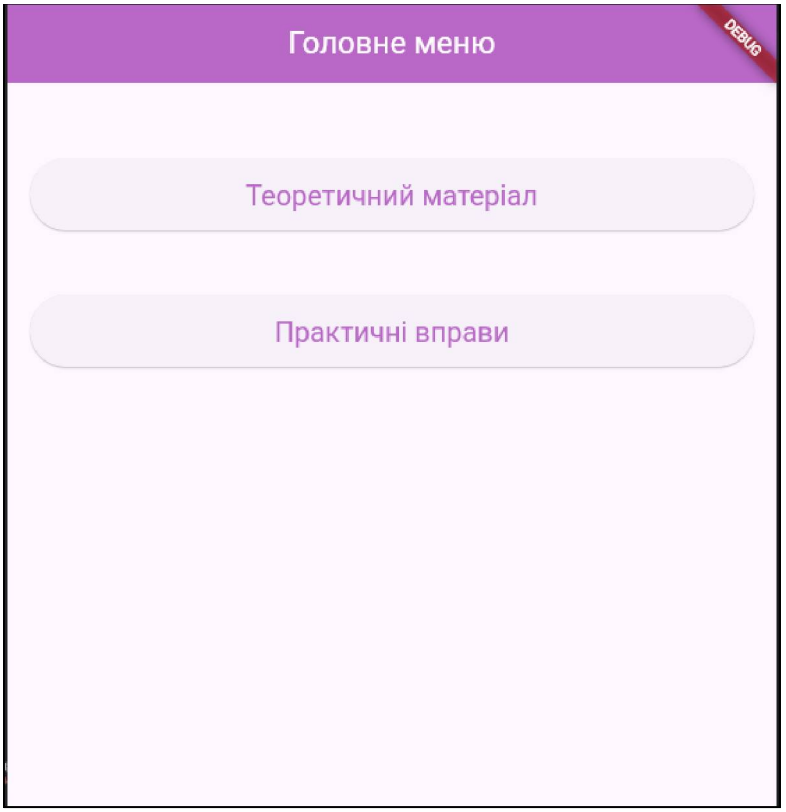


Рисунок В.1 – Успішно запущена програма.

Висновок: перевірка запуску пройшла успішно.

Тест 2: Перевірка теоретичного розділу

Перевіряється відображення навчальних матеріалів та перехід між сторінками. Результат можна побачити на рисунках В.2-В.11.

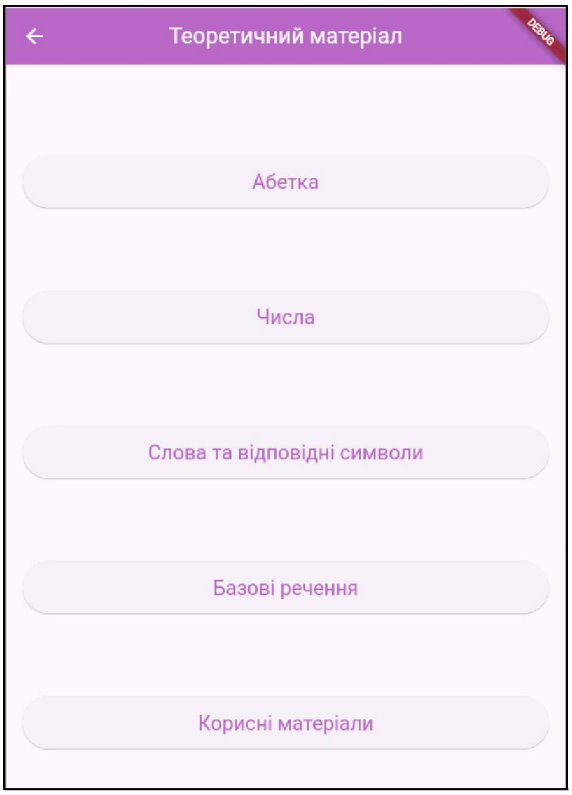


Рисунок В.2 – Меню теоретичного матеріалу.

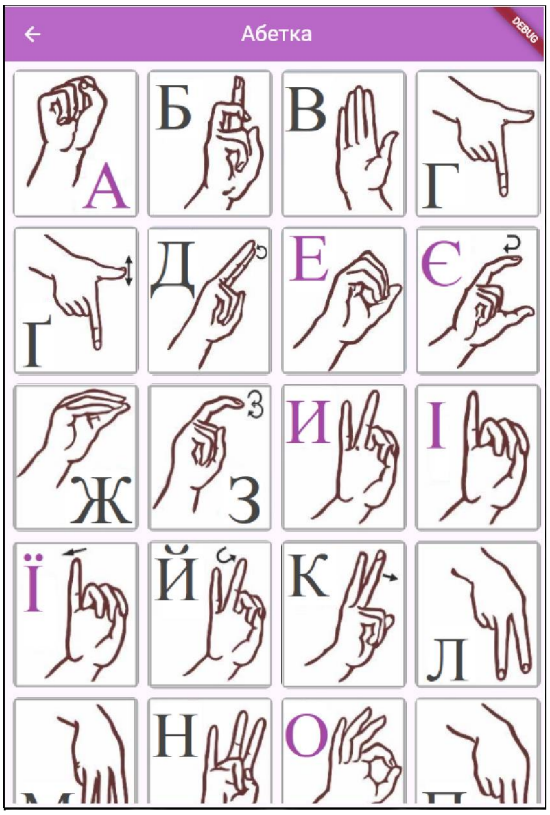


Рисунок В.3 – Підрозділ “Абетка”.



Рисунок В.4 – Меню підрозділу “Числа”.

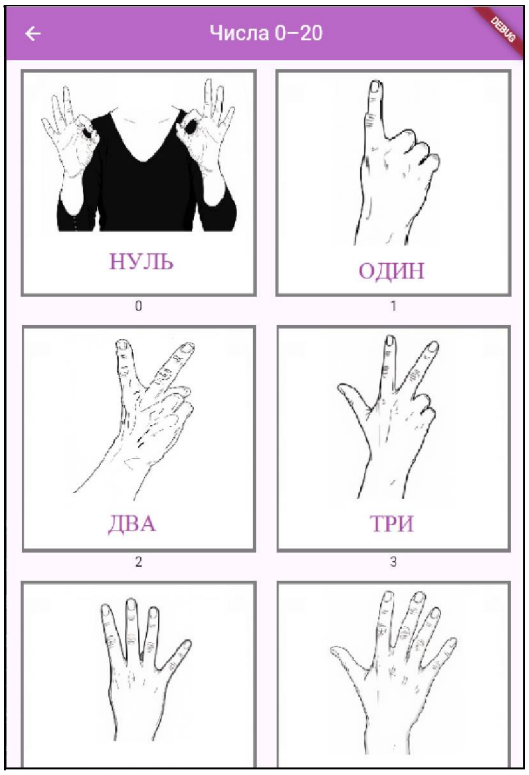


Рисунок В.5 – Підрозділ “Від 0 до 20”.

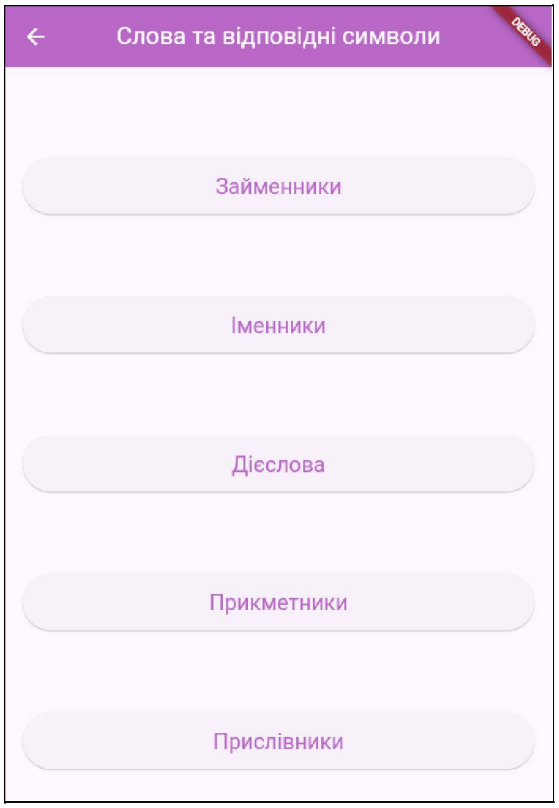


Рисунок В.6 – Меню підрозділу “Слова та відповідні символи”.

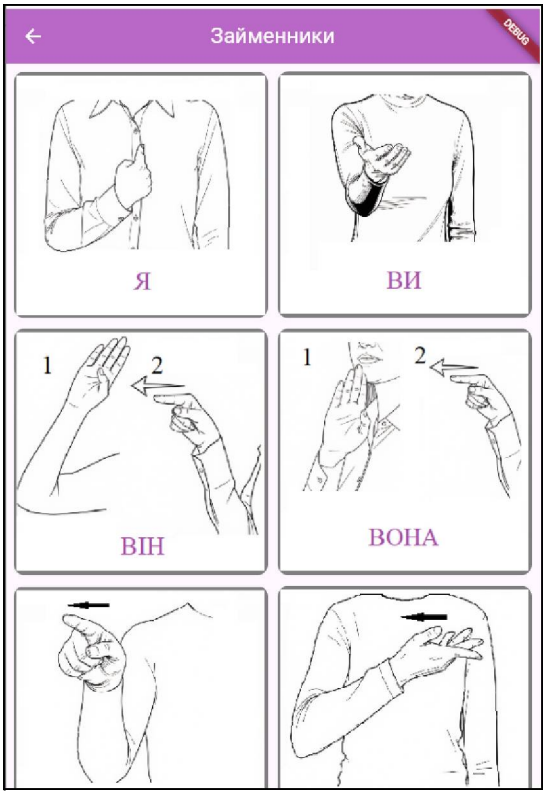


Рисунок В.7 – Підрозділ “Займенники”.

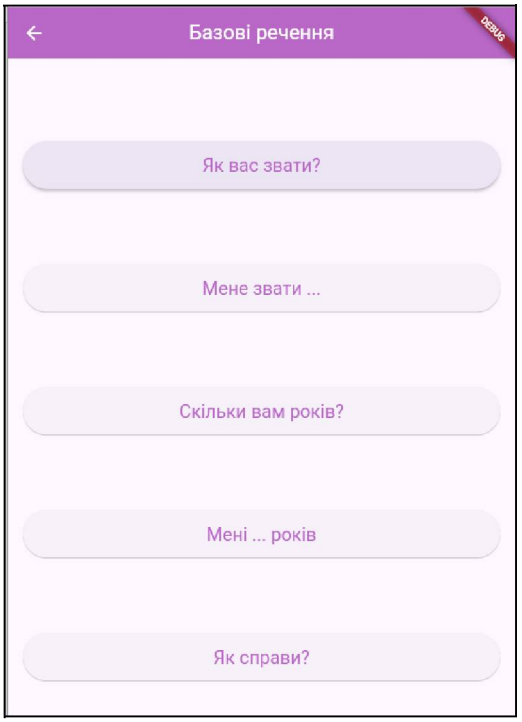


Рисунок В.8 – Меню підрозділу “Базові речення”.

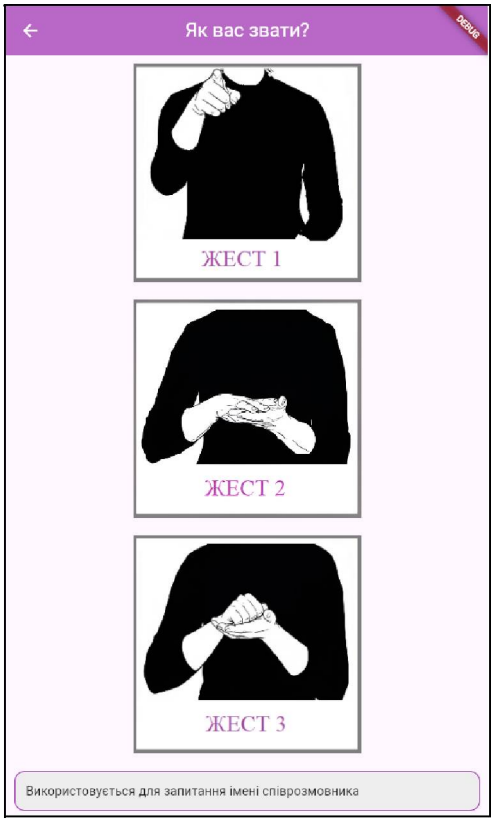


Рисунок В.9 – Підрозділ “Як вас звати?”.



Рисунок В.10 – Підрозділ “Корисні матеріали”.

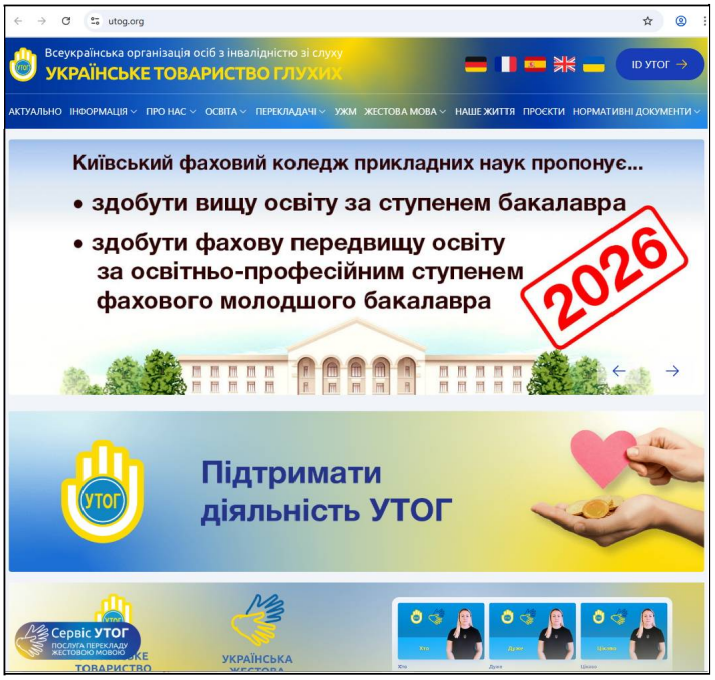


Рисунок В.11 – Відкрите посилання після натискання кнопки “Офіційний сайт українського товариства глухих”.

Висновок: усі підрозділи відображають інформацію належним чином, а проблеми під час переходів між сторінками відсутні, отже, перевірку пройдено успішно.

Тест 3: Перевірка практичного розділу

Перевіряється коректне відображення сторінок практичного розділу та перехід між ними. На рисунках В.12-В.18 наведено результати перевірки.

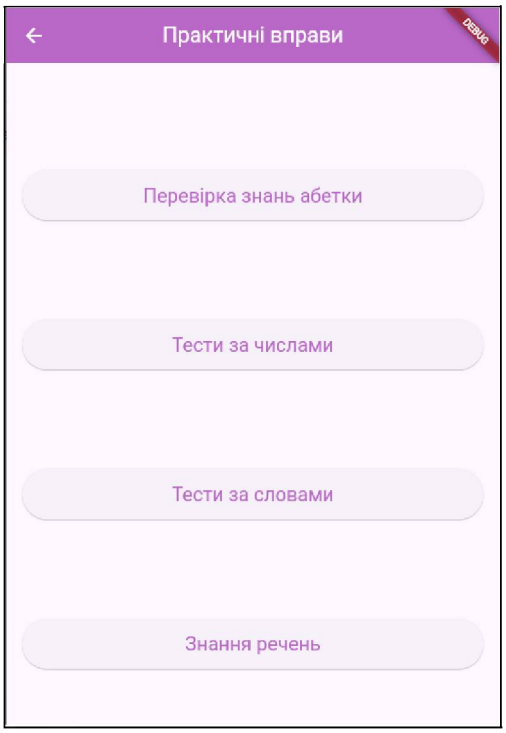


Рисунок В.12 – Меню розділу “Практичні вправи”.

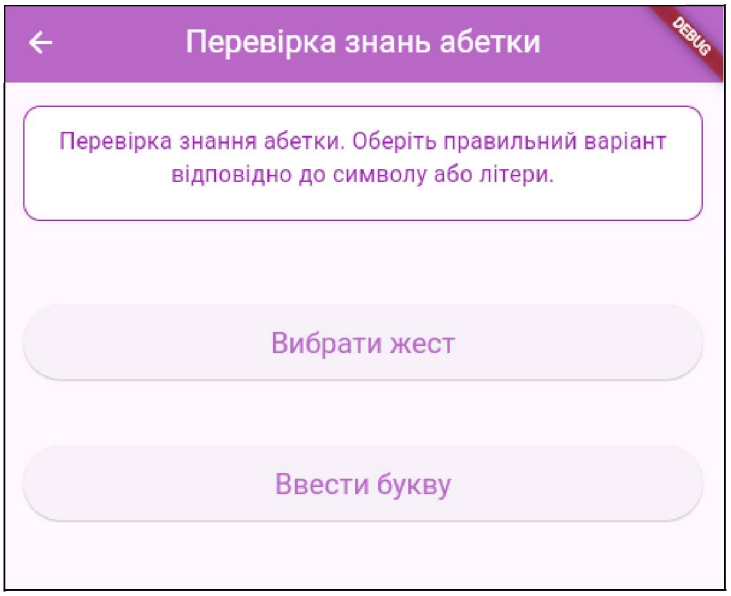


Рисунок В.13 – Підрозділ вибору типу тестування “Перевірка знань абетки”.

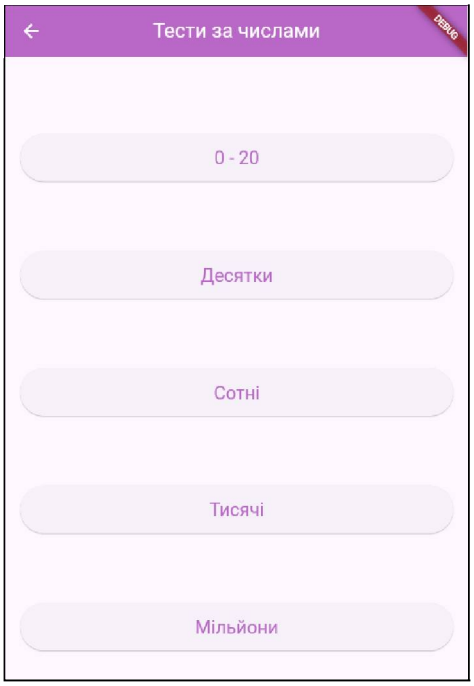


Рисунок В.14 – Меню підрозділу “Тести за числами”.

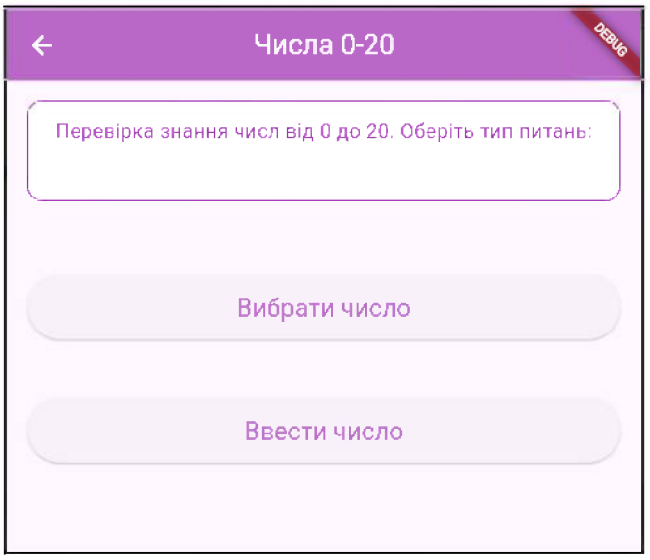


Рисунок В.15 – Підрозділ вибору типу тестування “Від 0 до 20”.

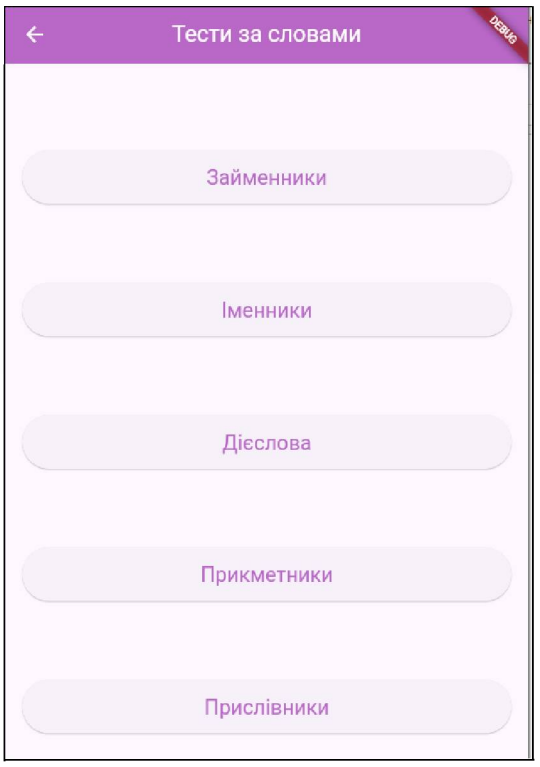


Рисунок В.16 – Меню підрозділу “Тести за словами”.

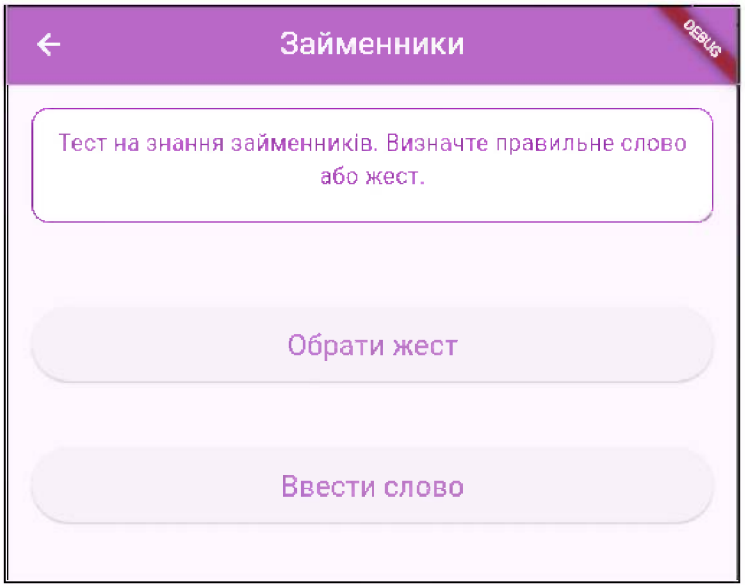


Рисунок В.17 – Підрозділ вибору типу тестування “Займенники”.

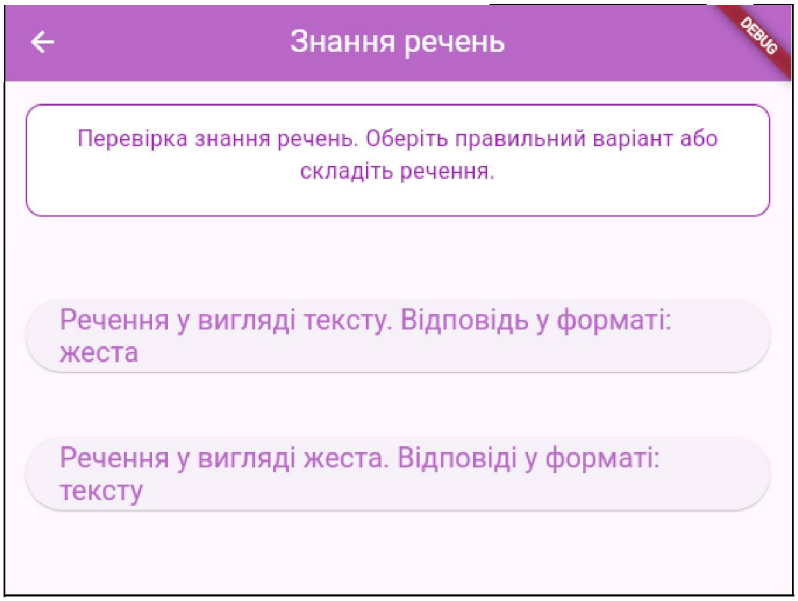


Рисунок В.18 – Підрозділ вибору типу тестування “Знання речень”.

Висновок: усі підрозділи відображають інформацію належним чином, а проблеми під час переходів між сторінками відсутні, отже, перевірку пройдено успішно.

Тест 4: Перевірка працездатності тестування

Перевіряється можливість обрати тип тестування, вибір відповіді, коректність підсвічування відповідей та вивід підсумкового результату після завершення тестування на рисунках В.19-В.22 можна побачити результат перевірки.

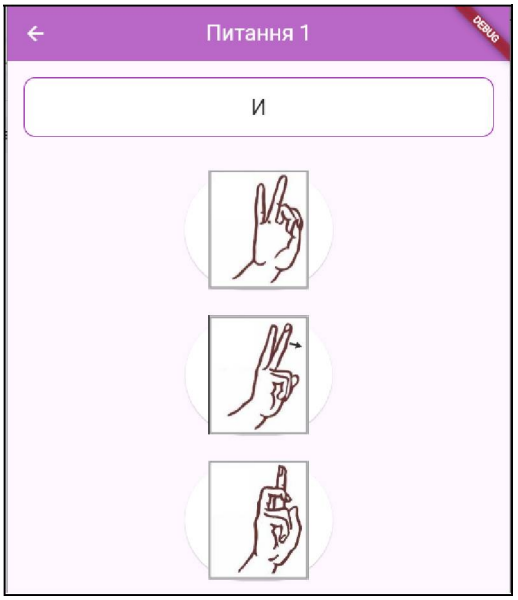


Рисунок В.19 – Початок тестування після натискання кнопки “Вибрати жест”.

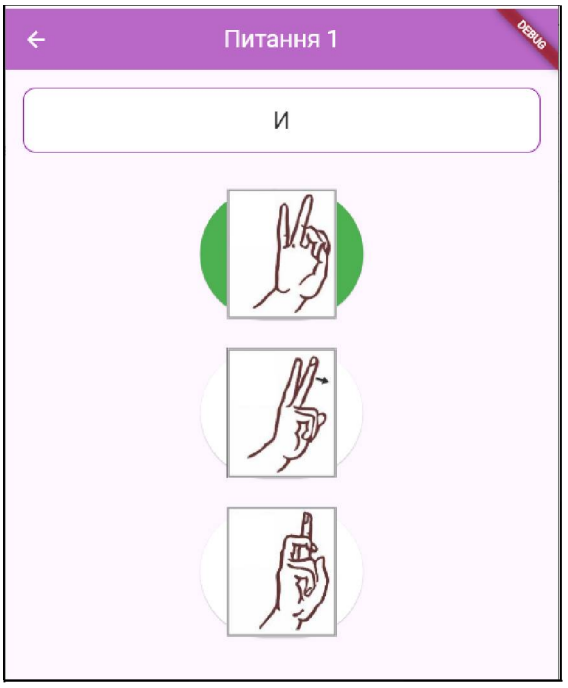


Рисунок В.20 – Обрано правильну відповідь.

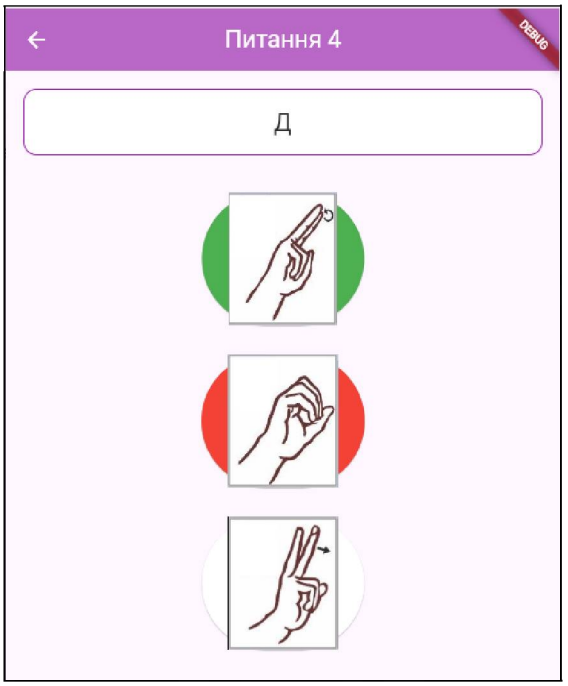


Рисунок В.21 – Обрано неправильну відповідь.

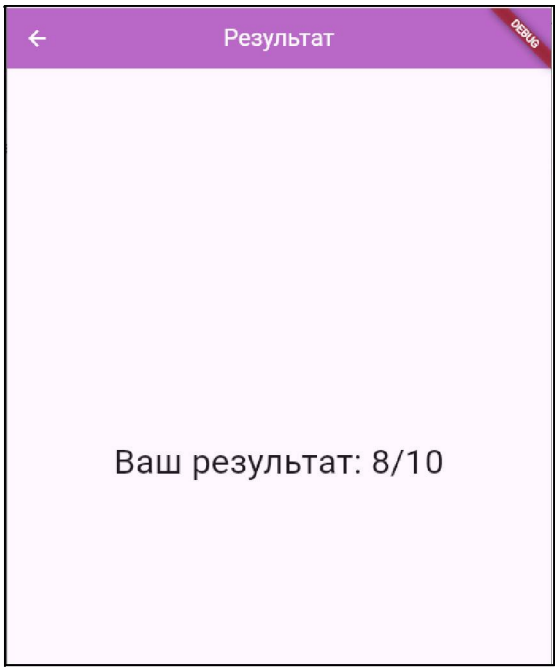


Рисунок В.22 – Результат тестування.

Висновок: за час тестування усі функції працювали коректно, отже, перевірку можна вважати успішною.

Виконавець: студент групи КУ-41, Карпенко К.Р. _____
(підпис)

Додаток Г

Код створення навігації

```
import 'package:flutter/material.dart';
import 'package:teaching_the_manual_components_of_sign_language/navigation/menu/presentation/pages/practice_menu_screen.dart';
import '../widgets/menu_button.dart';
import 'theory_menu_screen.dart';

class MainMenuScreen extends StatelessWidget {
 const MainMenuScreen({super.key});

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: const Text("Головне меню",
    style: TextStyle(color: Colors.white)
    ),
    backgroundColor: Colors.purple[300],
    centerTitle: true,
   ),
   body: Padding(
    padding: const EdgeInsets.all(16.0),
    child: Column(
     children: [

      const SizedBox(height: 30),

      MenuButton(
       title: "Теоретичний матеріал",
       onPressed: () {
```

```
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (_) => const TheoryMenuScreen(),
              ),
            );
          },
        ),

        const SizedBox(height: 30),

        MenuButton(
          title: "Практичні вправи",
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (_) => const PracticeMenuScreen(),
              ),
            );
          },
        ),
      ],
    ),
  ),
 );
 }
}
```

Лістинг Г.1 – Сторінка головного меню (MainMenuScreen).

```
import 'package:flutter/material.dart';
import '../../../../features/nouns/presentation/pages/nouns_screen.dart';
import '../../../../features/numbers/presentation/pages/numbers_screen.dart';
import '../../../../features/pronouns/presentation/pages/pronouns_screen.dart';
import '../../../../features/verbs/presentation/pages/verbs_screen.dart';
import '../widgets/menu_button.dart';

class NumbersMenuScreen extends StatelessWidget {
 const NumbersMenuScreen({super.key});

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: const Text("Числа"),
     centerTitle: true,
     backgroundColor: Colors.purple[300],
     foregroundColor: Colors.white
   ),
   body: Padding(
    padding: const EdgeInsets.all(16.0),
    child: Column(
     mainAxisAlignment: MainAxisAlignment.spaceEvenly,
     children: [
      MenuButton(
       title: "Від 0 до 20",
       onPressed: () {
        Navigator.push(
         context,
         MaterialPageRoute(
          builder: (_) => const NumbersScreen(
           category: "0_20",
```

```
          title: "Числа 0–20",
        ),
      ),
    );
  },
),
MenuButton(
  title: "Десятки",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const NumbersScreen(
          category: "tens",
          title: "Десятки",
        ),
      ),
    );
  },
),
MenuButton(
  title: "Сотні",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const NumbersScreen(
          category: "hundreds",
          title: "Сотні",
        ),
      ),
    );
```

```
    },
  ),
  MenuButton(
    title: "Тисячі",
    onPressed: () {
      Navigator.push(
        context,
        MaterialPageRoute(
          builder: (_) => const NumbersScreen(
            category: "thousands",
            title: "Тисячі",
          ),
        ),
      );
    },
  ),
  MenuButton(
    title: "Мільйони",
    onPressed: () {
      Navigator.push(
        context,
        MaterialPageRoute(
          builder: (_) => const NumbersScreen(
            category: "millions",
            title: "Мільйони",
          ),
        ),
      );
    },
  ),
 ],
),
```

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

Лістинг Г.2 – Сторінка вибору теми теоретичного розділу “Числа” (NumbersMenuScreen).

```
import 'package:flutter/material.dart';
import '../../../../features/exercises/domain/enums/exercise_type.dart';
import '../../../../features/exercises/presentation/pages/exercise_type_screen.dart';
import '../../../../features/exercises/presentation/pages/number_category_screen.dart';
import '../../../../features/exercises/presentation/pages/word_category_screen.dart';
import '../widgets/menu_button.dart';

class PracticeMenuScreen extends StatelessWidget {
 const PracticeMenuScreen({super.key});

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: const Text("Практичні вправи"),
    centerTitle: true,
    backgroundColor: Colors.purple[300],
    foregroundColor: Colors.white,
   ),
   body: Padding(
    padding: const EdgeInsets.all(16),
    child: Column(
     mainAxisAlignment: MainAxisAlignment.spaceEvenly,
     children: [
```

```
MenuButton(
  title: "Перевірка знань абетки",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => ExerciseTypeScreen(
          type: ExerciseType.alphabet,
          title: "Перевірка знань абетки",
          firstTitle: "Вибрати жест",
          secondTitle: "Ввести букву",
        ),
      ),
    );
  },
),

MenuButton(
  title: "Тести за числами",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const NumberCategoryScreen(),
      ),
    );
  },
),

MenuButton(
  title: "Тести за словами",
  onPressed: () {
```

```
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (_) => const WordCategoryScreen(),
              ),
            );
          },
        ),

        MenuButton(
          title: "Знання речень",
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (_) => ExerciseTypeScreen(
                  type: ExerciseType.sentences,
                  title: "Знання речень",
                  firstTitle: "Скласти речення",
                  secondTitle: "Розпізнати речення",
                ),
              ),
            );
          },
        ),
      ],
    ),
  ),
 );
 }
}
```

Лістинг Г.3 – Сторінка вибору теми тестування (PracticeMenuScreen).

```
import 'package:flutter/material.dart';
import '../../../../features/sentences/presentation/pages/sentences_screen.dart';
import '../widgets/menu_button.dart';

class SentencesMenuScreen extends StatelessWidget {
 const SentencesMenuScreen({super.key});

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: const Text("Базові речення"),
     centerTitle: true,
     backgroundColor: Colors.purple[300],
     foregroundColor: Colors.white
   ),
   body: Padding(
    padding: const EdgeInsets.all(16.0),
    child: Column(
     mainAxisAlignment: MainAxisAlignment.spaceEvenly,
     children: [
      MenuButton(
       title: "Як вас звати?",
       onPressed: () {
        Navigator.push(
         context,
         MaterialPageRoute(
          builder: (_) => const SentenceScreen(
           sentenceKey: "name_question"
          ),
         ),
        );
```

```
  },
),
MenuButton(
  title: "Мене звати ...",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const SentenceScreen(
          sentenceKey: "my_name_is"
        ),
      ),
    );
  },
),
MenuButton(
  title: "Скільки вам років?",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const SentenceScreen(
          sentenceKey: "how_old_are_you"
        ),
      ),
    );
  },
),
MenuButton(
  title: "Мені ... років",
  onPressed: () {
    Navigator.push(
```

```
            context,
            MaterialPageRoute(
              builder: (_) => const SentenceScreen(
                sentenceKey: "my_age"
              ),
            ),
          );
        },
      ),
      MenuButton(
        title: "Як справи?",
        onPressed: () {
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (_) => const SentenceScreen(
                sentenceKey: "how_are_you"
              ),
            ),
          );
        },
      ),
    ],
  ),
 ),
);
 }
}
```

Лістинг Г.4 – Сторінка вибору речення для навчання (SentencesMenuScreen).

```
import 'package:flutter/material.dart';
import 'package:teaching_the_manual_components_of_sign_language/navigation/menu/presentation/pages/sentences_menu_screen.dart';
import '../../../../features/alphabet/presentation/pages/alphabet_screen.dart';
import '../../../../features/resources/presentation/pages/resources_screen.dart';
import '../widgets/menu_button.dart';
import 'numbers_menu_screen.dart';
import 'words_menu_screen.dart';

class TheoryMenuScreen extends StatelessWidget {
  const TheoryMenuScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text("Теоретичний матеріал"),
        centerTitle: true,
          backgroundColor: Colors.purple[300],
          foregroundColor: Colors.white

      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            MenuButton(
              title: "Абетка",
```

```
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const AlphabetScreen(),
      ),
    );
  },
),
MenuButton(
  title: "Числа",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const NumbersMenuScreen(),
      ),
    );
  },
),
MenuButton(
  title: "Слова та відповідні символи",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const WordsMenuScreen(),
      ),
    );
  },
),
MenuButton(
```

```
            title: "Базові речення",
            onPressed: () {
              Navigator.push(
                context,
                MaterialPageRoute(
                  builder: (_) => const SentencesMenuScreen(),
                ),
              );
            },
          ),
          MenuButton(
            title: "Корисні матеріали",
            onPressed: () {
              Navigator.push(
                context,
                MaterialPageRoute(
                  builder: (_) => const ResourcesScreen(),
                ),
              );
            },
          ),
        ],
      ),
    ),
  );
 }
}
```

Лістинг Г.5 – Сторінка вибору теми для навчання (TheoryMenuScreen).

```
import 'package:flutter/material.dart';
import '../../../../features/adjective/presentation/pages/adjectives_screen.dart';
import '../../../../features/adverbs/presentation/pages/adverbs_screen.dart';
import '../../../../features/nouns/presentation/pages/nouns_screen.dart';
import '../../../../features/pronouns/presentation/pages/pronouns_screen.dart';
import '../../../../features/verbs/presentation/pages/verbs_screen.dart';
import '../widgets/menu_button.dart';

class WordsMenuScreen extends StatelessWidget {
 const WordsMenuScreen({super.key});

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: const Text("Слова та відповідні символи"),
     centerTitle: true,
     backgroundColor: Colors.purple[300],
     foregroundColor: Colors.white
   ),
   body: Padding(
    padding: const EdgeInsets.all(16.0),
    child: Column(
     mainAxisAlignment: MainAxisAlignment.spaceEvenly,
     children: [
      MenuButton(
       title: "Займенники",
       onPressed: () {
        Navigator.push(
         context,
         MaterialPageRoute(
```

```
        builder: (_) => const PronounsScreen(),
      ),
    );
  },
),
MenuButton(
  title: "Іменники",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const NounsScreen(),
      ),
    );
  },
),
MenuButton(
  title: "Дієслова",
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => const VerbsScreen(),
      ),
    );
  },
),
MenuButton(
  title: "Прикметники",
  onPressed: () {
    Navigator.push(
      context,
```

```
            MaterialPageRoute(
              builder: (_) => const AdjectivesScreen(),
            ),
          );
        },
      ),
      MenuButton(
        title: "Прислівники",
        onPressed: () {
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (_) => const AdverbsScreen(),
            ),
          );
        },
      ),
    ],
  ),
 ),
);
 }
}
```

Лістинг Г.6 – Сторінка вибору теми теоретичного розділу “Слова та відповідні символи” (WordsMenuScreen).

```
import 'package:flutter/material.dart';

class MenuButton extends StatelessWidget {
 final String title;
 final VoidCallback onPressed;

 const MenuButton({
  super.key,
  required this.title,
  required this.onPressed,
 });

 @override
 Widget build(BuildContext context) {
  return Padding(
   padding: const EdgeInsets.symmetric(vertical: 8.0),
   child: ElevatedButton(
    onPressed: onPressed,
    style: ElevatedButton.styleFrom(
     minimumSize: const Size(double.infinity, 60),
     textStyle: const TextStyle(fontSize: 20),
    ),
    child: Text(title,
      style: TextStyle(color: Colors.purple[300])
    ),
   ),
  );
 }
}
```

Лістинг Г.7 – Віджет (кнопка) MenuButton.

Код модуля alphabet

```
import '../../domain/entities/letter.dart';

abstract class AlphabetLocalDataSource {
  List<Letter> getAlphabet();
}

class AlphabetLocalDataSourceImpl implements AlphabetLocalDataSource {
  @override
  List<Letter> getAlphabet() {
    return List.generate(
      33,
        (index) => Letter(
        id: index,
        imagePath: 'assets/images/alphabet/$index.png',
      ),
    );
  }
}
```

Лістинг Г.8 – Лістинг файлу alphabet_local_datasource.dart.

```
import '../../domain/entities/letter.dart';
import '../../domain/repositories/alphabet_repository.dart';
import '../datasources/alphabet_local_datasource.dart';

class AlphabetRepositoryImpl implements AlphabetRepository {
  final AlphabetLocalDataSource datasource;

  AlphabetRepositoryImpl(this.datasource);

  @override
```

```
  List<Letter> getAlphabet() {
    return datasource.getAlphabet();
  }
}
```

Лістинг Г.9 – Лістинг файлу alphabet_repository_impl.

```
class Letter {
  final int id;
  final String imagePath;

  Letter({
    required this.id,
    required this.imagePath,
  });
}
```

Лістинг Г.10 – Лістинг файлу letter.dart.

```
import '../entities/letter.dart';

abstract class AlphabetRepository {
  List<Letter> getAlphabet();
}
```

Лістинг Г.11 – Лістинг файлу alphabet_repository.dart.

```
import '../entities/letter.dart';
import '../repositories/alphabet_repository.dart';

class GetAlphabet {
  final AlphabetRepository repository;

  GetAlphabet(this.repository);

  List<Letter> call() {
    return repository.getAlphabet();
```

```
  }
}
```

Лістинг Г.12 – Лістинг файлу get_alphabet.dart.

```
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import '../providers/alphabet_provider.dart';

class AlphabetScreen extends ConsumerWidget {
  const AlphabetScreen({super.key});

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    final alphabet = ref.watch(alphabetProvider);

    return Scaffold(
      appBar: AppBar(
        title: const Text("Абетка"),
        centerTitle: true,
        backgroundColor: Colors.purple[300],
        foregroundColor: Colors.white
      ),
      body: Padding(
        padding: const EdgeInsets.all(8),
        child: GridView.builder(
          itemCount: alphabet.length,
          gridDelegate: const SliverGridDelegateWithFixedCrossAxisCount(
            crossAxisCount: 4,
            crossAxisSpacing: 8,
            mainAxisSpacing: 8,
            childAspectRatio: 168 / 200,
          ),
          itemBuilder: (context, index) {
```

```
          final letter = alphabet[index];

          return ClipRRect(
            borderRadius: BorderRadius.circular(8),
            child: Image.asset(
              letter.imagePath,
              fit: BoxFit.contain,
            ),
          );
        },
      ),
    ),
  );
 }
}
```

Лістинг Г.13 – Лістинг файлу alphabet_screen.dart.

```
import 'package:flutter_riverpod/flutter_riverpod.dart';
import '../../data/datasources/alphabet_local_datasource.dart';
import '../../data/repositories/alphabet_repository_impl.dart';
import '../../domain/entities/letter.dart';
import '../../domain/usecases/get_alphabet.dart';

final alphabetProvider = Provider<List<Letter>>((ref) {
 final datasource = AlphabetLocalDataSourceImpl();
 final repository = AlphabetRepositoryImpl(datasource);
 final getAlphabet = GetAlphabet(repository);

 return getAlphabet();
});
```

Лістинг Г.14 – Лістинг файлу alphabet_provider.dart.

Код модуля sentences

```
import '../../domain/entities/sentences.dart';

abstract class SentencesLocalDataSource {
  SentenceItem getSentence(String key);
}

class SentencesLocalDataSourceImpl implements SentencesLocalDataSource {
  final List<SentenceItem> sentences = [
    SentenceItem(
      key: "name_question",
      title: "Як вас звати?",
      description: "Використовується для запитання імені співрозмовника",
      images: [
        'assets/images/sentences/what_is_your_name/First.png',
        'assets/images/sentences/what_is_your_name/Second.png',
        'assets/images/sentences/what_is_your_name/Third.png'
      ],
    ),
    SentenceItem(
      key: "my_name_is",
      title: "Мене звати",
      description: "Після використання усих символів потрібно показати своє ім'я по буквам за дактильною абеткою",
      images: [
        'assets/images/sentences/my_name_is/First.png',
        'assets/images/sentences/my_name_is/Second.png',
        'assets/images/sentences/my_name_is/Third.png'
      ],
    ),
    SentenceItem(
```

```
      key: "how_old_are_you",
      title: "Скільки тобі років?",
      description: "При виконанні першого символа важливо зробити один оберт навколо вуха, рухаючись від низу до верху",
      images: [
        'assets/images/sentences/how_old_are_you/First.png'
      ],
    ),
    SentenceItem(
      key: "my_age",
      title: "Мені ... років",
      description: "При виконанні другого символа важливо зробити один оберт навколо вуха, рухаючись від низу до верху, а після показати цифру відповідаючу вашому віку",
      images: [
        'assets/images/sentences/my_age/First.png',
        'assets/images/sentences/my_age/Second.png'
      ],
    ),
    SentenceItem(
      key: "how_are_you",
      title: "Як справи?",
      description: "При виконанні другого символа важливо рухати пальцями за вказаними напрямками, а після зробити прямі долоні та провести їх уперед",
      images: [
        'assets/images/sentences/how_are_you/First.png',
        'assets/images/sentences/how_are_you/Second.png'
      ],
    ),
  ];

  @override
  SentenceItem getSentence(String key) {
    return sentences.firstWhere(
```

```
        (element) => element.key == key,
    );
  }
}
```

Лістинг Г.15 – Лістинг файлу sentences_local_datasource.dart.

```
import '../../domain/entities/sentences.dart';
import '../../domain/repositories/sentences_repository.dart';
import '../datasources/sentences_local_datasource.dart';

class SentencesRepositoryImpl implements SentencesRepository {
  final SentencesLocalDataSource datasource;

  SentencesRepositoryImpl(this.datasource);

  @override
  SentenceItem getSentence(String key) {
    return datasource.getSentence(key);
  }
}
```

Лістинг Г.16 – Лістинг файлу sentences_repository_impl.dart.

```
class SentenceItem {
  final String key;
  final String title;
  final String description;
  final List<String> images;

  SentenceItem({
    required this.key,
    required this.title,
    required this.description,
    required this.images,
```

```
  });
}
```

Лістинг Г.17 – Лістинг файлу sentences.dart.

```
import '../entities/sentences.dart';

abstract class SentencesRepository {
  SentenceItem getSentence(String key);
}
```

Лістинг Г.18 – Лістинг файлу sentences_repository.dart.

```
import '../entities/sentences.dart';
import '../repositories/sentences_repository.dart';

class GetSentence {
  final SentencesRepository repository;

  GetSentence(this.repository);

  SentenceItem call(String key) {
    return repository.getSentence(key);
  }
}
```

Лістинг Г.19 – Лістинг файлу get_sentences.dart.

```
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import '../providers/sentences_provider.dart';

class SentenceScreen extends ConsumerWidget {
  final String sentenceKey;

  const SentenceScreen({
    super.key,
```

```
  required this.sentenceKey,
});

@override
Widget build(BuildContext context, WidgetRef ref) {
  final sentence = ref.watch(sentenceProvider(sentenceKey));

  return Scaffold(
    appBar: AppBar(
      title: Text(sentence.title),
        centerTitle: true,
        backgroundColor: Colors.purple[300],
        foregroundColor: Colors.white
    ),
    body: Padding(
      padding: const EdgeInsets.all(10),
      child: ListView(
        children: [
          // Всі зображення
          ...sentence.images.map(
              (image) => Padding(
            padding: const EdgeInsets.only(bottom: 20),
            child: ClipRRect(
              borderRadius: BorderRadius.circular(12),
              child: Image.asset(
                image,
                height: 250,
                fit: BoxFit.contain,
              ),
            ),
          ),
        ),
```

```
          // Опис
          Container(
            padding: const EdgeInsets.all(12),
            decoration: BoxDecoration(
              color: Colors.grey[200],
              borderRadius: BorderRadius.circular(12),
              border: Border.all(color: Colors.purple),
            ),
            child: Text(sentence.description),
          ),
        ],
      ),
    ),
  );
 }
}
```

Лістинг Г.20 – Лістинг файлу sentences_screen.dart.

```
import 'package:flutter_riverpod/flutter_riverpod.dart';

import '../../data/datasources/sentences_local_datasource.dart';
import '../../data/repositories/sentences_repository_impl.dart';
import '../../domain/entities/sentences.dart';
import '../../domain/usecases/get_sentences.dart';

final sentenceProvider =
Provider.family<SentenceItem, String>((ref, key) {
 final datasource = SentencesLocalDataSourceImpl();
 final repository = SentencesRepositoryImpl(datasource);
 final usecase = GetSentence(repository);

 return usecase(key);
```

```
});
```

Лістинг Г.21 – Лістинг файлу sentences_provider.dart.

Код модуля resources

```
import '../../domain/entities/resource.dart';


class ResourceLocalDataSource {
  List<Resource> getResources() {
    return [
      Resource(
        title: 'Офіційний сайт українського товариства глухих ',
        url: 'https://utog.org/',
      ),
      Resource(
        title: 'Уроки жестової мови від НБУ ім. Ярослава Мудрого (для початківців)',
        url:
'https://youtube.com/playlist?list=PLdrm7ELIh2RzeoSmnxpfqRPlwrY1eElJt&si=xr4IG5-SPPdIUcI0',
      ),
      Resource(
        title: 'Сайт з функцією пошуку потрібного слова або речення',
        url: 'https://spreadthesign.com/uk.ua/search/',
      ),
      Resource(
        title: 'Слова за темою "Природа і природні явища" жестовою мовою від УТОГ',
        url:
'https://youtube.com/playlist?list=PLyA8pE-tvVH6MkhEwzoEgoU6t6n7jXOFF&si=637Ba11LC3AKuTOs',
      ),
      Resource(
        title: 'Слова за темою "Календар" жестовою мовою від УТОГ',
        url:
'https://youtube.com/playlist?list=PLyA8pE-tvVH62LaiGuzq056Qji_6ityU2&si=MQIKYMG4d0gYH678',
      ),
      Resource(
```

```
            title: 'Слова за темою "Фрукти й ягоди" жестовою мовою від УТОГ',
            url:
'https://youtube.com/playlist?list=PLyA8pE-tvVH62E1u4NguUFef4KMOpwvm-&si=NcUq4mX3swLpHnQp',
          ),
          Resource(
            title: 'Слова за темою "Овочі" жестовою мовою від УТОГ',
            url:
'https://youtube.com/playlist?list=PLyA8pE-tvVH7fhj93FONBxwn-__OfeSeV&si=ayMrqgS_yvy-4sma',
          ),
          Resource(
            title: 'Слова за темою "Соціум" жестовою мовою від УТОГ',
            url:
'https://youtube.com/playlist?list=PLyA8pE-tvVH5CnGxjSHeu1vRZvkF2a23B&si=aeG7mDIZpPmOebzp',
          ),
          Resource(
            title: 'Слова за темою "Родина" жестовою мовою від УТОГ',
            url:
'https://youtube.com/playlist?list=PLyA8pE-tvVH5frMxrJOkwt29m-8tVxjZa&si=HlIU2ybaKLeDDZFW',
          ),
          Resource(
            title: 'Слова за темою "Питання" жестовою мовою від УТОГ',
            url:
'https://youtube.com/playlist?list=PLyA8pE-tvVH63G-sRsg52y2Iew6J4ZsFZ&si=M-wp4wJXnKlAVIwN',
          ),
          Resource(
            title: 'Слова за темою "Займенники" жестовою мовою від УТОГ',
            url:
'https://youtube.com/playlist?list=PLyA8pE-tvVH45beYnFB8TOCe8ayvW5Y4A&si=Y8Hx2qUkQ-rOQvIU
',
          ),
        ];
      }
    }
```

Лістинг Г.22 – Лістинг файлу resources_local_datasource.dart.

```
import '../../domain/entities/resource.dart';
import '../../domain/repositories/resource_repository.dart';
import '../datasources/resources_local_datasource.dart';

class ResourceRepositoryImpl implements ResourceRepository {
  final ResourceLocalDataSource dataSource;

  ResourceRepositoryImpl(this.dataSource);

  @override
  List<Resource> getResources() {
    return dataSource.getResources();
  }
}
```

Лістинг Г.23 – Лістинг файлу resources_repository_impl.dart.

```
class Resource {
  final String title;
  final String url;

  Resource({
    required this.title,
    required this.url,
  });
}
```

Лістинг Г.24 – Лістинг файлу resource.dart.

```
import '../entities/resource.dart';

abstract class ResourceRepository {
  List<Resource> getResources();
}
```

Лістинг Г.25 – Лістинг файлу resource_repository.dart.

```
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:url_launcher/url_launcher.dart';
import '../../widgets/choose_button.dart';
import '../providers/resources_provider.dart';

class ResourcesScreen extends ConsumerWidget {
 const ResourcesScreen({super.key});

 Future<void> _openLink(String url) async {
  final uri = Uri.parse(url);

  if (!await launchUrl(uri)) {
   throw 'Не вдалося відкрити $url';
  }
 }

 @override
 Widget build(BuildContext context, WidgetRef ref) {
  final resources = ref.watch(resourceProvider);

  return Scaffold(
   appBar: AppBar(
    title: const Text('Корисні матеріали'),
    centerTitle: true,
    backgroundColor: Colors.purple[300],
    foregroundColor: Colors.white,
   ),
   body: Padding(
    padding: const EdgeInsets.all(16),
    child: ListView.builder(
     itemCount: resources.length,
```

```
        itemBuilder: (context, index) {
          final item = resources[index];

          return ChooseButton(
            text: item.title,
            onPressed: () => _openLink(item.url),
          );
        },
      ),
    ),
  );
 }
}
```

Лістинг Г.26 – Лістинг файлу resources_screen.dart.

```
import 'package:flutter_riverpod/flutter_riverpod.dart';
import '../../data/datasources/resources_local_datasource.dart';
import '../../data/repositories/resources_repository_impl.dart';
import '../../domain/entities/resource.dart';

final resourceProvider = Provider<List<Resource>>((ref) {
 final datasource = ResourceLocalDataSource();
 final repository = ResourceRepositoryImpl(datasource);

 return repository.getResources();
});
```

Лістинг Г.27 – Лістинг файлу resources_provider.dart.

```
import 'package:flutter/material.dart';

class ChooseButton extends StatelessWidget {
 final String text;
 final VoidCallback onPressed;
```

```
const ChooseButton({
  super.key,
  required this.text,
  required this.onPressed
});

@override
Widget build(BuildContext context) {
  return Padding(
    padding: const EdgeInsets.symmetric(vertical: 8),
    child: SizedBox(
      width: double.infinity,
      height: 50,
      child: ElevatedButton(
        style: ElevatedButton.styleFrom(
          backgroundColor: Colors.white,
        ),
        onPressed: onPressed,
        child: Text(
          text,
          style: TextStyle(
            fontSize: 18,
            color: Colors.purple[300],
          ),
          textAlign: TextAlign.center,
          maxLines: 2,
          overflow: TextOverflow.ellipsis,
        ),
      ),
    ),
  );
```

```
  }
}
```

Лістинг Г.28 – Лістинг файлу choose_button.dart.

Код модуля exercises

```
import '../../domain/enums/exercise_type.dart';

class ExerciseDescriptions {
  static String getDescription(ExerciseType type) {
    switch (type) {
      case ExerciseType.alphabet:
        return 'Перевірка знання абетки. Оберіть правильний варіант відповідно до символу або літери.';

      case ExerciseType.pronouns:
        return 'Тест на знання займенників. Визначте правильне слово або жест.';

      case ExerciseType.nouns:
        return 'Тест на знання іменників. Оберіть відповідне слово або символ.';

      case ExerciseType.verbs:
        return 'Тест на знання дієслів. Визначте правильну дію або її позначення.';

      case ExerciseType.adjectives:
        return 'Тест на знання прикметників. Оберіть правильну характеристику.';

      case ExerciseType.adverbs:
        return 'Тест на знання прислівників. Визначте спосіб дії.';

      case ExerciseType.sentences:
        return 'Перевірка знання речень. Оберіть правильний варіант або складіть речення.';

      case ExerciseType.numbers_0_20:
```

```
      return 'Перевірка знання числ від 0 до 20. Оберіть тип питань:';

    case ExerciseType.numbers_tens:
      return 'Перевірка знання десятків. Оберіть тип питань:';

    case ExerciseType.numbers_hundreds:
      return 'Перевірка знання сотень. Оберіть тип питань:';

    case ExerciseType.numbers_thousands:
      return 'Перевірка знання тисяч. Оберіть тип питань:';

    case ExerciseType.numbers_millions:
      return 'Перевірка знання мільйонів. Оберіть тип питань:';
  }
 }
}
```

Лістинг Г.29 – Лістинг файлу exercise_descriptions.dart.

```
import '../../domain/entities/question.dart';
import '../../domain/enums/exercise_mode.dart';
import '../../domain/enums/exercise_type.dart';

abstract class ExerciseLocalDataSource {
 List<Question> getQuestions(
    ExerciseType type,
    ExerciseMode mode,
    );
}
```

Лістинг Г.30 – Лістинг файлу exercise_local_datasource.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';
```

```
class AlphabetQuestions {
  static List<Question> getQuestions({int count = 10}) {
    return QuestionGenerator.generateAlphabet(count: count);
  }
}
```

Лістинг Г.31 – Лістинг файлу alphabet_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class HundredQuestions {
  static const _numbers = [
    '100','200','300','400','500','600','700','800','900',
  ];

  static List<Question> getQuestions() {
    return QuestionGenerator.generate(
      words: _numbers,
      basePath: 'assets/images/exercises/numbers_hundreds_exercise',
    );
  }
}
```

Лістинг Г.32 – Лістинг файлу hundred_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class MillionQuestions {
  static const _numbers = [
    // Від 1 000 000 до 20 000 000
    "1000000", "2000000", "3000000", "4000000", "5000000", "6000000", "7000000", "8000000",
"9000000", "10000000",
    "11000000", "12000000", "13000000", "14000000", "15000000", "16000000", "17000000",
"18000000", "19000000", "20000000",
```

```
    // Від 30 000 000 до 90 000 000 (десятки)
    "30000000", "40000000", "50000000", "60000000", "70000000", "80000000", "90000000",

    // Від 100 000 000 до 900 000 000 (сотні)
    "100000000", "200000000", "300000000", "400000000", "500000000", "600000000",
"700000000", "800000000", "900000000"
  ];

  static List<Question> getQuestions() {
    return QuestionGenerator.generate(
      words: _numbers,
      basePath: 'assets/images/exercises/numbers_millions_exercise',
    );
  }
}
```

Лістинг Г.33 – Лістинг файлу million_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class Numbers_0_20_Questions {
  static const _numbers = [
    '0','1','2','3','4','5','6','7','8','9',
    '10','11','12','13','14','15','16','17','18','19','20',
  ];

  static List<Question> getQuestions() {
    return QuestionGenerator.generate(
      words: _numbers,
      basePath: 'assets/images/exercises/numbers_0_20_exercise',
    );
  }
```

```
}
```

Лістинг Г.34 – Лістинг файлу numbers_0_20_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class TensQuestions {
  static const _numbers = [
    '10','20','30','40','50','60','70','80','90','100',
  ];

  static List<Question> getQuestions() {
    return QuestionGenerator.generate(
      words: _numbers,
      basePath: 'assets/images/exercises/numbers_tens_exercise',
    );
  }
}
```

Лістинг Г.35 – Лістинг файлу tens_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class ThousandQuestions {
  static const _numbers = [
    "1000", "2000", "3000", "4000", "5000", "6000", "7000", "8000", "9000", "10000",
    "11000", "12000", "13000", "14000", "15000", "16000", "17000", "18000", "19000", "20000",

    // Від 30 000 до 90 000 (десятки)
    "30000", "40000", "50000", "60000", "70000", "80000", "90000",

    // Від 100 000 до 900 000 (сотні)
    "100000", "200000", "300000", "400000", "500000", "600000", "700000", "800000", "900000"
```

```
  ];

  static List<Question> getQuestions() {
    return QuestionGenerator.generate(
      words: _numbers,
      basePath: 'assets/images/exercises/numbers_thousands_exercise',
    );
  }
}
```

Лістинг Г.36 – Лістинг файлу thousand_questions.dart.

```
class SentenceData {
  final String text;
  final String imageName;

  const SentenceData({
    required this.text,
    required this.imageName,
  });
}

String getImageNameByText(String text) {
  return sentences
      .firstWhere((e) => e.text == text)
      .imageName;
}

const sentences = [
  SentenceData(
    text: 'Як вас звати?',
    imageName: 'what_is_your_name',
  ),
  SentenceData(
```

```
    text: 'Мене звати ...',
    imageName: 'my_name_is',
  ),
  SentenceData(
    text: 'Як справи?',
    imageName: 'how_are_you',
  ),
  SentenceData(
    text: 'Скільки вам років?',
    imageName: 'how_old_are_you',
  ),
  SentenceData(
    text: 'Мені ... років',
    imageName: 'my_age',
  ),
];

final Map<String, String> sentenceImageMap = {
  for (var item in sentences) item.text: item.imageName,
};
```

Лістинг Г.37 – Лістинг файлу sentence_data.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';
import 'sentence_data.dart';

class SentenceQuestions {
  static List<Question> getQuestions({int count = 5}) {
    return QuestionGenerator.generateFromData(
      data: sentences,
      basePath: 'assets/images/exercises/sentences_exercise',
      count: count,
    );
```

```
  }
}
```

Лістинг Г.38 – Лістинг файлу sentence_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class AdjectiveQuestions {
  static List<Question> getQuestions() {
    const words = [
      'ВИСОКИЙ',
      'ГОЛОДНИЙ',
      'НИЗЬКИЙ',
    ];

    return QuestionGenerator.generate(
      words: words,
      basePath: 'assets/images/exercises/adjectives_exercise',
    );
  }
}
```

Лістинг Г.39 – Лістинг файлу adjective_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class AdverbQuestions {
  static List<Question> getQuestions() {
    const words = [
      'БУДЬ-ЛАСКА',
      'ВИСОКО',
      'ДОБРЕ',
      'ПОГАНО',
```

```
    ];

    return QuestionGenerator.generate(
      words: words,
      basePath: 'assets/images/exercises/adverbs_exercise',
    );
  }
}
```

Лістинг Г.40 – Лістинг файлу adverb_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class NounQuestions {
  static List<Question> getQuestions() {
    const words = [
      'ВОДА',
      'БАТЬКО',
      'БІБЛІОТЕКА',
      'ДІМ',
      'ЇЖА',
      'КНИГА',
      'МАТИ',
      'УКРАЇНА',
      'ХЛІБ',
      'ШКОЛА',
    ];

    return QuestionGenerator.generate(
      words: words,
      basePath: 'assets/images/exercises/nouns_exercise',
    );
  }
```

```
}
```

Лістинг Г.41 – Лістинг файлу noun_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class PronounQuestions {
  static List<Question> getQuestions() {
    const words = [
      'Я',
      'ВИ',
      'ВІН',
      'ВОНА',
      'МИ',
      'ВОНИ',
    ];

    return QuestionGenerator.generate(
      words: words,
      basePath: 'assets/images/exercises/pronouns_exercise',
    );
  }
}
```

Лістинг Г.42 – Лістинг файлу pronoun_questions.dart.

```
import '../../../domain/entities/question.dart';
import '../../../utils/question_generator.dart';

class VerbQuestions {
  static List<Question> getQuestions() {
    const words = [
      'ВИКОНАТИ',
      'ГОВОРИТИ',
```

```
    'ДО КОГОСЬ',
    'РАХУВАТИ',
    'ЧИТАТИ',
   ];

   return QuestionGenerator.generate(
    words: words,
    basePath: 'assets/images/exercises/verbs_exercise',
   );
  }
}
```

Лістинг Г.43 – Лістинг файлу verb_questions.dart.

```
import '../../domain/entities/question.dart';
import '../../domain/enums/exercise_mode.dart';
import '../../domain/enums/exercise_type.dart';
import '../../domain/repositories/exercise_repository.dart';
import '../datasources/exercise_local_datasource.dart';

class ExerciseRepositoryImpl implements ExerciseRepository {
 final ExerciseLocalDataSource datasource;

 ExerciseRepositoryImpl(this.datasource);

 @override
 List<Question> getQuestions(
   ExerciseType type,
   ExerciseMode mode,
   ) {
  return datasource.getQuestions(type, mode);
 }
}
```

Лістинг Г.44 – Лістинг файлу exercise_repository_impl.dart.

```
import '../../domain/entities/question.dart';
import '../../domain/enums/exercise_mode.dart';
import '../../domain/enums/exercise_type.dart';
import '../datasources/exercise_local_datasource.dart';
import '../models/numbers/hundred_questions.dart';
import '../models/numbers/million_questions.dart';
import '../models/numbers/numbers_0_20_questions.dart';
import '../models/numbers/tens_questions.dart';
import '../models/numbers/thousand_questions.dart';
import '../models/words/adjective_questions.dart';
import '../models/words/adverb_questions.dart';
import '../models/alphabet/alphabet_questions.dart';
import '../models/words/noun_questions.dart';
import '../models/words/pronoun_questions.dart';
import '../models/sentences/sentence_questions.dart';
import '../models/words/verb_questions.dart';

class ExerciseLocalDataSourceImpl implements ExerciseLocalDataSource {
 @override
 List<Question> getQuestions(
    ExerciseType type,
    ExerciseMode mode,
    ) {
   switch (type) {
    case ExerciseType.alphabet:
      return AlphabetQuestions.getQuestions();

    case ExerciseType.pronouns:
      return PronounQuestions.getQuestions();

    case ExerciseType.nouns:
      return NounQuestions.getQuestions();
```

```
      case ExerciseType.verbs:
        return VerbQuestions.getQuestions();

      case ExerciseType.adjectives:
        return AdjectiveQuestions.getQuestions();

      case ExerciseType.adverbs:
        return AdverbQuestions.getQuestions();

      case ExerciseType.sentences:
        return SentenceQuestions.getQuestions();

      case ExerciseType.numbers_0_20:
        return Numbers_0_20_Questions.getQuestions();

      case ExerciseType.numbers_tens:
        return TensQuestions.getQuestions();

      case ExerciseType.numbers_hundreds:
        return HundredQuestions.getQuestions();

      case ExerciseType.numbers_thousands:
        return ThousandQuestions.getQuestions();

      case ExerciseType.numbers_millions:
        return MillionQuestions.getQuestions();
    }
  }
}
```

Лістинг Г.45 – Лістинг файлу exercise_repository_local_impl.dart.

```
import 'question.dart';

class ExerciseState {
  final List<Question> questions;
  final int currentIndex;
  final int score;
  final String? selectedAnswer;

  ExerciseState({
    required this.questions,
    required this.currentIndex,
    required this.score,
    this.selectedAnswer,
  });

  bool get isFinished => currentIndex >= questions.length;

  ExerciseState copyWith({
    List<Question>? questions,
    int? currentIndex,
    int? score,
    String? selectedAnswer,
  }) {
    return ExerciseState(
      questions: questions ?? this.questions,
      currentIndex: currentIndex ?? this.currentIndex,
      score: score ?? this.score,
      selectedAnswer: selectedAnswer,
    );
  }
}
```

Лістинг Г.46 – Лістинг файлу exercise_state.dart.

```
class Question {
  final String questionText;
  final String correctAnswer;
  final List<String> options;
  final String? imagePath;

  Question({
    required this.questionText,
    required this.correctAnswer,
    required this.options,
    this.imagePath,
  });
}
```

Лістинг Г.47 – Лістинг файлу question.dart.

```
enum ExerciseMode {
  symbolToLetter, // показуємо жест → треба вибрати букву
  letterToSymbol, // показуємо слово → треба вибрати жест
}
```

Лістинг Г.48 – Лістинг файлу exercise_mode.dart.

```
enum ExerciseType {
  alphabet,

  // слова
  pronouns,
  nouns,
  verbs,
  adjectives,
  adverbs,

  // числа
  numbers_0_20,
```

```
  numbers_tens,
  numbers_hundreds,
  numbers_thousands,
  numbers_millions,

  // речення
  sentences,
}
```

Лістинг Г.49 – Лістинг файлу exercise_type.dart.

```
import '../entities/question.dart';
import '../enums/exercise_mode.dart';
import '../enums/exercise_type.dart';

abstract class ExerciseRepository {
  List<Question> getQuestions(
      ExerciseType type,
      ExerciseMode mode,
      );
}
```

Лістинг Г.50 – Лістинг файлу exercise_repository.dart.

```
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import '../../domain/enums/exercise_mode.dart';
import '../../domain/enums/exercise_type.dart';
import '../../utils/image_path_helper.dart';
import '../providers/exercise_provider.dart';
import '../widgets/image_option_button.dart';
import '../widgets/text_option_button.dart';

class ExerciseScreen extends ConsumerWidget {
  final ExerciseType type;
```

```
final ExerciseMode mode;

const ExerciseScreen({
  super.key,
  required this.type,
  required this.mode,
});

@override
Widget build(BuildContext context, WidgetRef ref) {
  final state = ref.watch(
    exerciseProvider((type: type, mode: mode)),
  );

  final notifier = ref.read(
    exerciseProvider((type: type, mode: mode)).notifier,
  );

  if (state.isFinished) {
    return Scaffold(
      appBar: AppBar(title: const Text("Результат"),
        centerTitle: true,
        backgroundColor: Colors.purple[300],
        foregroundColor: Colors.white,
      ),
      body: Center(
        child: Text(
          "Ваш результат: ${state.score}/${state.questions.length}",
          style: const TextStyle(fontSize: 32),
        ),
      ),
    );
```

```
}

final question = state.questions[state.currentIndex];

return Scaffold(
 appBar: AppBar(
  title: Text("Питання ${state.currentIndex + 1}"),
  centerTitle: true,
  backgroundColor: Colors.purple[300],
  foregroundColor: Colors.white,
 ),
 body: Center(
  child: Padding(
   padding: const EdgeInsets.all(16),
   child: Column(

    children: [

     /// TEXT QUESTION
     if (mode == ExerciseMode.symbolToLetter)
      Container(
       height: 60,
       width: double.infinity,
       padding: const EdgeInsets.all(12),
       margin: const EdgeInsets.only(bottom: 20),
       decoration: BoxDecoration(
        color: Colors.white,
        borderRadius: BorderRadius.circular(12),
        border: Border.all(color: Colors.purple),
       ),
       child: Center(
        child: Text(
```

```
          question.questionText,
          style: TextStyle(
            fontSize: 22,
            color: Colors.grey[850]
          ),
          textAlign: TextAlign.center,
        ),
      ),
    ),

/// IMAGE QUESTION
if (mode == ExerciseMode.letterToSymbol &&
    question.imagePath != null)
  Image.asset(
    question.imagePath!,
    height: 180,
  ),

/// OPTIONS
...question.options.map((option) {
  final isSelected = state.selectedAnswer == option;
  final isCorrect = option == question.correctAnswer;

  Color? color;

  if (state.selectedAnswer != null) {
    if (isCorrect) {
      color = Colors.green;
    } else if (isSelected) {
      color = Colors.red;
    }
  }
```

```
            return Padding(
              padding: const EdgeInsets.symmetric(vertical: 5),

              child: mode == ExerciseMode.symbolToLetter
              /// КНОПКИ-КАРТИНКИ
                ? ImageOptionButton(
              imagePath: getImagePath(type: type, value: option),
              onPressed: () => notifier.answer(option),
              color: color,
            )

              /// КНОПКИ З ТЕКСТОМ
                : TextOptionButton(
              text: option,
              onPressed: () => notifier.answer(option),
              color: color,
            ),
          );
        }),
      ],
    ),
  ),
),
);
}
}
```

Лістинг Г.51 – Лістинг файлу exercise_screen.dart.

```
import 'package:flutter/material.dart';
import '../../data/datasources/exercise_descriptions.dart';
import '../../../../navigation/menu/presentation/widgets/menu_button.dart';
import '../../domain/enums/exercise_mode.dart';
import '../../domain/enums/exercise_type.dart';
import 'exercise_screen.dart';

class ExerciseTypeScreen extends StatelessWidget {
 final String title;
 final String firstTitle;
 final String secondTitle;
 final ExerciseType type;

 const ExerciseTypeScreen({
  super.key,
  required this.title,
  required this.firstTitle,
  required this.secondTitle,
  required this.type,
 });

 void _navigate(BuildContext context, ExerciseMode mode) {
  Navigator.push(
   context,
   MaterialPageRoute(
    builder: (_) => ExerciseScreen(
     type: type,
     mode: mode,
    ),
   ),
  );
 }
```

```
@override
Widget build(BuildContext context) {
 return Scaffold(
  appBar: AppBar(
   title: Text(title),
   centerTitle: true,
   backgroundColor: Colors.purple[300],
   foregroundColor: Colors.white,
  ),
  body: Padding(
   padding: const EdgeInsets.all(16),
   child: Column(
    children: [
     Container(
      height: 80,
      width: double.infinity,
      padding: const EdgeInsets.all(12),
      margin: const EdgeInsets.only(bottom: 20),
      decoration: BoxDecoration(
       color: Colors.white,
       borderRadius: BorderRadius.circular(12),
       border: Border.all(color: Colors.purple),
      ),
      child: Text(
       ExerciseDescriptions.getDescription(type),
       style: TextStyle(
        fontSize: 16,
        color: Colors.purple,
       ),
       textAlign: TextAlign.center,
      ),
```

```
          ),

          const SizedBox(height: 30),

          MenuButton(
            title: firstTitle,
            onPressed: () => _navigate(
              context,
              ExerciseMode.symbolToLetter,
            ),
          ),

          const SizedBox(height: 30),

          MenuButton(
            title: secondTitle,
            onPressed: () => _navigate(
              context,
              ExerciseMode.letterToSymbol,
            ),
          ),

          const Spacer(),
        ],
      ),
    ),
  );
 }
}
```

Лістинг Г.52 – Лістинг файлу exercise_type_screen.dart.

```
import 'package:flutter/material.dart';
import '../../../../navigation/menu/presentation/widgets/menu_button.dart';
import '../../domain/enums/exercise_type.dart';
import 'exercise_type_screen.dart';

class NumberCategoryScreen extends StatelessWidget {
 const NumberCategoryScreen({super.key});

 void _navigate(BuildContext context, String title, ExerciseType type) {
  Navigator.push(
   context,
   MaterialPageRoute(
    builder: (_) => ExerciseTypeScreen(
     type: type,
     title: title,
     firstTitle: "Число у вигляді тексту. Відповідь у форматі: жеста",
     secondTitle: "Число у вигляді жеста. Відповіді у форматі: тексту",
    ),
   ),
  );
 }

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: const Text("Тести за числами"),
    centerTitle: true,
    backgroundColor: Colors.purple[300],
    foregroundColor: Colors.white,
   ),
   body: Padding(
```

```
padding: const EdgeInsets.all(16),
child: Column(
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,
 children: [
  MenuButton(
   title: "0 - 20",
   onPressed: () => _navigate(
    context,
    "Числа 0-20",
    ExerciseType.numbers_0_20,
   ),
  ),
  MenuButton(
   title: "Десятки",
   onPressed: () => _navigate(
    context,
    "Десятки",
    ExerciseType.numbers_tens,
   ),
  ),
  MenuButton(
   title: "Сотні",
   onPressed: () => _navigate(
    context,
    "Сотні",
    ExerciseType.numbers_hundreds,
   ),
  ),
  MenuButton(
   title: "Тисячі",
   onPressed: () => _navigate(
    context,
```

```
            "Тисячі",
            ExerciseType.numbers_thousands,
          ),
        ),
        MenuButton(
          title: "Мільйони",
          onPressed: () => _navigate(
            context,
            "Мільйони",
            ExerciseType.numbers_millions,
          ),
        ),
      ],
    ),
  ),
 );
 }
}
```

Лістинг Г.53 – Лістинг файлу number_category_screen.dart.

```
import 'package:flutter/material.dart';
import '../../../../navigation/menu/presentation/widgets/menu_button.dart';
import '../../domain/enums/exercise_type.dart';
import 'exercise_type_screen.dart';

class WordCategoryScreen extends StatelessWidget {
 const WordCategoryScreen({super.key});

 void _navigate(BuildContext context, ExerciseType type, String title) {
  Navigator.push(
   context,
   MaterialPageRoute(
    builder: (_) => ExerciseTypeScreen(
```

```
        type: type,
        title: title,
          firstTitle: "Слово у вигляді тексту. Відповідь у форматі: жеста",
          secondTitle: "Слово у вигляді жеста. Відповіді у форматі: тексту"
      ),
    ),
  );
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text("Тести за словами"),
      centerTitle: true,
      backgroundColor: Colors.purple[300],
      foregroundColor: Colors.white,
    ),
    body: Padding(
      padding: const EdgeInsets.all(16),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.spaceEvenly,
        children: [
          MenuButton(
            title: "Займенники",
            onPressed: () => _navigate(context, ExerciseType.pronouns, "Займенники"),
          ),
          MenuButton(
            title: "Іменники",
            onPressed: () => _navigate(context, ExerciseType.nouns, "Іменники"),
          ),
          MenuButton(
```

```
            title: "Дієслова",
            onPressed: () => _navigate(context, ExerciseType.verbs, "Дієслова"),
          ),
          MenuButton(
            title: "Прикметники",
            onPressed: () => _navigate(context, ExerciseType.adjectives, "Прикметники"),
          ),
          MenuButton(
            title: "Прислівники",
            onPressed: () => _navigate(context, ExerciseType.adverbs, "Прислівники"),
          ),
        ],
      ),
    ),
  );
 }
}
```

Лістинг Г.54 – Лістинг файлу word_category_screen.dart.

```
import 'package:flutter_riverpod/flutter_riverpod.dart';
import '../../domain/entities/exercise_state.dart';
import '../../domain/entities/question.dart';

class ExerciseNotifier extends StateNotifier<ExerciseState> {
 ExerciseNotifier(List<Question> questions)
    : super(ExerciseState(
  questions: questions,
  currentIndex: 0,
  score: 0,
 ));

 void answer(String answer) {
  final currentQuestion = state.questions[state.currentIndex];
```

```
    state = state.copyWith(selectedAnswer: answer);

    Future.delayed(const Duration(seconds: 1), () {
      final isCorrect = answer == currentQuestion.correctAnswer;

      state = state.copyWith(
        currentIndex: state.currentIndex + 1,
        score: isCorrect ? state.score + 1 : state.score,
        selectedAnswer: null, // скидаємо
      );
    });
  }
}
```

Лістинг Г.55 – Лістинг файлу exercise_notifier.dart.

```
import 'package:flutter_riverpod/flutter_riverpod.dart';
import '../../data/repositories/exercise_repository_impl.dart';
import '../../data/repositories/exercise_repository_local_impl.dart';
import '../../domain/entities/exercise_state.dart';
import '../../domain/enums/exercise_mode.dart';
import '../../domain/enums/exercise_type.dart';
import 'exercise_notifier.dart';

final exerciseProvider = StateNotifierProvider.autoDispose.family<
    ExerciseNotifier,
    ExerciseState,
    ({ExerciseType type, ExerciseMode mode})>((ref, params) {

  final datasource = ExerciseLocalDataSourceImpl();
  final repository = ExerciseRepositoryImpl(datasource);

  final questions = repository.getQuestions(
```

```
    params.type,
    params.mode,
  );

  return ExerciseNotifier(questions);
});
```

Лістинг Г.56 – Лістинг файлу exercise_provider.dart.

```
import 'package:flutter/material.dart';

class ImageOptionButton extends StatelessWidget {
  final String imagePath;
  final VoidCallback onPressed;
  final Color? color;

  const ImageOptionButton({
    super.key,
    required this.imagePath,
    required this.onPressed,
    this.color,
  });

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.symmetric(vertical: 8),
      child: ElevatedButton(
        style: ElevatedButton.styleFrom(
          fixedSize: const Size(double.infinity, 120),
          backgroundColor: color ?? Colors.white,
        ),
        onPressed: onPressed,
        child: Image.asset(imagePath),
```

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

Лістинг Г.57 – Лістинг файлу image_option_button.dart.

```
import 'package:flutter/material.dart';

class TextOptionButton extends StatelessWidget {
 final String text;
 final VoidCallback onPressed;
 final Color? color;

 const TextOptionButton({
  super.key,
  required this.text,
  required this.onPressed,
  this.color,
 });

 @override
 Widget build(BuildContext context) {
  return Padding(
   padding: const EdgeInsets.symmetric(vertical: 8),
   child: SizedBox(
    width: double.infinity,
    height: 50,
    child: ElevatedButton(
     style: ElevatedButton.styleFrom(
      backgroundColor: color ?? Colors.white,
     ),
     onPressed: onPressed,
     child: Text(
```

```
          text,
          style: TextStyle(
            fontSize: 18,
            color: color != null ? Colors.white : Colors.purple[300],
          ),
          textAlign: TextAlign.center,
          maxLines: 2,
          overflow: TextOverflow.ellipsis,
        ),
      ),
    ),
  );
 }
}
```

Лістинг Г.58 – Лістинг файлу text_option_button.dart.