ДОДАТОК А

Специфікації програмного забезпечення

Software Requirements Specification

for

ReadEasy

Version 1.0 approved

Prepared by Височин Ілля

здобувач ПЗПІ-21-7

07.04.2025

1. Introduction

1.1 Purpose

Цей документ є специфікацією програмних вимог (SRS) до мобільного додатка "ReadEasy", версія 1.0, який розробляється як самостійний програмний продукт. Додаток призначений для читання електронних книг із підтримкою автоматичного озвучування тексту та перекладу окремих слів у реальному часі.

Метою даного програмного забезпечення є покращення доступності літератури для широкого кола користувачів, включно з людьми з порушенням зору або тими, хто вивчає іноземні мови. Додаток надаватиме можливості перегляду електронних книг у різних форматах, персоналізації інтерфейсу, голосового озвучення та перекладу текстових фрагментів без виходу з книги.

Ця специфікація охоплює повний функціональний обсяг першої версії додатка , включаючи основні та додаткові можливості, які будуть реалізовані в межах початкового релізу. Документ описує як функціональні, так і нефункціональні вимоги до системи, а також зовнішні інтерфейси та обмеження проекту.

1.2 Intended Audience and Reading Suggestions

Цей документ призначений для широкого кола осіб, задіяних у розробці, тестуванні, використанні та супроводі мобільного програмного додатка для читання електронних книг з функціями озвучування та перекладу тексту. Основними категоріями аудиторії є:

Розробники (frontend) — використовуватимуть специфікацію для розробки програмного забезпечення відповідно до функціональних та нефункціональних вимог, описаних у розділах 2–5.

UI/UX дизайнери — ознайомляться з описом користувацьких класів, функціональних сценаріїв та інтерфейсів для створення інтуїтивного, зручного і доступного дизайну.

Тестувальники — використовуватимуть опис функціональних і нефункціональних вимог для підготовки тест-кейсів та проведення тестування функціоналу та стабільності системи.

Керівники проектів і бізнес-аналітики — використовуватимуть документ для контролю за дотриманням вимог замовника, розробки плану проекту та формування критеріїв приймання.

Користувачі — хоча кінцеві користувачі зазвичай не читають SRS-документацію, окремі підрозділи (наприклад, опис функціоналу або інтерфейсів) можуть бути корисними для складання інструкцій користувача.

Технічні автори — використовуватимуть цей документ як основу для створення довідкових матеріалів, опису функцій додатку, навчальних посібників тощо.

Документ організовано так, щоб спочатку ознайомити читача з призначенням і контекстом проєкту (розділ 1), надати загальний опис системи (розділ 2), після чого переходити до детального опису функціональних можливостей (розділ 3), зовнішніх інтерфейсів (розділ 4) та нефункціональних вимог (розділ 5). Додаткові технічні деталі та глосарій наведені в додатках.

Рекомендована послідовність читання:

- Для загального ознайомлення: розділи 1 і 2.
- Для розробників: розділи 2, 3 і 4.
- Для тестувальників: розділи 3 і 5.
- Для дизайнерів: розділи 2.3, 4.1 і додатки.
- Для менеджерів: розділи 1.4, 2.2, 2.7, 5

1.3 Project Scope

Цей документ визначає вимоги до розробки мобільного кросплатформеного програмного додатка для читання електронних книг із функціями озвучування та перекладу тексту. Додаток створюється з метою покращення зручності доступу до книжкового контенту, полегшення вивчення іноземних мов та забезпечення інклюзивності для користувачів із порушенням зору.

Програмне забезпечення дозволятиме:

- завантажувати та відкривати книги у різних популярних форматах (TXT, FB2, EPUB, MOBI);
- озвучувати текст у реальному часі з можливістю вибору голосу, швидкості та тональності;
- перекладати окремі слова або фрагменти тексту в інтерактивному режимі;
- персоналізувати інтерфейс читання відповідно до потреб користувача;
- зберігати прогрес читання та надавати статистику активності.

Очікувані переваги системи:

- підвищення доступності літератури для широкої аудиторії;
- інтеграція в процес навчання іноземним мовам;
- зручність користування навіть у складних умовах (у дорозі, при порушеннях зору);
- унікальне поєднання озвучування, перекладу та читання в одному інтерфейсі.

Цілі проекту:

- створення стабільного та функціонального додатка з підтримкою основних платформ (на першому етапі – Android та неповна функціональність на IOS);
- впровадження технологій машинного навчання для покращення якості TTS та перекладу;
- забезпечення відповідності вимогам щодо безпеки, продуктивності та масштабованості.

Проект відповідає поточній стратегії цифрової трансформації та розвитку освітніх та інклюзивних технологій, спрямованих на створення комфортного, адаптивного та багатофункціонального інформаційного середовища. У перспективі можливе розширення функціональності, мультиплатформенність та комерційна експансія через преміум-функції та партнерські інтеграції.

2. Overall Description

2.1 Product Perspective

Розроблюваний програмний продукт є новим, самостійним мобільним додатком, що не входить до складу існуючих програмних лінійок, але потенційно може інтегруватися з зовнішніми платформами (хмарні сервіси, онлайн-бібліотеки, сервіси озвучення, платіжні системи). Додаток створюється з нуля на базі сучасних технологій (React Native, TypeScript, ML-інструменти для TTS та перекладу), із урахуванням потреб декількох типів користувачів: студентів, викладачів, людей з вадами зору, поліглотів та загального читача.

Додаток не є продовженням або заміною існуючих продуктів, але створюється як альтернатива популярним додаткам на ринку, які не поєднують усіх потрібних функцій в одному рішенні. Таким чином, розробка спрямована на усунення обмежень, притаманних конкурентам: відсутність персоналізації, уривчаста підтримка перекладу або озвучення.

Зовнішні інтерфейси, з якими взаємодіє система:

- TTS-сервіси (Google Text-to-Speech, Amazon Polly) — для озвучення тексту.
- Хмарні перекладачі (Google Translate API, DeepL API) — для перекладу виділеного фрагмента.
- Хмарні сховища (Google Drive) — для завантаження книг.
- Платіжні шлюзи (Google Pay) — для реалізації преміум-доступу.

Основні модулі системи:

- Інтерфейс користувача — реалізований через React Native UI.
- Модуль озвучення — інтегрується з ML-TTS сервісами.
- Модуль перекладу — обробляє запити до API та виводить результати.
- Менеджер книг — відповідає за збереження, відображення та структуру книг.
- Аналітика та налаштування — збір статистики, прогресу та індивідуальні параметри.

Додаток проектується як масштабований, з перспективою доповнення функціональності в наступних релізах, включаючи монетизацію, різні мови інтерфейсу, підтримку голосових команд та інтеграцію з освітніми платформами (Coursera, Duolingo).

2.2 Product Features

Мобільний додаток для читання електронних книг із функціями озвучення та перекладу надає користувачам інноваційний, адаптивний та доступний інструмент для роботи з текстовим контентом. Основні функціональні можливості продукту згруповані за логічними блоками:

1) Робота з електронними книгами
   - Завантаження книг з пам’яті пристрою або хмарних сховищ (Google Drive, Dropbox).
   - Підтримка популярних форматів: TXT, EPUB, FB2, MOBI.
   - Відображення тексту з налаштуванням шрифтів, розміру, кольорових тем, режимів (денний/нічний).
2) Озвучення тексту
   - Синтез мовлення (TTS) на основі ML-моделей.
   - Вибір голосу, мови, швидкості та тональності озвучення.
   - Озвучення як окремих слів, так і абзаців чи всієї книги.
3) Миттєвий переклад
   - Виділення тексту для перекладу в реальному часі.
   - Підтримка кількох мов (у тому числі української, англійської, німецької, французької тощо).
   - Збереження перекладених слів до особистого словника.
4) Інклюзивність
   - Озвучення для користувачів із порушеннями зору.
   - Великий шрифт, контрастні теми, доступність клавіш управління.
   - Робота без підключення до інтернету (офлайн-режим для вже завантажених книг).

5) Персоналізація та статистика

- Збереження прогресу читання, останньої прочитаної сторінки.
- Ведення статистики: кількість прочитаних сторінок, час використання.
- Індивідуальні налаштування інтерфейсу, словника та бібліотеки.

6) Монетизація та преміум-функції (у перспективі)

- Можливість підписки на додаткові функції (наприклад, преміум-голоси).
- Інтеграція з платіжними сервісами для транзакцій (Google Pay, Stripe).
- Показ реклами у безкоштовній версії (опціонально).

2.3 User Classes and Characteristics

Програмний продукт орієнтований на широке коло користувачів із різними цілями, технічним рівнем підготовки та потребами. Основні класи користувачів виділені на основі аналізу функціональних сценаріїв, мотивацій та особливостей використання.

1) Студенти та молодь, що навчається

Ціль: читання книжок іноземною мовою, покращення словникового запасу, вивчення вимови.

Частота використання: висока.

Функціонал: переклад слів, озвучення фрагментів, збереження прогресу, робота з інтерактивним словником.

Рівень підготовки: середній, впевнено користуються мобільними додатками.

Особливі вимоги: простий інтерфейс, доступ до офлайн-режиму, статистика прогресу.

2) Викладачі та освітяни

Ціль: використання додатку для підготовки матеріалів, практики з учнями, демонстрації текстів і аудіо.

Частота використання: середня.

Функціонал: точне озвучення тексту, підготовка фрагментів, збереження уривків, зміна голосу.

Рівень підготовки: високий рівень володіння матеріалом, середній технічний рівень.

Особливі вимоги: контрольовані параметри TTS, експортування уривків, стабільність.

3) Професіонали та IT-фахівці

Ціль: читання технічної літератури, переклад термінів, озвучення у фоновому режимі.

Частота використання: нерегулярна, але тривала.

Функціонал: переклад великих фрагментів, робота з PDF, озвучення під час роботи.

Рівень підготовки: високий.

Особливі вимоги: швидкодія, точність перекладу, зручність у багатозадачності.

4) Люди з порушеннями зору

Ціль: прослуховування книжок, доступ до інформації в голосовому форматі.

Частота використання: висока.

Функціонал: TTS, голосове управління, мінімум візуального інтерфейсу.

Рівень підготовки: від низького до середнього.

Особливі вимоги: адаптивність інтерфейсу, великі елементи, контрастні теми, доступність з клавіатури/жестів.

5) Поліглоти та мовні ентузіасти

Ціль: читання мовою оригіналу, переклад складних речень, поповнення словника.

Частота використання: регулярна.

Функціонал: багатомовний переклад, збереження лексики, історія перекладів.

Рівень підготовки: середній.

Особливі вимоги: точність, можливість порівняння варіантів перекладу, інтеграція з іншими мовними ресурсами.

Пріоритетні класи користувачів для першої версії додатку:

студенти та молодь (через обсяг і активність використання),

користувачі з вадами зору (через соціальну значимість і фокус на доступності).

Інші класи враховуються як перспективні напрями розвитку, особливо для розширення функціональності та комерціалізації продукту в наступних релізах.

2.4 Operating Environment

Мобільний додаток розробляється для функціонування в сучасному середовищі мобільних пристроїв і використовує актуальні технології для забезпечення стабільної та ефективної роботи. Нижче наведено опис середовища, в якому програмне забезпечення буде працювати:

1) Апаратна платформа

Смартфони та планшети.

Мінімальні технічні характеристики:

- Оперативна пам’ять: від 2 ГБ.
- Процесор: ARMv8-A (64-біт) або новіший.
- Дисплей: від 4.7 дюймів, підтримка сенсорного керування.
- Аудіовивід: динаміки або навушники для голосового відтворення.

2) Операційна система

Android:

- Мінімальна підтримувана версія: Android 8.0 (Oreo).
- Рекомендована версія: Android 10 і вище.

IOS:

- Мінімальна підтримувана версія: IOS 14.0.
- Рекомендована версія: IOS 13.0 і вище.

3) Програмне середовище

Розробка: React Native з використанням мови TypeScript.

Компіляція та тестування через Android Studio.

Сервіси:

- Google Text-to-Speech або інші TTS API.

- Google Translate API або DeepL API.

4) Системні залежності

Необхідність стабільного інтернет-з’єднання для:

- перекладу текстів через API,
- озвучення в реальному часі через API,
- завантаження книг з хмарних джерел.

Робота в офлайн-режимі можлива для книг, попередньо завантажених користувачем.

2.5 Design and Implementation Constraints

Розробка мобільного додатка для читання електронних книг із функціями озвучення та перекладу супроводжується низкою технічних, організаційних та функціональних обмежень, які впливають на архітектурні рішення, вибір технологій і реалізацію продукту:

1) Платформенні обмеження

- Перша версія додатка орієнтована на платформу Android (версія 8.0 і вище), ,більша частина функціоналу доступна буде на IOS.
- Враховується обмежена продуктивність пристроїв із малим об’ємом оперативної пам’яті (від 3 ГБ).

2) Вибір технологій

- Фреймворк: React Native, як кросплатформенне рішення.
- Мова програмування: TypeScript.
- Інструменти: Android Studio, Expo CLI, REST API.
- API для озвучення та перекладу: Google Text-to-Speech API, DeepL Translate API, що створює залежність від сторонніх сервісів.
- Усі ML-функції реалізуються як через сторонні сервіси, так і локально.

3) Обмеження апаратного середовища

- Програма має ефективно працювати на бюджетних пристроях, тому оптимізація швидкодії та використання пам’яті є критичною.

- Озвучення та переклад в реальному часі можуть вимагати суттєвих ресурсів, що потребує асинхронної обробки й оптимізації запитів.

4) Залежність від мережі

- Для кращого перекладу та синтезу мовлення необхідне стабільне підключення до Інтернету.
- У разі втрати з’єднання частина функцій має працювати в обмеженому режимі (наприклад, читання вже завантажених книг).

5) Безпека та конфіденційність

- Передбачено шифрування даних користувача при передачі на сервери сторонніх сервісів.
- Використання лише офіційних API для обробки текстів, що гарантує дотримання вимог безпеки Google/Apple.
- Персональні налаштування та прогрес зберігаються локально, без передачі в хмару (якщо не вказано інше).

6) Вимоги до інтерфейсу

- Інтерфейс має бути доступним для користувачів із порушеннями зору (контрастність, масштабованість, озвучення).
- Прийнято рішення дотримуватись принципів адаптивного та доступного дизайну (WCAG 2.1).

7) Інші обмеження

- Підтримка лише текстових форматів книг (EPUB, FB2, TXT, MOBI) без графічних або мультимедійних вкладень.
- Обмеження вільного функціоналу у безкоштовній версії програми, повний доступ – лише за підпискою.

2.6 Assumptions and Dependencies

У процесі розробки програмного забезпечення було зроблено низку припущень, які впливають на вимоги, описані в цьому документі. Також проєкт має зовнішні залежності, які необхідно враховувати на всіх етапах реалізації.

Припущення:

- Наявність стабільного інтернет-з’єднання у користувача для використання функцій перекладу та озвучення в реальному часі.
- Використання Android-пристроїв із підтримкою версії ОС не нижче 8.0 (Oreo) або IOS із підтримкою версії ОС не нижче 13.0 є достатнім для повноцінної роботи додатка .
- Доступність і стабільність роботи зовнішніх API, таких як DeepL Translate, Google Text-to-Speech, що забезпечують ключовий функціонал.
- Зовнішні хмарні сервіси не змінюють політику безкоштовного доступу або принципів API, що може вплинути на вартість експлуатації системи.
- Користувачі володіють базовими навичками використання мобільних додатків, що дозволяє спростити навчальні функції та інтерфейс.

Залежності:

- DeepL Translate API – для реалізації перекладу тексту в реальному часі.
- Google Text-to-Speech API – для озвучування фрагментів тексту.
- Моделі ML для цих сервісів – для функціоналу при відсутності мережі.
- Бібліотеки React Native та супутні плагіни – для кросплатформеної мобільної розробки.

Якщо одне з вищезазначених припущень буде порушено або зміниться статус зовнішніх компонентів, це може суттєво вплинути на функціональність, архітектуру або бізнес-модель програмного продукту. У такому разі вимоги потребуватимуть перегляду.

3. System Features

3.1 Функція читання електронної книги

3.1.1 Опис та пріоритет

Ця функція забезпечує основну можливість взаємодії користувача з текстовим контентом – відкривати, переглядати та налаштовувати електронні книги у зручному інтерфейсі.

Пріоритет: Високий

Benefit: 9 / Penalty: 8 / Cost: 4 / Risk: 2

3.1.2 Послідовність дій користувача та реакція системи

1) Користувач відкриває додаток → система відображає головну сторінку з бібліотекою.

2) Користувач вибирає книгу зі списку або імпортує нову з пам’яті чи хмарного сховища → система завантажує книгу.

3) Користувач відкриває книгу → система переходить у режим читання.

4) Користувач перегортає сторінки свайпом або натисканням → система відображає наступну/попередню сторінку.

5) Користувач змінює налаштування тексту → система динамічно оновлює інтерфейс.

6) При виході система автоматично зберігає останню прочитану сторінку.

3.1.3 Функціональні вимоги

REQ-1: Додаток має дозволяти відкриття електронних книг у форматах TXT, FB2, EPUB, MOBI.

REQ-2: Повинна бути реалізована можливість імпорту книги з пам’яті пристрою або хмарного сховища.

REQ-3: Інтерфейс перегляду повинен підтримувати навігацію сторінками (свайпи або кнопки).

REQ-4: Додаток повинен надавати користувачу можливість змінювати розмір шрифту, інтервал між рядками, шрифт, тему оформлення (світла/темна).

REQ-5: При виході з книги додаток повинен автоматично зберігати останню відкриту сторінку для продовження читання.

REQ-6: У разі завантаження пошкодженого або несумісного файлу система повинна відобразити повідомлення про помилку з рекомендацією.

REQ-7: Якщо книга велика (понад 1000 сторінок), система повинна працювати без помітного зниження швидкодії (в межах 200 мс на зміну сторінки).

3.2 Озвучення тексту (TTS)

3.2.1 Опис та пріоритет

Функція озвучення тексту (Text-to-Speech) дозволяє користувачам слухати фрагменти або повний текст електронної книги, що забезпечує доступність для людей із порушеннями зору, зручність використання під час подорожей або занять іншими справами.

Пріоритет: Високий

Benefit: 9 / Penalty: 7 / Cost: 5 / Risk: 3

3.2.2 Послідовність дій користувача та реакція системи

1) Користувач відкриває книгу в режимі читання.
2) Обирає фрагмент тексту або натискає кнопку "Озвучити".
3) Система надсилає запит до сервісу TTS (наприклад, Google Text-to-Speech).
4) Сервіс генерує звуковий файл і передає його додатку.
5) Система відтворює голосовий супровід за вибраними параметрами (швидкість, голос, мова).
6) Користувач має можливість призупинити, перемотати або змінити налаштування голосу.

3.2.3 Функціональні вимоги

REQ-TTS-1: Система повинна дозволяти озвучення як окремих слів/фрагментів, так і повного тексту книги.

REQ-TTS-2: Система має підтримувати використання сторонніх TTS API (Google Cloud TTS).

REQ-TTS-3: Користувач повинен мати змогу налаштовувати мову озвучення.

REQ-TTS-4: У разі втрати інтернет-з’єднання, система повинна інформувати користувача про неможливість озвучення (або перейти в офлайн-режим).

REQ-TTS-5: Голосове озвучення повинно запускатися не більше ніж через 2 секунди після дії користувача.

REQ-TTS-6: Система повинна відтворювати озвучення через внутрішній динамік або підключені навушники.

REQ-TTS-7: У разі помилки генерації мовлення (недоступність API, помилка тексту) має виводитися зрозуміле повідомлення користувачу.

REQ-TTS-8: Система повинна підтримувати озвучення багатьма мовами відповідно до налаштувань користувача.

3.3 Переклад тексту

3.3.1 Опис та пріоритет

Функція перекладу дозволяє користувачам виділяти окремі слова або фрагменти тексту для отримання миттєвого перекладу, що особливо корисно для вивчення іноземних мов або читання літератури мовою оригіналу.

Пріоритет: Високий

Benefit: 8 / Penalty: 6 / Cost: 4 / Risk: 3

3.3.2 Послідовність дій користувача та реакція системи

1) Користувач виділяє слово або фразу в тексті книги.
2) Натискає кнопку перекладу.
3) Система надсилає запит до перекладацького API (DeepL).
4) Отриманий переклад виводиться у спливаючому вікні.
5) Користувач може прослухати переклад або зберегти його до словника.

3.3.3 Функціональні вимоги

REQ-TR-1: Система повинна дозволяти переклад виділеного тексту з будь-якої підтримуваної мови на іншу.

REQ-TR-2: Система має підтримувати інтеграцію з API сторонніх сервісів перекладу.

REQ-TR-3: Переклад повинен повертатися та відображатися не довше ніж за 2 секунди після запиту.

REQ-TR-4: У разі втрати підключення до Інтернету, система має виводити повідомлення про неможливість перекладу.

REQ-TR-5: Переклад має відображатися у спливаючому вікні, без необхідності переходу на інший екран.

REQ-TR-6: Користувач повинен мати змогу зберігати перекладені слова до особистого словника.

REQ-TR-7: Система має забезпечувати підтримку щонайменше 20 популярних мов.

REQ-TR-8: У випадку недоступності API або помилки перекладу, має виводитись повідомлення про помилку.

3.4 Управління бібліотекою книг

3.4.1 Опис та пріоритет

Функція управління бібліотекою дозволяє користувачеві додавати, зберігати, організовувати та видаляти книги. Вона забезпечує централізований доступ до всіх електронних книг користувача, що робить додаток самодостатнім рідером.

Пріоритет: Високий

Benefit: 7 / Penalty: 5 / Cost: 3 / Risk: 2

3.4.2 Послідовність дій користувача та реакція системи

1) Користувач відкриває розділ "Бібліотека".
2) Обирає варіант: завантажити нову книгу або переглянути вже наявні.
3) У разі імпорту – обирає файл із пам’яті або хмарного сховища.
4) Система додає книгу до списку бібліотеки.
5) Користувач має змогу переглянути деталі книги, видалити її або почати читання.

3.4.3 Функціональні вимоги

REQ-LIB-1: Система повинна дозволяти додавання книг з локальної пам’яті пристрою.

REQ-LIB-2: Система має підтримувати завантаження книг з хмарних сервісів (Google Drive).

REQ-LIB-3: Бібліотека має відображати список книг з назвою, автором та прогресом читання.

REQ-LIB-4: Користувач повинен мати змогу видаляти книги зі свого списку.

REQ-LIB-5: Повинен бути пошук за назвою, автором або ключовими словами.

REQ-LIB-6: Система має зберігати список книг у локальному сховищі пристрою.

REQ-LIB-7: У разі пошкодженого або несумісного файлу має з’являтися повідомлення про помилку.

3.5 Персоналізація інтерфейсу

3.5.1 Опис та пріоритет

Функція персоналізації інтерфейсу забезпечує можливість адаптації зовнішнього вигляду додатка до індивідуальних потреб користувача. Це сприяє покращенню юзабіліті, зручності роботи з текстом, а також враховує особливі потреби (наприклад, користувачів із порушеннями зору).

Пріоритет: Середній

Benefit: 7 / Penalty: 5 / Cost: 3 / Risk: 2

3.5.2 Послідовність дій користувача та реакція системи

1) Користувач відкриває меню налаштувань зовнішнього вигляду.

2) Обирає бажану тему інтерфейсу (світлу або темну), тип та розмір шрифту, колір фону та відступів.

3) Зміни застосовуються до всіх розділів додатку в реальному часі.

4) Система зберігає обрані параметри для наступного сеансу.

3.5.3 Функціональні вимоги

REQ-UI-1: Користувач має змогу вибирати між світлою та темною темами.

REQ-UI-2: Повинна бути реалізована зміна розміру та типу шрифту тексту книги.

REQ-UI-3: Користувач може змінювати міжрядковий інтервал та ширину полів.

REQ-UI-4: Всі налаштування мають зберігатися локально для подальшого використання.

REQ-UI-5: Зміни повинні застосовуватись без необхідності перезавантаження додатка .

REQ-UI-6: Інтерфейс має бути адаптивним до різних розмірів екранів і типів пристроїв.

3.6 Система налаштувань

3.6.1 Опис та пріоритет

Система налаштувань забезпечує користувачеві можливість контролювати поведінку програми, її мову, параметри озвучення та перекладу, а також загальні функціональні аспекти.

Пріоритет: Середній

Benefit: 6 / Penalty: 4 / Cost: 2 / Risk: 1

3.6.2 Послідовність дій користувача та реакція системи

1) Користувач відкриває розділ «Налаштування».

2) Вибирає мову інтерфейсу, мову озвучення, швидкість мовлення, цільову мову для перекладу тощо.

3) Система застосовує змінені налаштування до відповідних модулів.

4) Усі зміни зберігаються автоматично.

3.6.3 Функціональні вимоги

REQ-SET-1: Додаток повинен дозволяти вибір мови інтерфейсу з-поміж доступних (наприклад, українська, англійська).

REQ-SET-2: Повинна бути можливість вибору мови озвучення та мови перекладу окремо.

REQ-SET-3: Усі зміни повинні зберігатися локально та застосовуватись одразу.

REQ-SET-4: Повинна бути реалізована кнопка «Скинути до стандартних налаштувань».

3.7 Робота в офлайн-режимі

3.7.1 Опис та пріоритет

Функціональність офлайн-режиму забезпечує доступ до основних можливостей програми без підключення до Інтернету. Це критично важливо для користувачів, які хочуть читати книги у будь-який час, незалежно від мережевого покриття.

Пріоритет: Високий

Benefit: 9 / Penalty: 7 / Cost: 4 / Risk: 3

3.7.2 Послідовність дій користувача та реакція системи

1) Користувач завантажує книгу під час роботи онлайн.
2) При втраті Інтернету відкриває вже завантажену книгу.
3) Система автоматично переходить у режим офлайн.
4) Користувач читає, переглядає зміст, використовує локальні функції.
5) Якщо переклад або озвучення недоступні – система повідомляє про обмеження.

3.7.3 Функціональні вимоги

REQ-OFF-1: Система повинна дозволяти читання книг, що були попередньо завантажені, без підключення до Інтернету.

REQ-OFF-2: Усі персональні налаштування, інтерфейс і збереження прогресу мають працювати в офлайн-режимі.

REQ-OFF-3: Якщо функції перекладу або озвучення недоступні через відсутність мережі, користувач має отримати відповідне повідомлення.

REQ-OFF-4: Має бути можливість використовувати локальні моделі для отримання функціональності офлайн.

REQ-OFF-5: Усі локальні зміни повинні синхронізуватись після відновлення з’єднання.

4 External Interface Requirements

4.1 User Interfaces

Програмний продукт включає графічний інтерфейс користувача (GUI), орієнтований на мобільні пристрої з сенсорним керуванням (переважно Android-смартфони та планшети). Інтерфейс розробляється відповідно до сучасних стандартів мобільної юзабіліті (Google Material Design, рекомендації WCAG 2.1 щодо доступності).

Основні компоненти інтерфейсу:

1) Головна сторінка (бібліотека книг)

– Список книг із обкладинками, назвою, автором, індикатором прогресу.

– Кнопка «+» для додавання нової книги з пам’яті або хмарного сховища.

– Панель пошуку за назвою/автором.

– Бокове меню навігації: «Бібліотека», «Статистика», «Налаштування».

2) Режим читання

– Відображення тексту книги з можливістю гортання сторінок свайпом.

– Випадаюче меню перекладу при виділенні тексту.

– Кнопка озвучення фрагменту або повного тексту.

– Верхнє меню: переклад, пошук по тексту, озвучення, перехід на сторінку.

– Нижній індикатор прогресу читання.

3) Меню налаштувань

– Налаштування інтерфейсу (тема, шрифт, мова).

– Параметри озвучення (швидкість, голос, тональність).

– Мова перекладу та цільова мова.

– Скидання до стандартних налаштувань.

4) Інтерфейс перекладу та словника

– Спливаюче вікно з перекладом слова/фрази.

– Кнопка «Додати до словника».

– Перегляд збережених слів і статистики в окремому розділі.

5) Елементи універсального доступу (доступність)

– Великі кнопки та шрифти.

– Контрастні теми з можливістю перемикання.

– Озвучення тексту інтерфейсу за запитом користувача (інтеграція з Android Accessibility).

Стандарти дизайну та поведінки:

– Всі повідомлення про помилки повинні бути короткими, чіткими, з поясненням причини та порадою щодо усунення (наприклад: «Неможливо з’єднатися з сервером. Перевірте підключення до Інтернету»).

– Кожен екран має кнопку «Назад» (віртуальна або в навігаційній панелі).

– Підтримка жестів для навігації (свайп, утримання, подвійний тап).

– Екрани оптимізовані для вертикальної орієнтації.

Компоненти, що потребують UI:

– Модуль читання книг.

– Модуль перекладу та озвучення.

– Налаштування програми.

– Модуль словника/історії перекладів.

– Модуль бібліотеки книг.

– Інтерфейс повідомлень та підтверджень дій.

4.2 Hardware Interfaces

Програмне забезпечення є мобільним додатком, що функціонує виключно на мобільних пристроях із сенсорним екраном, які працюють під управлінням операційної системи Android. Додаток не вимагає спеціалізованого апаратного забезпечення, однак взаємодіє з низкою стандартних апаратних компонентів мобільного пристрою.

Підтримувані типи пристроїв:

– Смартфони та планшети з операційною системою Android 8.0 або вище та IOS 13.0 або вище.

– Пристрої з ARM-процесорами (64-бітна архітектура).

– Пристрої з мінімум 4 ГБ оперативної пам’яті.

Основні апаратні інтерфейси:

1) Сенсорний екран

Користувач взаємодіє із додатком через сенсорні жести (дотик, свайп, утримання).

Підтримуються стандартні жести Android UI.

2) Динаміки та навушники

Виведення озвучення (TTS) здійснюється через вбудовані динаміки або аудіовихід.

Система має автоматично перемикатися між динаміками і підключеними навушниками.

3) Інтернет-з’єднання (Wi-Fi, мобільна мережа)

Необхідне для роботи з API озвучення, перекладу, хмарного сховища.

Відсутність з’єднання повинна визначатися програмно з відповідною реакцією інтерфейсу.

4) Сховище пристрою

Додаток має доступ до файлової системи користувача (лише в межах дозволів Android) для імпорту книг і збереження прогресу.

Використовуються стандартні шляхи доступу до внутрішнього або зовнішнього сховища (через SAF — Storage Access Framework).

5) Мікрофон (опціонально, у майбутніх версіях)

У перспективі можливе використання для голосового керування або розпізнавання мови.

Комунікаційні протоколи:

– HTTP/HTTPS – для взаємодії з хмарними сервісами (TTS, перекладач, Google Drive).

– Android Media API – для обробки та відтворення аудіо.

– Android Storage API / SAF – для взаємодії з файловою системою.

– Bluetooth / USB (опціонально) – для відтворення озвучення через зовнішні пристрої.

4.3 Software Interfaces

Мобільний додаток взаємодіє з низкою зовнішніх програмних компонентів, сервісів, бібліотек і API для забезпечення основної функціональності: озвучення тексту, перекладу, роботи з файлами та аналітики. Нижче наведено основні інтеграційні точки.

Операційна система:

– Android 8.0 (API level 26) і вище.

– IOS 13.0 і вище.

Системні компоненти: Android Permissions, MediaPlayer API, Connectivity Manager, Storage Access Framework.

Використовувані бібліотеки та фреймворки:

– React Native 0.7x — основний фреймворк для кросплатформенної розробки.

– TypeScript 4.x — типізована мова програмування для клієнтської логіки.

– React Navigation — маршрутизація між екранами.

– Zustand / AsyncStorage — управління станом і локальне збереження даних.

Зовнішні сервіси та API:

1) Google Text-to-Speech API

– Призначення: генерування аудіо з тексту (TTS).

– Вхідні дані: текст, мова, тип голосу.

– Вихідні дані: аудіофайл у форматі MP3/WAV.

– Канал зв’язку: HTTPS.

2) DeepL API

– Призначення: переклад виділеного фрагмента тексту.

– Вхідні дані: текст для перекладу, мова-джерело, цільова мова.

– Вихідні дані: перекладена фраза.

– Канал зв’язку: HTTPS.

Обмін даними між компонентами:

– Усі внутрішні модулі програми (читання, озвучення, переклад, словник, налаштування) обмінюються даними через глобальний стан (Redux Store).

– Дані про книги, переклади, прогрес зберігаються в локальному сховищі (AsyncStorage) та можуть кешуватися для офлайн-режиму.

– Механізм передачі даних — події, API виклики, реактивні оновлення.

Вимоги до реалізації обміну:

– Обмін даними між компонентами повинен бути асинхронним, із підтримкою обробки помилок.

– Персональні дані користувача повинні зберігатися локально без передачі стороннім сервісам, окрім випадків прямих API-запитів (переклад, озвучення).

– Всі API-запити мають реалізовуватись із використанням HTTPS для шифрування трафіку.

– API-ключі повинні зберігатися в зашифрованому вигляді або через зовнішній секретний сервер.

4.4 Communications Interfaces

Мобільний додаток для читання електронних книг з функціями озвучення та перекладу активно використовує мережеву взаємодію для забезпечення доступу до зовнішніх сервісів, завантаження контенту, синхронізації даних та обробки транзакцій. Усі комунікації здійснюються за допомогою сучасних безпечних протоколів та з урахуванням вимог до конфіденційності.

1) Протоколи обміну даними

HTTPS (Hypertext Transfer Protocol Secure) — основний протокол для передачі даних між додатком і зовнішніми API:

– Переклад текстів (DeepL API).

– Озвучення текстів (Google Cloud TTS).

2) Формат переданих повідомлень

JSON — основний формат обміну даними між додатком і серверними API (запити до TTS, перекладача).

3) Механізми синхронізації

Обробка запитів здійснюється асинхронно з використанням fetch.

Повторна спроба з’єднання виконується при тимчасовій втраті мережі.

Прогрес користувача, словник та історія перекладів зберігаються локально з подальшою можливістю синхронізації при відновленні з’єднання.

4) Безпека комунікацій

Усі передані дані шифруються за допомогою TLS 1.2 або вище.

Конфіденційна інформація користувача (платіжні дані, облікові токени) не зберігається в додатка .

Доступ до зовнішніх сервісів обмежується через API-ключі, які зберігаються у змінних середовища CI/CD.

5) Обмеження передачі даних

Максимальний розмір тексту для перекладу або озвучення в одному запиті: 1 000 символів.

5 Other Nonfunctional Requirements

5.1 Performance Requirements

Додаток повинен забезпечувати високу продуктивність у роботі з текстовими даними, мережевими запитами та інтерфейсною взаємодією на мобільних пристроях з різними характеристиками. Основна мета — забезпечити плавний, швидкий та надійний досвід користувача, навіть за умов обмежених ресурсів пристрою або нестабільного інтернет-з’єднання.

1) Загальні вимоги до швидкодії

Запуск додатка не повинен перевищувати 3 секунд після натискання іконки.

Час відкриття книги – не більше 2 секунд для файлів обсягом до 5 МБ.

Перегортання сторінки має відбуватись із затримкою не більше 200 мс.

2) Озвучення тексту (TTS)

Час очікування генерації озвучення (після запиту на фрагмент до 300 символів) — не більше 2 секунд при стабільному інтернет-з’єднанні.

Озвучення має починатися безперервно та без підвисань, навіть при паралельній роботі інших процесів.

3) Переклад тексту

Час на отримання перекладу фрагменту (до 50 слів) — не більше 1.5 секунд.

Відповіді з API повинні кешуватись локально при повторному запиті на той самий текст.

4) Робота в офлайн-режимі

Відкриття книг з локальної пам’яті має бути доступне без затримок навіть при повній відсутності інтернету.

Додаток повинен підтримувати безперервну роботу в офлайн-режимі щонайменше 1 годину без суттєвого споживання ресурсів.

5) Робота на пристроях з обмеженими ресурсами

Система повинна стабільно працювати на пристроях з 4 ГБ ОЗП без аварійного завершення сесій.

Середній рівень завантаження процесора під час озвучення або перекладу не повинен перевищувати 60%.

6) Паралельність процесів

Читання, озвучення та переклад повинні працювати одночасно, без потреби закриття будь-якої з функцій.

Програма повинна підтримувати асинхронне виконання запитів без блокування основного потоку інтерфейсу (UI thread).

5.2 Safety Requirements

Хоча мобільний додаток для читання електронних книг не є критичною системою, яка безпосередньо впливає на здоров’я чи фізичну безпеку користувача, певні аспекти безпеки все ж мають бути враховані. Зокрема, це стосується захисту даних користувача, стабільної роботи програми, а також психо-фізіологічного комфорту при тривалому використанні додатка .

1) Захист від втрати даних

Додаток повинен автоматично зберігати прогрес читання при зміні сторінки.

Усі збережені налаштування та словники повинні зберігатися локально з можливістю резервного копіювання.

У разі аварійного завершення програми, при наступному запуску має відбуватись відновлення останнього стану сесії.

2) Попередження перевантаження пристрою

Озвучення великих обсягів тексту повинно виконуватись порційно, з попередженням у разі надмірного навантаження.

Система повинна контролювати використання пам’яті та запобігати виходу за межі допустимого обсягу RAM (встановлений поріг — 75% доступної оперативної пам’яті).

3) Зменшення ризику зорової втоми

Програма повинна включати нічний режим (темна тема) та налаштування шрифтів для зручного читання в умовах поганого освітлення.

4) Взаємодія з користувачами з порушеннями зору

Інтерфейс має відповідати мінімальним вимогам WCAG 2.1 рівня AA (контрастність, масштабованість).

Повинні бути універсальні елементи керування з чітким аудіоозвученням (інтеграція з Android Accessibility Services).

5) Запобігання некоректним діям

При спробі завантажити файл із непідтримуваним форматом або перевищенням допустимого розміру має відображатися повідомлення з підтримуваними форматами.

Усі запити на озвучення або переклад мають перевірятись на коректність вхідних даних перед надсиланням до API.

6) Відповідність політикам та сертифікаціям

Додаток має відповідати вимогам Google Play Developer Policies щодо поводження з персональними даними.

При використанні озвучення для освітніх цілей у навчальних закладах — дотримання принципів цифрової інклюзії.

5.3 Security Requirements

Додаток опрацьовує персоналізовану інформацію користувача (настройки, прогрес, словник), взаємодіє з хмарними сервісами, а також може здійснювати фінансові транзакції. Тому до системи пред’являються підвищені вимоги щодо інформаційної безпеки, захисту персональних даних і конфіденційності.

1) Ідентифікація та автентифікація користувача

Усі маркери доступу (access tokens) повинні зберігатися в зашифрованому вигляді та автоматично оновлюватись при завершенні терміну дії.

2) Захист даних користувача

Персональні дані (наприклад, словник, історія читання, мова інтерфейсу) повинні зберігатися в локальному сховищі пристрою і не передаватися третім сторонам.

У разі реалізації синхронізації з хмарними сервісами — дані мають передаватися лише через захищене з’єднання (HTTPS з TLS ≥ 1.2).

Усі ключові дії (завантаження файлів, запити до API) повинні перевірятись на наявність небезпечного коду або некоректних даних.

3) Безпека мережевої взаємодії

Усі запити до зовнішніх сервісів (переклад, озвучення) здійснюються тільки через HTTPS.

Не дозволяється кешування персональних або фінансових запитів сторонніми API-платформами.

4) Обмеження доступу

Доступ до локальних даних додатка повинен бути обмежений тільки самим додатком.

5) Відповідність зовнішнім політикам

Додаток має відповідати:

– Політиці конфіденційності Google Play;

– Загальному регламенту захисту даних (GDPR) для користувачів з ЄС;

6) Додаткові заходи

Повинна бути реалізована можливість повного видалення персональних даних на запит користувача.

Повідомлення про збої, що стосуються безпеки, мають бути чітко і зрозуміло сформульовані.

5.4 Software Quality Attributes

Якість програмного продукту є критично важливою як для кінцевих користувачів, так і для розробників та замовників. Нижче визначено основні характеристики якості, які повинні бути досягнуті під час розробки, тестування та підтримки мобільного додатка .

1) Юзабіліті (Usability)

Інтерфейс повинен бути інтуїтивно зрозумілим для користувачів з мінімальним досвідом у мобільних технологіях.

Користувач повинен навчитися основним функціям програми за ≤ 5 хвилин без зовнішньої інструкції.

Усі дії (переклад, озвучення, навігація) мають бути доступні в 2–3 кліки.

2) Надійність (Reliability)

Додаток має працювати стабільно при ≥ 99% усіх сесій, без зависань або аварійного завершення.

У разі несподіваного завершення сесії система повинна відновити останній стан прогресу читання.

3) Мобільність/Портативність (Portability)

Програма повинна бути сумісною з 95% сучасних Android-пристроїв (версії 8.0 і вище) та IOS пристроїв (версії 13.0 і вище).

4) Масштабованість і гнучкість (Scalability & Flexibility)

Архітектура має підтримувати додавання нових мов, голосів, функцій без повної переробки ядра.

Система повинна легко інтегрувати сторонні API (наприклад, альтернативні сервіси TTS або перекладачі).

5) Підтримуваність (Maintainability)

Код повинен бути структурованим і відповідати стандартам ESLint/TypeScript.

Модульна структура забезпечує простоту оновлення окремих компонентів без впливу на інші.

Очікується, що новий розробник може ознайомитися з базовою структурою проекту протягом ≤ 1 тижня.

6) Тестованість (Testability)

Усі ключові функції повинні покриватися автоматизованими юніт-тестами.

Інтерфейс має бути протестований вручну на основних розмірах екранів.

7) Інтероперабельність (Interoperability)

Система повинна взаємодіяти з хмарними сервісами (Google API) через відкриті протоколи REST.

Дані повинні бути сумісні з іншими системами через стандартизовані формати: JSON, MP3, EPUB, TXT.

8) Адаптивність (Adaptability)

Інтерфейс має автоматично підлаштовуватись під орієнтацію екрана, розмір пристрою, шрифти та мову інтерфейсу.

Підтримується повноцінна робота у темному та світлому режимах.

ДОДАТОК Б

Код алгоритму розбиття книги на сторінки

Код реалізації:

```
// Helper function to measure text
const measureText = (text: string, settings: { fontSize: number;
lineSpacing: number; marginWidth: number }) => {
  const headerHeight = 80;
  const nameHeight = 95;
  const footerHeight = 22;
  const { width: screenWidth, height: screenHeight } =
Dimensions.get('window');
  const availableWidth = screenWidth - (settings.marginWidth * 2);
  const lineHeight = settings.fontSize * settings.lineSpacing;
  const availableHeight = screenHeight - headerHeight - nameHeight -
footerHeight - 16*2;
  const linesPerPage = Math.floor(availableHeight / lineHeight);
  return {
    availableWidth,
    linesPerPage,
  };
};

// Helper function to wrap text to width
const wrapText = (text: string, maxWidth: number, fontSize: number) => {
  const words = text.match(/[^\s\n]+|\n/g) || [];
  const lines: string[] = [];
  let currentLine: string[] = [];
  let currentWidth = 0;
  const defaultCharWidth = 0.63;
  const uppercaseCharWidth = 0.75;
  let isNewLine = true;

  const getCharWidth = (char: string): number => {
    // Check if character is uppercase
    if (char === char.toUpperCase() && char !== char.toLowerCase() &&
CHAR_WIDTHS[char] === undefined) {
      return uppercaseCharWidth * fontSize;
    }
    return (CHAR_WIDTHS[char] || defaultCharWidth) * fontSize;
  };

  const indentWidth = getCharWidth(' ') * 3;

  const getWordWidth = (word: string) => {
    let width = 0;
    for (let i = 0; i < word.length; i++) {
      width += getCharWidth(word[i]);
    }
    return width;
  };

  for (const word of words) {
    // If we encounter a newline, start a new line
```

```
    if (word === '\n') {
      if (currentLine.length > 0) {
        lines.push(currentLine.join(' '));
        currentLine = [];
        currentWidth = 0;
      }
      isNewLine = true;
      continue;
    }

    const wordWidth = getWordWidth(word);
    // If adding this word would exceed the line width
    if (currentWidth + wordWidth + (currentLine.length > 0 ? getCharWidth('
') : 0) + (isNewLine ? indentWidth : 0) > maxWidth) {
      if (currentLine.length > 0) {
        lines.push(currentLine.join(' '));
        currentLine = [];
        currentWidth = 0;
      }

      // If a single word is longer than the line, split it
      if (wordWidth > maxWidth) {
        let remainingWord = word;
        while (remainingWord.length > 0) {
          let charsThatFit = 0;
          let widthSoFar = 0;

          // Calculate how many characters can fit
          while (charsThatFit < remainingWord.length) {
            const charWidth = getCharWidth(remainingWord[charsThatFit]);
            if (widthSoFar + charWidth > maxWidth) break;

            widthSoFar += charWidth;
            charsThatFit++;
          }

          lines.push(remainingWord.slice(0, charsThatFit));
          remainingWord = remainingWord.slice(charsThatFit);
        }
        continue;
      }
    }

    // Add 3 spaces before the first word after a newline
    if (isNewLine) {
      currentLine.push('   ' + word);
      currentWidth += indentWidth;
      isNewLine = false;
    } else {
      currentLine.push(word);
    }
    currentWidth += wordWidth + (currentLine.length > 1 ? getCharWidth(' ')
: 0);
  }

  // Handle the last line
  if (currentLine.length > 0) {
```

```
    lines.push(currentLine.join(' '));
  }

  return lines;
};
```

Функція виклику:

```
processContent: async (content: string) => {
    const currentBook = get().book;
    if (!currentBook) {
      console.log('No book available for processing content');
      return;
    }

    try {
      console.log('Processing content, length:', content.length);

      // Get actual settings from settings store
      const settings = useSettingsStore.getState();
      const textSettings = {
        fontSize: settings.fontSize,
        lineSpacing: settings.lineSpacing,
        marginWidth: settings.marginWidth,
      };

      console.log('Using settings:', textSettings);

      const { availableWidth, linesPerPage } = measureText(content,
textSettings);
      console.log('Page settings:', { availableWidth, linesPerPage });

      // Split content into paragraphs
      const paragraphs = content.split(/\n\s*\n/).filter(p =>
p.trim().length > 0);
      console.log('Number of paragraphs:', paragraphs.length);

      // Create pages
      const pages: string[] = [];
      let currentPageLines: string[] = [];
      let currentLineCount = 0;

      for (const paragraph of paragraphs) {
        // Wrap paragraph to width
        const wrappedLines = wrapText(paragraph, availableWidth,
textSettings.fontSize);

        for (const line of wrappedLines) {
          // If adding this line would exceed the page limit
          if (currentLineCount >= linesPerPage) {
            pages.push(currentPageLines.join('\n'));
            currentPageLines = [];
            currentLineCount = 0;
          }

          currentPageLines.push(line);
```

```
        currentLineCount++;
      }

      // Add paragraph spacing if not at the end of a page
      if (currentLineCount < linesPerPage) {
        currentPageLines.push('');
        currentLineCount++;
      }
    }

    // Add the last page if it has content
    if (currentPageLines.length > 0) {
      pages.push(currentPageLines.join('\n'));
    }

    console.log('Pages created:', pages.length);

    const updatedBook = {
      ...currentBook,
      content,
      totalPages: pages.length,
      pages,
      lastRead: new Date().toISOString()
    };
    console.log('Updating book with processed content');
    set({ book: updatedBook });
    await storageUtils.set(STORAGE_KEY, updatedBook);
    await storageUtils.set(LAST_BOOK_KEY, updatedBook.id);
    console.log('Book updated successfully');
  } catch (error) {
    console.error('Error processing content:', error);
  }
}
```

ДОДАТОК В

Код хука сторінки читання

```
const book = useBookStore((state) => state.book);
  const updateCurrentPage = useBookStore((state) => state.setCurrentPage);
  const loadBookContent = useBookStore((state) => state.loadBookContent);
  const updatePages = useBookStore((state) => state.updatePages);
  const loadLastBook = useBookStore((state) => state.loadLastBook);
  const { width } = useWindowDimensions();
  const { colors } = useTheme();
    const { fontSize, fontFamily, lineSpacing, marginWidth } =
useSettingsStore();

  const [currentPage, setCurrentPage] = useState(0);
  const [isLoading, setIsLoading] = useState(true);
  const translateX = useSharedValue(0);
  const context = useSharedValue({ x: 0 });



  const changePage = useCallback((direction: number) => {
    if (!book?.pages) {
      console.log('No pages available for navigation');
      return;
    }

    const nextPage = currentPage + direction;
    if (nextPage >= 0 && nextPage < book.pages.length) {
      console.log('Changing page to:', nextPage);
      setCurrentPage(nextPage);
      updateCurrentPage(nextPage);
    } else {
      console.log('Cannot change page: out of bounds');
    }
  }, [book?.pages, currentPage, updateCurrentPage]);

  const gesture = Gesture.Pan()
    .onStart(() => {
      context.value = { x: translateX.value };
    })
    .onUpdate((e) => {
      translateX.value = e.translationX + context.value.x;
    })
    .onEnd((e) => {
      const threshold = width / 4;
       const direction = e.translationX < -threshold ? 1 : e.translationX >
threshold ? -1 : 0;
      if (direction !== 0) {
        runOnJS(changePage)(direction);
      }
      translateX.value = withSpring(0);
    });

  const animatedStyle = useAnimatedStyle(() => ({
    transform: [{ translateX: translateX.value }]
```

```
}));

// Load last book when component mounts
useEffect(() => {
  const loadBook = async () => {
    setIsLoading(true);
    try {
      await loadLastBook();
    } catch (error) {
      console.error('Error loading last book:', error);
    } finally {
      setIsLoading(false);
    }
  };
  loadBook();
}, []);

// Load book content when book changes
useEffect(() => {
  const loadContent = async () => {
    if (book?.filePath) {
      console.log('Loading book content from:', book.filePath);
      setIsLoading(true);
      try {
        await loadBookContent();
      } catch (error) {
        console.error('Error loading book content:', error);
      } finally {
        setIsLoading(false);
      }
    }
  };
  loadContent();
}, [book?.filePath]);

// Update pages when text settings change
useEffect(() => {
  const updatePageSettings = async () => {
    if (book?.content) {
      console.log('Updating pages with new settings');
      setIsLoading(true);
      try {
        await updatePages({ fontSize, lineSpacing, marginWidth });
      } catch (error) {
        console.error('Error updating pages:', error);
      } finally {
        setIsLoading(false);
      }
    }
  };
  updatePageSettings();
}, [fontSize, lineSpacing, marginWidth]);

// Update current page when book changes
useEffect(() => {
  console.log('Book changed:', {
    hasBook: !!book,
```

```
      hasPages: !!book?.pages,
      pagesLength: book?.pages?.length,
      currentPage: book?.currentPage
    });

    if (book?.currentPage !== undefined) {
      setCurrentPage(book.currentPage);
    }

    if (book?.pages && book.pages.length > 0) {
      setIsLoading(false);
    }
  }, [book]);
```

ДОДАТОК Г
Слайди презентації

Мобільний додаток для читання та прослуховування електронних книг

Височин Ілля Максимович ПЗПІ-21-7
Керівник: доц. кафедри ПІ Кравець Наталя Сергіївна

16 червня 2025

Рисунок Г.1 – Слайд презентації 1

Мета роботи

Створення кросплатформенного мобільного додатка для читання електронних книг з підтримкою автоматичного озвучування тексту та перекладу окремих слів у режимі реального часу

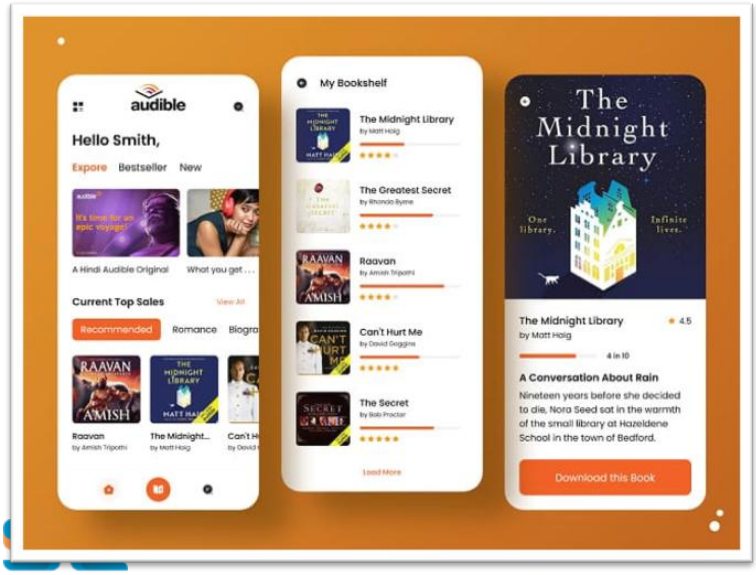
Актуальність роботи

У сучасному світі важливою частиною інформаційних технологій є оптимізація процесу доступу до знань та інформації. Зі зростанням обсягів споживання цифрового контенту, особливо літератури, виникає потреба у зручних засобах читання електронних книг, які не лише надають базовий функціонал, а й враховують потреби широкого кола користувачів: від людей із вадами зору до тих, хто вивчає іноземні мови.

2

Рисунок Г.2 – Слайд презентації 2

Аналіз проблеми (аналіз існуючих рішень)



Audible

Переваги:

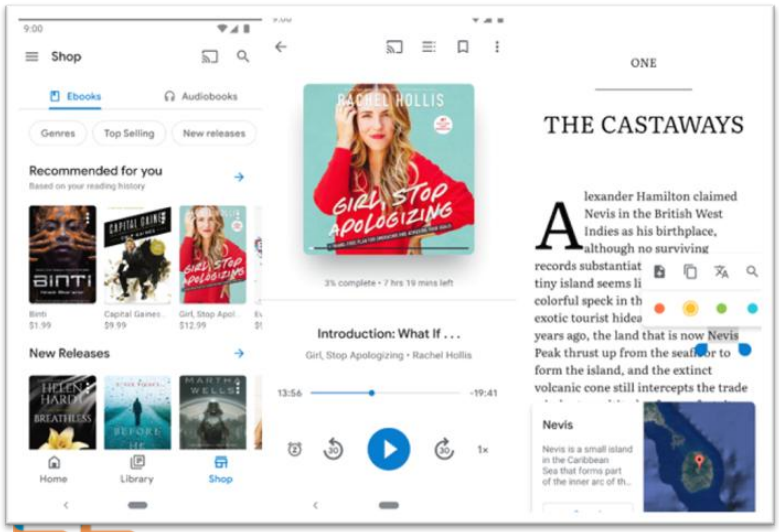
- Висока якість озвучення професійними дикторами;
- Інтеграція з Alexa, підтримка закладок та зміни швидкості;
- Зручний інтерфейс для прослуховування.

Недоліки:

- Відсутність підтримки TTS або динамічного озвучення тексту;
- Немає перекладу «на льоту»;
- Переважно платний контент;
- Обмеження для користувачів, що вивчають мову.

Рисунок Г.3 – Слайд презентації 3

Аналіз проблеми (аналіз існуючих рішень)



Google Play Books

Переваги:

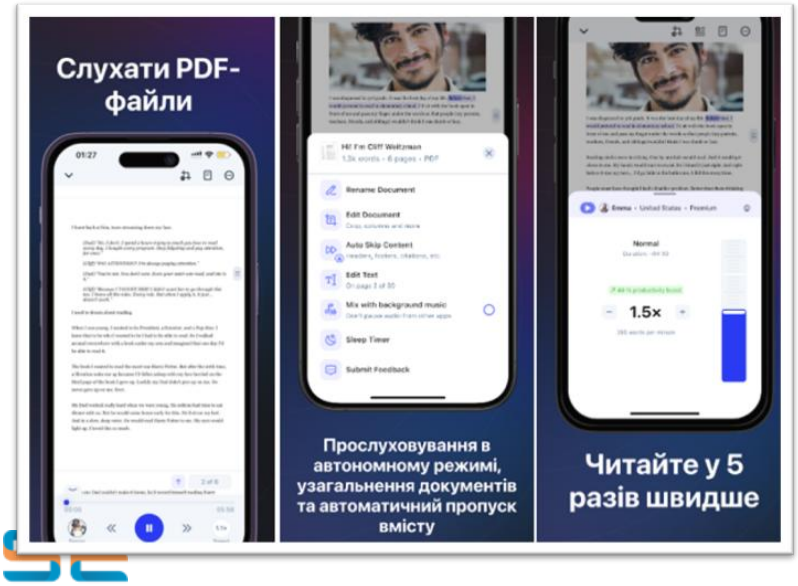
- Інтеграція з Google Translate для перекладу під час читання;
- Підтримка функції озвучення (TTS);
- Простий та зрозумілий інтерфейс.

Недоліки:

- Немає офлайн-перекладу;
- Низька якість голосів для менш поширених мов;
- Відсутність історії перекладів або словника;
- Обмежена адаптація під мовне навчання.

Рисунок Г.4 – Слайд презентації 4

Аналіз проблеми (аналіз існуючих рішень)



Speechify

Переваги:

- Якісний TTS для людей із дислексією або порушеннями зору;
- Можливість імпорту з PDF, Google Docs, вебсторінок;
- Персоналізація голосу, швидкості, інтонації.

Недоліки:

- Розширений функціонал доступний лише в платній версії;
- Немає перекладу та збереження до словника;
- Відсутність повної офлайн-підтримки;
- Немає інтеграції з перегортанням сторінок і статистикою.

5

Рисунок Г.5 – Слайд презентації 5

Постановка задачі та опис системи

- Підтримка багатоформатного імпорту, створення бібліотеки та читання електронних книг
- Реалізація автоматичного озвучування тексту з підтримкою кількох мов
- Інтеграція функції перекладу окремих слів або фраз
- Інклюзивність для користувачів з особливими потребами
- Надання можливостей для індивідуального налаштування інтерфейсу, збереження історії прочитаного, налаштування мови

6

Рисунок Г.6 – Слайд презентації 6

Постановка задачі та опис системи

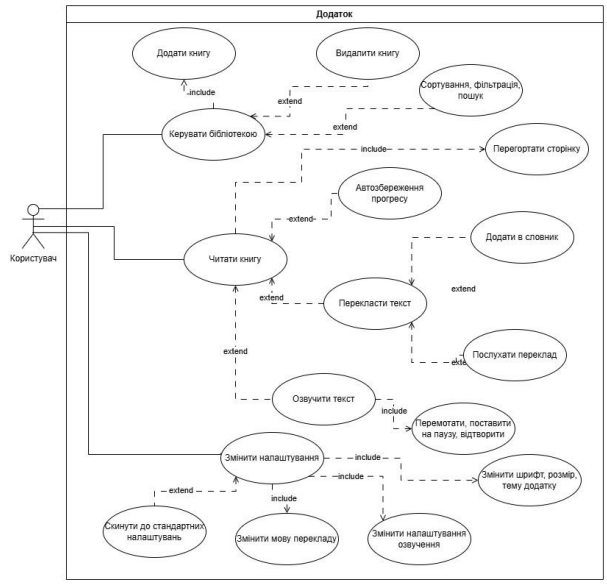


Рисунок Г.7 – Слайд презентації 7

Діаграма станів книги та діаграма розгортання

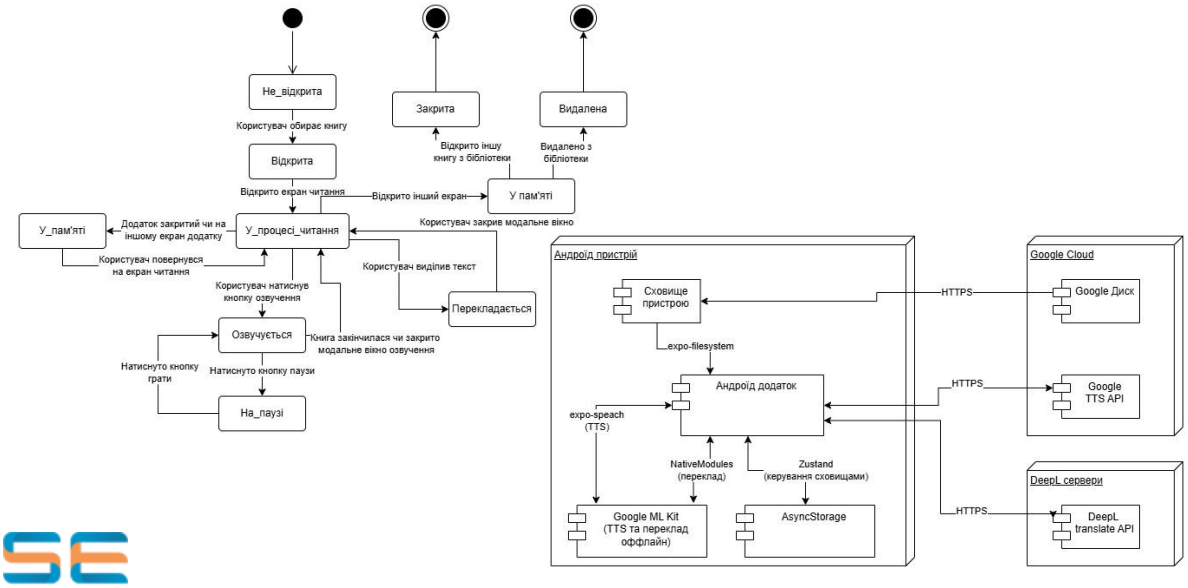
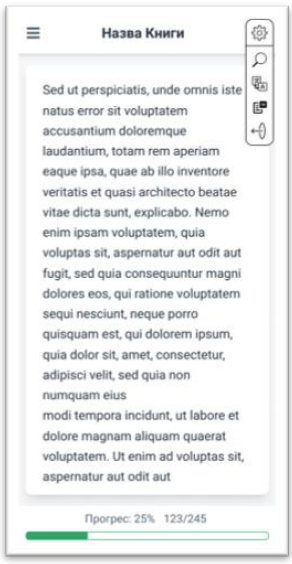


Рисунок Г.8 – Слайд презентації 8

Макет екрану «Бібліотека» та дизайн «Читати»

Макет екрану «Бібліотека» для телефона та планшета



Дизайн екрану «Читати»

9

Рисунок Г.9 – Слайд презентації 9

Вибір технологій розробки

React Native – кросплатформена технологія розробки, як основа додатку

TypeScript - мова програмування для безпечної типізації та кращої читабельності коду

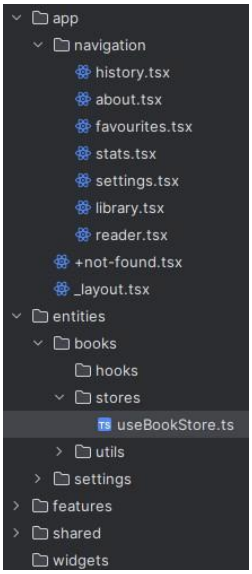
Gradle – інструмент автоматизації збирання для андроїд додатків

10

Рисунок Г.10 – Слайд презентації 10

Архітектура створенного програмного забезпечення

- Використовується Feature-Sliced Design (FSD)
- У структурі проєкту всі модулі поділяються на шари: app, widgets, features, entities, shared
- Логіка екранів знаходиться в хуках
- Глобальні стани знаходяться в сторах
- Допоміжні функції знаходиться в хелперах



11

Рисунок Г.11 – Слайд презентації 11

Опис програмного забезпечення, що було використано у дослідженні

Expo – платформа, спрощує збірку, тестування та розгортання додатків на React Native також надає великий вибір з зручних бібліотек.

Zustand - легка та продуктивна бібліотека для керування глобальними і локальними станами.

Expo-speech – бібліотека для легкого використання вбудованого в пристрій TTS.

Google ML Kit – бібліотека для використання моделей машинного навчання від гугл, використовуються моделі для перекладу.

12

Рисунок Г.12 – Слайд презентації 12

Дизайн системи

Методи

- Проєктування, орієнтоване на користувача (User-Centered Design, UCD)
- Принципи WCAG 2.1 для доступності
- Feature-Sliced Design для структурування архітектури інтерфейсу
- Тестування прототипів із цільовими користувачами

Послідовність

- Аналіз потреб користувачів (персони та сценарії)
- Побудова інформаційної архітектури (екрани, навігація)
- Створення макетів і прототипів (yFigma)
- Розробка UI за допомогою React Native
- Впровадження темізації, адаптивності, озвучення
- Проведення тестування (ручного та автоматизованого)

Рисунок Г.13 – Слайд презентації 13

Схема структури зберігання даних

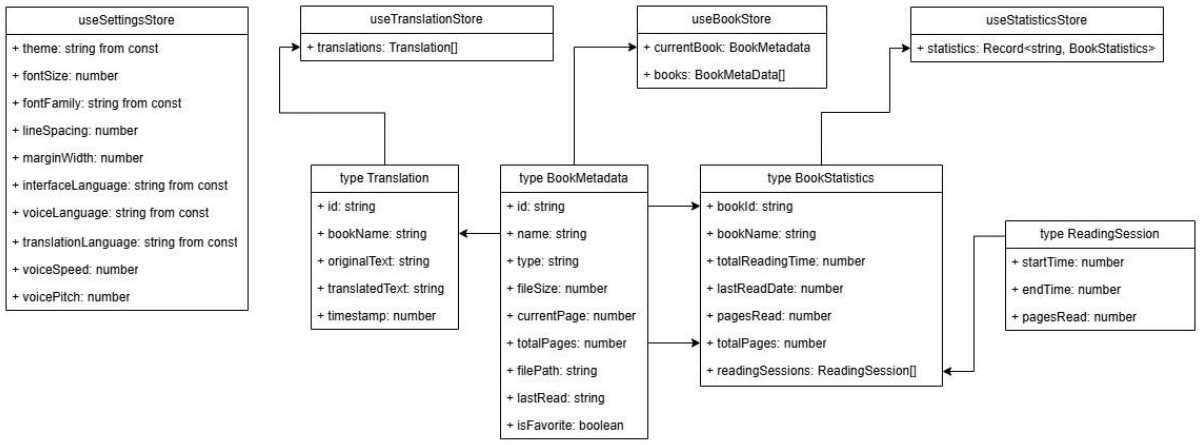


Рисунок Г.14 – Слайд презентації 14

Приклад реалізації: збереження статистики

```
const startReadingSession = (
  bookId: string,
  bookName: string,
  totalPages: number,
) => {
  const currentStats = statistics[bookId] || {
    bookId,
    bookName,
    totalReadingTime: 0,
    pagesRead: 0,
    totalPages,
    readingSessions: [],
  };

  const newStats: BookStatistics = {
    ...currentStats,
    lastReadDate: Date.now(),
    readingSessions: [
      ...currentStats.readingSessions,
      {
        startTime: Date.now(),
        endTime: 0,
        pagesRead: 0,
      },
    ],
  };

  setBookStatistics(bookId, newStats);
};
```

```
  const endReadingSession = (bookId: string, currentPage: number) => {
    const currentStats = statistics[bookId];
    if (!currentStats) {
      return;
    }

    const lastSession =
      currentStats.readingSessions[currentStats.readingSessions.length -
1];
    if (!lastSession || lastSession.endTime !== 0) {
      return;
    }

    const sessionDuration = (Date.now() - lastSession.startTime) / 1000;

    const pagesReadBeforeSession = currentStats.pagesRead;
    const sessionPagesRead = currentPage - pagesReadBeforeSession;

    const newStats: BookStatistics = {
      ...currentStats,
      totalReadingTime: currentStats.totalReadingTime + sessionDuration,
      pagesRead: currentPage,
      readingSessions: currentStats.readingSessions.map((session, index)
=>
        index === currentStats.readingSessions.length - 1
          ? { ...session, endTime: Date.now(), pagesRead:
sessionPagesRead }
          : session,
      ),
    };

    setBookStatistics(bookId, newStats);
  };
```

software engineering

15

Рисунок Г.15 – Слайд презентації 15

Приклад реалізації: озвучення книги

```
const handleResetPosition =
() => {
  if (isSpeaking) {
    Speech.stop();
    setIsSpeaking(false);
  }
  setCurrentWordIndex(0);
  setPausedPosition(0);
};
```

```
const handleSpeech = async () => {
  if (player.currentStatus.playing ||
isSpeaking) {
    player.pause();
    await Speech.stop();
    setIsSpeaking(false);

setPausedPosition(currentWordIndex);

    return;
  }
  const online = await isOnline();
  if (online) {
    try {
      await playGoogleTTS();
    } catch (error) {
      Toast.error('error' + error);
      await handleSpeechOffline();
    }
  } else {
    await handleSpeechOffline();
  }
};
```

```
const handleSpeechOffline = async () => {
  const baseText =
    selectedText || currentPageContent.replace('\\newLine', '
');

  const textToSpeak =
    pausedPosition !== 0
      ? baseText.split(/\s+/).slice(pausedPosition).join(' ')
      : baseText;
  setIsSpeaking(true);

  const speechHandlers = {
    ...(voiceLanguage.toString() !== 'auto' && { language:
voiceLanguage }),
    onDone: () => {
      setIsSpeaking(false);
      setPausedPosition(0);

      if (currentPage < (currentBook?.totalPages || 1) - 1) {
        changePage(1);
        setTimeout(() => handleSpeech(), 500);
      }
    },

    onError: () => {
      setIsSpeaking(false);
      setPausedPosition(0);
    },

    onStopped: () => {
      setIsSpeaking(false);
    },

    onBoundary: (boundaries: { charIndex: number; charLength:
number }) => {
      const { charIndex, charLength } = boundaries;
      const word = textToSpeak.substring(charIndex, charIndex
+ charLength);
      const wordIndex = words.findIndex((w) =>
w.includes(word));

      if (wordIndex !== -1) {
        setCurrentWordIndex(wordIndex);
      }
    },
  };

  Speech.speak(textToSpeak, speechHandlers);
};
```

software engineering

16

Рисунок Г.16 – Слайд презентації 16

Приклад реалізації: перекладу тексту

```
const handleTranslation =
async () => {
  const online = await
isOnline();
  if (online) {
    try {

handleOnlineTranslation();
    } catch (error) {
      Toast.error('error' +
error);

handleOfflineTranslation();
    }
  } else {

handleOfflineTranslation();
  }
};
```

```
const handleOnlineTranslation = async () => {
  if (selectedText) {
    const apiKey =
process.env.EXPO_PUBLIC_DEEPL_API_KEY;
    const params = new URLSearchParams();
    params.append('text',
selectedText.replace('\\newLine', ' '));
    params.append('target_lang',
translationLanguage.substring(0, 2));
    const response = await fetch('https://api-
free.deepl.com/v2/translate', {
      method: 'POST',
      headers: {
        Authorization: 'DeepL-Auth-Key ' + apiKey,
        'Content-Type': 'application/x-www-form-
urlencoded',
      },
      body: params.toString(),
    }).catch((error) => Toast.error('Error:', error));
    const result = await response?.json();
    if (!result.translations) {
      Toast.error('result', result);
      throw new Error(result.error?.message || 'Failed
to get translation');
    }
    setTranslatedText(result.translations[0].text);
  }
};
```

```
const handleOfflineTranslation =
() => {
  if (selectedText) {

NativeModules.Translator.translat
eWord(

selectedText.replace('\\newLine',
' '),
      translationLanguage,
    )
      .then((translated: string)
=> {

setTranslatedText(translated);
      })
      .catch((error: Error) => {
        Toast.error('Translation
error:' + error);

setTranslatedText('Translation
failed');
      });
  }
};
```


Рисунок Г.17 – Слайд презентації 17

Схема роботи алгоритму пагінації

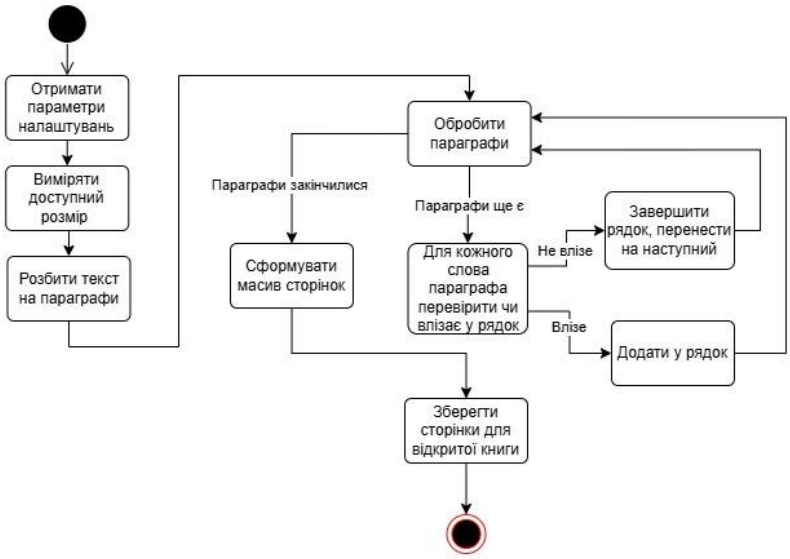


Рисунок Г.18 – Слайд презентації 18

Інтерфейс користувача

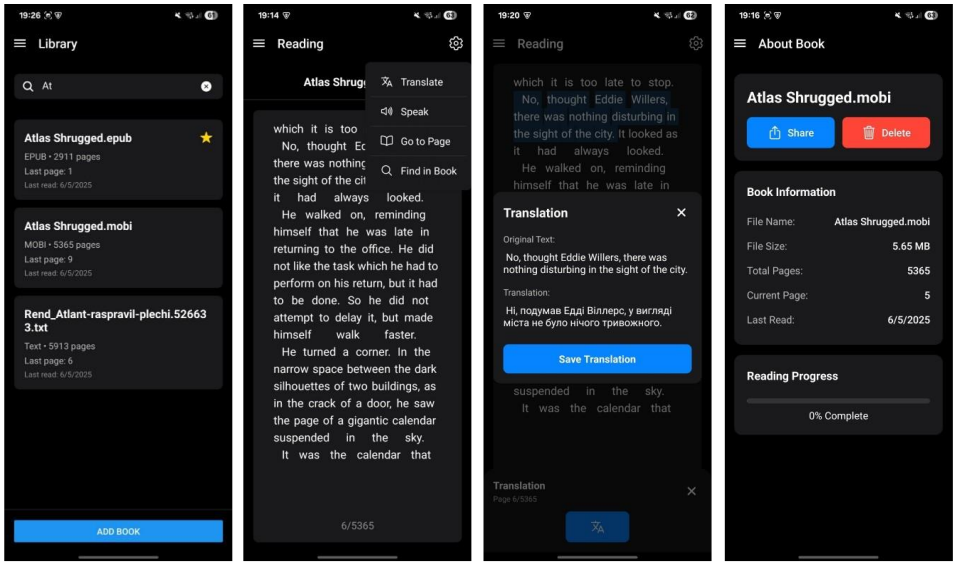


Рисунок Г.19 – Слайд презентації 19

Інтерфейс користувача

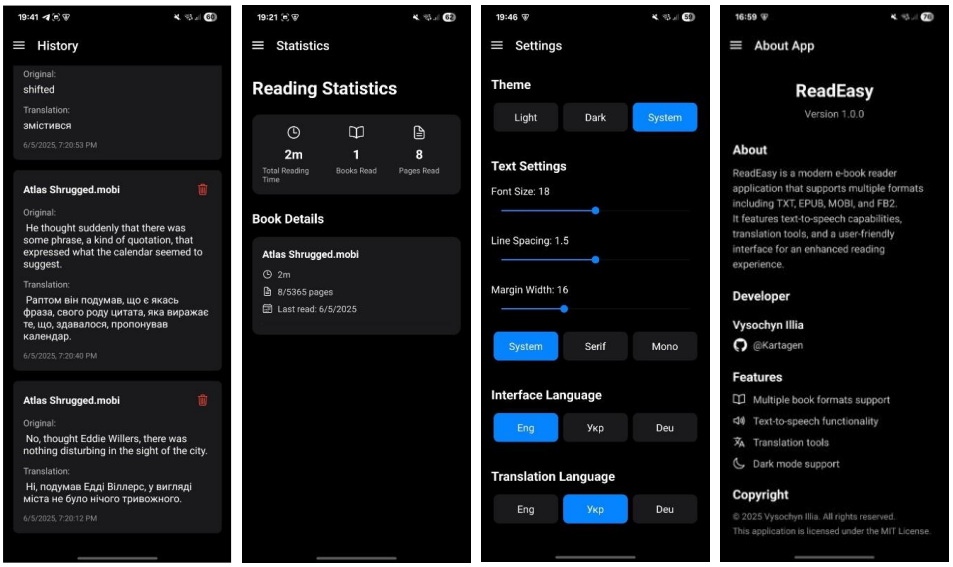


Рисунок Г.20 – Слайд презентації 20

Тестування

Мануально протестовано:

- UI/UX: інтерфейс, навігація, світла/темна теми
- Бібліотека книг: імпорт різних форматів (TXT, EPUB, FB2, MOBI), управління колекцією
- Читання: форматування тексту, жести перегортання, налаштування шрифтів
- Озвучення: онлайн/офлайн режими, виділення поточного слова
- Переклад: онлайн/офлайн режими, збереження перекладів

Критичні сценарії:

- Стабільність: робота з великими текстами (>5000 сторінок), багаторазові операції
- Виняткові ситуації: розрив з'єднання, пошкоджені файли, видалення під час читання
- Збереження прогресу: автозбереження, відновлення позиції читання

Написано тести за допомогоюJest для:

- Додавання та видалення перекладу до словника .
- Додавання, обрання улюбленою, видалення книги у бібліотеці.
- Збереження початку та кінця сесії читання для отримання статистики, оновлення кількості прочитаних сторінок.

Рисунок Г.21 – Слайд презентації 21

Публікація результатів

1. Vysochyn Illya, Michkivskyy Sergiy USING MACHINE LEARNING FOR TRANSLATION AND SPEECH GENERATION IN E-BOOK READING APPLICATIONS // Держава, регіони, підприємництво: інформаційні, суспільно-правові, соціально-економічні аспекти розвитку: збірник матеріалів VI Міжнародної конференції (5 грудня 2024 р., м. Київ): Університет "КРОК", 2024 – URL: https://conf.krok.edu.ua/SRE/SRE-2024/paper/view/2661
2. Височин І. М. Модульна архітектура у React Native-додатках / І. М. Височин, Н. С. Кравець // Радіоелектроніка та молодь у XXI столітті : матеріали 29-го Міжнар. молодіж. форуму, 16–19 квітня 2025 р. – Харків : ХНУРЕ, 2025. – Т. 6 – С. 241-243. – URL: https://openarchive.nure.ua/handle/document/30757

Рисунок Г.22 – Слайд презентації 22

Підсумки

Розроблений мобільний додаток є реалістичним та надзвичайно корисним інструментом, що ефективно долає існуючі обмеження популярних рішень для читання електронних книг. Він забезпечує інтегрований функціонал озвучування тексту та перекладу в реальному часі, а також широкі можливості персоналізації. Гібридна модель роботи з мовними сервісами (онлайн та офлайн) гарантує високу доступність та стабільність додатка за будь-яких умов підключення до інтернету

Можливості використання:

- Розширений досвід читання
- Вивчення іноземних мов
- Інклюзивність та доступність
- Широка цільова аудиторія

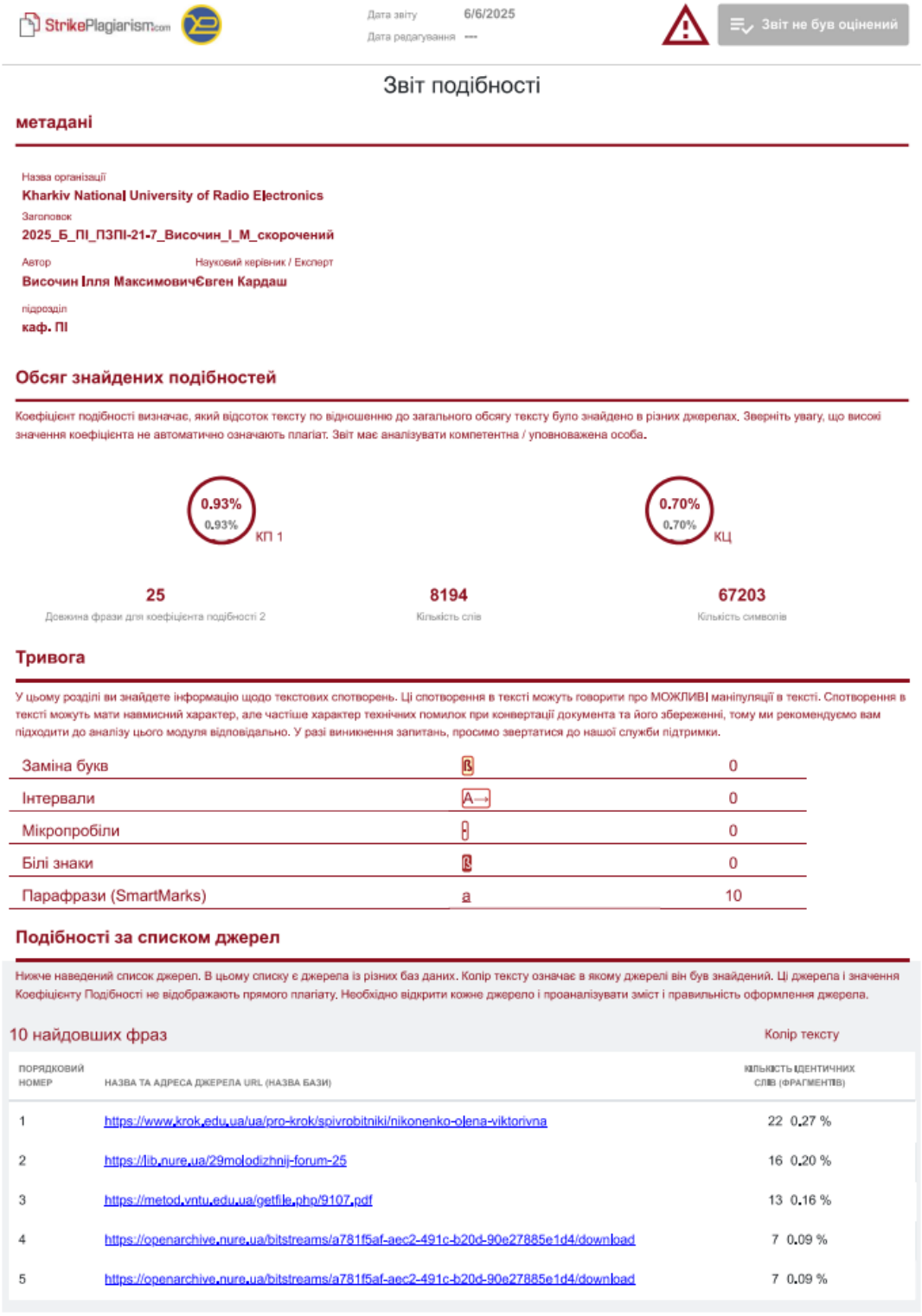
Можливий розвиток програмного забезпечення:

- Подальша інтеграція ML
- Хмарна синхронізація
- Розширення функціоналу
- Розширена мовна підтримка

software engineering

Рисунок Г.23 – Слайд презентації 23

ДОДАТОК Д
Звіт з результатами перевірки на унікальність тексту



StrikePlagiarism.com

Дата звіту 6/6/2025
Дата редагування ---

Звіт не був оцінений

Звіт подібності

метадані

Назва організації
Kharkiv National University of Radio Electronics
Заголовок
2025_Б_ПІ_ПЗПІ-21-7_Височин_І_М_скорочений
Автор
Височин Ілля Максимович
Науковий керівник / Експерт
Євген Кардаш
підрозділ
каф. ПІ

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

0.93% 0.93% КП 1

0.70% 0.70% КЦ

25	8194	67203
Довжина фрази для коефіцієнта подібності 2	Кількість слів	Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		0
Інтервали		0
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)	a	10

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://www.krok.edu.ua/ua/pro-krok/spivrobitniki/nikonenko-olena-viktorivna	22 0.27 %
2	https://lib.nure.ua/29molodizhnij-forum-25	16 0.20 %
3	https://metod.vntu.edu.ua/getfile.php/9107.pdf	13 0.16 %
4	https://openarchive.nure.ua/bitstreams/a781f5af-aec2-491c-b20d-90e27885e1d4/download	7 0.09 %
5	https://openarchive.nure.ua/bitstreams/a781f5af-aec2-491c-b20d-90e27885e1d4/download	7 0.09 %

6	https://openarchive.nure.ua/bitstreams/a781f5af-aec2-491c-b20d-90e27885e1d4/download	6 0.07 %
7	https://lib.nure.ua/29molodizhnij-forum-25	5 0.06 %

з бази даних RefBooks (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з домашньої бази даних (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з програми обміну базами даних (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з Інтернету (0.93 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://www.krok.edu.ua/ua/pro-krok/spivrobitniki/nikonenko-olena-viktorivna	22 (1) 0.27 %
2	https://lib.nure.ua/29molodizhnij-forum-25	21 (2) 0.26 %
3	https://openarchive.nure.ua/bitstreams/a781f5af-aec2-491c-b20d-90e27885e1d4/download	20 (3) 0.24 %
4	https://metod.vntu.edu.ua/getfile.php/9107.pdf	13 (1) 0.16 %

Список прийнятих фрагментів (немає прийнятих фрагментів)