Міністерство освіти і науки України
Рівненський державний гуманітарний університет
Кафедра інформаційних технологій та моделювання

Кваліфікаційна робота

за освітнім ступенем «бакалавр»

на тему:

МОБІЛЬНИЙ ЗАСТОСУНОК ДЛЯ ЧИТАННЯ КНИГ З ФУНКЦІЄЮ ОЗВУЧУВАННЯ ТЕКСТУ

Виконала:
здобувачка IV курсу
групи ІПЗ-41
спеціальності 121 «Інженерія
програмного забезпечення»
Гай Вікторія Іванівна

Науковий керівник:
к.т.н., Шевцова Н. В.

Рівне – 2026

ЗМІСТ

ВСТУП

Актуальність роботи. У сучасному світі цифрові технології активно розвивають способи споживання інформації. Зокрема зростає популярність електронного читання, що забезпечує мобільність, доступність та зручність використання літератури на різних пристроях.

Одним із перспективних напрямів розвитку цифрових сервісів є поєднання текстового контенту з технологіями синтезу мовлення. Функція озвучування тексту розширює можливості користувачів, дозволяючи слухати аудіоконтент під час навчання або повсякденної діяльності, а також підвищує доступність книг для людей із порушеннями зору або труднощами сприйняття друкованого тексту. В цілому, попит на інструменти для створення аудіокниг зростає разом із розвитком дистанційного навчання та самонавчання.

У зв’язку з цим розробка мобільного застосунку для читання книг із функцією озвучування тексту є **актуальним завданням**, що відповідає сучасним тенденціям розвитку інформаційних технологій та потребам користувачів.

Метою роботи є: розробка мобільного застосунку для читання електронних книг із функцією озвучування тексту на основі технологій синтезу мовлення, що забезпечує зручність використання, доступність та ефективність сприйняття інформації користувачами.

Для досягнення сформульованої мети були поставлені та вирішені наступні **завдання**:

- проаналізувати сучасні підходи до створення мобільних застосунків для читання електронних книг;
- дослідити технології синтезу мовлення та можливості їх інтеграції в мобільні платформи;
- реалізувати програмну частину застосунку;
- оцінити ефективність роботи розробленого рішення.

Об’єкт дослідження: процес організації електронного читання текстової інформації з використанням мобільних технологій.

Предмет дослідження: методи та засоби розробки мобільного застосунку для читання книг із функцією синтезу мовлення.

Методи дослідження:

- аналіз наукових та технічних джерел;
- порівняння існуючих програмних рішень;
- методи проєктування програмного забезпечення.

Практичне значення дослідження. Створення мобільного застосунку, який може використовуватися для зручного читання та прослуховування електронних книг, що розширює можливості користувачів та підвищує доступність цифрового контенту.

Апробація і впровадження результатів дослідження. Основні теоретичні та практичні результати дослідження даної кваліфікаційної роботи обговорювалися на XIX Всеукраїнській науково-практичній конференції здобувачів вищої освіти і молодих учених «Наука, освіта, суспільство очима молодих» (м. Рівне, 14 травня 2026 р.). Також результати дослідження доповідалися на звітній конференції викладачів, аспірантів та здобувачів освіти РДГУ за 2024 рік (м. Рівне, 14 травня 2026 р.).

Структура роботи складається зі вступу, трьох розділів, висновків, списку використаних джерел, додатків. У першому розділі кваліфікаційної роботи проведено аналіз предметної області та сучасних TTS-систем. У другому розділі здійснено аналіз вимог до застосунку та обґрунтовано вибір технічних рішень для його реалізації. У третьому розділі описано процес практичної реалізації застосунку.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ РІШЕНЬ

1.1. Електронне читання та аудіокниги як сучасні форми споживання літературного контенту

Електронне читання є сучасною формою взаємодії користувача з текстовою інформацією, що здійснюється за допомогою пристроїв та програмних платформ. Воно передбачає використання електронних книг, які являють собою текстові документи у цифровому форматі, адаптовані для відображення на екранах комп'ютерів, планшетів, смартфонів або спеціалізованих пристроїв для читання.

Перші спроби комерціалізації ідеї цифрової книги пов'язані з появою таких пристроїв, як SoftBook та Rocket E-Book на початку 1990-х років. Попри інноваційність, ці моделі не здобули комерційного успіху.

Ситуація кардинально змінилася лише у 2006-2007 роках із виходом Sony Reader та, особливо, Amazon Kindle. Останній став революційним завдяки інтеграції з найбільшим у світі магазином електронних книг. Подальша поява Barnes & Noble Nook (2009) та багатофункціонального Apple iPad (2010) остаточно сформувала структуру сучасного ринку [3].

Статистичні дані свідчать про стрімку популяризацію формату: лише за 2011 рік було реалізовано близько 60 млн пристроїв. Кількість виробництва призвела до суттєвого зниження вартості як самих рідерів, так і контенту [3].

Сучасні застосунки для читання пропонують користувачу можливості, які були недоступні в епоху традиційного читання книг:

- здатність компактного портативного пристрою зберігати тисячі видань одночасно;
- вбудовані словники для швидкого перекладу, повнотекстовий пошук, система гіперпосилань та гнучке налаштування, наприклад, зміна шрифтів, інтервалів;
- впровадження програм синтезу мовлення, анімаційних елементів та можливості відтворення аудіоконтенту;

- низька вартість розповсюдження;
- екологічність [2].

Ефективність програмної обробки електронної книги залежить від її формату та внутрішньої структури. Класифікацію видань доцільно проводити за наступними ознаками:

1. *За логічною структурою формату:*

- адаптивні – формати, що базуються на мовах розмітки (XML/HTML);
- статичні – формати, що зберігають точне позиціонування елементів незалежно від пристрою;
- пропрієтарні – закриті формати, розроблені для власних екосистем.

2. *За рівнем інтерактивності та типом контенту:* текстові, інтерактивні (включають мультимедіа), аудіокниги [1].

Розвиток цифрових технологій та електронного читання сприяв появі нових форматів споживання текстової інформації. Одним із найбільш перспективних напрямків стали аудіокниги. Цей формат забезпечує користувачам можливість сприймати літературний контент у звуковому вигляді, що сприяє не обмежуватися традиційним візуальним читанням та навіть знімає навантаження на органи зору.

На сьогоднішній день аудіокниги суттєво набирають популярності серед різних верств населення. Вони дедалі частіше розглядаються як зручна та мобільна альтернатива не лише друкованим виданням, а й навіть електронним. Сучасні дослідження показують, що прослуховування аудіокниг є більш ефективним з точки зору економії часу. Це зумовлено тим, що сприйняття тексту на слух дає змогу поєднувати процес ознайомлення літературою з іншими видами діяльності, наприклад, виконанням побутових справ чи заняттям спортом.

Розвиток цього формату розпочався завдяки компанії Audible, яка у 1990-х роках створила портативний пристрій для прослуховування аудіокниг. Початок XXI століття ознаменувався стрімким переходом аудіокниг у цифровий формат, адаптований до мобільних пристроїв [4].

В Україні історія аудіокниг має власні особливості розвитку. Перші записи створювалися з метою забезпечення доступу до літератури для людей із

порушеннями зору. Уже у 1960-х роках у бібліотеці Українського товариства сліпих існувала значна колекція аудіокниг, яка систематично поповнювалася протягом наступних десятиліть [4].

Традиційний процес створення аудіокниги є високо витратним, оскільки передбачає залучення професійних дикторів. Однак, розвиток інформаційних технологій відкрив шлях до автоматизації цього процесу. Впровадження систем синтезу мовлення дозволяє трансформувати текстові масиви у звукові файли, що значно знижує собівартість створення контенту та робить його доступним у режимі реального часу.

Підсумовуючи аналіз предметної області, можна зробити висновок, що сучасний ринок літературного контенту характеризується поєднанням текстових та аудіо форматів. У сучасних умовах ці дві форми споживання контенту дедалі більше інтегруються, зокрема, за допомогою технологій синтезу мовлення, які дозволяють автоматично перетворювати текстові матеріали на звуковий формат та інтегрувати цю функцію безпосередньо у застосунки для читання.

1.2. Технологічні стандарти та формати цифрового контенту

Для розробки програмного забезпечення для читання ключовим аспектом є підтримка кросплатформних форматів, що забезпечують коректну візуалізацію контенту на різних типах пристроїв. *Формати eBook* – це типи цифрових файлів, у яких зберігаються книги для читання на електронних пристроях. Різні формати краще працюють на певних пристроях або платформах і відрізняються можливостями, наприклад, інтерактивністю, переформатованим (reflowable) текстом чи фіксованим макетом [10].

EPUB є відкритим стандартом, що базується на технологіях веб-розмітки (XML, HTML, CSS). З точки зору архітектури, файл формату EPUB являє собою ZIP-архів, що містить: набір XHTML-файлів, адаптує текст під розмір екрана користувача; стилі CSS, які забезпечують гнучке керування шрифтами, відступами; файл з інформацією про структуру книг, автора, мову (OPF); навігаційний файл (NCX) – програмна інструкція [11].

На відміну від EPUB, *PDF* базується на мові опису сторінок PostScript. Основною перевагою формату є фіксована верстка, що гарантує ідентичне відображення макета незалежно від програмної платформи. Проте для мобільних додатків цей формат є складнішим у використанні через відсутність адаптивності до малих екранів, що вимагає впровадження додаткових алгоритмів масштабування.

Формат *MOBI* є популярним серед користувачів пристроїв Kindle. Він простий, має компактний розмір файлу, підтримує базові зображення. Його особливістю є підтримка інструментів для створення нотаток та закладок на рівні файлової структури. Однак, формат не працює з розширеними можливостями, такими як відео чи складні макети [10]. Через закритість формату та поступовий перехід Amazon до AZW3, підтримка MOBI у сторонніх додатках часто реалізується за залишковим принципом.

AZW та *AZW3* – це сучасні пропрієтарні формати Amazon, які є оновленою версією MOBI. Вони підтримують складну верстку, вбудовані шрифти та мультимедійні елементи, аналогічні до EPUB, але захищені DRM-системами (Digital Rights Management), що обмежує їх використання в універсальних читалках. AZW3 добре підходить для складних eBook із зображеннями чи інтерактивними елементами, але обмежений екосистемою Amazon. [10].

TXT та *RTF* є найпростішими текстовими форматами, що не мають структурованих метаданих та засобів розмітки. Їх перевагою є мінімальний розмір файлу та абсолютна сумісність з будь-якою ОС проте вони не забезпечують необхідного рівня комфорту для електронного читання.

Формат *FB2* використовують для електронних книг на багатьох рідерах і мобільних додатках. Його популярність серед розробників мобільних застосунків зумовлена використанням мови розмітки XML, що забезпечує високу гнучкість при парсингу та рендерингу тексту. FB2 не підтримує складне форматування, проте його розмітка дозволяє зберігати логіку тексту та адаптувати його під різні пристрої [11].

Для обґрунтування вибору технології розроблюваного додатка було

проведено порівняльний аналіз найбільш поширених форматів за ключовими критеріями: технічна база, адаптивність та зручність інтеграції в мобільні системи.

Таблиця 1.1

Технологічні параметри форматів електронних книг

Критерій	**Формати**		
	EPUB	PDF	FB2
Базова архітектура	XML, HTML5, CSS	PostScript	XML
Тип верстки	Адаптивна	Фіксована	Адаптивна
Підтримка метаданих	Висока	Середня	Висока
Мультимедіа	Повна підтримка	Обмежена	Відсутня
Складність парсингу	Середня	Висока	Низька
Збереження стилів	Повне	Повне	Базове

За аналізом результатів порівняння, визначено, що EPUB є пріоритетним форматом для розробки, оскільки він реалізовує функціонал Text-to-Speech найбільш ефективно за рахунок чіткої розмітки тексту. PDF залишається необхіним для підтримки через його універсальність, хоча він створює додаткове навантаження на апаратні ресурси смартфона при рендерингу великих документів. FB2 є допоміжним форматом, що підходить для реалізації легкого режиму читання.

На відміну від текстових форматів, де ключовим є метод розмітки (XML/HTML), формати аудіокниг класифікуються за методами кодування аудіосигналу та підтримкою специфічних функцій метаданих (закладки, глави, обкладинки).

MP3 – найбільш поширений формат, що базується на алгоритмі стиснення з втратами. Перевагою є універсальність підтримки будь-якими медіаплеєрами, малий розмір файлів за умови прийнятної якості для мовлення [12]. Проте відсутня

вбудована підтримка ієрархічної структури глав (аудіокнига зазвичай розбивається на десятки окремих файлів). Але на відміну від форматів WAV або FLAC, формат MP3 не потребує значного обсягу пам’яті, що робить його більш придатним для прикладних користувацьких застосунків.

M4B – спеціалізований формат для аудіокниг, що використовується переважно в екосистемі Apple. Використовує кодек AAC (Advanced Audio Coding) [13], що забезпечує вищу якість звуку при меншому бітрейті порівняно з MP3. Головною відмінністю є підтримка метаданих глав та збереження «точки зупинки» (bookmarks) всередині одного великого файлу. Це надає змогу зберігати всю книгу в одному файлі без втрати навігації.

Таким чином, проведений аналіз технологічних стандартів електронного та аудіо контенту свідчить про значну фрагментацію форматів, кожен з яких має свої архітектурні особливості. Зокрема, можливість програмного витягу тексту дає змогу перейти від візуального сприйняття до генерації аудіоконтенту за допомогою синтезу мовлення (Text-to-Speech).

1.3. Технології синтезу мовлення у сучасних інформаційних системах

Сучасні аудіокниги дедалі частіше стають популярними завдяки своїй зручності та мобільності. Якщо на початку розвитку аудіокниг використовувалися фізичні носії та записи живого голосу, то з розвитком цифрових технологій з’явилася можливість автоматичного створення мовлення з книги за допомогою спеціальних програмних систем. Цей процес отримав назву синтез мовлення з тексту або *Text-to-Speech (TTS).* Більшість сучасних рішень реалізуються у вигляді прикладних програмних інтерфейсів (API), що дозволяє інтегрувати їх у мобільні та веб застосунки.

Технологія синтезу мовлення є процесом автоматичного перетворення тексту на усну мову. Вона приймає текстові дані в будь-якій формі, від окремого речення, до цілого документа, і генерує звуковий сигнал, який сприймається як мова. Сучасні системи TTS здебільшого намагаються створити голос, максимально наближений до людського, хоча рівень природності може відрізнятися залежно від

конкретного продукту. У ранніх системах синтезу мовлення голос мав певну роботизованість: перші електричні синтезатори мови з'явилися приблизно в 1930-х роках, мали обмежені можливості та в цілому були складними в експлуатації [5].

З появою комп'ютерів програмісти, починаючи з кінця 1950-х років, працювали над алгоритмами, здатними використовувати великі бази даних аудіофайлів як джерело звуків. Ці алгоритми могли знаходити звукові відповідності для фрагментів тексту та складати з них елементи мовлення. Спочатку синтезований голос звучав механічно. У міру того, як моделювання мови ставало точнішим, алгоритми перетворення тексту в мову вдосконалювалися [5].

Подібно до інших технологій, TTS еволюціонує під впливом розвитку алгоритмів штучного інтелекту та машинного навчання. Впровадження методів глибокого навчання та нейронних мереж дозволило моделювати звукові хвилі безпосередньо на основі реальних записів. Це призвело до появи високоякісних голосів, які важко відрізнити від людських.

Процес синтезу мовлення умовно поділяють на декілька основних етапів (рис. 1.1).

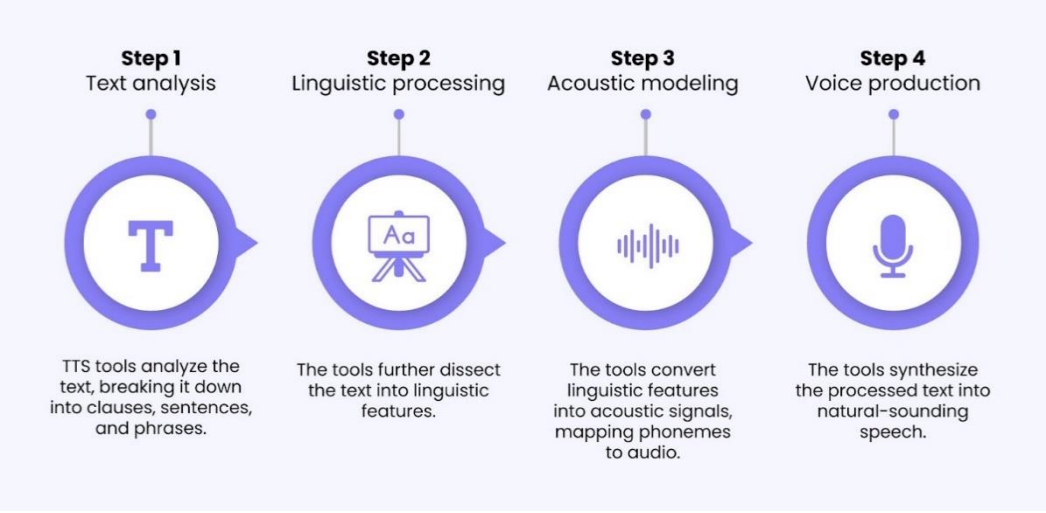

Рисунок 1.1. Основні етапи синтезу мовлення (Text-to-Speech)

На початковому етапі роботи системи TTS йде підготовка тексту до мовлення. Цей процес включає аналіз тексту – система сканує текст для визначення його структури, враховуючи знаки пунктуації, абревіатури, числа та інші елементи.

Це дає змогу правильно зрозуміти контекст і уникнути неоднозначностей. Наприклад абревіатура «Dr.» коректно розпізнається як «Доктор», а не «Драйв». Далі – фонетичне розбиття – слова розкладаються на найменші одиниці звуку – фонеми. Даний крок забезпечує точність вимови. Наприклад, слово «cat» поділяється на три фонеми: /k/, /æ/ і /t/. Наступним кроком є контенстуальна обробка – система визначає правильну вимову слів залежно від контексту. Наприклад, слово «lead» вимовляється по-різному в словосполученнях «lead a team» і «lead pipe» [6].

Наступним етапом є синтез мовлення, що передбачає генерацію звуку. Синтез перетворює письмовий текст на людську мову, розбиваючи його на менші одиниці звуку та призначаючи їм звукові параметри. Параметри звуку містять інформацію, необхідну для генерації форми сигналу, яка визначає вихідну мову. Програмне забезпечення для перетворення тексту в мовлення містить необхідну інформацію для створення мови, яка має ритми та інтонації, які є правдоподібно людськими.

Існують два основних підходи – *конкатенативний синтез* та *Neural TTS* (нейронний синтез). Конкатенативний синтез – традиційний метод, який передбачає об’єднання попередньо записаних фрагментів людської мови у повні речення. Наприклад, для фрази «Hello, world» система поєднує окремі записи слів «Hello» та «world». Основним недоліком цього методу є уривчастість або роботизованість мови при складних конструкціях. Нейронний синтез, в свою чергу, є сучасним методом, що використовує штучний інтелект та глибоке навчання для генерації мовлення «з нуля». Такий підхід дозволяє створювати природну та емоційну мову. Наприклад, фраза «Привіт, світе» буде відтворена в природному темпі та тоні, з емоційною забарвленістю, наближеною до людської [6].

Завершальний етап – поліпшення та додаткові ефекти. На заключному етапі система TTS вносить коригування для підвищення природності мовлення: тон і висота голосу – допомагають передати емоційне забарвлення або акцент, наприклад, хвилювання виражається вищим тоном, серйозність – нижчим. Темп мовлення – регулює швидкість проголошення слів відповідно до природних

мовних патернів. Дихання та паузи – сучасні системи імітують природні паузи та дихальні звуки, що значно підвищує реалістичність синтезованого мовлення [6].

Отже, технологія TTS пройшла складний шлях розвитку: від простих роботизованих голосів до сучасних нейронних систем, здатних відтворювати природне мовлення з високим рівнем емоційності та виразності.

1.4. Аналіз мобільних застосунків для читання та прослуховування книг

У процесі розробки мобільного застосунку для читання книг з функцією озвучування тексту, доцільно проаналізувати існуючі рішення з огляду на їх функціональні можливості та відповідність потребам користувача. Огляд популярних застосунків дозволяє визначити підходи до організації бібліотеки, реалізацію синтезу мовлення, механізми синхронізації даних, а також вимоги до інтерфейсу користувача.

Одним з прикладів багатофункціонального мобільного застосунку для читання електронних книг є PocketBook Reader. Даний програмний продукт забезпечує роботу з електронним контентом різних форматів та поєднує функції читання і прослуховування тексту. Застосунок підтримує 26 популярних форматів, серед яких FB2, PDF, EPUB, DJVU, MOBI, TXT, DOCX, а також формати коміксів (CBZ і CBR) та аудіофайлів (MP3, M4B) [7]. Завдяки широкій підтримці форматів можна працювати з різними типами цифрового контенту без необхідності конвертації.

Функціональні можливості застосунку включають реалізацію синтезу мовлення, що забезпечує озвучування тексту. Додатково передбачено роботу зі словниками та перекладачами, створення закладок, написання нотаток і обміну ними через месенджери. Усі матеріали можуть зберігатися у хмарному сховищі з можливістю синхронізації між пристроями. Реалізована інтеграція з популярними сервісами, такими як Dropbox та Google Drive, що дає можливість створити єдину цифрову бібліотеку [7].

У застосунку реалізовано механізми пошуку файлів у пам’яті пристрою, пошук книг за ISBN-кодом, функції сортування, фільтрації та створення

тематичних колекцій. Інтерфейс застосунку характеризується мінімалістичним дизайном та широкими можливостями персоналізації.

Таким чином, PocketBook Reader є прикладом комплексного рішення, що поєднує функції читання, прослуховування та управління електронною бібліотекою.

NaturalReader є одним із найпоширеніших безкоштовних застосунків для перетворення тексту в мовлення, що пропонує широкий спектр інструментів для створення чіткого та природного аудіоконтенту. Система підходить для створення навчальних матеріалів, інтернет-контенту та аудіокниг.

NaturalReader має велику базу реалістичних голосів. Застосунок пропонує сотні штучно створених голосів із підтримкою більш ніж 50 мов. Доступні різні стилі звучання, що дозволяє адаптувати аудіоконтент під специфіку цільової аудиторії. У платних версіях NaturalReader підтримується створення індивідуальних голосових моделей, що може бути корисним для брендованих озвучень, подкастів, відеоматеріалу або професійних аудіокниг. Вбудована студія забезпечує високоякісний рендерінг аудіофайлів. Додаток може працювати з різними форматами документів, включно з PDF, Word та вебсторінками. Також підтримує технологію оптичного розпізнавання тексту (OCR) для оцифрування друкованих матеріалів. Доступним є спеціальний ШІ-фільтр, що автоматично пропускає цитати, таблиці, виноски чи другорядні елементи під час прослуховування наукових текстів. Це підвищує зручність роботи з великими академічними документами та сприяє концентрації на основному змісті [9].

З недоліків NaturalReader є те, що під час опрацювання великих проєктів можливе зниження продуктивності, зокрема зависання або затримки в роботі програми. Безкоштовна версія не дозволяє використовувати синтезоване мовлення для комерційних цілей. Також відсутня можливість експорту аудіо у форматі MP3 у безкоштовному тарифі.

Bookmate – це мобільний застосунок, що надає доступ до великої електронної бібліотеки текстових та аудіокниг різними мовами. Застосунок підтримує завантаження власних файлів у форматах EPUB та FB2, що розширює можливості

формування персональної бібліотеки. Завантажені матеріали доступні як у режимі онлайн, так і офлайн [7].

Функціональні можливості Bookmate включають систему персоналізованих рекомендацій, що формується на основі аналізу читацької активності користувача. Для навігації бібліотекою реалізовано тематичні добірки та механізми пошуку. Передбачено створення нотаток, збереження цитат, формування власних книжкових підбірок.

Важливою особливістю сервісу є наявність соціальних елементів взаємодії: користувачі мають можливість публікувати цитати, ділитися враженнями, формувати рекомендації та взаємодіяти між собою в межах платформи.

Незважаючи на широкий функціонал та наявність аудіоконтенту, застосунок не реалізує повноцінну функцію синтезу мовлення для озвучування довільного тексту електронних книг. Користувач має змогу прослуховувати лише готові аудіокниги з можливістю регулювання швидкості відтворення. Дані про прогрес читання, позицію в тексті та персональні налаштування синхронізуються між пристроями користувача.

Speechify – це популярний сервіс для перетворення тексту в мовлення, орієнтований на створення реалістичного аудіоряду для навчальних та бізнесових потреб. Платформа поєднує синтез мовлення з додатковими інструментами продуктивності на основі ШІ, що робить її універсальним рішенням для користувачів, які працюють з великими обсягами текстів. Speechify пропонує понад 1000 голосів на більш ніж 60 мовах, включно з акцентами, діалектами та спеціальними стилями озвучення. На основі алгоритмів ШІ Speechify може автоматично створювати короткі виклади великих документів та генерувати вікторини для перевірки розуміння тексту. Користувач може адаптувати швидкість та стиль озвучення, змінювати паузи та інтонації. Також Speechify дозволяє сканувати паперові документи, книги або сторінки за допомогою камери чи завантаження файлів. Розпізнаний текст можна одразу перетворити на аудіо. Сервіс підтримує PDF, Google Docs, вебсторінки та інші текстові формати [9].

Платформа має підтримку командної роботи та корпоративних функцій. Для

бізнесу доступні: SSO (Single Sing-On), відповідність стандарту безпеки SOC 2, командні робочі простори, необмежене завантаження відео. Завдяки цьому Speechify підходить підприємствам, що працюють з великими масивами різноманітних даних [9].

Недоліком є те, що у безкоштовному плані доступні лише базові голоси та обмежений функціонал. Інколи можуть траплятися помилки у вимові або темпі озвучення, особливо для рідкісних мов або безкоштовної моделі голосів.

Voice Aloud Reader – це мобільний застосунок для операційної системи Android, призначений для озвучування текстової інформації за допомогою технології синезу мовлення [8]. На відміну від стандартних TTS-програм, застосунок орієнтований не лише на відтворення окремих фрагментів тексту, а й на комплексну роботу з різними типами документів і джерел інформації.

Застосунок підтримує широкий спектр форматів, зокрема вебсторінки, електронні листи, PDF-документи, файли DOC, EPUB, MOBI та звичайні текстові файли. Завдяки цьому, програму можна використовувати як універсальний інструмент для озвучування електронних книг та інших текстових матеріалів.

Однією з ключових особливостей є можливість формування списків відтворення. Користувач може додавати декілька документів або статей для послідовного прослуховування без перерви, що є зручним при роботі з великими обсягами інформації.

Застосунок забезпечує гнучкі налаштування параметрів мовлення, зокрема регулювання швидкості, висоти тону та гучності голосу. Передбачений експорт озвученого тексту в аудіофайли форматів WAV або OGG для подальшого використання на інших пристроях.

Важливою особливістю є сумісініть із будь-яким встановленим у системі Android TTS-движком. Якість озвучування залежить від обраного механізму синтезу мовлення.

Додатково реалізована інтеграція з браузером та підтримка використання в автомобільних системах на базі Android Auto, що розширює сценарії застосування програми.

1.5. Порівняльний аналіз функціональних можливостей існуючих рішень

Для проведення порівняльного аналізу сучасних мобільних застосунків для електронного читання та озвучування тексту було визначено систему критеріїв оцінювання. Обрані критерії дозволяють комплексно врахувати які функціональні можливості застосунків, так і якість реалізації технології синтезу мовлення.

1. *Якість синтезованого мовлення:* важливою характеристикою є природність звучання голосу, правильність інтонації та чіткість вимови. Якісний механізм Text-to-Speech повинен забезпечувати комфортне тривале прослуховування тексту без надмірної монотонності або спотворень.
2. *Підтримка форматів електронних книг:* застосунок має забезпечувати коректну роботу з популярними форматами (EPUB, PDF, FB2 тощо), оскільки від цього залежить доступність контенту для користувача.
3. *Підтримка багатомовності:* наявність декількох мов інтерфейсу та можливість озвучування текстів різними мовами розширює сферу застосування програми та підвищує її універсальність.
4. *Зручність користувалького інтерфейсу:* інтерфейс повинен бути інтуїтивно зрозумілим та забезпечувати швидкий доступ до бібліотеки, налаштувань читання та функції озвучування тексту.
5. *Синхронізація та збереження даних:* збереження закладок, нотаток і позиції читання з подальшою синхронізацією між пристроями.
6. *Кросплатформність:* наявність версій для різних операційних систем або доступ через вебплатформу підвищує доступність застосунку.

У межах дослідження були протестовані найпоширеніші інструменти для електронного читання та аудіовідтворення. Їх оцінювання здійснювалося за шістьма перерахованими вище показниками. Такий підхід дозволив сформувати об’єктивне уявлення про ефективність кожного продукту та відібрати найбільш придатні рішення для розробки.

Таблиця 1.2

Порівняльна таблиця існуючих рішень

Критерій	**Застосунки**				
	PocketBook Reader	NaturalReader	Bookmate	Speechify	Voice Aloud Reader
Якість TTS	Середня	Висока	Відсутня	Висока	Залежить від TTS
Підтримка форматів	Дуже широка (FB2, PDF, EPUB тощо)	PDF, DOC, веб	EPUB, FB2	PDF, Google Docs, веб	Широка (веб, PDF, DOC, тощо)
Багатомовність	Присутня	50+ мов	Присутня	60+ мов	Залежить від TTS
Інтерфейс	Зручний	Середній	Дуже зручний	Сучасний	Функціональний
Синхронізація	Присутня	Обмежена	Присутня	Присутня	Відсутня
Кросплатформність	Android, IOS	Web, Widows, Android, IOS	Android, IOS	Web, Anrdoid, IOS	Android, IOS

Аналіз отриманих результатів показує, що універсальні застосунки для читання, наприклад, PocketBook Reader, поєднують широкий функціонал роботи з текстом, але мають обмежені можливості синтезу мовлення. У той же час спеціалізовані TTS-сервіси, такі як Speechify та NaturalReader, забезпечують високу якість озвучування, проте не мають повноцінної системи електронного читання.

Отож, існуючі рішення не забезпечують оптимального поєднання функцій читання та озвучування тексту, що обумовлює доцільність розробки власного мобільного застосунку.

РОЗДІЛ 2
ПРОЄКТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ ЧИТАННЯ КНИГ ІЗ ФУНКЦІЄЮ СИНТЕЗУ МОВЛЕННЯ

2.1. Формування вимог до програмної системи та вибір технологій для розробки

У процесі розробки мобільного застосунку для читання книг з функцією озвучування тексту було визначено основні вимоги до програмної системи. Дані вимоги формуються на основі аналізу потреб користувачів, сучасних тенденцій у сфері мобільних застосунків для читання, а також технічних можливостей обраної платформи Android.

До *функціональних вимог* належать основні можливості, які має реалізовувати застосунок:

- завантаження книг – користувач повинен мати змогу завантажувати книги у форматах EPUB, FB2, TXT та DOCX з пам’яті пристрою;
- збереження книг у бібліотеці – додані файли повинні зберігатися у внутрішній бібліотеці застосунку;
- відображення тексту книги – для текстових файлів реалізується відображення тексту у вигляді сторінок;
- озвучення тексту – користувач повинен мати можливість прослуховувати текст книги за допомогою вбудованого синтезу мовлення;
- створення аудіокниги – користувач може завантажити книгу та озвучити її за допомогою API;
- налаштування аудіокниги – користувач повинен мати можливість налаштовувати параметри озвучування аудіо, зокрема обирати голос синтезу мовлення та додаткові характеристики;
- формування аудіобібліотеки – озвучені книги повинні відображатися у відповідному розділі застосунку;
- керування відтворенням аудіо – присутність функції запуску, паузи,

перемотування та зміни швидкості.

Нефункціональні вимоги визначають якість та характеристики роботи системи:

- зручність використання – інтерфейс повинен бути простим, зрозумілим та інтуїтивним для користувача;
- продуктивність – застосунок має швидко відкривати файли та обробляти текст без значних затримок;
- стабільність роботи – система повинна коректно обробляти помилки;
- сумісність – застосунок повинен працювати на більшості сучасних Android-пристроїв;
- масштабованість – архітектура має дозволяти подальше розширення.

Окремою вимогою є зручність та візуальна привабливість інтерфейсу користувача. Інтерфейс застосунку має бути виконаний у єдиному стилі з чітким поділом основних функціональних зон. Візуальні елементи інтерфейсу повинні забезпечувати зворотній зв’язок з користувачем, зокрема інформувати про активний стан елементів управління виконання дій.

Під час розробки застосунку було здійснено вибір відповідних технологій та інструментів, які забезпечують ефективну реалізацію поставлених вимог, зручність розробки та подальшу масштабованість продукту. Вибір мови програмування Java обумовлюється тим, що вона є однією з основних мов для створення Android-застосунків. Дана мова має широку підтримку в середовищі Android Studio, велику кількість документації та прикладів. Java стабільна та перевірена роками мова у розробці мобільних застосунків та сумісна із більшістю Android API. Незважаючи на існування сучаснішої мови Kotlin, Java була обрана з огляду на її зрозумілість та наявний досвід використання.

У якості середовища розробки використовується Android Studio – офіційна інтегрована середа розробки для платформи Android. Вона забезпечує зручний редактор коду, інструменти для створення інтерфейсу. Також має вбудований емулятор Android-пристроїв.

Архітектура проєкту сприяла чіткому розподілу відповідальності між

активностями та моделями даних. Графічний інтерфейс користувача побудований на основі XML-розмітки, яка є стандартом для Android-розробки.

Для збереження метаданих про книги, шляхів до файлів та прогресу прослуховування було використано SharedPreferences. Дане рішення зберігає дані у форматі «ключ-значення».

Вибір вхідних форматів даних для застосунку базувався на детальному аналізі сучасних стандартів зберігання інформації та потреб цільової аудиторії, проведеному у першому розділі роботи. Результати дослідження підтвердили, що для досягнення зручності користування необхідно орієнтуватися на формати EPUB та FB2. Формати TXT і DOCX також підтримуються через їх універсальність та розповсюдженість.

Android TextToSpeech API – використовується для реалізації фукнції базового озвучування тексту кожної окремої сторінки під час читання книги.

Для реалізації функції перетворення книги в аудіокнигу в розроблюваному застосунку було обрано сервіс синтезу мовлення ElevenLabs. ElevenLabs є сучасною вебплатформою на базі штучного інтелекту, призначеною для синтезу мовлення з високим рівнем природності звучання. Сервіс спеціалізується на генерації голосу з тексту та використовує моделі глибокого навчання для відтворення інтонаційних, тембрових та ритмічних особливостей людської мови. Платформа підтримує роботу з великою кількістю голосів і мов, що дозволяє використовувати її для створення різноманітного аудіо, наприклад, аудіокниг, озвучення текстових матеріалів та навчальних ресурсів.

На відміну від класичних TTS-систем, функціонування ElevenLabs ґрунується на застосуванні алгоритмів машинного навчання та глибоких нейронних мереж, які навчаються на великому обсязі аудіоданих людської мови. Навчальні вибірки включають записи з різними акцентами, стилями мовлення та мовними особливостями, що дає змогу моделі генерувати більш природне та виразне звучання. Під час обробки тексту сервіс використовує методи обробки природної мови (Natural Language Processing, NLP) для аналізу структури речень, інтонаційних пауз та контексту. На основі отриманої інформації формується

аудіовихід, який максимально наближений до живої людської мови. Завдяки такому підходу ElevenLabs здатний відтворювати не лише окремі звуки, а й складні мовні конструкції з відповідною інтонацією та емоційним забарвленням.

ElevenLabs надає доступ до функцій синтезу мовлення через REST API, що робить сервіс незалежним від мови програмування клієнтського застосунку. Взаємодія із сервісом здійснюється за допомогою стандартних HTTP-запитів, що забезпечує повну сумісність з мовою програмування Java.

У межах даного проєкту інтеграція з ElevenLabs реалізована з використанням стандартних класів Java для роботи з мережею та потоками даних. Такий спосіб не потребує сторонніх бібліотек та зберігає контроль над процесом надсилання запитів і обробки відповіді. Тому, використання Java для роботи з ElevenLabs є правильним та доцільним з точки зору архітектури та підтримки коду.

Для доступу до функцій ElevenLabs використовується API-ключ, який користувач може створити після реєстрації на платформі. API-ключ передається у HTTP-заголовку запиту та використовується для автентифікації клієнта.

Ще однією з ключових переваг ElevenLabs є детальне налаштування параметрів синтезу мовлення. Перед генерацією аудіоряду, користувач може керувати швидкістю мовлення, стабільністю голосу, ступенем подібності до базового голосу та інтонаційними характеристиками.

Проте сервіс ElevenLabs має низку обмежень, які необхідно враховувати під час розробки застосунку:

- обмеження на кількість символів, що можуть бути передані для генерації (10 000 символів на місяць);
- вимоги до коректності тексту, що передається на синтез;
- обмеження на формат вихідного аудіо, який задається параметрами API.

Сервіс синтезу мовлення ElevenLabs надає можливість отримувати згенеровне аудіо у форматах MP3 та WAV. У межах даного проєкту було обрано формат MP3, так як він забезпечує оптимальне співвідношення між якістю звучання та розміром файлу, а також підтримується більшістю програмних та апаратних засобів без необхідності додаткової обробки.

Тому, вибір сервісу ElevenLabs для реалізації функції синтезу мовлення зумовлений такими перевагами:

- підтримка багатьох мов;
- висока природність звучання голосу;
- гнучне налаштування параметрів голосу;
- зручний API для інтеграції з Java-застосунками;
- можливість масштабування функціоналу без змін у клієнтській частині.

Відтворення аудіоконтенту реалізовано за допомогою системного класу MediaPlayer. Даний інструмент забезпечує повний контроль над медіапотоком, дозволяючи керувати стартом, паузою, перемотуванням, а також змінювати швидкість відтворення.

Отже, сформовані вимоги визначають основні функціональні можливості та характеристики мобільного застосунку. Вони слугують основою для подальшого проєктування архітектури та реалізації додатку. А обрані технології, що включають мову Java, Android SDK та сервіси синтезу мовлення, дозволяють створити сучасний, функціональний мобільний застосунок.

2.2. Проєктування архітектури та інтерфейсу користувача мобільного застосунку

Проєктування архітектури застосунку є одним із ключових етапів розробки програмного забезпечення, що визначає загальну структуру системи, взаємодію її компонентів та принципи організації даних. Від обраної архітектури залежить масштабованість, зручність розробки, продуктивність та можливість подальшого розширення функціоналу застосунку.

У рамках розробки мобільного застосунку для читання електронних книг із функцією озвучування тексту було обрано підхід, що базується на розподілі системи на окремі логічні компоненти. Даний підхід дає змогу забезпечити модульність системи та спростити процес тестування та модифікації проєкту.

Архітектура застосунку включає три основні рівні:

Рівень представлення – рівень, який відповідає за взаємодію з користувачем, відображення тексту книги, елементів керування, а також забезпечує доступ до функцій озвучування.

Рівень бізнес-логіки – реалізує основні функції застосунку, зокрема обробку текстових даних, управління процесом читання, керування відтворенням мовлення та обробку користувацьких дій.

Рівень даних – забезпечує збереження та завантаження інформації, включаючи бібліотеку книг, аудіокниг.

Для реалізації взаємодії між зазначеними рівнями доцільно використовувати архітектурний шаблон *Model-View-ViewModel*, який широко застосовується при розробці мобільних застосунків. У даному підході Model відповідає за роботу з даними та бізнес-логіку, View реалізує інтерфейс користувача, ViewModel виступає посередником між інтерфейсом і логікою, забезпечуючи обробку даних та оновлення UI. Взаємодія даного шаблону показана на рисунку 2.1.

Модель у застосунку представлена класами даних: Item, Voice, VoiceSettings. Це класи, які описують основні сутності застосунку. Клас Item зберігає інформацію про завантажені книги та створені аудіокниги. VoiceSettings зберігає параметри синтезу мовлення, зокрема обраний голос та налаштування озвучування. Клас Voice призначений для зберігання та обробки інформації про доступні голоси синтезу мовлення.

Рівень представлення (View) включає класи активностей: AddActivity, AudiobooksActivity, PlayerActivity, SettingsActivity, LibraryActivity, ReaderActivity та їхні XML-макети. Також він містить універсальний адаптер Adapter для книг та аудіокниг та PageAdapter для сторінок. Основна роль View – відображення даних користувачеві та перехоплення його дій.

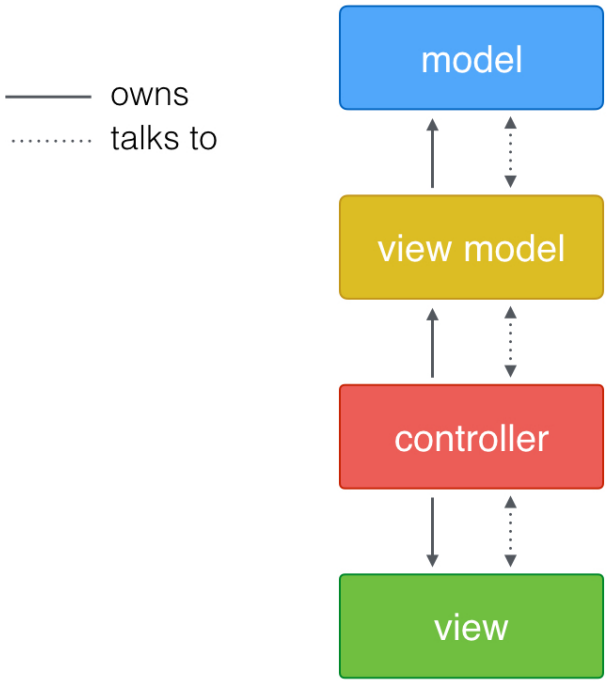

Рисунок 2.1. Схема архітектурного шаблону Model-View-ViewModel

Рівень ViewModel містить SettingsViewModel та ReaderViewModel, які виступають посередниками, що керують станом екрана. Клас ReaderViewModel отримує команду на завантаження книги, а SettingsViewModel, в свою чергу, реалізує цикл процесу генерації.

Також присутній пакет *utils*, який містить класи FileUtils та TTS. FileUtils відповідає за читання текстових документів, а TTS здійснює інтеграцію із зовнішнім сервісом синтезу мовлення та забезпечує перетворення тексту в аудіоформат.

Ще одним важливим етапом розробки є проєктування користувацького інтерфейсу та користувацького досвіду. Основною метою на цьому етапі було створення максимально спрощеного та функціонального дизайну.

Для проєктування структури екранів було створено структуру навігації між основними екранами системи (рис. 2.2).

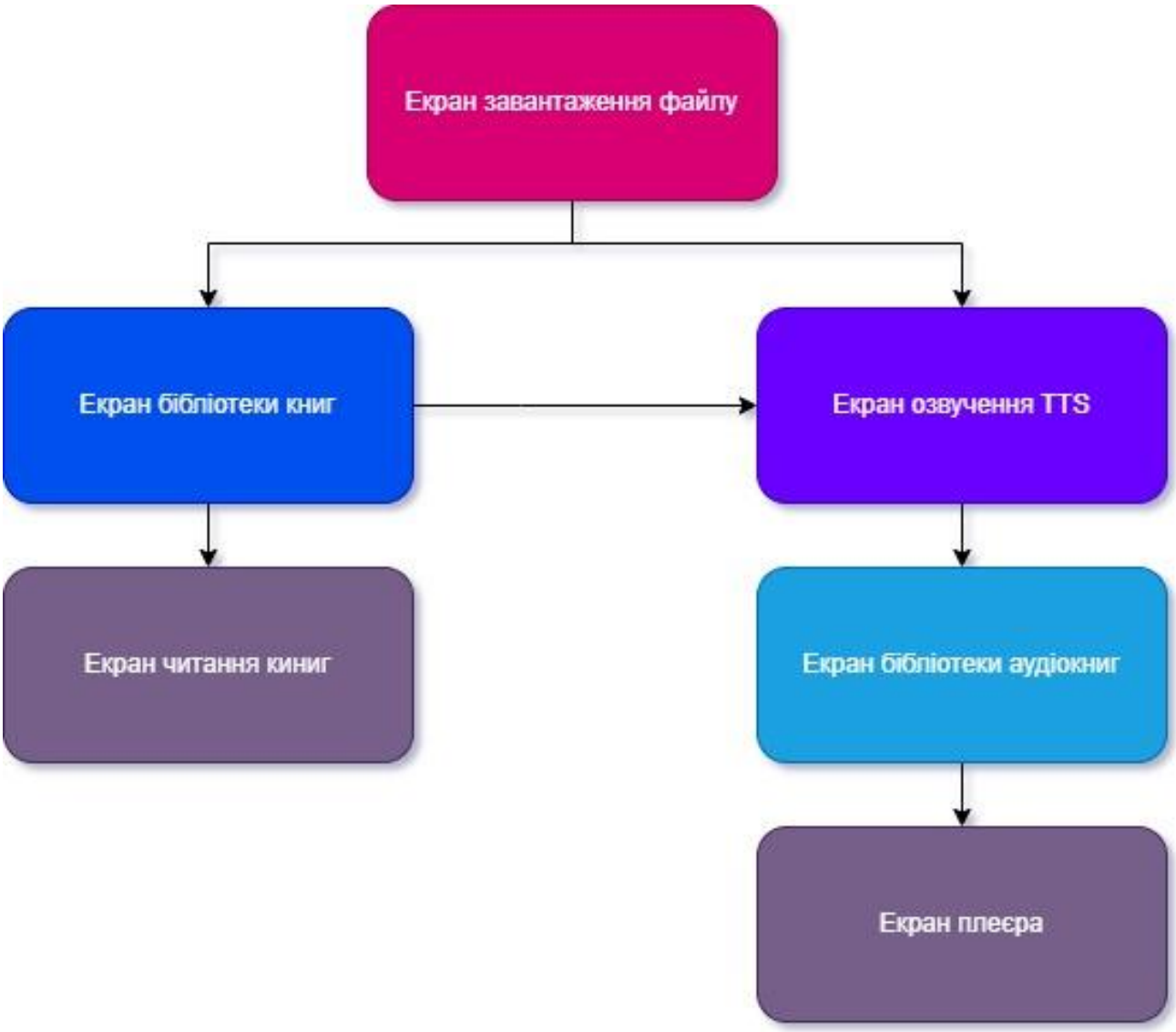

Рисунок 2.2. Діаграма навігації застосунку

Основними модулями є: екран додавання файлів, бібліотека книг, модуль читання, модуль озвучення тексту, бібіліотека аудіокниг, модуль плеєру. Користувач може завантажити текстовий файл, додати його до бібліотеки, відкрити для читання або озвучення. Для прослуховування використовується окремий екран аудіоплеєра.

Використана нижня панель керування, що забезпечує доступ до бібліотеки книг та аудіокниг в один клік. Ключовими принципами закладеними в проєкт інтерфейсу є:

- групування – схожі за функціями елементи згруповані в одному модальному вікні;

- консистентність – використання єдиної колірної схеми та шрифтів для всіх екранів застосунку;
- доступність – елементи мають збільшений розмір для зручного натискання.

Тому, запропонована архітектура програмного застосунку є логічною та структурованою, відповідає вимогам та забезпечує ефективну реалізацію основних задач. MVVM підхід сприяє підвищенню надійності додатку, зручності його використання та можливості подальшого розвитку.

РОЗДІЛ 3

РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ РОБОТИ МОБІЛЬНОГО ЗАСТОСУНКУ

3.1. Огляд основного функціоналу застосунку

Розробка інтерфейсу користувача є важливим етапом створення додатку, оскільки саме інтерфейс забезпечує взаємодію користувача з функціоналом системи. Основною метою розробки інтерфейсу даного застосунку є створення зрозумілого, зручного середовища для роботи з текстом, його читання та генерації аудіоконтенту на його основі.

Проєктування інтерфейсу застосунку базувалося на принципах, що забезпечують інтуїтивно зрозумілий користувацький досвід. Основний акцент було зроблено на мінімізації кількості кроків для доступу до ключових функцій: читання та генерації аудіоконтенту.

Головний екран (бібліотека) розроблений на основі компонента RecyclerView, який забезпечує ефективне керування списками даних за рахунок ViewHolder. Навігацію між модулями роботи застосунку (бібліотека електронних книг, завантаження, аудіобібліотека) реалізовано за допомогою LinearLayout, який розташований у нижній частині екрана. Також присутній SearchView для пошуку книг (рис. 3.1).

Рисунок 3.1. Головний екран бібліотеки користувача

Екран збереження створених аудіокниг побудований за аналогічною схемою (рис. 3.2), що дає змогу користувачеві швидко орієнтуватися у структурі застосунку.

Рисунок 3.2. Екран бібліотеки аудіокниг користувача

Модуль імпорту контенту (рис. 3.3) інтегрований із системним провідником Android, що дає користувачеві можливість обирати файли форматів FB2, EPUB, TXT, DOCX з пам’яті пристрою. Після вибору файлу (рис. 3.4), користувач може обрати додавання книги до бібліотеки для читання чи створення з неї аудіокниги за допомогою синтезу мовлення.

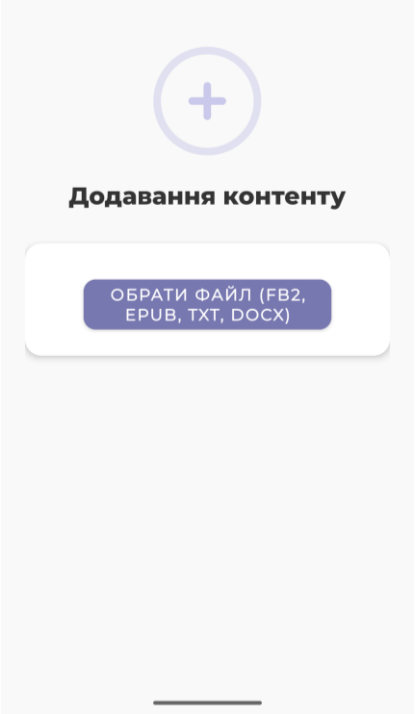

Рисунок 3.3. Екран для завантаження електронної книги

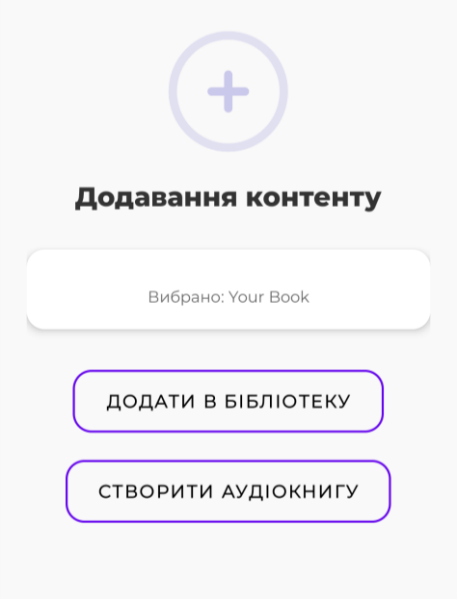

Рисунок 3.4. Екран вибору сценарію обробки контенту

Після успішного додавання електронної книги в бібліотеку, користувач отримує відповідне сповіщення та завантажена книга з’являється у RecyclerView бібліотеки (рис. 3.5). Взаємодія з уже доданими об’єктами реалізована через обробник подій onLongClick, який викликає showOptionsDialog (рис. 3.6). Це дозволяє реалізувати видалення електронної книги або створення аудіокниги з неї. У бібліотеці аудіокниг доступна лише функція видалення.

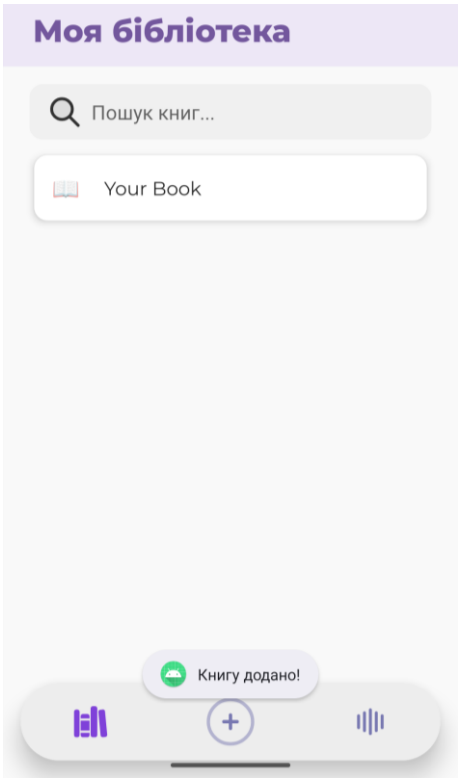

Рисунок 3.5. Процес додавання книги

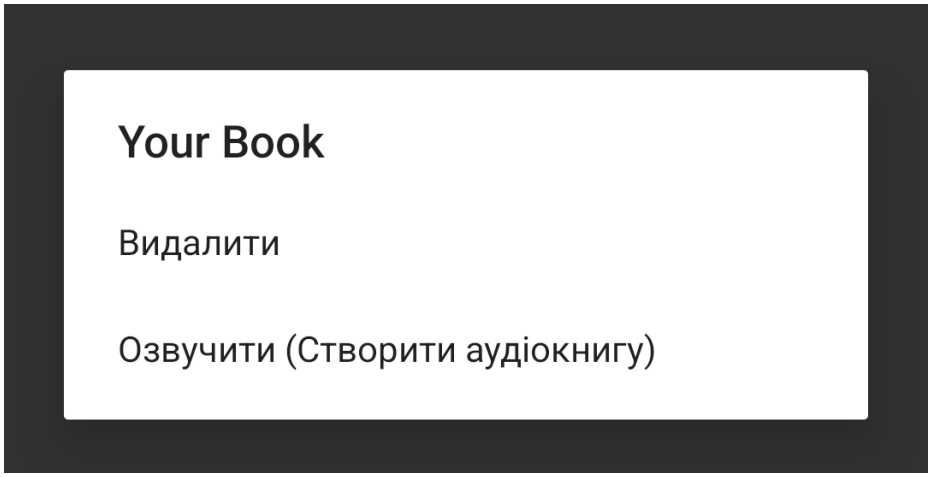

Рисунок 3.6. Контекстне вікно керування елементом бібліотеки

При натисканні на книгу в бібліотеці, відкривається екран для читання (рис. 3.7). Використано панелі керування, які не перекривають основний текст. Інтегровано SeekBar для відстеження прогресу поточної сторінки, присутні кнопки для озвучення сторінки, пошуку.

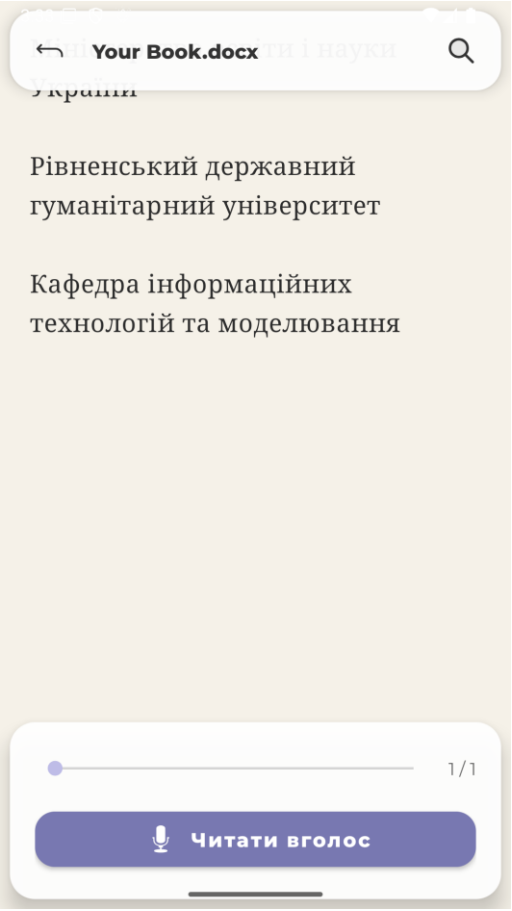

Рисунок 3.7. Екран читання

Особливу увагу приділено екрану створення аудіокниги (рис 3.8). Блок налаштувань дає можливість користувачеві керувати параметрами синтезу мовлення. До них належать: Spinner – для вибору доступних голосів (рис. 3.9); SeekBar – для налаштування параметрів швидкості, стабільності та подібності голосу. Також, окрім основної кнопки генерації аудіокниги, в даному блоці є можливість прослухати короткий семпл аудіо з поточними налаштуваннями голосу.

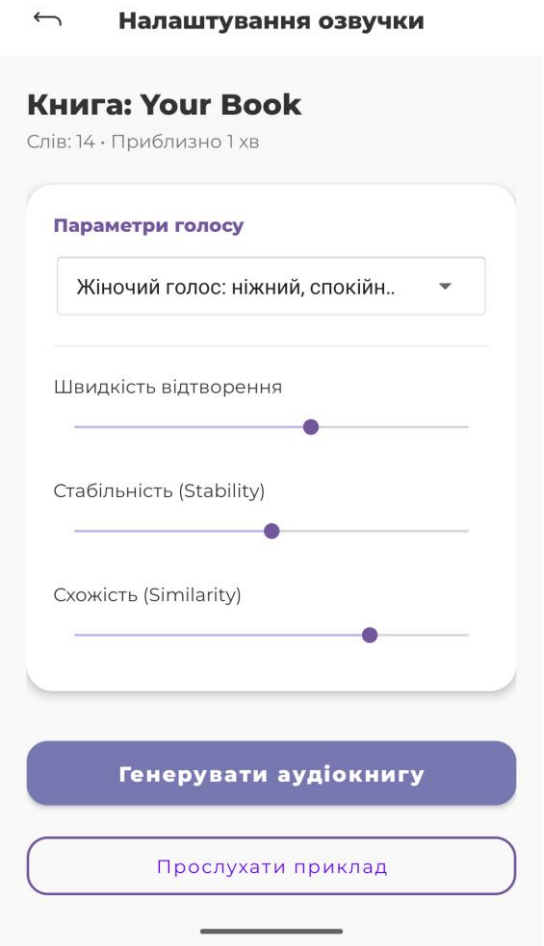

Рисунок 3.8. Екран налаштувань для створення аудіокниги

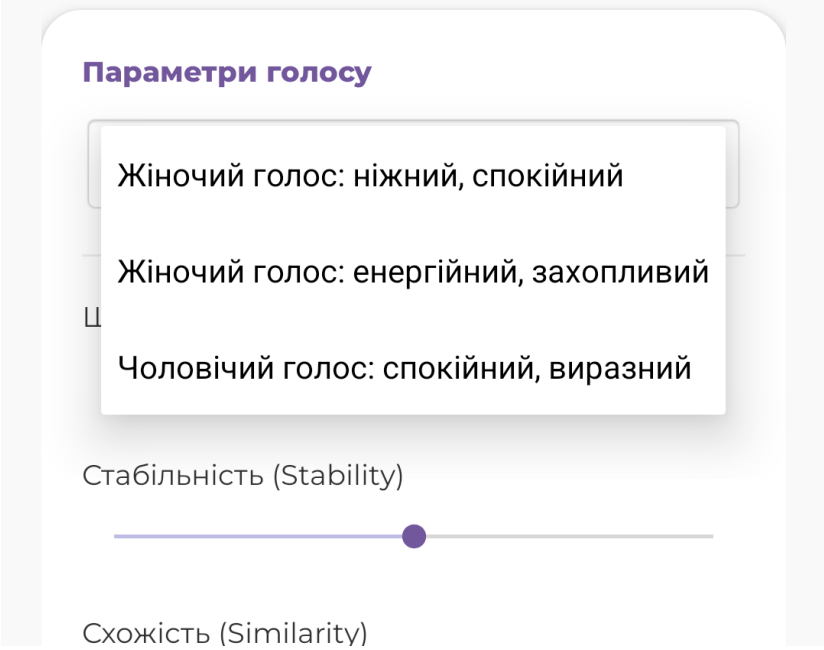

Рисунок 3.9. Випадаючий список вибору голосу

При натисканні на створену аудіокнигу в бібліотеці, користувач переходить на екран плеєру. Це повнофункціональний плеєр, з можливістю зміни швидкості відтворення (1.0x, 1.5x, 2.0x), перемотування на 15 секунд та збереження останньої відкритої позиції (рис. 3.10).

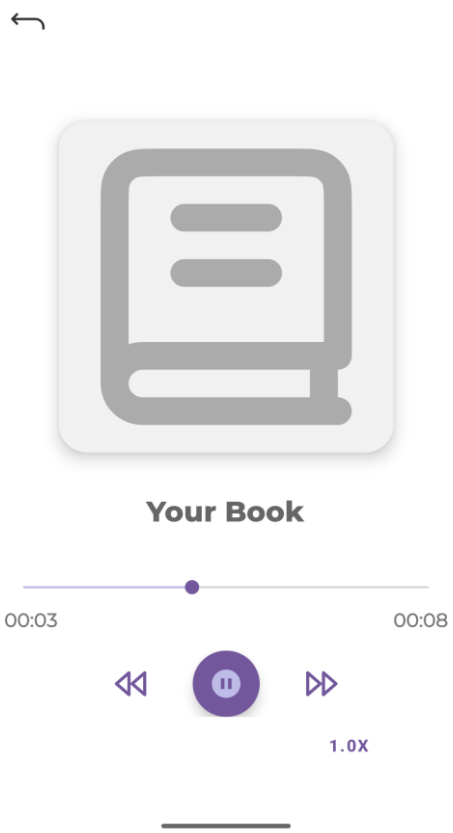

Рисунок 3.10. Екран плеєра застосунку

Розроблений інтерфейс користувача забезпечує зручну та інтуїтивну взаємодію з функціоналом застосунку, дозволяючи виконувати всі основні операції без зайвих дій. Структурованість інтерфейсу та логічний поділ на функціональні блоки роблять додаток придатним для використання навіть користувачами без технічної підготовки.

3.2. Опис програмної реалізації ключових компонентів

Процес додавання нового контенту реалізовано у класі AddActivity. Після вибору файлу застосунок отримує стійкий дозвіл на читання URI, що дає змогу

звертатися до книги при наступних запусках програми без повторного вибору користувачем. Ключовим технічним компонентом є клас утиліта FileUtils, який відповідає за читання тексту форматів EPUB, FB2, TXT, DOCX. Для формату FB2 здійснюється потокове читання XML-структури з вилученням вмісту тегів <p>. Для EPUB, оскільки цей формат є ZIP-архівом, застосунок використовує ZipInputStream. Після успішного аналізу файлу користувачеві надається вибір: додати книгу до бібліотеки або перейти до створення аудіокниги. Дані про шлях до файлу та метадані книги зберігаються за допомогою SharedPreferences.

Керування бібліотекою завантажених книг реалізовано у класі LibraryActivity. Центральним елементом інтерфейсу RecyclerView, який працює у зв'язці з універсальним адаптером Adapter. Для пошуку книг реалізовано метод filter(string text), який працює з копією повного списку (fullList). Через обробник onLongClick реалізовано виклик діалогового вікна, яке дає змогу видалити або озвучити об'єкт. Оновлення інтерфейсу виконується у методі onResume().

Екран читання ReaderActivity забезпечує візуалізацію тексту. Для ефекту гортання сторінок використано ViewPager2. Текст книги розбивається на окремі фрагменти, які передаються в PageAdapter. Також клас взаємодіє з ReaderViewModel (див. Додаток А). Використання LiveData дає змогу автоматично оновлювати інтерфейс, наприклад, при зміні поточної сторінки. SeekBar реалізує швидку навігацію по всій книзі, а діалогове вікно пошуку реалізує перехід на конкретну сторінку. Крім створення повноцінних аудіокниг, у модулі читання є функція миттєвого озвучення поточної сторінки за допомогою TextToSpeech.

Процес підготовки до генерації аудіокниги реалізовано у класі SettingsActivity. Користувачеві надається можливість налаштування характеристик голосу за допомогою:

- Spinner – для вибору конкретного голосу, які зберігаються у класі Voice;
- SeekBar – для налаштування темпу, стабільності та подібності голосу.

Присутня функція генерації короткого аудіо для перевірки обраних налаштувань.

Логіка синтезу винесена у SettingsViewModel (див. Додаток Б). Процес

генерації запускається в окремому потоці виконання, задля уникнення блокування додатку. Використання LiveData дає Activity змогу відстежувати стан виконання операції. Після завершення методу TTS.generateAudio, вихідний файл зберігається у внутрішньому сховищі застосунку, а дані записуються в окрему категорію SharedPreferences. Для відображення створеного аудіоконтенту використовується AudiobooksActivity, який реалізований аналогічним способом як LibraryActivity.

Для реалізації синтезу мовлення використовується зовнішній API сервісу ElevenLabs, який надає можливість генерувати високоякісне аудіо з урахуванням параметрів голосу.

Процес перетворення тексту в аудіо складається з таких етапів:

1. Отримання тексту імпортованого з файлу.
2. Отримання та перевірка налаштувань генерації.
3. Формування HTTP-запиту до сервісу синтезу мовлення.
4. Надсилання запиту та отримання аудіопотоку в форматі MP3.
5. Збереження згенеровного аудіофайлу на локальному носії.
6. Повідомлення користувача про результат виконання операції.

Параметри синтезу мовлення передаються у сервіс TTS відповідно до значень, заданих користувачем у налаштуваннях голосу.

Безпосередня генерація аудіофайлу виконується у методі generateAudio класу TTS (див. Додаток В). Даний метод формує HTTPS-з’єднання з API сервісу ElevenLabs та передає текст і параметри синтезу у форматі JSON. Отриманий у відповідь аудіопотік записується у аудіоформаті MP3, який надалі може бути прослуханий користувачем у плеєрі.

Для відтворення синтезованих аудіокниг було розроблено модуль PlayerActivity. Технічна реалізація базується на використанні системного класу MediaPlayer. Реалізовано метод changeSpeed(), який дає можливість змінювати параметр PlaybackParams. За допомогою методів saveProgress() та restoreProgress() застосунок фіксує поточну позицію відтворення, що відновлює прослуховування навіть після закриття застосунку.

3.3. Тестування мобільного застосунку

Після реалізації програмного застосунку було проведено функціональне тестування основних модулів системи. Метою тестування є перевірка працездатності розробленого додатка, виявлення дефектів та підтвердження відповідності вимогам. Перевірка здійснювалася шляхом запуску застосунку на емуляторі мобільного пристрою Pixel з операційною системою Android версії 16.0.

За результатами тестування було виявлено та усунуто помилки пов’язані із передачею URI файлів, відображенням елементів інтерфейсу та відтворенням тексту. Після внесення виправлень програмний застосунок стабільно виконує основні функції.

Для перевірки функціоналу було розроблено ряд тестів, результати яких наведено в таблиці 3.1.

Таблиця 3.1

Результати функціонального тестування

№	Функція, що тестується	Вхідні дані	Очікуваний результат	Статус
1	Парсинг файлу FB2	Файл .fb2	Вилучення тексту без XML-тегів	Пройдено
2	Розділення тексту	Довгий рядок тексту	Коректне розбиття	Пройдено
3	Обмеження генерації	Текст довжиною 6000 символів	Блокування операції	Пройдено
4	Збереження прогресу книги	Номер сторінки «12»	При перезапуску відкриваться 12 ст.	Пройдено
5	Робота з ZIP-архівами	Файл .zip з FB2 всередині	Успішне розархівування та читання внутрішнього файлу	Пройдено
6	Збереження прогресу аудіокниги	Зупинка на 01:05 хв.	При перезапуску відкривається 01:05 хв.	Пройдено
7	Завантаження EPUB файлу	Файл EPUB	Успішне читання файлу	Пройдено
8	Спроба озвучити пустий файл	Файл DOCX	Відповідне повідомлення	Пройдено

ВИСНОВКИ

У межах кваліфікаційної роботи було проведено комплексне дослідження та розробку мобільного застосунку для читання електронних книг із інтегрованою функцією синтезу мовлення.

У теоретичній частині роботи розглянуто поняття тексту в мовлення, основні підходи до його реалізації та роль сучасних нейромережевих моделей у забезпеченні природного звучання синтезованої мови. Було проаналізовано існуючі програмні рішення для читання книг з можливістю синтезу мовлення, що підтвердило необхідність створення альтернативного додатку. Аналіз предметної області підтвердив актуальність інтеграції технологій Text-to-Speech у мобільні рідери. Виявлено, що поєднання текстового та аудіоформатів є провідним трендом у споживанні контенту, оскільки це забепезпечує доступність та гнучкість навчання в сучасних умовах.

Практична частина реалізована як діючий мобільний застосунок для читання електронних книг з можливістю озвучування тексту, який демонструє застосування теоретичних принципів синтезу мовлення у реальному програмному продукті. Реалізація застосунку показала ефективність використання зовнішнього TTS-сервісу, ElevenLabs, для отримання якісного аудіо без необхідності реалізації складних алгоритмів на стороні клієнта.

Розроблена інформаційна система має низку переваг:

- універсальність споживання контенту, що дозволяє користувачеві вибір між візуальним читанням та прослуховуванням тексту;
- гнучкість налаштувань синтезу, яка дає змогу індивідуально адаптувати параметри голосу;
- оптимізація ресурів, що дозволяє ефективно використовувати ресурси пристрою без втрати швидкості інтерфейсу;
- інтуїтивність взаємодії, яку забезпечує зручна навігація.

Мету роботи було досягнуто: розроблено мобільний застосунок для читання та створення аудіокниг. Отримані результати підтверджують, що використання

сучасних нейромережевих технологій синтезу мовлення є перспективним напрямом для створення доступних та зручних інструментів озвучування текстового контенту. Розроблений застосунок може бути використаний для популяризації літератури та для підвищення ефективності навчання. Розробка також відкриває перспективи для використання як основи для подальших досліджень і розширення функціональних можливостей у сфері цифрової доступності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Електронна книга: що це таке, якими бувають та чому стають все більше актуальними. URL: https://taperowebschool.com.ua/elektronna-knyga-shho-cze-take-yakymy-buvayut-ta-chomu-stayut-vse-bilshe-aktualnymy/ (дата звернення: 13.02.2026).

2. Виникнення електронних видань. StudFiles. URL: https://studfile.net/preview/9102446/page:6/ (дата звернення: 13.02.2026).

3. Пашкова В. С., Ярошенко Т. О. Електронні книжки та електронні читанки (рідери) в бібліотеці: з чого почати? Київ : Укр. бібл. асоц., НПБ України, НБ НаУКМА, 2013. 8–9 с.

4. Христина Бойчук. Звідки взялись аудіокниги? Kl.informator.ua. URL: https://kl.informator.ua/uk/zvidky-vzyalys-audioknygy (дата звернення: 26.02.2026).

5. Charlotte Hu, Amanda Downie. What is text to speech? IBM. URL: https://www.ibm.com/think/topics/text-to-speech (дата звернення: 14.03.2026).

6. Що таке синтез мовлення? – Пояснення TTS. Shaip. URL: https://uk.shaip.com/blog/what-is-tts/ (дата звернення: 14.03.2026).

7. 7 кращих мобільних застосунків для читання книг. Kitapp. URL: https://kitapp.pro/uk/7-krashhih-mobilnih-zastosunkiv-dlya-chitannya-knig/ (дата звернення: 17.03.2026).

8. Best Text to Speech Apps for Android in 2025. Lemonfox.ai. URL: https://www.lemonfox.ai/blog/text-to-speech-apps-for-android?utm_source=chatgpt.com (дата звернення: 17.03.2026).

9. Berkay Kınacı. 7 найкращих БЕЗКОШТОВНИХ додатків для перетворення тексту в мовлення у 2025. Speaktor. URL: https://speaktor.com/uk/найкращі-безкоштовні-програми-для-перетворення-тексту-в-мовлення/ (дата звернення: 17.03.2026).

10. Bianca P. Найпопулярніші формати електронних книг для зручного читання. Pdf2go. URL: https://www.pdf2go.com/uk/blog/top-ebook-formats (дата звернення: 25.03.2026).

11. Верстка для видання книги у форматі epub, mobi, fb2. Кавун. URL: https://isbn.com.ua/verstka-u-formati-epub-mobi-fb2/ (дата звернення: 25.03.2026).

12. MP3. Вікіпедія. URL: https://uk.wikipedia.org/wiki/MP3 (дата звернення: 25.03.2026).

13. Advanced Audio Coding. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Advanced_Audio_Coding (дата звернення: 25.03.2026).

14. Копитко М.Ф., Іванків К.С. Основи програмування мовою Java. Львів : Видавничий центр ЛНУ ім. Івана Франка, 2002. 83 с.

15. MVVM. Brander. URL: https://brander.ua/technologies/mvvm (дата звернення: 02.04.2026).

16. ElevenLabs. URL: https://elevenlabs.io/app/home (дата звернення: 20.04.2026).

17. ElevenLabs Documentation. URL: https://elevenlabs.io/docs/overview/intro (дата звернення: 20.04.2026).

18. Гай В. І., Шевцова Н. В. Інтеграція технологій синтезу мовлення у мобільні системи для читання. *Наука, освіта, суспільство очима молодих* : матеріали XIX Всеукраїнської науково-практичної конференції здобувачів вищої освіти та молодих учених. м. Рівне, 14 травня 2026 р. Рівне, 2026.

ДОДАТКИ

ДОДАТОК А
class ReaderViewModel

```
public class ReaderViewModel extends AndroidViewModel {
    private final MutableLiveData<List<String>> pages = new MutableLiveData<>(new
ArrayList<>());
    private final MutableLiveData<Integer> currentPage = new MutableLiveData<>(0);
    private final MutableLiveData<Boolean> isLoading = new
MutableLiveData<>(false);
    private String currentUri;
    public ReaderViewModel(@NonNull Application application) {
        super(application);
    }
    public LiveData<List<String>> getPages() { return pages; }
    public LiveData<Integer> getCurrentPage() { return currentPage; }

    public void loadBook(String uriString) {
        if (uriString == null || uriString.equals(currentUri)) return;
        currentUri = uriString;
        isLoading.setValue(true);
        Executors.newSingleThreadExecutor().execute(() -> {
            String text = FileUtils.readTextFromUri(getApplication(),
Uri.parse(uriString));
            List<String> chunks = splitText(text);
            int lastPage = getApplication().getSharedPreferences("ReaderPrefs",
Context.MODE_PRIVATE)
                    .getInt("last_page_" + uriString, 0);
            new Handler(Looper.getMainLooper()).post(() -> {
                pages.setValue(chunks);
                currentPage.setValue(lastPage < chunks.size() ? lastPage : 0);
                isLoading.setValue(false);
            });
        });
    }
    public void saveProgress(int index) {
        if (currentUri == null) return;
        getApplication().getSharedPreferences("ReaderPrefs",
Context.MODE_PRIVATE).edit()
                .putInt("last_page_" + currentUri, index).apply();
        currentPage.setValue(index);
    }
    private List<String> splitText(String text) {
        List<String> list = new ArrayList<>();
        int length = text.length();
        int step = 1500;
        int start = 0;
        while (start < length) {
            int end = Math.min(start + step, length);
            if (end < length) {
                int lastSpace = text.lastIndexOf(' ', end);
                if (lastSpace > start) end = lastSpace;
            }
            list.add(text.substring(start, end).trim());
            start = end;
        }
        return list;
    }
}
```

ДОДАТОК Б
class SettingsViewModel

```
public class SettingsViewModel extends ViewModel {
    private final MutableLiveData<Boolean> isGenerating = new
MutableLiveData<>(false);
    private final MutableLiveData<Integer> progress = new MutableLiveData<>(0);
    private final MutableLiveData<Integer> maxProgress = new MutableLiveData<>(0);
    private final MutableLiveData<String> error = new MutableLiveData<>();
    private final MutableLiveData<Boolean> isFinished = new
MutableLiveData<>(false);

    public LiveData<Boolean> getIsGenerating() {
        return isGenerating;
    }

    public LiveData<Integer> getProgress() {
        return progress;
    }

    public LiveData<Integer> getMaxProgress() {
        return maxProgress;
    }

    public LiveData<String> getError() {
        return error;
    }

    public LiveData<Boolean> getIsFinished() {
        return isFinished;
    }

    public void startGeneration(String fullText, String bookName, VoiceSettings
settings, File baseDir) {
        if (fullText == null || fullText.isEmpty()) {
            error.setValue("Текст порожній!");
            return;
        }
        if (fullText.length() > 5000) {
            error.setValue("Текст занадто великий (макс. 5000 симв. для Free).
Будь ласка, оформите підписку.");
            return;
        }

        isGenerating.setValue(true);

        new Thread(() -> {
            try {
                File folder = new File(baseDir, "AudioBooks");
                if (!folder.exists()) folder.mkdirs();

                File outputFile = new File(folder, bookName.replaceAll("\\s+",
"_") + ".mp3");

                Log.d("DEBUG_GEN", "Починаю запис у: " +
outputFile.getAbsolutePath());

                TTS.generateAudio(fullText, outputFile, settings);

                if (outputFile.exists() && outputFile.length() > 0) {
                    Log.d("DEBUG_GEN", "ФАЙЛ СТВОРЕНО! Розмір: " +
outputFile.length());
```

```
                    isFinished.postValue(true);
                } else {
                    Log.e("DEBUG_GEN", "ФАЙЛ НЕ З'ЯВИВСЯ АБО ПУСТИЙ!");
                    error.postValue("Файл не записався на диск.");
                }
            } catch (Exception e) {
                Log.e("DEBUG_GEN", "КРИТИЧНА ПОМИЛКА: " + e.getMessage());
                error.postValue(e.getMessage());
            } finally {
                isGenerating.postValue(false);
            }
        }).start();
    }
}
```

ДОДАТОК В
class TTS

```
public class TTS {
    private static final String API_KEY = " API_KEY_PLACEHOLDER";

    public static void generateAudio(String text, File outputFile, VoiceSettings
settings) throws Exception {
        URL url = new URL("https://api.elevenlabs.io/v1/text-to-speech/" +
settings.getVoiceId());

        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setRequestMethod("POST");
        conn.setDoOutput(true);

        conn.setRequestProperty("Content-Type", "application/json");
        conn.setRequestProperty("xi-api-key", API_KEY);
        conn.setRequestProperty("Accept", "audio/mpeg");

        String cleanedText = sanitizeText(text).replace("\"", "'");

        String jsonBody = "{"
                + "\"text\":\"" + cleanedText + "\","
                + "\"voice_settings\": {"
                + "\"speed\": " + settings.getSpeed() + ","
                + "\"stability\": " + settings.getStability() + ","
                + "\"similarity_boost\": " + settings.getSimilarity()
                + "}"
                + "}";

        try (OutputStream os = conn.getOutputStream()) {
            os.write(jsonBody.getBytes(java.nio.charset.StandardCharsets.UTF_8));
        }

        int code = conn.getResponseCode();
        if (code != 200) {
            throw new Exception("Помилка сервісу: " + code);
        }

        try (InputStream is = conn.getInputStream();
             FileOutputStream fos = new FileOutputStream(outputFile)) {
            byte[] buffer = new byte[8192];
            int read;
            while ((read = is.read(buffer)) != -1) {
                fos.write(buffer, 0, read);
            }
        }
    }
    public static String sanitizeText(String text) {
        if (text == null) return "";
        text = text.replace("\n", " ")
                .replace("\r", " ")
                .replace("\t", " ")
                .replaceAll("\\s+", " ")
                .trim();
        return text;
    }
}
```