МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ANDROID ЗАСТОСУНКУ АНОТУВАННЯ ТЕКСТІВ»**

Виконав: студент 4 курсу, групи 6.1210-1пі
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
М.І. Завгородній
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.ф.-м.н. Кудін О.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент доцент кафедри фундаментальної та прикладної математики, доцент, к.ф.-м.н., Панасенко Є.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Завгородньому Марку Ігоровичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка Android застосунку анотування текстів

керівник роботи Кудін Олексій Володимирович, к.ф.-м.н., доцент
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 21 » грудня 2023 року № 2180-с

2. Строк подання студентом роботи 03.06.2024 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Проєктування та реалізація текстового анотатора для Android.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація за темою докладу

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 25.12.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	09.01.2024	
2.	Збір вихідних даних.	30.01.2024	
3.	Обробка методичних та теоретичних	27.02.2024	
	джерел.		
4.	Розробка першого та другого розділу.	23.04.2024	
5.	Розробка третього розділу.	20.05.2024	
6.	Оформлення та нормоконтроль	27.05.2024	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	16.06.2024	

Студент _____ М.І. Завгородній
(підпис) (ініціали та прізвище)

Керівник роботи _____ О.В. Кудін
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка Android застосунку анотування текстів»: 38 с., 9 рис., 1 табл., 5 джерел, 3 додатка.

АНОТАЦІЯ ТЕКСТУ, АНОТУВАННЯ, ІНСТРУМЕНТИ ДЛЯ РОБОТИ З ТЕКСТОМ, КОРИСТУВАЧІ, МОБІЛЬНІ ПРИСТРОЇ, ОБРОБКА ІНФОРМАЦІЇ, ПРОДУКТИВНІСТЬ, РОЗРОБКА ANDROID-ЗАСТОСУНКУ, РОЗРОБКА ЗАСТОСУНКІВ, СПІЛЬНА РОБОТА, СУЧАСНІ ТЕХНОЛОГІЇ, ТЕСТУВАННЯ ДОДАТКІВ, ТЕКСТОВИЙ КОНТЕНТ, ІНТЕРФЕЙС КОРИСТУВАЧА, УПРАВЛІННЯ ЗАЛЕЖНОСТЯМИ.

Об’єкт дослідження – процес розробки Android-застосунку для анотування тексту, включаючи аналіз потреб користувачів, огляд існуючих рішень, проєктування інтерфейсу користувача та реалізацію функціональності для роботи з текстовим контентом на мобільних пристроях.

Мета роботи: розробка Android-застосунку для анотування тексту, який забезпечить користувачам зручні та ефективні інструменти для виділення, коментування та організації текстового контенту на мобільних пристроях, що покращить їх продуктивність та зручність роботи з інформацією.

Метод дослідження – вибір теми та формулювання завдання, пошук і аналіз літератури, критичний і теоретичний аналіз матеріалів, а також написання та оформлення тексту роботи.

SUMMARY

Bachelor’s qualifying paper «Development of an Android Application for Text Summarization»: 38 pages, 9 figures, 1 table, 5 references, 3 supplements.

TEXT ANNOTATION, ANNOTATION, TOOLS FOR WORKING WITH TEXT, USERS, MOBILE DEVICES, INFORMATION PROCESSING, PRODUCTIVITY, ANDROID APPLICATION DEVELOPMENT, APPLICATION DEVELOPMENT, COLLABORATION, MODERN TECHNOLOGIES, APPLICATION TESTING, TEXT CONTENT, USER INTERFACE, DEPENDENCY MANAGEMENT.

The object of study is the process of developing an Android application for text annotation, including analysis of user needs, review of existing solutions, design of the user interface, and implementation of functionality for working with text content on mobile devices.

The aim of the study is to develop an Android application for text annotation that will provide users with convenient and effective tools for highlighting, commenting and organizing text content on mobile devices, which will improve their productivity and convenience of working with information.

The method of research is choosing a topic and formulating a task, searching and analyzing literature, critical and theoretical analysis of materials, as well as writing and formatting the text of the paper.

ЗМІСТ

ВСТУП

Щоб орієнтуватися у неперервному величезному потоці інформації, своєчасно і швидко вирішувати проблеми, що виникають в сучасному цифровому світі важливо приймати рішення спираючись на факти та достовірну інформацію. Але інформаційний потік з будь-яких питань невичерпний, швидко мінливий, може містити велику кількість незначних слів в загальному обсязі документу, мати багато слів, які не несуть інформаційне навантаження. Обмеженість у часі та невичерпність інформації, а також необхідність виділення значущих для користувача даних або фактів серед щоденного лавиноподібного інформаційного потоку документів, книг, статей зумовило необхідність безперервно і швидко обробляти інформацію, яку вони містять, розкриваючи логічну структуру та отримуючи стислий зміст інформації – анотувати інформацію.

Для того, щоб швидко виконувати пошук значущої стислої інформації, необхідних значущих фактів і робити це постійно, користувачеві необхідно мати зручні та ефективні інструменти для роботи з текстовим контентом. Одним із таких інструментів є застосунок для анотування тексту, що дозволяє користувачам виділяти, коментувати та організовувати інформацію з метою подальшого аналізу, навчання та співпраці.

Розробка застосунків для мобільних пристроїв, зокрема під платформу Android, стала актуальною завдяки швидкому розвитку технологій та зростанню популярності смартфонів та планшетів. Цей розвиток відкриває безліч можливостей для створення нових інструментів, спрямованих на поліпшення продуктивності та зручності користувачів.

Ця кваліфікаційна робота спрямована на розробку Android застосунку анотування тексту з метою створення зручного, функціонального та інтуїтивно зрозумілого інструменту для роботи з текстовим контентом на мобільних пристроях. Робота буде включати вивчення сучасних підходів та

технологій у розробці Android-додатків, аналіз існуючих рішень у галузі анотування тексту, проєктування та реалізацію застосунку, оцінку його ефективності та можливості подальшого розвитку.

Метою даної кваліфікаційної роботи є розробка Android-застосунку анотування тексту, спрямованого на полегшення роботи з текстовим контентом на мобільних пристроях. Цей застосунок має на меті забезпечити користувачам зручні та ефективні інструменти для виділення, коментування та організації тексту, що допоможе покращити їх продуктивність та зручність роботи з інформацією.

Для досягнення поставленої мети визначені наступні об’єкти дослідження: аналіз потреб користувачів, що включає оцінку потреб та очікувань щодо функціональності застосунку для анотування тексту на мобільних пристроях, вивчення типових сценаріїв використання, особливостей спільної роботи та індивідуальних потреб; огляд існуючих рішень з оглядом Android-застосунків для анотування тексту, аналізом інтерфейсу користувача, функціональності та можливостей спільної роботи; проєктування інтуїтивно зрозумілого та зручного інтерфейсу користувача, що включає визначення структури екранів, розміщення елементів керування та дизайн; реалізація базової функціональності застосунку для анотування тексту з можливістю виділення тексту, додавання коментарів, створення категорій та організацію анотацій; та реалізація функціональності спільної роботи, яка передбачає підтримку обміну анотаціями між користувачами та синхронізацію даних.

Реалізація цих об’єктивів дослідження дозволить досягти поставленої мети та створити ефективний та зручний застосунок для анотування тексту на платформі Android.

Ця кваліфікаційна робота доводить широку актуальність обраної теми, вказавши на потребу в розробці ефективних інструментів для роботи з текстом на мобільних пристроях та на можливості, що відкриваються завдяки застосуванню сучасних технологій розробки програмного забезпечення.

Кваліфікаційна робота має таку структуру: вступ, три розділи, висновки, перелік посилань із 5 найменувань, три додатки.

Перший розділ роботи присвячений огляду теоретичної частини роботи, а саме визначення проблеми, мети та існуючі рішення, яку за допомогою цієї кваліфікаційної роботи ми будемо вирішувати.

У другому розділі розглядаються інструменти, які дозволять нам порівняти їх та обрати потрібні інструменти для вирішення поставленої задачі у цій кваліфікаційній роботі.

В межах третього розділу роботи будуть розроблено та розглянуті шляхи вирішення поставленої мети інструментами розглянутими у другому розділі цієї роботи.

1 ТЕОРЕТИЧНИЙ ОГЛЯД ТЕМИ РОБОТИ

1.1 Визначення проблеми

Швидкий розвиток сучасного інформаційного суспільства зумовив постійне суттєве зростання обсягу інформації, яку цей розвиток продукує. В свою чергу стрімкий розвиток інтернет-технологій та мобільних пристроїв зумовив легкий та швидкий доступ до цього безмежжя інформації. Через це у користувачів передбачувано виникла проблема ефективного управління та обробки цієї інформації. Найбільш розповсюдженою формою передачі вербальної інформації, що дозволяє її упорядковувати є текстовий контент. Від статей та книг до нотаток та електронних документів, текст становить важливий елемент навчання, роботи та спілкування. Проте, збільшення обсягу текстової інформації також викликає проблеми з її організацією, аналізом та зрозумінням.

Необхідність швидко та ефективно опрацьовувати велику кількість постійно оновлюваної текстової інформації виявилася однією з основних проблем, з якою стикаються користувачі. Вирішення цих проблем потребує застосування відповідних інструментів, які допомагають виокремлювати, організовувати та аналізувати важливу інформацію. Також, індивідуальні потреби користувачів можуть вимагати різних методів роботи з текстом, включаючи підкреслення, позначки, коментарі, зберігання посилань тощо.

Крім того слід зважати на те, що наступною проблемою що потребує вирішення є необхідність забезпечення можливості спільної роботи з текстовими документами. При розробці великих проєктів для роботи кількох учасників важко переоцінити необхідність мати зручний інструмент для спільного анотування та коментування тексту для забезпечення ефективної комунікації та співпраці.

Наступною проблемою з якою стикаються користувачі є проблема

зберігання та організації анотованого текстового контенту. З великою кількістю анотацій та коментарів стає складно вести систематизовану базу знань та забезпечувати доступ до неї у зручному форматі.

Одним зі способів вирішення цих проблем є створення застосунків для анотування тексту, які дозволяють користувачам ефективно взаємодіяти з текстовою інформацією, виокремлювати важливе, робити коментарі та організовувати контент. Такі застосунки можуть сприяти підвищенню продуктивності, поліпшенню якості роботи та зручності користування інформацією.

Загальна проблема полягає у розробці Android-застосунку анотування тексту, який би враховував потреби користувачів у роботі з текстовою інформацією на мобільних пристроях. Такий застосунок повинен бути зручним у використанні, мобільним, функціональним та забезпечувати можливість спільної роботи з текстом.

1.2 Огляд літератури та існуючих рішень

Заради вирішення проблеми, що стає все більш важливою для різних сфер життя суспільства, а саме, забезпечення виконання структурування та опрацювання текстового контенту сучасного інформаційного світу, існує багато інструментів та рішень для анотування тексту на різних платформах. Android – одна з найрозповсюдженіших платформ, яка може використовуватися для вирішення цієї проблеми. Огляд існуючих рішень допоможе зрозуміти тенденції у розробці застосунків для анотування тексту та визначити кращі практики та можливі напрямки для власного дослідження.

Одним з найпопулярніших інструментів для анотування тексту є “Evernote”. Цей застосунок дозволяє користувачам створювати нотатки, додавати до них текст, зображення, аудіо та інші медіафайли, а також робити позначки, підкреслення та коментарі. Evernote має зручний інтерфейс та

підтримує синхронізацію даних між різними пристроями, що робить його популярним серед користувачів, що шукають універсальний інструмент для роботи з нотатками та анотаціями. Розглянемо приклад інтрефейсу програми (рис. 1.1).

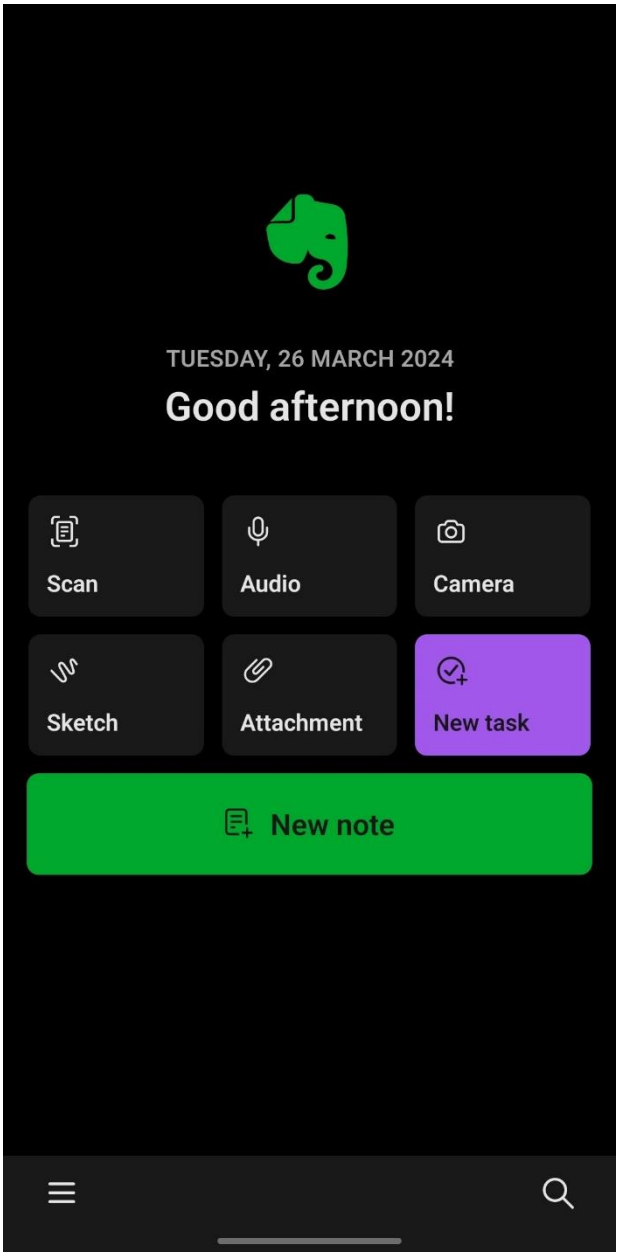

Рисунок 1.1 – Приклад інтерфейсу програми Evernote

Ще одним відомим інструментом є “Microsoft OneNote”. Це додаток, що дозволяє користувачам створювати нотатки, організовувати їх у структуровані блокноти, додавати до них різноманітні елементи (текст, зображення, відео,

файли тощо) та робити позначки та анотації. OneNote також підтримує синхронізацію між пристроями та інтеграцію з іншими сервісами Microsoft, що робить його зручним для користувачів, що працюють з екосистемою Microsoft. Розглянемо приклад інтрефейсу програми (див. рис. 1.2).

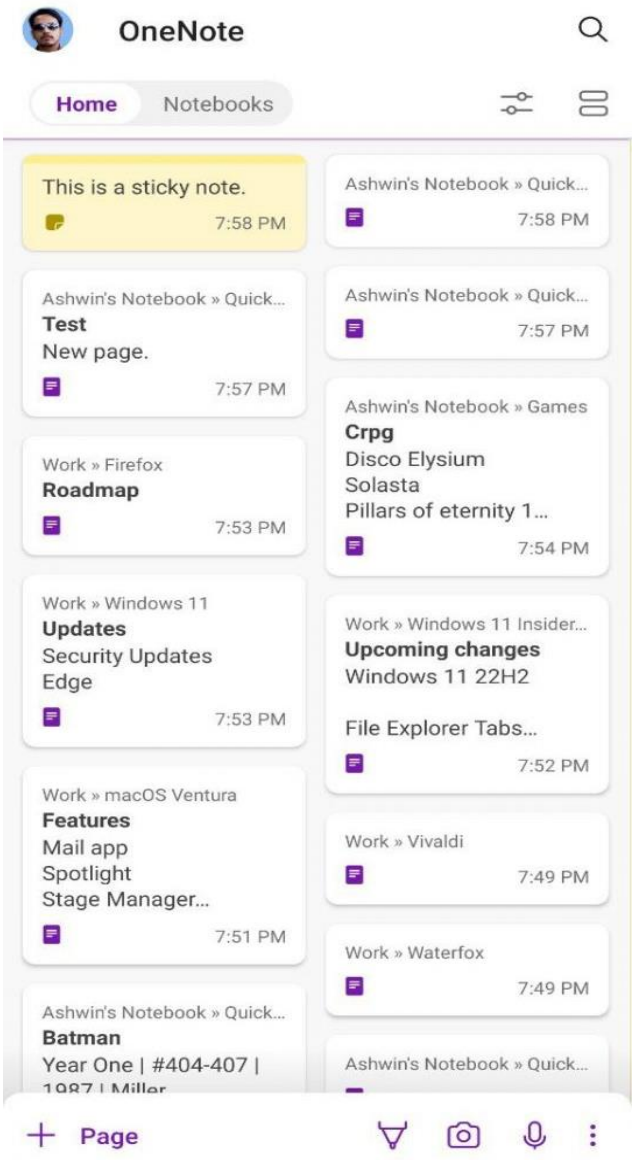

Рисунок 1.2 – Приклад інтерфейсу програми Microsoft OneNote

У сфері мобільних додатків для анотування тексту важливим інструментом є “Google Keep”. Цей додаток дозволяє користувачам створювати короткі нотатки, додавати до них тексти, зображення, список завдань та інші елементи, а також робити позначки та анотації. Google Keep

інтегрований з іншими сервісами Google, що дозволяє користувачам легко синхронізувати свої нотатки з різними пристроями та іншими програмами Google. Розглянемо приклад інтрефейсу програми (див. рис. 1.3).

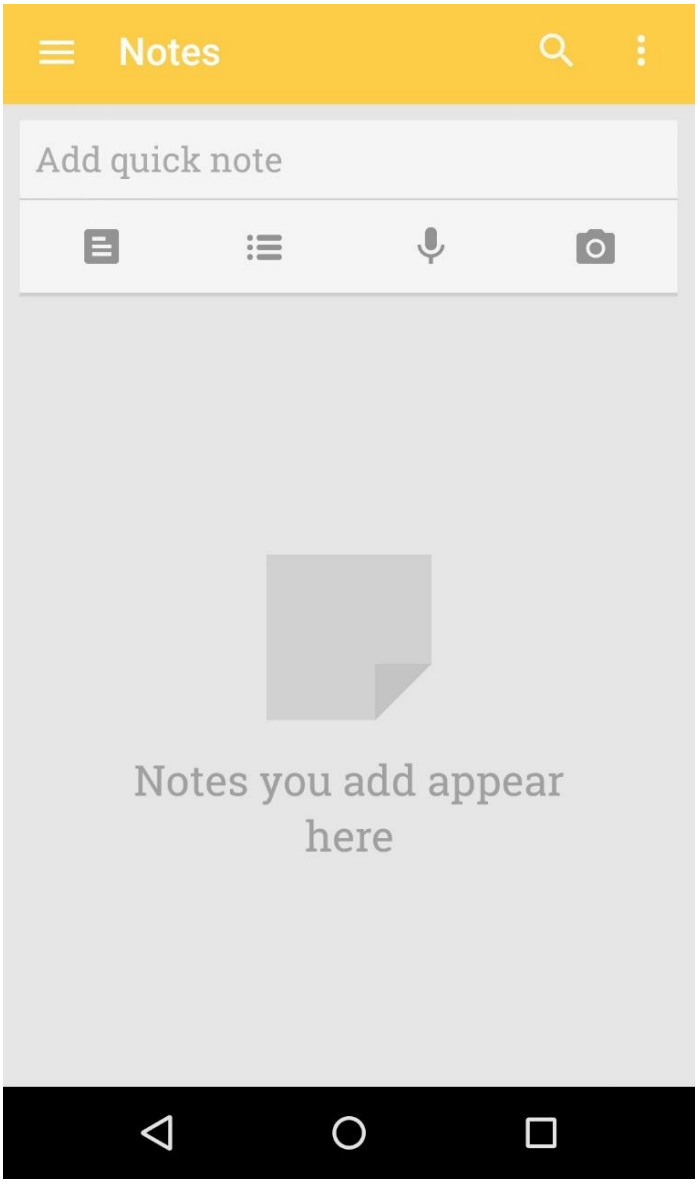

Рисунок 1.3 – Приклад інтерфейсу програми Google Keep

Однак, незважаючи на наявність цих популярних рішень, багато користувачів виявляють певні недоліки у них, такі як обмежена функціональність, складний інтерфейс або нестабільна робота. Тому існує попит на нові та покращені застосунки для анотування тексту, зокрема на Android-платформі.

Важливо враховувати, що на Android-платформі існує також безліч інших додатків для анотування тексту, які можуть мати свої унікальні особливості та переваги. Огляд існуючих рішень допомагає зрозуміти тенденції та потреби користувачів, а також визначити можливості для створення конкурентоспроможного та ефективного застосунку для анотування тексту на платформі Android.

2 ОГЛЯД ТЕХНОЛОГІЙ ТА ІНСТРУМЕНТІВ

У цьому розділі буде проведено докладний огляд технологій та інструментів, які можуть бути використані для розробки Android-застосунку для анотування тексту. Будуть розглянуті мови програмування, фреймворки, бібліотеки та інші інструменти, які можуть допомогти в створенні функціонального та ефективного застосунку.

Java є основною мовою програмування для створення Android-додатків, проте **Kotlin** стає все більш популярною альтернативою. Kotlin пропонує кілька переваг, таких як більш зручний та безпечний синтаксис, вбудована підтримка нульових значень та інші функції, що роблять його привабливим для розробників.

Android Software Development Kit (SDK) – це набір інструментів та бібліотек для розробки Android-додатків. SDK включає інструменти для створення, компіляції та тестування додатків, а також набір бібліотек для взаємодії з різними компонентами Android-платформи, такими як активності, фрагменти, сервіси та інші.

Android Studio – це офіційне інтегроване середовище розробки (IDE) для Android-платформи, яке базується на IntelliJ IDEA. Воно надає зручний інтерфейс для розробки, різноманітні корисні інструменти для створення та налагодження додатків, а також підтримку автоматичних підказок та виявлення помилок.

Цей огляд технологій та інструментів надає важливу основу для подальшого розгортання розробки Android-застосунку для анотування тексту. Вибір оптимальних технологій та інструментів допоможе забезпечити ефективну та продуктивну розробку додатку. Тому після огляду технологій ми розглянемо вступ до розробки Android-додатків, основні концепції та архітектурні принципи.

2.1 Вступ до розробки Android-додатків

Розробка додатків для Android вимагає розуміння різних аспектів програмування, дизайну та взаємодії з мобільними пристроями, що є складним та захоплюючим процесом. У цьому розділі ми детальніше розглянемо огляд розробки додатків для Android, включаючи основні концепції, інструменти та технології, необхідні для успішної розробки мобільних додатків.

Перш ніж розпочати розробку нашого додатку для анотації тексту, важливо зрозуміти основні концепції та архітектурні принципи, що лежать в основі платформи Android.

Основними концепціями є Activity, Fragment, Layouts, Intents, і Manifest file [1].

Activity представляє собою один екран користувацького інтерфейсу, є основною одиницею взаємодії з користувачем, і зазвичай містить інтерфейс користувача та логіку взаємодії з ним.

Fragment є частиною інтерфейсу або поведінки, яку можна вставити в Activity, що дозволяє створювати більш гнучкі та повторно використовувані компоненти користувацького інтерфейсу.

Layouts визначають структуру і розташування елементів інтерфейсу в Activity або Fragment, використовуючись для визначення розміщення елементів на екрані.

Intents застосовуються для взаємодії між компонентами програми або між різними додатками Android, і можуть бути використані для запуску нової Activity, передачі даних між Activity або запуску зовнішнього додатка.

Manifest file (AndroidManifest.xml) містить інформацію про додаток, включаючи перелік всіх його компонентів, права доступу, конфігурації та інше.

Для розробки програми для Android нам потрібно використовувати спеціалізовані інструменти, які допоможуть нам ефективно створювати, тестувати та налагоджувати додаток. Розглянемо основні інструменти.

Android Studio є офіційним інтегрованим середовищем розробки (IDE) для створення Android-додатків. Воно забезпечує зручний інтерфейс розробки, автоматичне завершення коду, візуальний редактор користувацького інтерфейсу та інші корисні інструменти.

Android SDK Manager дозволяє керувати версіями Android SDK, завантажувати необхідні компоненти SDK та оновлювати інструменти розробки. Для тестування та налагодження додатків ви можете використовувати емулятор Android або фізичний пристрій.

ADB дозволяє взаємодіяти з підключеними Android-пристроями для виконання команд, встановлення додатків та налагодження.

AVD Manager дає змогу створювати та керувати віртуальними пристроями Android для емуляції різних конфігурацій пристроїв та версій Android.

Android-розробники мають можливість вибирати між двома основними мовами програмування: Java та Kotlin. Java є традиційною мовою програмування для Android та має велику кількість бібліотек та інструментів, але Kotlin набирає популярності завдяки своєму сучасному синтаксису та ряду корисних функцій. Тому у ході цієї роботи буде розглядатися процес розробки Android-додатку для анотування тексту за допомогою мови програмування Kotlin [2].

Розуміння основних концепцій, використання правильних інструментів та вибір відповідної мови програмування допоможе нам розпочати розробку Android-додатків з впевненістю та ефективністю. Після цього розділу ми пропонуємо розглянути існуючі рішення для анотування тексту для розуміння того, як правильно нам потрібно писати власний додаток.

2.2 Аналіз інструментів для анотування тексту

Анотування тексту стало важливим засобом ефективного забезпечення швидкого обміну новою інформацією, саме воно забезпечує істотне

скорочення часу користувачів на обробку інформації. Анотування тексту є важливою функцією для багатьох користувачів, особливо для тих, хто працює з великими обсягами науково-технічної текстової інформації, таких як дослідники, студенти, журналісти та інші професіонали. У цьому розділі ми проаналізуємо існуючі інструменти для анотування тексту, що доступні для різних платформ, включаючи Android, та оцінимо їхні переваги та недоліки (див. таб. 2.1).

Таблиця 2.1 – Існуючі інструменти для анотування тексту

Функціонал	**Evernote**	**OneNote**	**Google Keep**	**Наш додаток**
Платформи	Android, iOS, Web, Windows, MacOS	Android, iOS, Web, Windows, MacOS	Android, iOS, Web	Android
Синхроніза-ція	Так	Так	Так	Можливо, через сторонні сервіси
Можливість роботи офлайн	Так	Так	Так	Так
Підтримка мульти-медіа	Текст, зображення, аудіо, відео, файли	Текст, зображення, аудіо, відео, файли	Текст, зображення, аудіо	Ні
Пошук по тексту	Так	Так	Так	Так
Мітки та категорії	Так (теги, нотатки у зошити)	Так (розділи, сторінки)	Так (теги, кольори)	Так
Спільний доступ	Так	Так	Так	Планується

Продовження таблиці 2.1

Функціонал	**Evernote**	**OneNote**	**Google Keep**	**Наш додаток**
Інтерфейс користувача	Інтуїтивний, Функціональний	Інтуїтивний, функціональний	Простий, мінімалістичний	Інтуїтивний, мінімалістичний
Робота з рукописним текстом	Так	Так	Ні	Можливо, через сторонні сервіси
Інтеграції з іншими сервісами	Так (Google Drive, Slack, Outlook та ін.)	Так (Office 365, Outlook, Teams та ін.)	Так (Google Workspace)	Можливо, через сторонні сервіси
Ціна	Безкоштовно преміум-підписка	Безкоштовно преміум-підписка	Безкоштовно	Безкоштовно, опенсорс
Безпека даних	Шифрування двофакторна автентифікація	Шифрування двофакторна автентифікація	Шифрування двофакторна автентифікація	Шифрування

У першому розділі ми ознайомились з аналогами програм для анотування тексту, розглянули їх інтерфейс. У цьому розділі ми проаналізуємо їх переваги. Розглянемо таблицю з порівнянням

Перед багатьма користувачами постійно виникає проблема необхідності компресії тексту, для опрацювання великих обсягів інформації у стислі терміни, для вирішення цієї проблеми користувачі вдаються до анотування тексту. Анотування тексту – це важлива функція для багатьох користувачів, існує багато інструментів для анотування тексту на різних платформах, включаючи Android. Кожен з цих інструментів має свої переваги та недоліки, і вибір конкретного інструменту залежить від потреб користувача та особливостей конкретної ситуації.

2.3 Вибір технологій та визначення стеку розробки

Вибір технологій для розробки Android-застосунку для анотування тексту є ключовим етапом, який визначить продуктивність, якість та майбутнє розвиток проєкту. У цьому розділі ми розглянемо різні аспекти вибору технологій та визначимо оптимальний стек розробки для нашого застосунку.

Першим кроком у виборі технологій є вибір мови програмування. У нас є два основних варіанти для розробки Android-додатків: Java та Kotlin.

Java є традиційною мовою програмування для Android-розробки. Вона має широку підтримку, багатий набір бібліотек та добре документовану спільноту розробників.

Kotlin є новішою мовою програмування, яка стала дуже популярною в останні роки. Вона пропонує сучасний та безпечний синтаксис, вбудовану підтримку нульових значень та інші переваги.

У нашому випадку ми виберемо Kotlin як основну мову програмування.

Для розробки Android-додатків найпопулярнішим інтегрованим середовищем розробки є Android Studio.

Android Studio є офіційним інтегрованим середовищем розробки для платформи Android. Воно забезпечує всі необхідні інструменти для створення, тестування та налагодження додатків, а також пропонує широкий набір допоміжних сервісів та плагінів.

Для ефективної розробки Android-додатків рекомендується використовувати модульну, чисту та розширювану архітектуру. Одним з популярних архітектурних підходів є архітектура Model-View-ViewModel (MVVM).

MVVM є архітектурним шаблоном, що дозволяє відокремлювати логіку додатку від його представлення та моделей даних. Він забезпечує чітке розділення відповідальностей та полегшує тестування і підтримку коду.

Android Jetpack є набором компонентів та бібліотек, розроблених Google для спрощення розробки Android-додатків. Він включає такі

компоненти, як Lifecycle, ViewModel, Room та інші, що полегшують роботу з Android-платформою.

Dagger та Hilt є фреймворками для виконання залежностей у Android-додатках. Вони допомагають керувати залежностями та реалізовувати принципи інверсії управління.

Вибір технологій та визначення стеку розробки є ключовим етапом у розробці Android-додатку для анотування тексту. Використання сучасних технологій та підходів, таких як Kotlin, Android Jetpack та MVVM, допоможе забезпечити ефективну та продуктивну розробку додатку, який буде відповідати потребам користувачів і забезпечувати зручний та надійний функціонал. Після детального аналізу існуючих варіантів шляхів розробки було обрано мову програмування Kotlin для написання додатку для анотації тексту.

3 РОЗРОБКА ТА РЕАЛІЗАЦІЯ ANDROID ЗАСТОСУНКУ ДЛЯ АНОТУВАННЯ ТЕКСТІВ

У цьому розділі буде детально розглянуто процес розробки та реалізації Android застосунку для анотування текстів. Першочергове завдання – створення функціонального, зручного у використанні та ефективного додатку, який відповідає вимогам сучасних користувачів.

На основі аналізу існуючих рішень, таких як Evernote, OneNote та Google Keep, було визначено ключові функції та особливості, які повинні бути реалізовані в нашому застосунку. У цьому розділі ми розглянемо архітектуру застосунку, проєктування бази даних, розробку користувацького інтерфейсу, реалізацію основних функцій та інтеграцію додаткових можливостей. Також буде представлено результати тестування та порівняння з існуючими рішеннями.

Розробка застосунку починається з вибору архітектурного підходу. Для забезпечення розширюваності, підтримки та тестованості додатку ми обрали архітектуру MVVM (Model-View-ViewModel) [3]. Цей підхід дозволяє чітко розділити відповідальності між різними компонентами, що спрощує розробку та подальшу підтримку.

Модель (Model) – відповідає за бізнес-логіку додатку та взаємодію з базою даних. У нашому випадку це класи, які описують структуру анотацій та операції з ними.

Вид (View) – відповідає за відображення даних користувачу та взаємодію з ним. Це XML-розмітки екранів, які описують зовнішній вигляд та розташування елементів інтерфейсу.

ViewModel – забезпечує зв’язок між Моделлю та Відом. Він отримує дані з Моделі, обробляє їх та передає Виду для відображення. ViewModel також обробляє події користувача та взаємодії з інтерфейсом.

3.1 Розробка користувацького інтерфейсу

Розробка користувацького інтерфейсу (UI) є одним з ключових етапів створення будь-якого програмного продукту, зокрема мобільних застосунків. Від інтерфейсу залежить зручність використання, естетична привабливість та загальне враження користувачів від застосунку. У цьому розділі ми розглянемо основні екрани нашого Android застосунку для анотування текстів, обговоримо макети екранів та взаємодію між ними за допомогою діаграм навігації, а також розглянемо використання XML для розмітки інтерфейсу.

Для початку розглянемо на UML-діаграмі принцип роботи додатку (див. рис. 3.1).

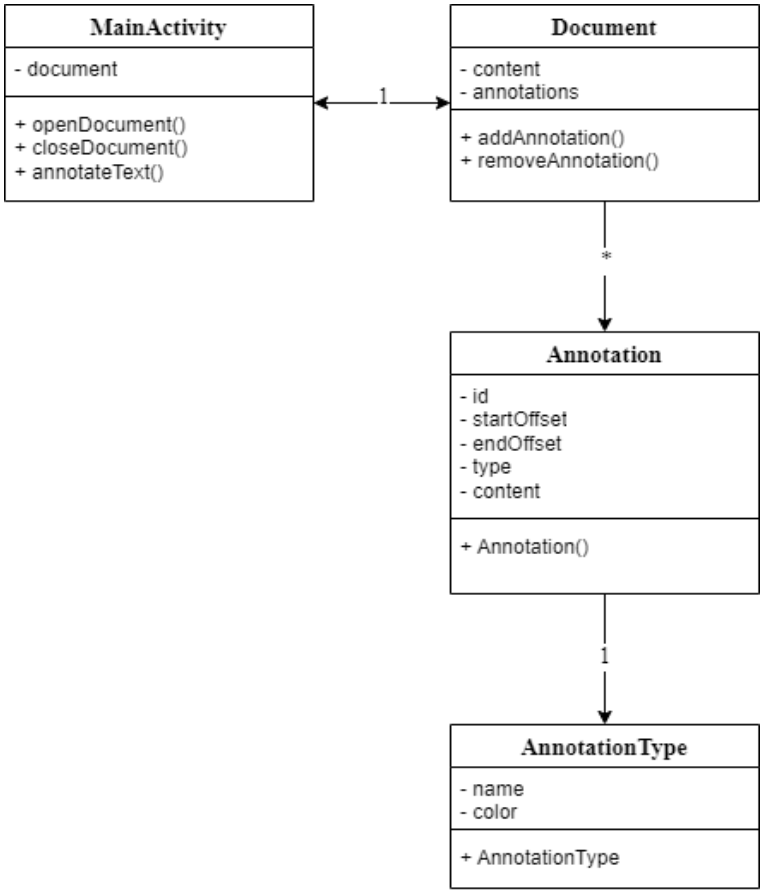

Рисунок 3.1 – UML – діаграма принцип роботи додатку

Огляд основних екранів застосунку. Основні екрани застосунку включають екран головного меню, екран створення/редагування анотацій та

екран перегляду списку анотацій. Кожен з цих екранів виконує певну функцію та забезпечує користувачу доступ до необхідних даних та можливостей.

Екран головного меню. Екран головного меню є основним екраном застосунку, на якому користувач бачить список всіх створених анотацій. На цьому екрані користувач може додавати нові анотації за допомогою кнопки «Додати» та видаляти існуючі анотації за допомогою кнопки «Видалити». Головний екран має бути інтуїтивно зрозумілим, зручним у використанні та забезпечувати швидкий доступ до основних функцій.

Екран створення/редагування анотацій. Екран створення/ редагування анотацій дозволяє користувачу створювати нові анотації або редагувати існуючі. Цей екран має Текстове поле для введення основного тексту анотації текстове поле для введення основного тексту анотації.

Проєктування макетів екранів є важливим етапом у розробці користувацького інтерфейсу, оскільки макети дозволяють візуалізувати розташування елементів на екрані та визначити, як користувач буде взаємодіяти з різними компонентами інтерфейсу. Головний екран має текстовий рядок, розташований у верхній частині, який дозволяє користувачу шукати анотації за ключовими словами, та кнопку «Додати», розташовану в нижньому правому куті екрану, яка відкриває екран створення нової анотації. На екрані створення анотацій є поле для заголовка у верхній частині, що дозволяє ввести заголовок анотації, і текстове поле для основного тексту, розташоване під полем для заголовка, яке дозволяє вводити основний текст анотації, а також кнопки додавання та видалення у нижній частині екрану, що дозволяють зберегти анотацію або відмінити зміни. Екран перегляду списку анотацій відображає основний текст анотації під заголовком і має кнопки анотування та видалення у нижній частині екрану.

Використання XML для розмітки інтерфейсу. Для створення користувацького інтерфейсу в Android ми використовуємо XML (eXtensible Markup Language), яка дозволяє чітко та структуровано описати розташування елементів інтерфейсу на екрані [5]. Використання XML для розмітки

інтерфейсу має кілька переваг: чіткість та структурованість, оскільки XML забезпечує чітке розділення логіки від представлення, що полегшує підтримку та розширення інтерфейсу; легкість редагування, адже XML-файли легко редагувати, що дозволяє швидко вносити зміни до інтерфейсу; можливість попереднього перегляду, оскільки Android Studio надає можливість попереднього перегляду XML-розміток, що дозволяє бачити зміни в інтерфейсі в режимі реального часу. У додатку А та у додатку Б будуть наведені приклади скриншотів розроблених нами користувацьких інтерфейсів.

3.2 Реалізація основних функцій

У цьому розділі ми зосередимося на детальному описі реалізації основних функцій нашого Android застосунку для анотування текстів. Зокрема, ми розглянемо модуль для роботи з текстом, який включає створення, редагування та видалення анотацій. Ці функції є основними для забезпечення коректної роботи застосунку та зручності користувачів.

Опис реалізації модуля для роботи з текстом. Реалізація модуля для роботи з текстом складається з кількох основних етапів: створення анотацій, редагування анотацій та видалення анотацій.

Створення нових анотацій є однією з основних функцій нашого застосунку (див. рис. 3.2). Для реалізації цієї функції необхідно забезпечити зручний інтерфейс для введення тексту та зберігання даних у базі даних [4].

Після опису основних етапів пропонуємо розглянути реалізацію цих етапів у нашому додатку.

На цьому скриншоті зображено написаний текст для анотації (див. рис. 3.3). Натиснувши на кнопку анотувати текст отримуємо скорочений текст.

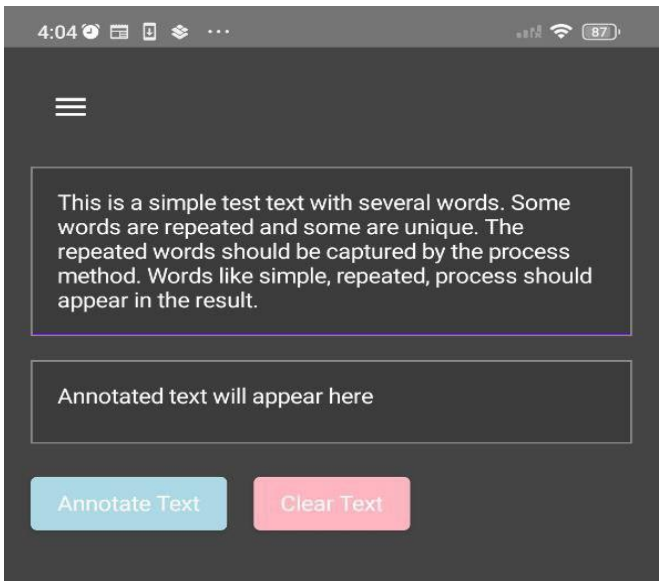

Рисунок 3.2 – Приклад роботи додатку

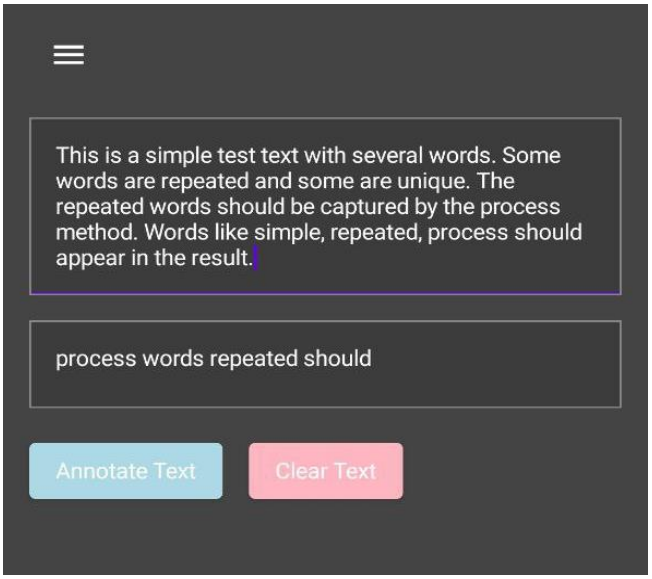

Рисунок 3.3 – Інтерфейс додатку

На інтерфейсі користувача для створення анотацій повинні бути наявні наступні елементи: поле для введення заголовка анотації, текстове поле для введення основного тексту анотації, опціональне поле для додавання міток, а також кнопки для збереження та відміни змін.

Редагування анотацій дозволяє користувачу оновлювати зміст уже створених анотацій. Ця функція включає завантаження даних анотації для редагування, внесення змін та збереження оновлених даних у базі.

Інтерфейс для видалення анотацій реалізується через кнопку на екрані

перегляду анотацій. Коли користувач натискає кнопку «Видалити», анотація видаляється з бази даних.

Після опису інтерфейсу, пропонуємо розглянути розроблений нами інтерфейс.

У цьому розділі ми детально розглянули реалізацію основних функцій Android застосунку для анотування текстів, включаючи створення, редагування та видалення анотацій. Ми розробили інтерфейси користувача для кожної з цих функцій, використовуючи XML-розмітку

Створення анотацій включає зручний інтерфейс для введення тексту та зберігання даних у базі даних. Редагування анотацій дозволяє користувачам оновлювати існуючі анотації, завантажуючи дані для редагування та зберігаючи оновлену інформацію. Видалення анотацій дозволяє користувачам видаляти непотрібні анотації з бази даних.

Реалізація цих функцій забезпечує повноцінний функціонал для роботи з текстовими анотаціями в нашому застосунку, надаючи користувачам зручний та ефективний інструмент для створення, редагування та видалення анотацій.

ВИСНОВКИ

У процесі розробки були створені наступні основні компоненти: інтерфейс користувача, де основна увага приділялася розробці зручного та інтуїтивно зрозумілого інтерфейсу, що дозволяє користувачам швидко орієнтуватися та виконувати необхідні дії, використовуючи сучасні практики UI/UX дизайну для забезпечення комфортного використання застосунку; функціонал анотування, що включає можливість виділення текстових фрагментів різними кольорами та додавання до них коментарів, з можливістю збереження та перегляду анотацій у зручному форматі; синхронізація та зберігання даних, яка дозволяє зберігати анотації локально на пристрої та передбачає можливість синхронізації з хмарним сервісом для доступу до анотацій з різних пристроїв; та тестування і оптимізація, що включало тестування застосунку на різних пристроях для виявлення та виправлення можливих помилок, а також оцінку продуктивності для оптимізації швидкодії та споживання ресурсів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Android Developers. Офіційна документація Android. *Android Developers*. URL: https://developer.android.com/docs (дата звернення: 18.02.2024).
2. Гугл. Офіційна документація мови програмування Kotlin. *Kotlinlang*. URL: https://kotlinlang.org/docs/home.html (дата звернення: 20.02.2024).
3. Kotlinlang. Офіційний репозиторій Kotlin на GitHub. *GitHub*. URL: https://github.com/JetBrains/kotlin (дата звернення: 21.02.2024).
4. Android Developers Blog. Блог розробників Android, де часто публікуються цікаві та корисні статті щодо розробки на Android. *Android Developers Blog*. URL: https://android-developers.googleblog.com/ (дата звернення: 17.03.2024).
5. Kotlin Programming Language. Офіційний блог мови Kotlin, де можна знайти інформацію про останні новини та оновлення. *JetBrains Blog*. URL: https://blog.jetbrains.com/kotlin/ (дата звернення: 20.04.2024).

ДОДАТОК А

Код файлу annotationwindow.kt

```
package com.example.summarizeapp
import androidx.compose.foundation.border
import androidx.compose.foundation.isSystemInDarkTheme
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.text.BasicTextField
import androidx.compose.material.*
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.Menu
import androidx.compose.material3.DrawerDefaults.scrimColor
import androidx.compose.runtime.Composable
import androidx.compose.runtime.MutableState
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.runtime.rememberCoroutineScope
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import kotlinx.coroutines.launch
val LightBlue = Color(173, 216, 230)
val LightRed = Color(255, 182, 193)
val White = Color(255, 255, 255)
```

```
@Composable
fun AnnotationWindow(isDarkTheme: MutableState<Boolean>) {
    val originalText = remember { mutableStateOf("") }
    val annotatedText = remember { mutableStateOf("") }
    val annotationProcessor = AnnotationProcessor()
    val colors = if (isDarkTheme.value)
MaterialTheme.colors.copy(background = Color.DarkGray) else
MaterialTheme.colors.copy(background = Color.White)
    val textColor = if (isDarkTheme.value) Color.White else Color.Black
    val drawerState = rememberDrawerState(DrawerValue.Closed)
    val scope = rememberCoroutineScope()
    ModalDrawer(
        drawerState = drawerState,
        drawerContent = {
            Surface(color = colors.background) {
                Column(modifier = Modifier.padding(16.dp)) {
                    Text("Settings", style = TextStyle(color = textColor, fontSize =
24.sp))
                    Spacer(modifier = Modifier.height(24.dp))
                    Switch(
                        checked = isDarkTheme.value,
                        onCheckedChange = { isDarkTheme.value = it },
                        colors = SwitchDefaults.colors(
                            checkedThumbColor = textColor,
                            checkedTrackColor = textColor.copy(alpha = 0.6f),
                            uncheckedThumbColor = textColor,
                            uncheckedTrackColor = textColor.copy(alpha = 0.6f)
                        )
                    )
                    Text(if (isDarkTheme.value) "Dark Mode" else "Light Mode",
```

```
style = TextStyle(color = textColor))
            }
        }
    },
    content = {
        Surface(color = colors.background) {
            Column(
                modifier = Modifier
                    .fillMaxSize()
                    .padding(16.dp),
                verticalArrangement = Arrangement.spacedBy(16.dp)
            ) {
                Row(
                    modifier = Modifier.fillMaxWidth(),
                    horizontalArrangement = Arrangement.Start
                ) {
                    IconButton(onClick = {
                        scope.launch {
                            drawerState.open()
                        }
                    }) {
                        Icon(Icons.Filled.Menu, contentDescription = "Menu", tint
= textColor)
                    }
                }


                Text("Original Text")
                TextField(
                    value = originalText.value,
```

```
                onValueChange = { originalText.value = it },
                modifier = Modifier
                    .fillMaxWidth()
                    .border(width = 1.dp, color = Color.Gray),
                textStyle = TextStyle(color = textColor),
                placeholder = { Text("Enter original text here", style =
TextStyle(color = textColor)) }
            )
            Text("Annotated Text")
            TextField(
                value = annotatedText.value,
                onValueChange = { annotatedText.value = it },
                modifier = Modifier
                    .fillMaxWidth()
                    .border(width = 1.dp, color = Color.Gray),
                textStyle = TextStyle(color = textColor),
                placeholder = { Text("Annotated text will appear here", style
= TextStyle(color = textColor)) }
            )
            Row {
                Button(
                    onClick = { annotatedText.value =
annotationProcessor.process(originalText.value) },
                    colors = ButtonDefaults.buttonColors(backgroundColor =
LightBlue, contentColor = textColor)
                ) {
                    Text("Annotate Text", style = TextStyle(color =
textColor))
                }
                Spacer(modifier = Modifier.width(16.dp))
```

```
                    Button(
                        onClick = {
                            originalText.value = ""
                            annotatedText.value = ""
                        },
                        colors = ButtonDefaults.buttonColors(backgroundColor =
LightRed, contentColor = textColor)
                    ) {
                        Text("Clear Text", style = TextStyle(color = textColor))
                    }
                }
            }
        },
        scrimColor = colors.background.copy(alpha =
DrawerDefaults.ScrimOpacity)
    )
}
@Preview
@Composable
fun PreviewAnnotationWindow() {
    val isDarkTheme = remember { mutableStateOf(false) }
    AnnotationWindow(isDarkTheme)
}
```

ДОДАТОК Б

Код файлу androidmanifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.SummarizeApp"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="@string/app_name"
            android:theme="@style/Theme.SummarizeApp">
            <intent-filter>
              <action android:name="android.intent.action.MAIN" />
              <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

ДОДАТОК В

Скриншоти роботи додатку

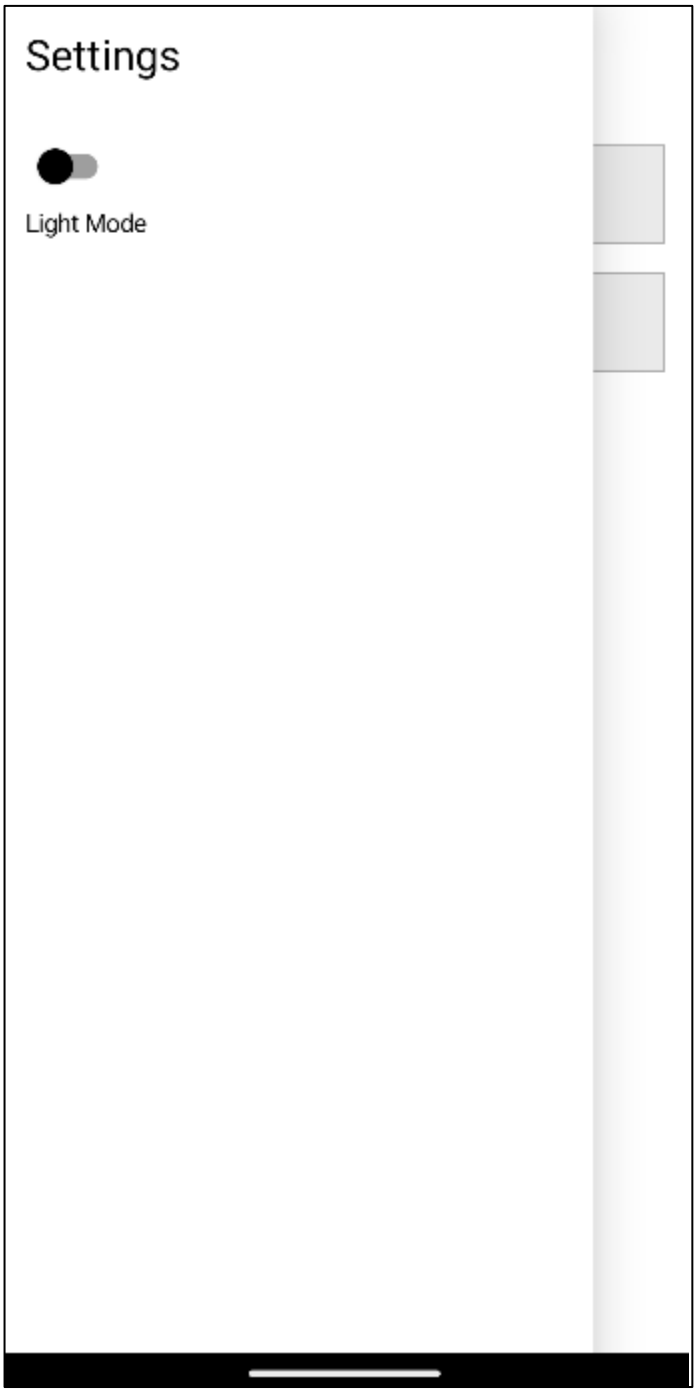

Рисунок В.1 – Меню додатку

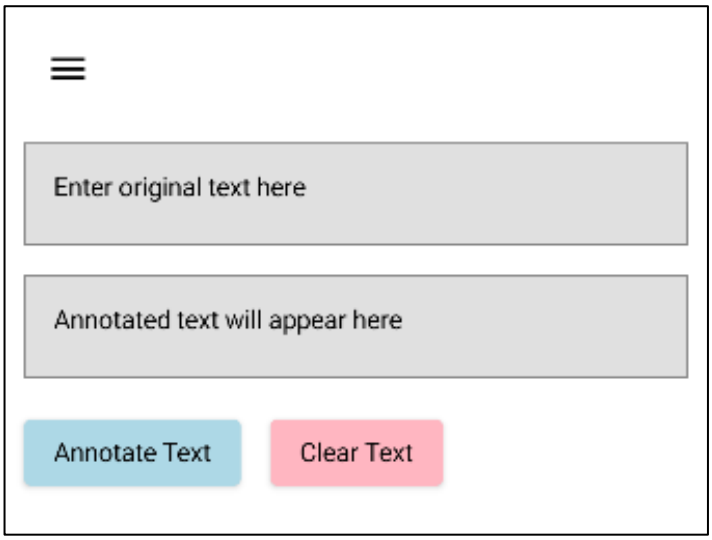

Рисунок В.2 – Приклад світлої теми

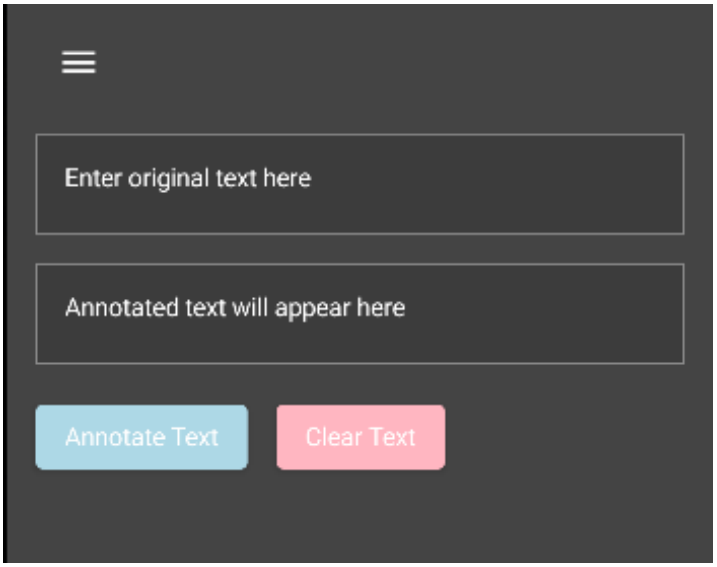

Рисунок В.3 – Приклад темної теми