МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ЕКОНОМІКО-ПРАВНИЧИЙ ФАХОВИЙ КОЛЕДЖ ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ»

Циклова комісія математичних дисциплін та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «ПРОЄКТУВАННЯ ТА РОЗРОБКА ЗАСТОСУНКУ ДЛЯ КЕРУВАННЯ НОТАТКАМИ»

Виконав: здобувач освіти 4 курсу, групи К121-20
Спеціальність 121 Інженерія програмного забезпечення
(шифр і спеціальність)
Михайло СОСНОВСЬКИЙ
(ім'я та ПРІЗВИЩЕ)
Керівник Юлія ЛИМАРЕНКО
(ім'я та ПРІЗВИЩЕ)

Рецензент доцент кафедри програмної інженерії ЗНУ, доцент, к.ф.-м.н. Олексій КУДІН
(посада, вчене звання, науковий ступінь, ім'я та ПРІЗВИЩЕ)

Запоріжжя
2024

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ВСП «Економіко-правничий фаховий коледж ЗНУ»

Освітньо–кваліфікаційний рівень фаховий молодший бакалавр

Спеціальність 121 – Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ

Голова циклової комісії математичних дисциплін та інформаційних технологій

_____ Т.М. Смолянкова
(підпис)

“ 14 ” червня 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

СОСНОВСЬКОМУ Михайлу Валерійовичу
(ПРІЗВИЩЕ, ім’я та по– батькові)

1. Тема роботи «Проєктування та розробка застосунку для керування нотатками»

Керівник роботи к.т.н., ЛИМАРЕНКО Юлія Олексіївна
(науковий ступінь, вчене звання, прізвище, ім’я, по батькові)

Затверджені наказом ВСП ЕПФК ЗНУ від « 30 » листопада 2023 р. № 2004-с

2. Строк подання студентом роботи 01.06.2024

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

4. Зміст розрахунково– пояснювальної записки (перелік питань, які потрібно розробити)
1. 1.Аналіз існуючих рішень для створення нотаток на Android, огляд сучасних технологій та інструментів для розробки мобільних застосунків.
2. Проектування, вимоги до програмного застосунку.
3.Реалізація програмного застосунку
презентація до захисту

6. Консультанти розділів роботи

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

7. Дата видачі завдання _____ 30.11.2023

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи:		
	Постановка задачі	Грудень 2023	виконано
2.	Збір вихідних даних, обробка методичних		
	та теоретичних джерел	Лютий 2024	виконано
3.	Розробка першого розділу:		
	Аналіз існуючих рішень для створення нотаток на Android, огляд сучасних технологій та інструментів для розробки мобільних застосункі	Лютий 2024	виконано
4.	Розробка другого розділу:		
	Проектування, вимоги до програмного застосунку	Березень 2024	виконано
5.	Розробка третього розділу:		
	Реалізація програмного застосунку	Квітень 2024	виконано
6.	Оформлення і нормоконтроль кваліфікаційної роботи та перевірка на плагіат	Червень 2024	виконано
7.	Захист кваліфікаційної роботи	21.06.2024	виконано

Здобувач освіти _____ Михайло СОСНОВСЬКИЙ
(підпис) (ім'я ПРІЗВИЩЕ)

Керівник роботи _____ Юлія ЛИМАРЕНКО
(підпис) (ім'я ПРІЗВИЩЕ)

Нормоконтроль пройдено

Нормоконтролер _____ Юлія БОРИСОВСЬКА
(підпис) (ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Кваліфікаційна робота: 36 сторінок, 1 таблиця, 10 ілюстрацій, 8 позицій у переліку посилань.

Об’єкт дослідження – Аспекти розробки, реалізації та тестування мобільного додатка, призначеного для створення, збереження та організації текстових нотаток на мобільних пристроях, що працюють під управлінням операційної системи Android.

Предмет дослідження – програмний застосунок "Нотатки" для платформи Android.

Мета дослідження – розробка програмного застосунку "Нотатки" для платформи Android.

НОТАТКИ, ЗАСТОСУНОК, JAVA.

SUMMARY

Qualification Work: 36 pages, 1 table, 10 Illustrations, 8 items in the list of links.

The object of research is aspects of developing, implementing, and testing a mobile application designed to create, save, and organize text notes on mobile devices running the Android operating system.

The subject of the research is the notes software application for the Android platform.

The aim of the study is to develop the notes software application for the Android platform.

NOTES, APPLICATION, JAVA.

ЗМІСТ

ВСТУП

У сучасному світі мобільні додатки відіграють важливу роль у повсякденному житті мільйонів користувачів.

Зростаюча популярність смартфонів та планшетів зумовлює підвищений попит на різноманітні програмні рішення, які допомагають організовувати особисту та робочу діяльність. Один із найбільш затребуваних типів мобільних додатків — це програми для створення та управління нотатками. Вони дозволяють користувачам зберігати важливу інформацію, ідеї, завдання та нагадування в зручному цифровому форматі. У зв'язку з цим, я вирішив розробити ефективний і функціональний застосунок "Нотатки" для платформи Android, бо вважаю це актуальним завданням.

Метою даного дослідження є розробка програмного застосунку "Нотатки" для Android, який забезпечує користувачам зручний інструмент для створення та організації текстових нотаток. Основні завдання дослідження включають:

- Аналіз існуючих рішень та визначення ключових вимог до додатку.
- Вибір відповідних технологій та інструментів для розробки.
- Проектування архітектури програмного забезпечення та користувацького інтерфейсу.
- Реалізація функціоналу додатку згідно з вимогами.
- Тестування додатку для забезпечення його якості та надійності.
- Оптимізація продуктивності та забезпечення можливостей для подальшого розвитку.

Об’єктом дослідження є програмний застосунок "Нотатки" для платформи Android. Предметом дослідження є процеси проектування, розробки, тестування та оптимізації мобільного додатку для створення та управління текстовими нотатками.

Для досягнення мети дослідження використовуються наступні методи:

• Аналіз літератури та існуючих рішень для виявлення ключових вимог та кращих практик.

• Проектування архітектури та інтерфейсу на основі сучасних методологій розробки програмного забезпечення.

• Реалізація програмного застосунку за допомогою обраних технологій та інструментів.

• Тестування додатку з використанням різних видів тестування (функціонального, регресійного, навантажувального).

• Аналіз результатів тестування та оптимізація додатку.

Завдяки цьому підходу, було створено ефективний та надійний програмний застосунок "Нотатки" для Android, який відповідатиме потребам користувачів.

1 СУЧАСНІ МЕТОДИ АНАЛІЗУ

У контексті розробки програмного застосунку "Нотатки" для Android, сучасні методи аналізу охоплюють декілька ключових аспектів: аналіз вимог, аналіз користувацького досвіду, аналіз ринку та конкурентів, а також технічний аналіз [1].

1.1 Аналіз вимог

Аналіз вимог включає визначення та документування потреб і очікувань користувачів від майбутнього додатку. До основних методів належать:

• Збір вимог: інтерв'ю з потенційними користувачами, опитування, фокус-групи, аналіз відгуків на схожі продукти.

• Формалізація вимог: використання технік як User Stories (користувацькі історії), Use Cases (випадки використання), створення діаграм вимог.

• Пріоритезація вимог: методи MSCW (Must have, Should have, Could have, Won't have), Kano Model для визначення пріоритетності функцій (див. рис. 1.1).

Рисунок 1.1 – MSCW метод

1.2 Аналіз користувацького досвіду (UX)

- Аналіз користувацького досвіду спрямований на створення зручного та інтуїтивно зрозумілого інтерфейсу для користувачів:
- Персоналізація та сценарії використання: створення персони користувача (User Persona) та сценаріїв використання (User Scenarios).
- Прототипування: розробка прототипів інтерфейсу (Wireframes, Mockups), які дозволяють протестувати взаємодію користувача з додатком на ранніх етапах.
- Юзабіліті тестування: проведення тестувань з реальними користувачами для оцінки зручності інтерфейсу та виявлення можливих проблем.

1.3 Аналіз ринку та конкурентів

Цей метод допомагає вивчити існуючі рішення на ринку та зрозуміти, як новий додаток може виділитися:

- SWOT-аналіз: аналіз сильних (Strengths), слабких (Weaknesses) сторін, можливостей (Opportunities) та загроз (Threats) для розробки додатку.
- Benchmarking: порівняння характеристик, функціональності та продуктивності конкурентних додатків.
- Аналіз відгуків та рейтингів: вивчення відгуків користувачів на платформах, таких як Google Play, для виявлення сильних та слабких сторін існуючих рішень.

1.4 Технічний аналіз

Технічний аналіз включає вибір технологій, архітектури та інструментів для реалізації додатку:

- Аналіз технологій: оцінка мов програмування (Java, Kotlin), фреймворків (Android SDK, Jetpack), бібліотек (Room, Retrofit).
- Аналіз архітектури: вибір архітектурного підходу (MVVM, MVP), створення діаграм архітектури (компонентна діаграма, діаграма класів).
- Аналіз продуктивності: оцінка потенційної продуктивності додатку, вибір оптимальних рішень для роботи з базою даних, оптимізація швидкості завантаження.

Застосування цих методів аналізу дозволяє створити добре обґрунтований, функціональний та конкурентоспроможний додаток "Нотатки" для Android, який відповідає потребам користувачів і вимогам сучасного ринку.

1.5 Етапи створення застосунку

Створення програмного застосунку "Нотатки" для Android включає кілька ключових етапів, кожен з яких є важливим для досягнення кінцевого результату. Нижче наведені основні етапи розробки:

1.5.1 Планування та аналіз

- Визначення мети та цілей: Формулювання основної мети проекту, визначення ключових цілей та завдань.
- Збір вимог: Проведення інтерв'ю, опитувань, фокус-груп для визначення потреб користувачів та бізнес-вимог.
- Аналіз вимог: Документування вимог, створення користувацьких історій (User Stories) та випадків використання (Use Cases).
- Технічне завдання: Створення детального технічного завдання на основі зібраних вимог.

1.5.2 Проектування

- Архітектурне проектування: Вибір архітектурного підходу (MVVM, MVP), створення діаграм архітектури.
- Проектування бази даних: Вибір моделі бази даних, проектування структури таблиць, визначення зв’язків між таблицями.
- Проектування інтерфейсу користувача (UI/UX): Створення прототипів (Wireframes, Mockups), розробка дизайну інтерфейсу, проведення юзабіліті тестувань.

1.5.3 Розробка

- Налаштування середовища розробки: Вибір та налаштування інструментів та середовища розробки (Android Studio, Git).
- Реалізація функціональності: Написання коду додатку, створення основних модулів, інтеграція з базою даних (Room, SQLite).
- Розробка інтерфейсу користувача: Реалізація користувацького інтерфейсу, налаштування анімацій та переходів.
- Інтеграція сторонніх сервісів: Впровадження сторонніх API та сервісів (наприклад, хмарне зберігання, синхронізація даних).

1.5.4 Тестування

- Функціональне тестування: Перевірка основних функцій додатку на відповідність вимогам.
- Тестування користувацького інтерфейсу: Перевірка зручності та інтуїтивності інтерфейсу.
- Регресійне тестування: Перевірка стабільності системи після внесення змін або додавання нових функцій.
- Навантажувальне тестування: Оцінка продуктивності додатку під час високих навантажень.

1.5.5 Оптимізація

- Оптимізація продуктивності: Поліпшення швидкодії додатку, зниження використання пам'яті та ресурсів пристрою.
- Поліпшення користувацького досвіду: Внесення змін до інтерфейсу на основі зворотного зв'язку користувачів та результатів тестувань.

- Підготовка до релізу: Створення релізних збірок, підготовка документації, опису додатку для магазину.
- Публікація в Google Play: Завантаження додатку в Google Play, налаштування сторінки додатку, проходження процесу перевірки.
- Маркетинг та просування: Проведення рекламних кампаній, взаємодія з користувачами через соціальні мережі та інші канали.

1.5.6 Підтримка та розвиток

- Підтримка користувачів: Відповіді на запитання користувачів, вирішення технічних проблем.
- Оновлення та виправлення помилок: Регулярне оновлення додатку, виправлення знайдених помилок, додавання нових функцій.
- Аналіз зворотного зв'язку: Збір та аналіз відгуків користувачів для подальшого покращення додатку.

Кожен з цих етапів є важливим для створення якісного, функціонального та зручного програмного застосунку "Нотатки" для Android.

1.6 Аналіз існуючих рішень для створення нотаток на Android

1.6.1 Google Keep

Google Keep — це офіційний додаток для нотаток від Google (див. рис. 1.2), який пропонує:

- Функціональність: Створення текстових та голосових нотаток, списків, нагадувань, можливість додавання зображень.

- Інтеграція: Синхронізація з Google Drive, підтримка роботи на різних пристроях.
- Інтерфейс: Простий та інтуїтивний дизайн, різнокольорові нотатки для кращої організації.
- Переваги: Безкоштовний доступ, інтеграція з екосистемою Google, швидкий доступ через веб-версію.

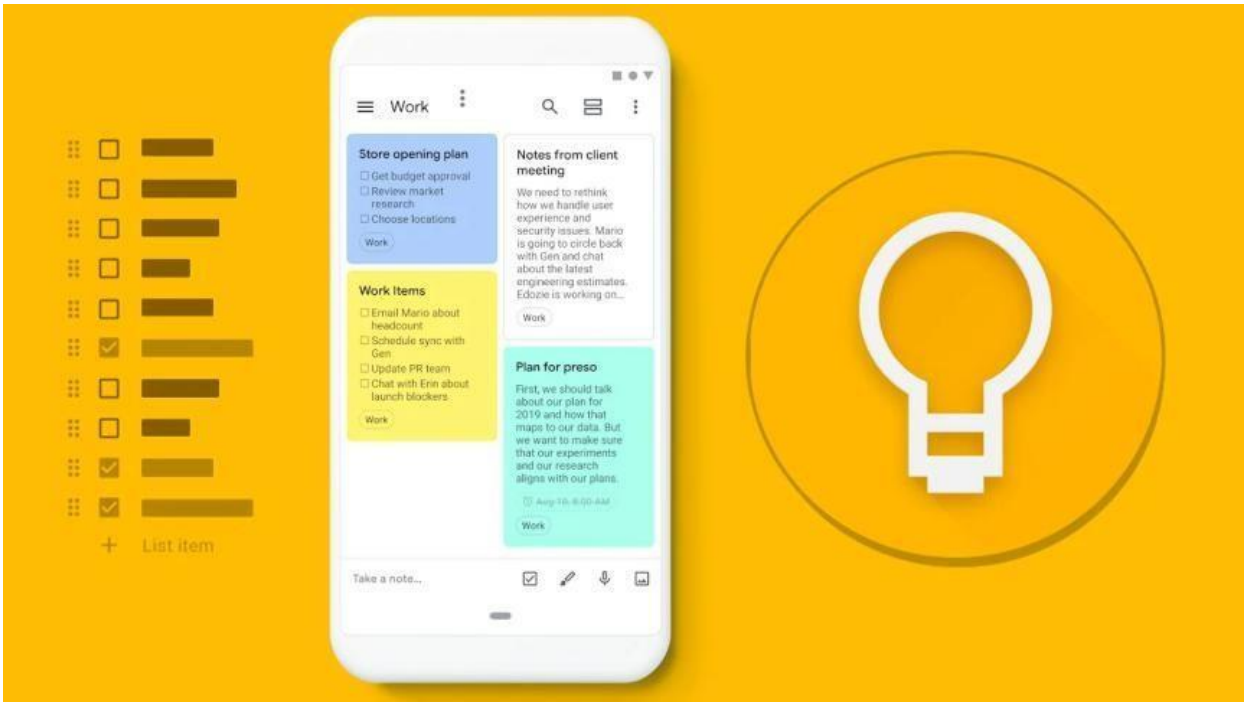

Рисунок 1.2 – Інтерфейс Google Keep

1.6.2 Microsoft OneNote

Microsoft OneNote — це потужний інструмент для нотаток (див. рис. 1.3), який є частиною пакету Microsoft Office:

- Функціональність: Створення текстових нотаток, малюнків, таблиць, можливість впорядковувати нотатки в блокноти та розділи.
- Інтеграція: Синхронізація з OneDrive, інтеграція з іншими додатками Microsoft Office.

- Інтерфейс: Багатофункціональний та гнучкий дизайн, підтримка стилуса для рукописних нотаток.
- Переваги: Глибока інтеграція з Microsoft Office, безкоштовний доступ, великий набір функцій для організації.

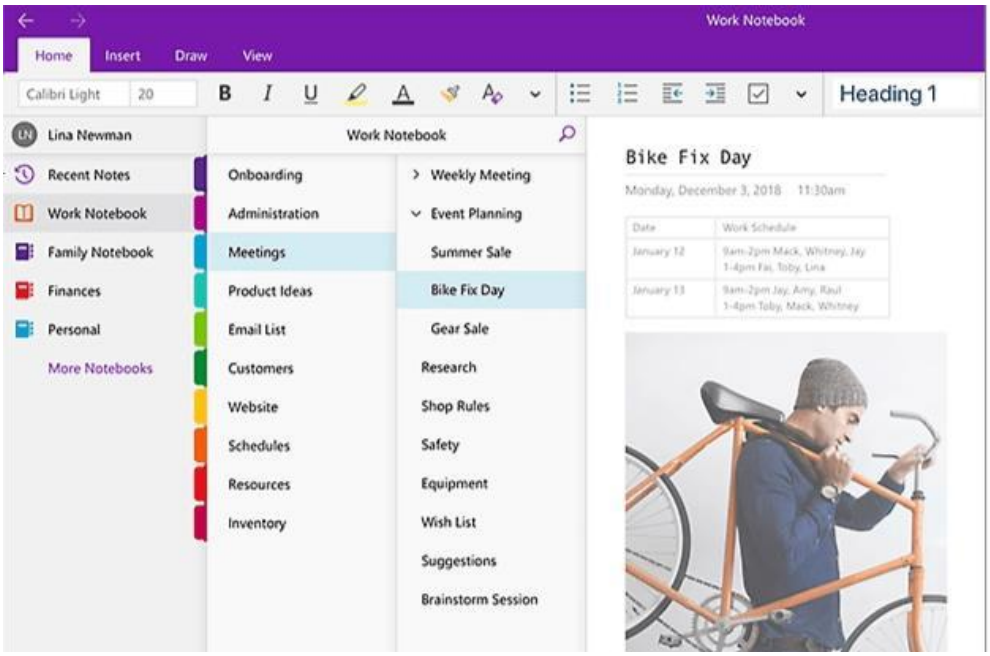

Рисунок 1.3 – Інтерфейс Microsoft OneNote

1.6.3 Evernote

Evernote — популярний додаток для управління нотатками та завданнями (див. рис. 1.4):

- Функціональність: Створення текстових, аудіо та фото нотаток, можливість організовувати нотатки у блокноти, теги.
- Інтеграція: Синхронізація з різними платформами, інтеграція з хмарними сервісами та іншими додатками.
- Інтерфейс: Потужний та багатофункціональний дизайн, підтримка сканування документів та рукописних нотаток.
- Переваги: Можливість зберігання великих об'ємів інформації, широкі можливості для організації та пошуку.

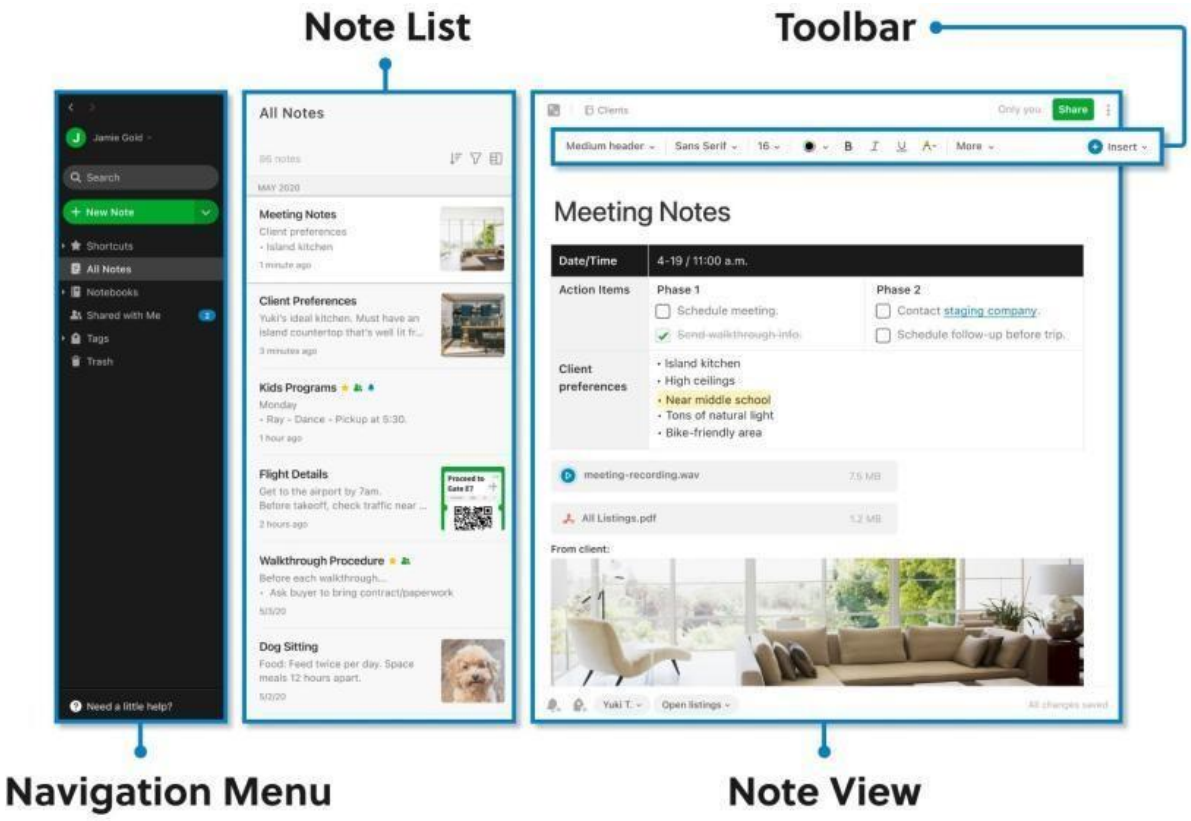

Рисунок 1.4 – Інтерфейс Evernote

1.6.4 Simplenote

Simplenote — легкий та швидкий додаток для створення простих текстових нотаток (див. рис. 1.5):

- Функціональність: Створення та редагування текстових нотаток, підтримка синхронізації між пристроями.
- Інтеграція: Синхронізація з веб-версією та іншими платформами.
- Інтерфейс: Мінімалістичний та простий дизайн, зручний для швидкого створення нотаток.
- Переваги: Безкоштовний доступ, швидкість та простота використання.

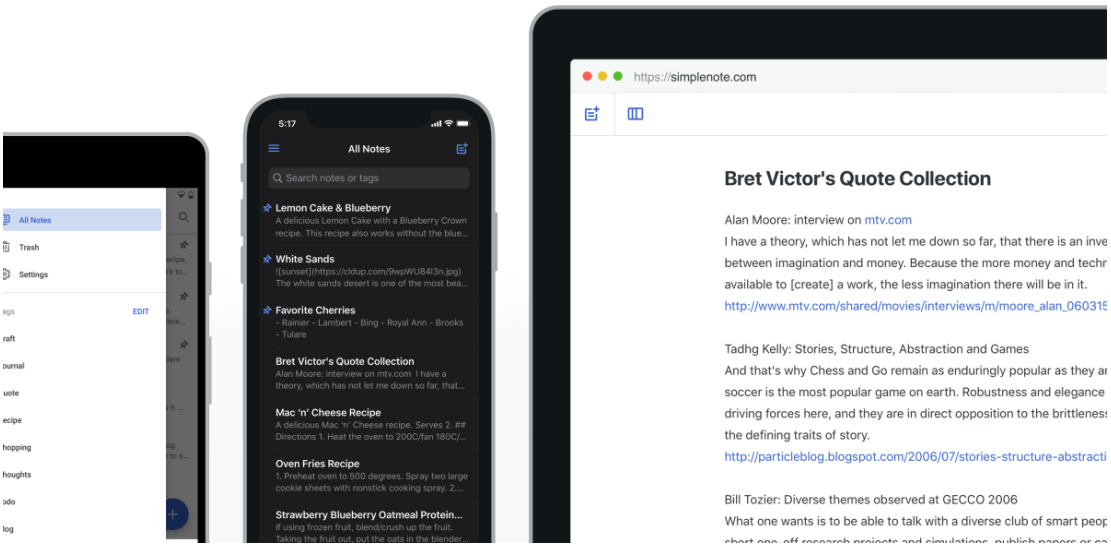

Рисунок 1.5 – Інтерфейс Simplenote

1.7 Огляд сучасних технологій та інструментів для розробки мобільних застосунків

1.7.1 Мови програмування

- Java: Традиційна мова для розробки під Android, підтримується з моменту створення платформи [4].
- Kotlin: Офіційно рекомендована мова для Android від Google, сучасна, безпечна та зручна у використанні [1].

1.7.2 Фреймворки та бібліотеки

- Android SDK: Офіційний набір інструментів для розробки додатків під Android [5].
- Android Jetpack: Набір бібліотек, які допомагають спростити розробку, включаючи компоненти для роботи з архітектурою, UI, навігацією, базами даних [5].

- Room: Бібліотека для роботи з базами даних SQLite, спрощує роботу з даними та забезпечує зручний API [5].
- Retrofit: Бібліотека для роботи з мережевими запитами, підтримка REST API [5].

1.7.3 Інструменти розробки

- Android Studio: Офіційне середовище розробки від Google, включає всі необхідні інструменти для створення, налагодження та тестування додатків.
- Gradle: Система автоматизації складання проектів, яка використовується в Android Studio.
- Firebase: Платформа від Google для розробки мобільних додатків, включає базу даних у реальному часі, аутентифікацію, аналітику.
- 4. Інструменти для дизайну
- Sketch: Інструмент для створення прототипів та дизайну інтерфейсів.
- Figma: Онлайн-платформа для дизайну та спільної роботи над макетами.
- Adobe XD: Інструмент для дизайну та прототипування користувацького інтерфейсу.

Аналіз існуючих рішень для створення нотаток на Android показує, що ринок вже насичений потужними інструментами, кожен з яких має свої переваги та унікальні особливості.

Для розробки конкурентоспроможного додатку "Нотатки" важливо використовувати сучасні технології та інструменти, які забезпечують високу продуктивність, зручність використання та гнучкість у реалізації функціоналу. Використання Kotlin, Android Jetpack, Room та інших сучасних технологій дозволить створити якісний та функціональний продукт.

2 ПРОЕКТУВАННЯ ТА ВИМОГИ ДО ПРОГРАМНОГО ЗАСТОСУНКУ

Функціональні вимоги — це опис функцій, які програмний застосунок повинен виконувати. Вони визначають поведінку системи у відповідь на певні вхідні дані, а також умови, за яких ця поведінка відбувається. Основна мета функціональних вимог полягає в тому, щоб забезпечити зрозумілий і точний опис того, що система повинна робити.

2.1 Створення нотаток

- Користувач має можливість створювати текстові нотатки.
- Нотатки включають основний текст.
- Підтримка додавання заголовків до нотаток для кращої організації.

2.2 Видалення нотаток

- Користувач може видаляти вже створені нотатки.

2.3 Інтерфейс користувача

- Інтуїтивно зрозумілий та зручний інтерфейс.
- Приємне та привабливе оформлення застосунку

2.4 Сумісність

- Підтримка різних версій Android (мінімальна версія SDK 21 і вище).
- Оптимізація для різних розмірів екрану.

2.5 Опис функціоналу та користувацького інтерфейсу

2.5.1 Головний екран

- Відображення усіх нотаток у списку.
- Кнопка для створення нової нотатки.
- Кнопка видалення нотатки.
- Поле пошуку для швидкого знаходження нотаток.
- Фільтри для відображення нотаток за назвою.

2.5.2 Створення нотатки

1. Користувач натискає на поле "Enter Note".
2. Користувач вводить текст нотатки.
3. Користувач натискає кнопку "Add Note".
4. Нотатка додається в базу даних і з'являється на головному екрані.

2.5.3 Видалення нотатки

- Користувач натискає кнопку видалення біля нотатки на головному екрані.
- Застосунок виводить підтвердження видалення.
- Користувач підтверджує видалення.
- Застосунок видаляє нотатку з бази даних і оновлює список нотаток на головному екрані.

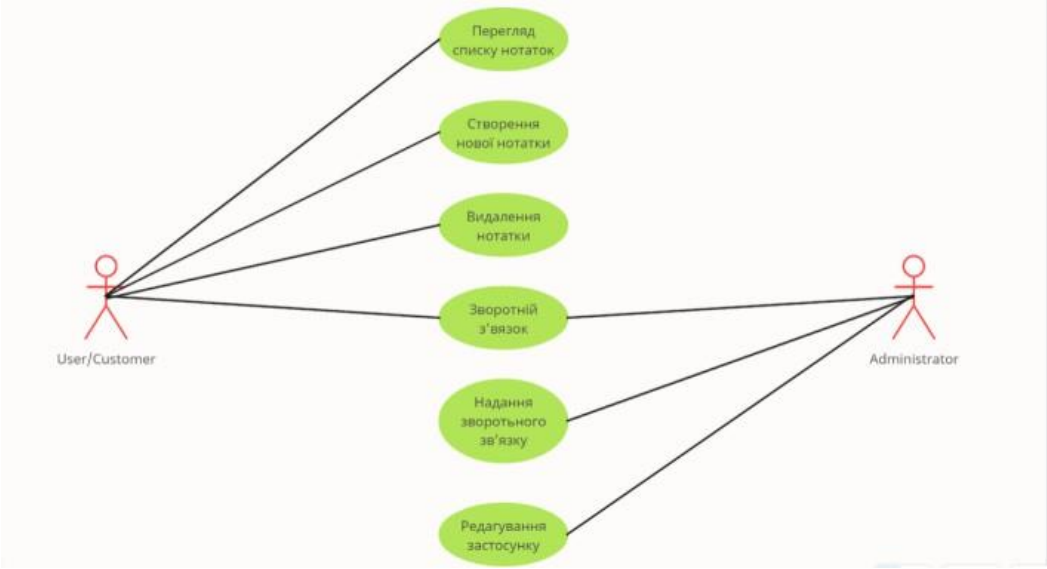

Рисунок 2.1 – UML Діаграма прецедентів

Ця діаграма прецедентів (див. рис. 2.1) допомагає визначити функціональність системи, показуючи, які дії можуть виконувати користувачі та адміністратори. Вона спрощує спілкування між розробниками, дизайнерами і менеджерами проекту, наочно демонструючи, хто може виконувати які дії в системі. Діаграма служить частиною проектної документації, корисною на всіх етапах розробки та підтримки системи. Також вона важлива при проектуванні архітектури системи і плануванні розробки, допомагаючи виявити всі необхідні компоненти і взаємодії для створення ефективного і зручного продукту.

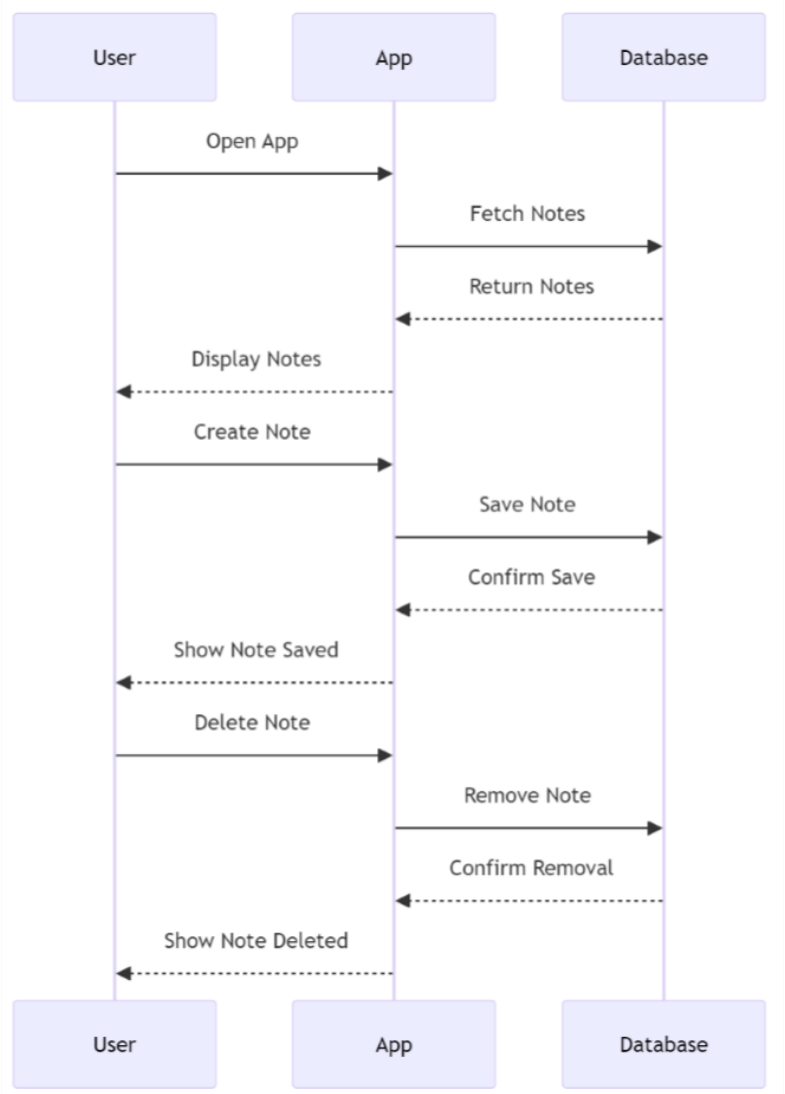

Рисунок 2.2 – UML Діаграма послідовності

Проектування застосунку "Нотатки" для Android включає визначення архітектури, вимог та функціональності.

Використання архітектури MVVM дозволяє чітко розділити відповідальності між компонентами, що забезпечує надійність та підтримуваність коду. Вимоги до застосунку включають основні функціональні можливості для роботи з нотатками(див. рис. 2.2), а також нефункціональні вимоги щодо продуктивності, інтерфейсу користувача, безпеки та сумісності. Опис функціоналу та користувацького інтерфейсу визначає основні екрани та їх елементи, що забезпечує зручність використання додатку.

3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАСТОСУНКУ

3.1 Вибір середовища розробки

Для реалізації програмної частини кваліфікаційної роботи я використовував IDE Android Studio від Google (див. рис. 3.1), так як на мою думку воно вважається найкращим рішенням на ринку в даний момент.

В Android Studio можна створювати додатки для останньої версії Android, а створений додаток можна відразу ж перевірити на наявність помилок, протестувати різними інструментами всі елементи застосунку і заздалегідь виявити всі можливі огріхи в її роботі [7].

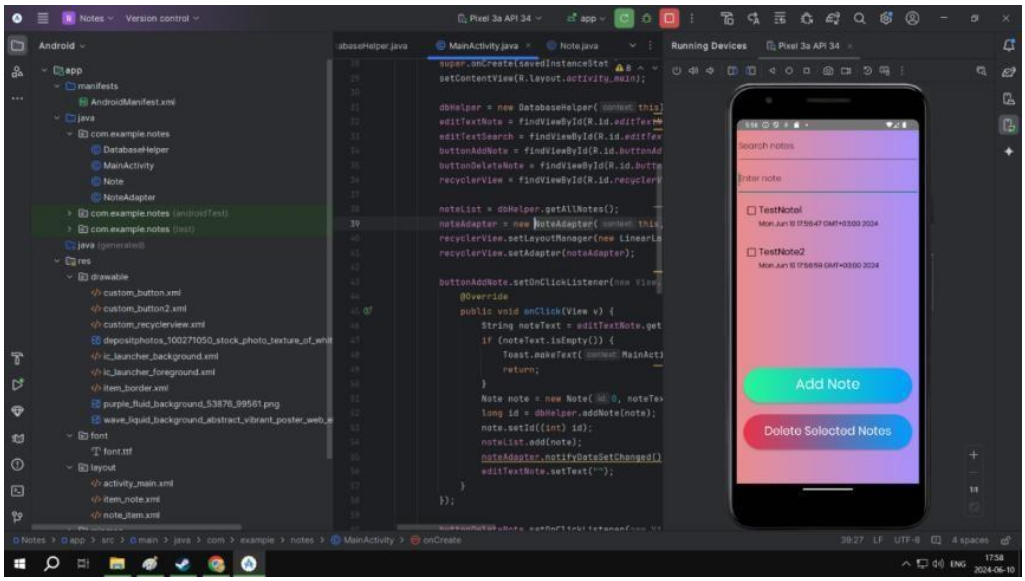

Рисунок 3.1 – Інтерфейс Android Studio

3.2 Програмна частина

Для розробки програмної частини мені було надано дві мови - Java і Kotlin, обидві мови є об'єктно-орієнтованими і таким чином підтримують поліморфізм, успадкування, інкапсуляція а так само абстракцію [3].

Я вибрав Java [8], оскільки вона вважається найпопулярнішою мовою.

9,16 % розробників віддають перевагу Kotlin, більш ніж 32 % — Java. Однак ці опитування (див. рис. 3.2) також показують, що Kotlin подобається програмістам більше, ніж Java, і швидко нарощує аудиторію [1].

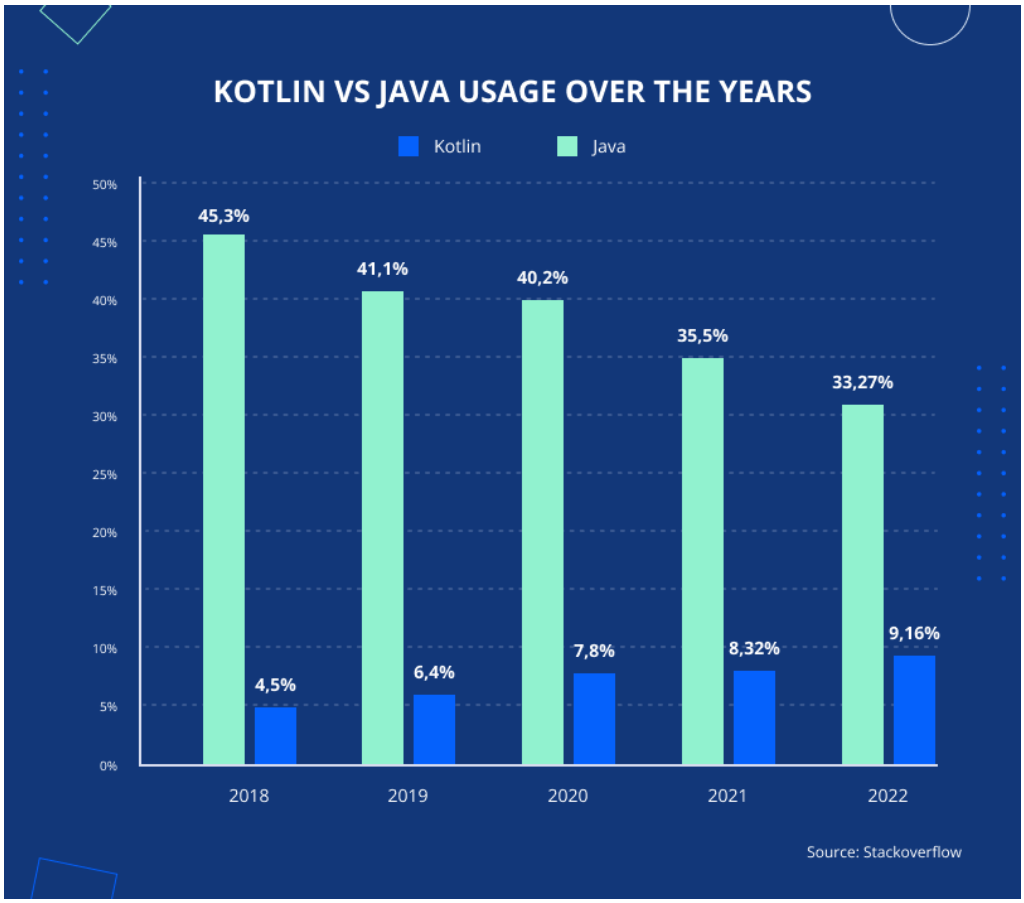

Рисунок 3.2 – Статистика використання Java та Kotlin

В якості бази даних я використовував реляційну базу даних Sqlite, мій вибір припав саме на цю БД так як це вбудована Кросплатформна система, яка забезпечує швидку обробку інформації. Для обробки не використовуються сторонні сервери, служби або бібліотеки. Все необхідне для роботи вже є [2].

Структура бази даних: У базі даних є одна таблиця з назвою "notes". Вона має три стовпці:

- id (типу INTEGER): Це унікальний ідентифікатор для кожного нотатку. Він автоматично збільшується (AUTOINCREMENT).
- text (типу TEXT): Це поле для зберігання тексту нотатку.

- date (типу INTEGER): Це поле для зберігання дати створення нотатку у форматі мілісекунд.

Створення нотатку: Коли користувач додає новий нотаток, його текст та дата зберігаються в базі даних за допомогою методу addNote() класу DatabaseHelper (див. рис. 3.3).

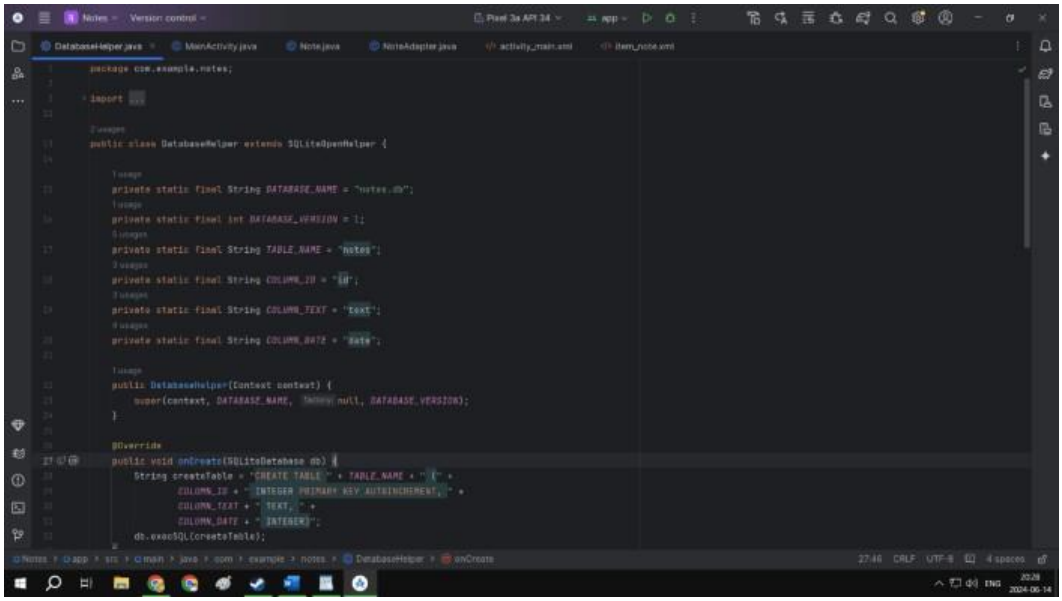

Рисунок 3.3 – Код DatabaseHelper

Видалення нотатку: Користувач може видаляти нотатки. При цьому викликається метод deleteNoteById(), який видаляє нотаток за його ідентифікатором (id).

Отримання всіх нотатків: Метод getAllNotes() повертає список усіх нотатків у базі даних (див. рис. 3.4). Він створює Cursor, який витягує дані з таблиці "notes", і перетворює кожен рядок у об'єкт класу Note.

Відображення нотатків у RecyclerView: У MainActivity, дані з бази даних завантажуються при запуску застосунку, а потім відображаються у списку (RecyclerView). Для цього використовується NoteAdapter, який приймає список нотатків та відображає їх у RecyclerView.

Id	Text	Date
1	Текст	1623309300000
2	Інша...	1623309500000
3	Ще один	1623309700000

Рисунок 3.4– Схема бази даних

Ця таблиця містить інформацію про три нотатки. Кожен нотаток має унікальний id, текст та дату створення.

3.3 Створення інтерфейсу

Щодо інтерфейсу, було використано Android-компоненти користувацького інтерфейсу, такі як EditText для введення тексту нотатки, Button для додавання нового нотатки та видалення, і RecyclerView для відображення списку нотаток у вигляді списку [6].

Також були використані інші компоненти, такі як CheckBox для відзначення нотаток для видалення та TextView для відображення тексту нотаток та їхньої дати створення.

Це дозволило створити ефективний та функціональний інтерфейс для нашого застосунку для зберігання нотаток.

3.3.1 Поле пошуку нотаток

Це поле дозволяє користувачеві шукати нотатки за ключовими словами.

- **EditText**: Використовується для введення тексту пошукового запиту.
- **Attributes**:
  - hint="Search notes": Текст-підказка, який з’являється, коли поле порожнє.
  - inputType="text": Тип введення тексту.
  - layout_width="match_parent": Ширина поля заповнює весь доступний простір.
  - layout_height="wrap_content": Висота поля залежить від його вмісту.

3.3.2 Поле введення нової нотатки

Це поле дозволяє користувачеві вводити текст нової нотатки.

- **EditText**: Використовується для введення тексту нотатки.
- **Attributes**:
  - hint="Enter note": Текст-підказка, який з’являється, коли поле порожнє.
  - inputType="text": Тип введення тексту.
  - layout_width="match_parent": Ширина поля заповнює весь доступний простір.
  - layout_height="wrap_content": Висота поля залежить від його вмісту.

3.3.3 Список нотаток

Список, що відображає всі створені нотатки.

- **RecyclerView**: Використовується для відображення списку нотаток.

- **Attributes**:
  - layout_width="match_parent": Ширина списку заповнює весь доступний простір.
  - layout_height="wrap_content": Висота списку залежить від його вмісту.

RecyclerView використовує адаптер для відображення даних. Кожен елемент списку містить текст нотатки та позначку часу.

- **Adapter**: Адаптер для RecyclerView, який відповідає за відображення кожного елемента списку.
- **ViewHolder**: Клас, що утримує посилання на компоненти інтерфейсу для кожного елемента списку.

3.3.4 Кнопка додавання нотатки

Кнопка, що дозволяє користувачеві додати нову нотатку.

- **Button**: Використовується для виконання дії додавання нотатки.
- **Attributes**:
  - text=”Add Note”: Текст на кнопці.
  - layout_width=”wrap_content”: Ширина кнопки залежить від її вмісту.
  - layout_height=”wrap_content”: Висота кнопки залежить від її вмісту.
  - backgroundTint: Встановлення кольору фону кнопки.

3.3.5 Кнопка видалення вибраних нотаток

Кнопка, що дозволяє користувачеві видалити вибрані нотатки.

- **Button**: Використовується для виконання дії видалення нотаток.

- **Attributes**:
  - text="Delete Selected Notes": Текст на кнопці.
  - layout_width="wrap_content": Ширина кнопки залежить від її вмісту.
  - layout_height="wrap_content": Висота кнопки залежить від її вмісту.
  - backgroundTint: Встановлення кольору фону кнопки.

3.4 Використані технології та інструменти

3.4.1 XML для розмітки інтерфейсу

Для створення користувацького інтерфейсу було використано XML-файли [5], які визначають компоненти інтерфейсу та їх атрибути. Наприклад, у файлі activity_main.xml можуть бути визначені наступні компоненти:

```
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   android:padding="16dp">

   <EditText
      android:id="@+id/searchField"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Search notes"
```

```
        android:inputType="text" />
    <EditText
        android:id="@+id/noteField"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter note"
        android:inputType="text" />
    <Button
        android:id="@+id/addNoteButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Add Note"
        android:backgroundTint="@color/colorPrimary" />
    <Button
        android:id="@+id/deleteNoteButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Delete Selected Notes"
        android:backgroundTint="@color/colorAccent" />
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

3.4.2 Android Jetpack Components

- **ViewModel**: Управляє даними UI у реактивний спосіб, зберігає стан даних при змінах конфігурації.
- **LiveData**: Дозволяє компонентам UI спостерігати за змінами у даних і оновлюватися автоматично.
- **Room**: Бібліотека для роботи з базами даних, яка забезпечує абстракцію над SQLite.

3.4.3 Gradle

Використовується для управління залежностями та збірки проєкту.

3.5 Взаємодія компонентів інтерфейсу

Поле пошуку дозволяє користувачеві вводити текст для пошуку нотаток. У відповідь на зміну тексту викликається метод searchNotes() у ViewModel, який оновлює список нотаток (див. рис. 3.5).

Поле введення нової нотатки дозволяє користувачеві вводити текст нової нотатки.

Кнопка додавання нотатки викликає метод addNote() у ViewModel, який додає нову нотатку до бази даних та оновлює список нотаток.

Кнопка видалення вибраних нотаток викликає метод deleteSelectedNotes() у ViewModel, який видаляє вибрані нотатки з бази даних.

Список нотаток (RecyclerView) автоматично оновлюється при зміні даних у LiveData.

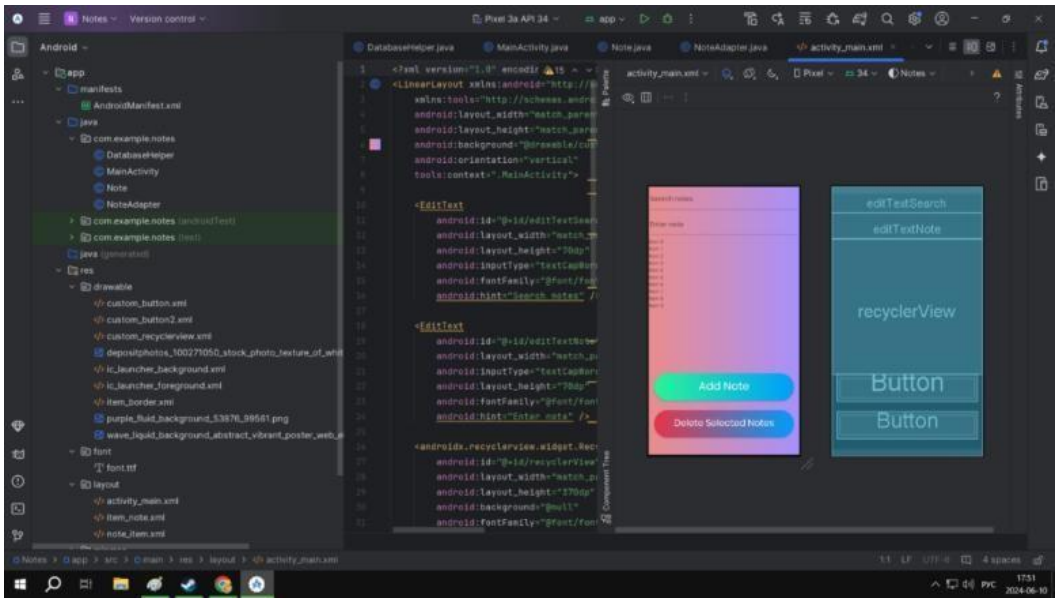

Рисунок 3.5 – Розробка інтерфейсу та вікно Palette

ВИСНОВОК

Під час розробки цього додатка для управління нотатками були реалізовані основні функції, які дозволяють користувачам ефективно створювати, редагувати, видаляти та шукати нотатки. Програма була створена з використанням новітніх технологій, таких як Android Studio і мова програмування Java, що забезпечує високу продуктивність і простоту використання.

Основним сховищем даних був SQLite, що дозволило мені організувати зберігання нотаток без необхідності підключення до зовнішнього сервера.

Рішення зробило додаток автономним і незалежним від підключення до Інтернету. Це величезна перевага для користувачів.

Інтерфейс застосунку розроблений з акцентом на простоту і зручність використання. Усі елементи керування, такі як поля введення тексту, кнопки та список нотаток, були ретельно доопрацьовані, щоб забезпечити інтуїтивно зрозумілий інтерфейс користувача. RecyclerView дозволяє ефективно переглядати нотатки у вигляді списку, який можна легко оновлювати та модифікувати.

Пошук нотаток за текстом був реалізований за допомогою TextWatcher, який дозволяє динамічно фільтрувати нотатки за текстом, введеним Користувачем. Це зробило процес пошуку швидким і зручним.

В цілому, розроблене додаток є надійним інструментом для збереження заміток і управління ними. Воно відповідає всім вимогам, що пред'являються на початку розробки, і демонструє високу функціональність і простоту використання. Впровадження цього додатка може значно спростити організацію особистих нотаток для користувачів і забезпечити зручний доступ до інформації в будь-який час.

Я ціную можливість працювати над цим проектом і впевнений, що отримані знання та досвід допоможуть моїй майбутній професійній діяльності.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Kotlin vs Java: Breaking Down the Differences for Business Leaders. Url: https://jaydevs.com/kotlin-vs-java/ (дата звернення: 21.04.2024).

2. Ranking of the most popular relational database management systems worldwide, as of September 2023. URL: https://www.statista.com/statistics/1131568/worldwide-popularity-ranking-relational-database-management-systems/ (дата звернення: 21.04.2024).

3. Kotlin vs. Java For Android Development Ultimate Comparison. URL: https://lampa.dev/blog/kotlin-vs-java-for-android-development-ultimate-comparison (дата звернення: 20.04.2024).

4. Java Tutorial. URL: https://www.geeksforgeeks.org/java/ (дата звернення: 21.04.2024).

5. Developer guides. URL: https://developer.android.com/guide (дата звернення: 21.04.2024).

6. The ultimate Android development guide: 50+ beginner and expert resources. URL: https://techbeacon.com/app-dev-testing/ultimate-android-development-guide-50-beginner-expert-resources (дата звернення: 21.04.2024).

7. Основи Android studio. URL: https://w3schoolsua.github.io/hyperskill/android-studio-basics.html#gsc.tab=0 (дата звернення: 21.04.2024).

8. Розробка на Java — з чого почати знайомство з мовою URL: https://dou.ua/lenta/articles/how-to-learn-java/ (дата звернення: 21.04.2024).

Декларація

академічної доброчесності

здобувача освіти ВСП «Економіко-правничого фахового коледжу ЗНУ»

Я, Сосновський Михайло Валерійович, здобувач освіти 4 курсу, спеціальності/освітньо-професійної програми інженерія програмного забезпечення, групи К 121-20, адреса електронної пошти vermihelion@gmail.com

- підтверджую, що написана мною дипломна робота на тему «Проєктування та розробка застосунку для керування нотатками» відповідає вимогам академічної доброчесності та не містить порушень, що визначені у ст. 42 Закону України «Про освіту», зі змістом яких ознайомлений/ознайомлена;

- заявляю, що надана мною для перевірки електронна версія роботи є ідентичною її друкованій версії;

- згоден/згодна на перевірку моєї роботи на відповідність критеріям академічної доброчесності у будь-який спосіб, у тому числі за допомогою інтернет-системи, а також на архівування моєї роботи в базі даних цієї системи.

Дата _____Підпис _____ Михайло СОСНОВСЬКИЙ

Дата _____Підпис _____ Юлія ЛИМАРЕНКО