МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ANDROID ЗАСТОСУНКУ В ОБЛАСТІ ФАРМАЦЕВТИКИ»**

Виконав: студент 4 курсу, групи 6.1219-1пі
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
О.В. Томін
(ініціали та прізвище)
Керівник доцент кафедри програмної інженерії, доцент, к.т.н. Лимаренко Ю.О.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент директор ТОВ «Голден-Тім» Калін М.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2023

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)

“ 07 ” 02 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Томіну Олексію Віталійовичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка Android застосунку в області фармацевтики

керівник роботи Лимаренко Юлія Олексіївна, к.т.н., доцент
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 26 » січня 2023 року № 102-с

2. Строк подання студентом роботи 07.06.2023

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. Розробка Android застосунку в області фармацевтики.

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 07.02.2023 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.	08.02.2023	
2.	Збір вихідних даних.	15.02.2023	
3.	Обробка методичних та теоретичних	22.02.2023	
	джерел.		
4.	Розробка першого та другого розділу.	05.04.2023	
5.	Розробка третього розділу.	10.05.2023	
6.	Оформлення та нормоконтроль	01.06.2023	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	21.06.2023	

Студент _____ О.В. Томін
(підпис) (ініціали та прізвище)

Керівник роботи _____ Ю.О. Лимаренко
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

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

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка Android застосунку в області фармацевтики»: 52 с., 19 рис., 12 джерел.

АНДРОІД, АРХІТЕКТУРА, ЗАСТОСУНОК, РОЗРОБКА, ФАРМАЦЕВТИКА, ФРЕЙМВОРК.

Об’єкт дослідження – розробка Android застосунку в області фармацевтики.

Мета роботи: розробити та протестувати Android застосунок в області фармацевтики.

Метод дослідження – теоретичні: теоретико-методологічний аналіз; емпіричні: спостереження, тестування, аналіз програмного забезпечення, проєктування.

Наукова новизна полягає у розробці та впровадженні Android застосунку, який буде простим у використанні для користувачів, гнучкою до використання у різних сферах ІТ виробництва, масштабованою, здатною до збільшення користувачів.

Галузь використання: медицина, фармацевтика.

SUMMARY

Bachelor’s qualifying paper «Development of a Pharmacy Android Application»: 52 pages, 19 figures, 12 references.

ANDROID, ARCHITECTURE, APPLICATION, DEVELOPMENT, PHARMACY, FRAMEWORK.

Object of the study is development of an Android application in the field of pharmacy.

Aim of the study: develop and test an Android application in the field of pharmacy.

Methods of research are theoretical and methodological analysis; empirical: observation, testing, software analysis, design.

The scientific novelty lies in the development and implementation of an android application that will be easy to use for users, flexible - able to be used in various areas of IT production, scalable – able to increase users.

Field of application: medicine, pharmacy.

ЗМІСТ

Завдання на кваліфікаційну роботу……...…2
Реферат……..…4
Summary……..…..5
Вступ..... 8
1 Аналіз предметної області..... 10
1.1 Опис предметного середовища..... 10
1.2 Опис можливих інструментів розробки 12
1.3 Опис основних етапів створення мобільного додатку..... 15
1.4 Огляд програмних засобів, що вирішують подібну задачу..... 16
2 Проєктування Android застосунку..... 20
2.1 Технічне завдання 20
2.1.1 Найменування та область застосування 20
2.1.2 Підстава для розробки..... 20
2.1.3 Призначення розробки 20
2.1.4 Вимоги до функціональних характеристик 21
2.1.5 Вимоги до надійності 22
2.1.6 Вимоги до складу і параметрів технічних засобів та програмної сумісності 23
2.2 Проєктування інтерфейсу..... 23
3 Розробка Android застосунку 29
3.1 Характеристика потенційної аудиторії проєкту 29
3.2 Вибір засобів розробки..... 29
3.3 Розробка програмного застосунку 32
3.3.1 Ознайомлення і підготовка середовища розробки..... 32
3.3.2 Розробка основних екранів 35
3.3.3 Розробка додаткових екранів..... 42
Висновки 49

Перелік посилань..... 51

ВСТУП

В сучасному світі смартфони та планшети займають важливе місце в повсякденному житті людей. Вони стали невід'ємною частиною комунікації, інформаційного пошуку та особистого управління. Отже, розробка Android застосунку в області фармацевтики має велике значення, оскільки це дозволить забезпечити зручний та швидкий доступ до фармацевтичних послуг для користувачів пристроїв на базі Android.

Такий застосунок може містити функції, такі як пошук ліків та інформації про них, дозування та можливість поділитися цією інформацією з іншими користувачами. Також застосунок буде мати можливість порахувати дозування для ліків.

Нажаль альтернатив подібним застосункам майже нема, а існуючі інструменти для доступу до інформації про продукти фармацевтики та керування прийомом ліків не завжди забезпечують повну функціональність або зручний інтерфейс користувача.

Саме тому розробка такого застосунку дозволить покращити доступність інформації про продукти фармацевтики, зекономити час та зусилля лікарів та їх пацієнтів, а також забезпечити більш точне та зручне управління прийомом ліків.

Предмет дослідження – процес та фактори розробки Android застосунку.

Мета: розробити та протестувати Android застосунок в області фармацевтики.

Завдання дослідження:

1) проаналізувати існуючі Android застосунки в області фармацевтики;

2) спроєктувати Android застосунок;

3) виконати вимоги до системи: повинна мати основні функції для перегляду фармацевтичних виробів та інформації про них, пошук фармацевтичних виробів, для деяких виробів повинен бути доступний

калькулятор дозування, можливість поділитися інформацією про фармацевтичний продукт або його дозування з калькулятора;

4) реалізувати Android застосунок з використанням Dart та Flutter.

Методи дослідження: теоретико-методологічний аналіз, проєктування, моделювання, аналіз програмного забезпечення.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

Мобільний застосунок – це програмне забезпечення, розроблене для використання на мобільних пристроях, таких як смартфони і планшети. Він призначений для виконання різних функцій, від ігор і розваг до соціальних мереж, електронної комерції, банківських операцій, подорожей та багато іншого. Мобільні застосунки можуть бути встановлені через офіційні магазини додатків, такі як App Store для iOS та Google Play Store для Android, або у випадку з Android, додатково можуть встановлюватись безпосередньо з веб-сайтів чи файлової системи Android [1].

Щоб отримати позитивне враження від використання мобільного застосунку, важливо врахувати кілька ключових факторів. Розглянемо їх нижче.

Інтуїтивний і привабливий інтерфейс: застосунок повинен мати зрозумілий і зручний інтерфейс, який дозволяє користувачеві легко орієнтуватися та взаємодіяти з функціями застосунку. Графічний дизайн повинен бути привабливим і сприяти позитивному сприйняттю.

Функціональність і продуктивність: застосунок повинен виконувати свої функції ефективно і швидко. Користувачі цінують мобільні застосунки, які забезпечують надійну та швидку роботу, без затримок або перебоїв. Оптимізація продуктивності та мінімальне споживання ресурсів пристрою також є важливими аспектами.

Оновлення і підтримка: регулярні оновлення застосунку з виправленням помилок, додаванням нових функцій і покращенням загальної роботи є важливими для створення позитивного враження. Користувачі хочуть бачити, що розробники активно працюють над вдосконаленням свого продукту і слухають зворотний зв'язок користувачів.

Безпека та конфіденційність: застосунок повинен забезпечувати надійний рівень безпеки даних користувача і забезпечувати конфіденційність інформації. Важливо використовувати шифрування та інші заходи безпеки для захисту приватності користувача та запобігання зловживанням.

Взаємодія і соціальні можливості: багато користувачів цінують мобільні застосунки, які надають можливості взаємодії з іншими користувачами, такі як соціальні мережі, чати або спільне виконання завдань. Це дозволяє створити спільноту користувачів і забезпечує більш залучений досвід використання застосунку.

Підтримка різних платформ і пристроїв: щоб забезпечити доступність для широкої аудиторії, мобільний застосунок повинен підтримувати різні платформи, такі як iOS та Android, а також різні пристрої з різною розміром екрану та характеристиками.

Зручність використання та персоналізація: користувачі хочуть мати можливість налаштувати застосунок під свої потреби і вимоги. Важливо забезпечити зручність використання та персоналізацію інтерфейсу, які дозволяють користувачам налаштовувати налаштування, пріоритети та вигляд застосунку.

В цілому, успіх мобільного застосунку залежить від поєднання зазначених факторів, а також від спрямованості на потреби і очікування користувачів. Якщо застосунок забезпечує зручність використання, високу функціональність, безпеку та підтримку, а також створює позитивний і сприятливий досвід для користувачів, ймовірність отримання позитивного враження від його використання значно зростає [2].

Порівнюючи мобільні застосунки з усіма вже звичними веб-сайтами, можна виділити низку переваг. Основні з них наведено нижче.

Зручність використання: мобільні застосунки спеціально розроблені для мобільних пристроїв і їхніх екранів. Інтерфейс та взаємодія з мобільними застосунками оптимізовані під розмір екрану та функціональність пристрою,

що робить їх більш зручними для використання на маленьких екранах смартфонів або планшетів.

Офлайн-режим: однією з важливих переваг мобільних застосунків є можливість працювати в офлайн-режимі. Користувачі можуть отримувати доступ до певних функцій і виконувати дії, навіть якщо вони не підключені до Інтернету. Це особливо корисно в ситуаціях з обмеженим або нестабільним з'єднанням з Інтернетом.

Краща продуктивність: мобільні застосунки часто мають кращу продуктивність, оскільки вони оптимізовані для конкретних операційних систем і пристроїв. Ізольований середовище мобільного застосунку дозволяє використовувати ресурси пристрою більш ефективно, що призводить до швидшої роботи і кращої відзивчивості.

Доступність: швидкість доступу до мобільних застосунків значно вища, ніж до веб-сайтів, завдяки тому, що у більшості людей смартфон завжди під рукою. Застосунок в свою чергу більшу частину графіки, розмітки та ін. частин сторінки завантажує з мережі Інтернет лише один раз, а потім за потреби завантажується з пристрою.

Використання особливостей пристроїв: деякі мобільні застосунки можуть використовувати датчики руху, віртуальну або доповнену реальність, що додає ще більше можливостей і інтерактивності до застосунків.

Хоча мобільні застосунки мають свої переваги, веб-сайти також залишаються важливими інструментами. Вони мають свої сильні сторони, такі як ширше охоплення різних платформ, більша доступність без вимог до встановлення, простота оновлення та крос-платформова сумісність [3].

1.2 Опис можливих інструментів розробки

Існує декілька шляхів для розробки мобільних застосунків, але основних усього два: крос-платформові та нативні.

Крос-платформові – такими застосунками називають програмне забезпечення, яке розроблюється з прицілом на декілька операційних систем, при чому не тільки мобільних. Робиться це задля здешевлення розробки, адже утримуючи одну команду розробників ви отримуєте змогу розробляти застосунок під Android, iOS, Windows або Linux та WEB одночасно, якщо упустити деякі нюанси.

Нативні застосунки – такими застосунками називають програмне забезпечення, яке розроблюється для конкретної платформи або операційної системи. Цей варіант використовують коли проблем з додатковим фінансуванням немає і треба забезпечити максимальну продуктивність і відповідність дизайну операційної системи [2].

Якщо йти по першому шляху, то розробляти застосунок під усі платформи одразу не обов’язково, це завжди можна буде зробити пізніше, але чим масштабніше стане застосунок, тим складніше потім буде його адаптувати під інші платформи.

Також є так звані «конструктори» мобільних застосунків, але їх існуванням ми нехтуємо, адже їх не можна розглядати як серйозну альтернативу першим двом варіантам.

Android Studio – інтегроване середовище розробки (IDE) від компанії Google, яке побудовано на базі вихідного коду іншого середовища розробки під назвою IntelliJ IDEA, яке в свою чергу розроблялося компанією JetBrains. Android Studio є потужним інструментом для розробки мобільних додатків на платформі Android (див. рис. 1.1). Він допомагає розробникам створювати високоякісні додатки з високою продуктивністю, функціональністю та сумісністю з різними пристроями Android.

У той же час, це програмне забезпечення можна використовувати і для розробки крос-платформових застосунків за допомогою Flutter SDK та мови програмування Dart. Саме цей варіант був обраним для розробки дипломного проєкту [3].

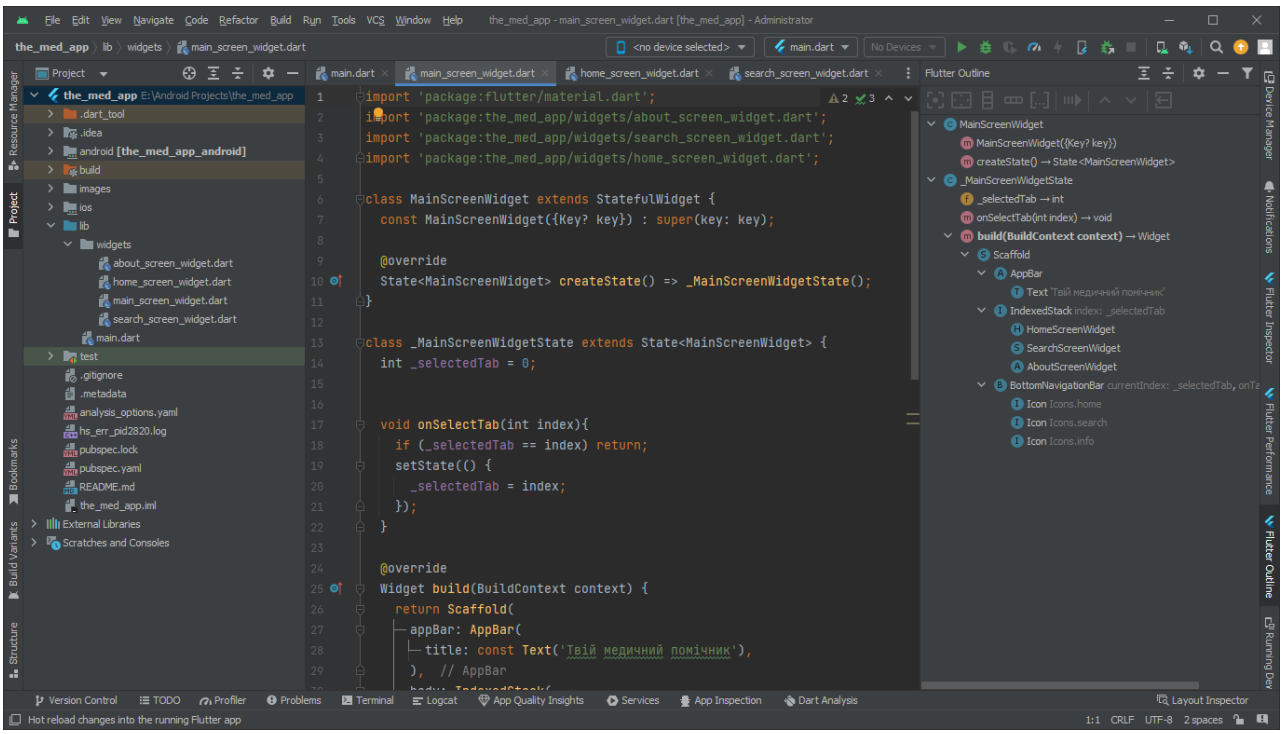

Рисунок 1.1 – Інтерфейс середовища розробки Android Studio

Visual Studio Code – безкоштовний редактор вихідного коду, який був створений компанією Microsoft у 2015 році (див. рис. 1.2).

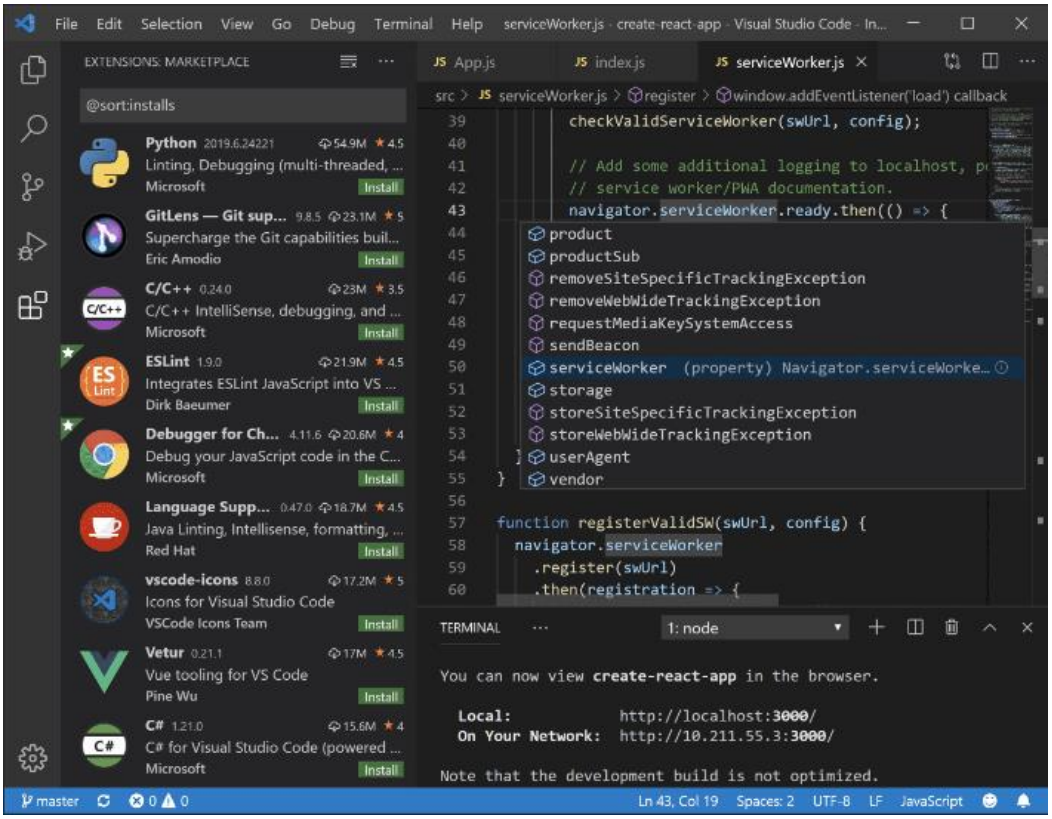

Рисунок 1.2 – Інтерфейс Visual Studio Code

На відміну від минулого середовища розробки, це програмне забезпечення призначене для розробки різноманітних типів програмного забезпечення і підтримує різні мови програмування, такі як: C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python, Rust та багато інших.

Також функціонал Visual Studio Code розширюється за допомогою різного роду розширень, які можуть як додати підтримку нової мови програмування, так і пришвидшити або спростити розробку для вже існуючих мов. Більшість розробників, які вже мали досвід у програмуванні, скоріш за все вже використовували VS Code, тому перехід з іншої мови програмування не повинно викликати у розробника ніяких проблем.

1.3 Опис основних етапів створення мобільного додатку

Створення мобільного застосунку включає кілька основних етапів, які можна узагальнити в декілька пунктів.

Планування і аналіз:

- визначення цілей та області застосунку: розуміння того, що саме ви хочете досягти з вашим мобільним застосунком та як він вирішуватиме проблеми або задовольнятиме потреби користувачів;
- дослідження ринку та аудиторії: вивчення ринкових умов, конкуренції та цільової аудиторії для визначення потреб і вимог користувачів;
- створення концепції: розроблення загального опису функціональності та інтерфейсу застосунку.

Проєктування і інтерфейс:

- створення інформаційної архітектури: організація контенту та інформаційних потоків в застосунку;
- створення скетчів та дизайну: визначення вигляду і взаємодії користувача з допомогою скетчів, прототипів або мокапів;
- розроблення інтерфейсу користувача (UI): реалізація дизайну інтерфейсу

з використанням інструментів для графічного дизайну або засобів розробки інтерфейсу.

Розробка:

– вибір платформи та технологій: визначення платформи (Android, iOS, крос-платформовий) та вибір технологій розробки (Java, Kotlin, Swift, React Native, Flutter тощо);
– інтеграція зовнішніх сервісів: підключення до сторонніх сервісів, таких як мапи, соціальні мережі, платіжні системи тощо;
– тестування та усунення помилок: виконання тестів для перевірки функціональності, стабільності та безпеки застосунку, виправлення помилок і вдосконалення продукту.

Розгортання і підтримка:

– розгортання на мобільних платформах: підготовка застосунку для публікації в магазинах додатків (Google Play, App Store);
– моніторинг та оптимізація: відстеження продуктивності та аналіз поведінки користувачів для покращення функціональності і забезпечення задоволення користувачів;
– підтримка та оновлення: надання підтримки користувачам, виправлення помилок і випуск нових версій застосунку з оновленнями та новими функціями.

Важливо також відзначити, що кожен етап може мати свої різноманітні підетапи та вимоги, і сам процес створення може досить значно варіюватись залежно від розміру проєкту, команди розробників і обраних технологій [4].

1.4 Огляд програмних засобів, що вирішують подібну задачу

За результатами пошуку альтернативних рішень вдалося знайти лише один застосунок, який виконує подібну до однієї з основних задач, що виконує застосунок дипломного проєкту. Інші функції, які не є основними, також

присутні в інших програмних засобах, але ними можна нехтувати через їх малу значущість. Назва цього застосунку «Дозування ліків».

«Дозування ліків» – це застосунок від розробника VNLab, який допоможе розрахувати необхідну дозу жарознижуючих, сиропів (суспензій) і таблеток, в залежності від ваги дитини або дорослого і кількості діючої речовини в ліках (див. рис. 1.3).

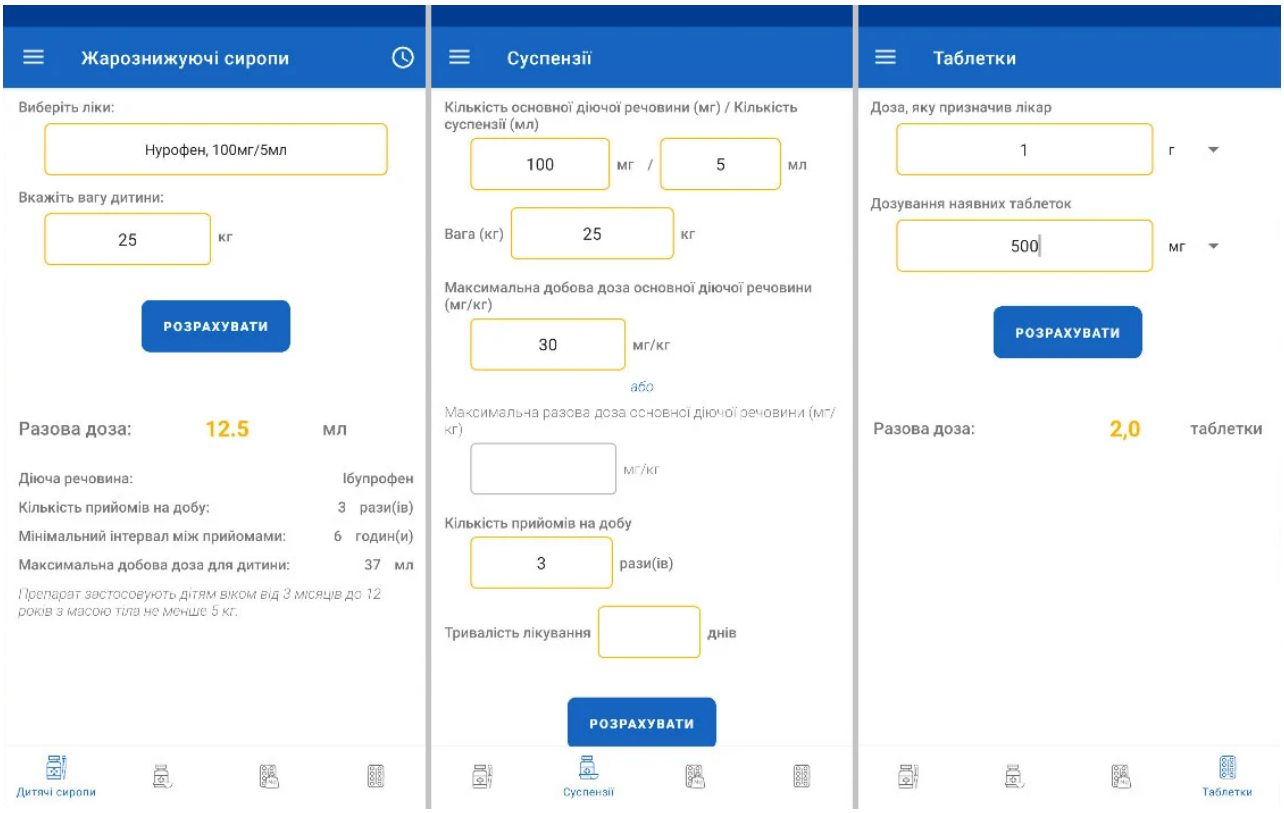

Рисунок 1.3 – Інтерфейс застосунку «Дозування ліків»

Додаток містить 5 розділів:

1) зберегти час прийому ліків – дозволяє зберігати час прийому ліків, і таймер покаже час, що минув після прийому останньої дози;
2) жарознижуючі дитячі сиропи – для розрахунку потрібної дози треба вибрати потрібні ліки зі списку і вказати вагу дитини;
3) дозування суспензій (сиропів) – тут можна розрахувати необхідну дозу будь-якого лікарського сиропу – всі дані для розрахунку вказуються в інструкції до препарату;

4) дозування таблеток в залежності від маси тіла – розрахунок необхідної кількості таблеток в залежності від ваги дитини або дорослого;
5) дозування таблеток – розрахунок необхідної кількості таблеток, враховуючи призначену дозу та дозу, що є в наявності.

Нажаль, застосунок збирає велику кількість аналітичних даних, які потім скоріш за все використовуються для таргетованої реклами або інших цілей, пов’язаних з рекламою. Проте розробник чесно вказує, які дані збираються і передаються третім сторонам.

Хоч ці дані і передаються в зашифрованому вигляді, але розробник не надає можливості надіслати запит на видалення даних, які були накопичені за час використання застосунку.

Застосунок має переважно гарні відгуки, має середню оцінку 4.8 на Google Play Store, останнє оновлення було ще 19 листопада 2022 року, що говорить або про те, що розробник досить слабо підтримує застосунок, або застосунок майже не потребує оновлень.

Нажаль, застосунок присутній лише на платформі Android, тобто користувачам iOS немає можливості скористатися цим застосунком хоча б у WEB і це безперечно мінус, який зможе закрити цей дипломний проєкт.

У даному розділі був проведений аналіз предметної області – фармацевтики та мобільних додатків. В ході аналізу було виявлено, що мобільні додатки мають значний потенціал для поліпшення доступності та ефективності фармацевтичних послуг. Вони можуть забезпечити швидкий доступ до інформації про ліки, допомагати в зручному прийомі ліків і підбору дозування. Такі додатки можуть бути особливо корисними для людей, які постійно стикаються з подібними задачами і завдяки дипломному проєкту зможуть прискорити свою роботу, а для пацієнтів спростити комунікацію між лікарем та пацієнтом.

В процесі аналізу були також виявлені деякі особливості та вимоги, які потрібно враховувати при розробці мобільного додатку в області фармацевтики. Зокрема, важливо забезпечити зручний та інтуїтивно зрозумілий

інтерфейс користувача, забезпечити безпеку та конфіденційність даних, а також враховувати регуляторні та правові вимоги, пов'язані з фармацевтичною галуззю.

Отже, аналіз предметної області підтвердив актуальність розробки мобільного додатку в області фармацевтики. Такий додаток може принести значну користь користувачам, полегшуючи їх доступ до фармацевтичних послуг та допомагаючи в підтримці оптимального здоров’я і добробуту.

2 ПРОЄКТУВАННЯ ANDROID ЗАСТОСУНКУ

2.1 Технічне завдання

Дане технічне завдання поширюється на розробку програмного мобільного Android застосунку в області фармацевтики.

2.1.1 Найменування та область застосування

Найменування програмного продукту: “LikApp”.
Область застосування: медичний довідник, пошуковик, калькулятор дозувань для пацієнтів і лікарів.

2.1.2 Підстава для розробки

Застосунок розробляється на підставі робочої програми по курсу «Дипломна робота», затвердженої наказом від 26 січня 2023р. №102-с.

2.1.3 Призначення розробки

Дана програма призначена для вирішення наступних завдань:

- зручний перегляд списку фармацевтичних виробів;
- перегляд опису для кожного з фармацевтичних виробів;
- перегляд інструкцій для кожного з фармацевтичних виробів;
- функція калькулятору дозувань для ліків;
- функція «поділитись з пацієнтом» для поширення інформації стосовно опису, інструкції або дозувань;

2.1.4 Вимоги до функціональних характеристик

Застосунок повинен забезпечувати можливість виконання наступних функцій:

1) введення початкових даних: користувач вводить данні до пошуку за найменуванням ліків; в калькуляторі дозувань користувач вводить вагу особи, для якої підбирається дозування;

2) обробка:

- перегляд списку доступних ліків;
- пошук ліків за найменуванням;
- перегляд опису для кожного з доступних найменувань;
- перегляд інструкції для кожного з доступних найменувань;
- калькулятор дозувань згідно з вагою особи, для якої підбирається дозування;
- поділитися переглянутою інформацією;

3) виведення результату:

- перелік результатів пошуку відповідно до заданого найменування;
- дозування для обраного найменування;

Організація вхідних і вихідних даних.

Вхідні дані – файл бази даних, що містить: інформацію про фармацевтичні вироби, їх детальну інформацію, інструкцію, дозування.

Вихідні дані:

- список всіх фармацевтичних виробів;
- результати пошуку;
- результати калькулятору дозування;

Діаграма прецедентів – це діаграма, на якій зображено відношення між акторами та прецедентами в системі. Також, перекладається як діаграма варіантів використання (див. рис. 2.1).

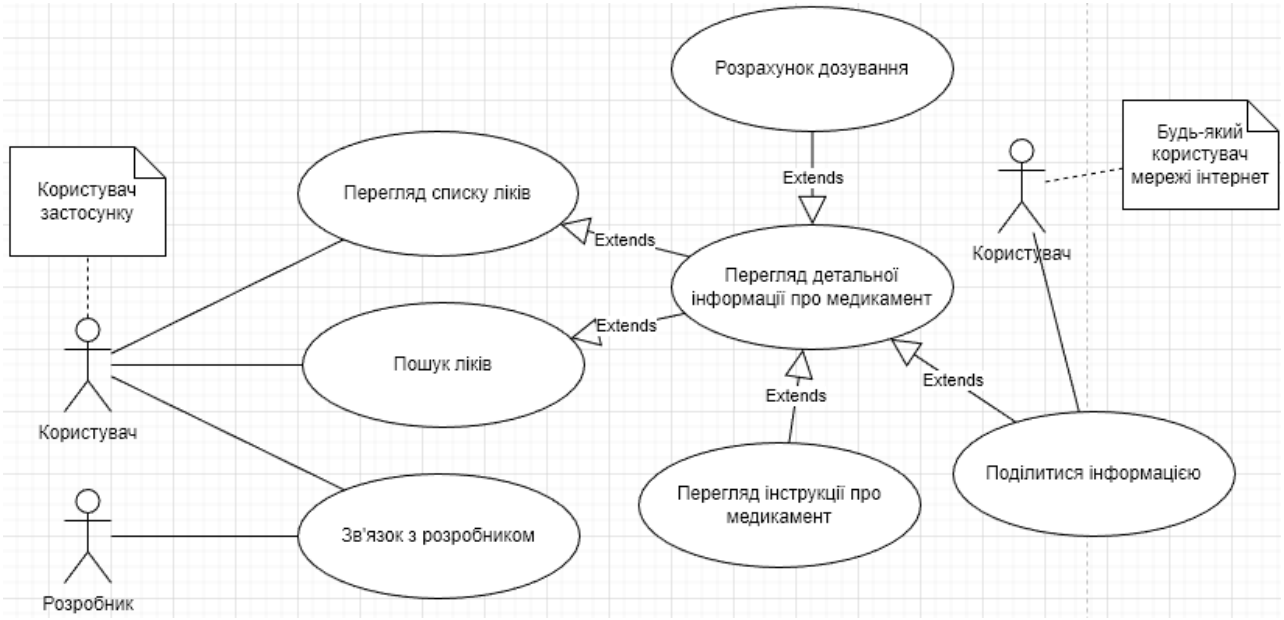

Рисунок 2.1 – Діаграма прецедентів

Діаграми прецедентів складають модель прецедентів (варіантів використання, use-cases). Прецедент – це функціональність системи, що дозволяє користувачеві отримати якийсь значущий для нього, відчутний і вимірний результат. Кожен прецедент відповідає окремому сервісу, що надається моделюючою системою у відповідь на запит користувача. Визначає спосіб використання цієї системи. Варіанти використання найчастіше застосовуються для специфікації зовнішніх вимог до проєктованої системи або для специфікації функціонального поведінки вже існуючої системи. Крім цього, варіанти використання неявно описують типові способи взаємодії користувача з системою, що дозволяють коректно працювати з наданими системою сервісами.

2.1.5 Вимоги до надійності

Передбачити контроль інформації, що вводиться, і блокування некоректних дій користувача при роботі з файлами.

Забезпечити цілісність інформації, що зберігається в базі даних.

2.1.6 Вимоги до складу і параметрів технічних засобів та програмної сумісності

Дана розробка є мобільним додатком та вимагає наступну мінімальну конфігурацію апаратного та програмного забезпечення: версія ОС Android: 7.0, оперативна пам’ять 1024 мб, 512 мб вільного простору на накопичувачі.

2.2 Проєктування інтерфейсу

Проєктування інтерфейсу мобільних застосунків – це процес створення користувацького інтерфейсу (UI) для мобільних додатків, який включає в себе планування, дизайн та розробку елементів інтерфейсу з урахуванням потреб користувачів і найкращих практик дизайну [10].

Під час проєктування інтерфейсу мобільних застосунків використовуються різні інструменти, зокрема графічні редактори. Одним з таких є Adobe Photoshop (див. рис. 2.2). Photoshop є одним з найпопулярніших графічних редакторів, що використовуються для створення візуальних елементів дизайну, таких як іконки, кнопки, фони тощо. Він надає багатий набір інструментів для маніпулювання зображеннями та кольорами, а також надає широкий спектр інструментів для редагування та маніпулювання зображеннями, що дозволяє створювати різні візуальні елементи для інтерфейсу. Програмне рішення від компанії Adobe дозволяє також використовувати шари, маски, фільтри та інші функції, що дозволяють детально працювати з дизайном інтерфейсу та вносити зміни за потребою. У випадку крупних або корпоративних проєктів, Photoshop як частина Adobe Creative Cloud, легко інтегрується з іншими програмами, такими як Illustrator або After Effects, що полегшує спільну роботу над проєктом та обмін даними.

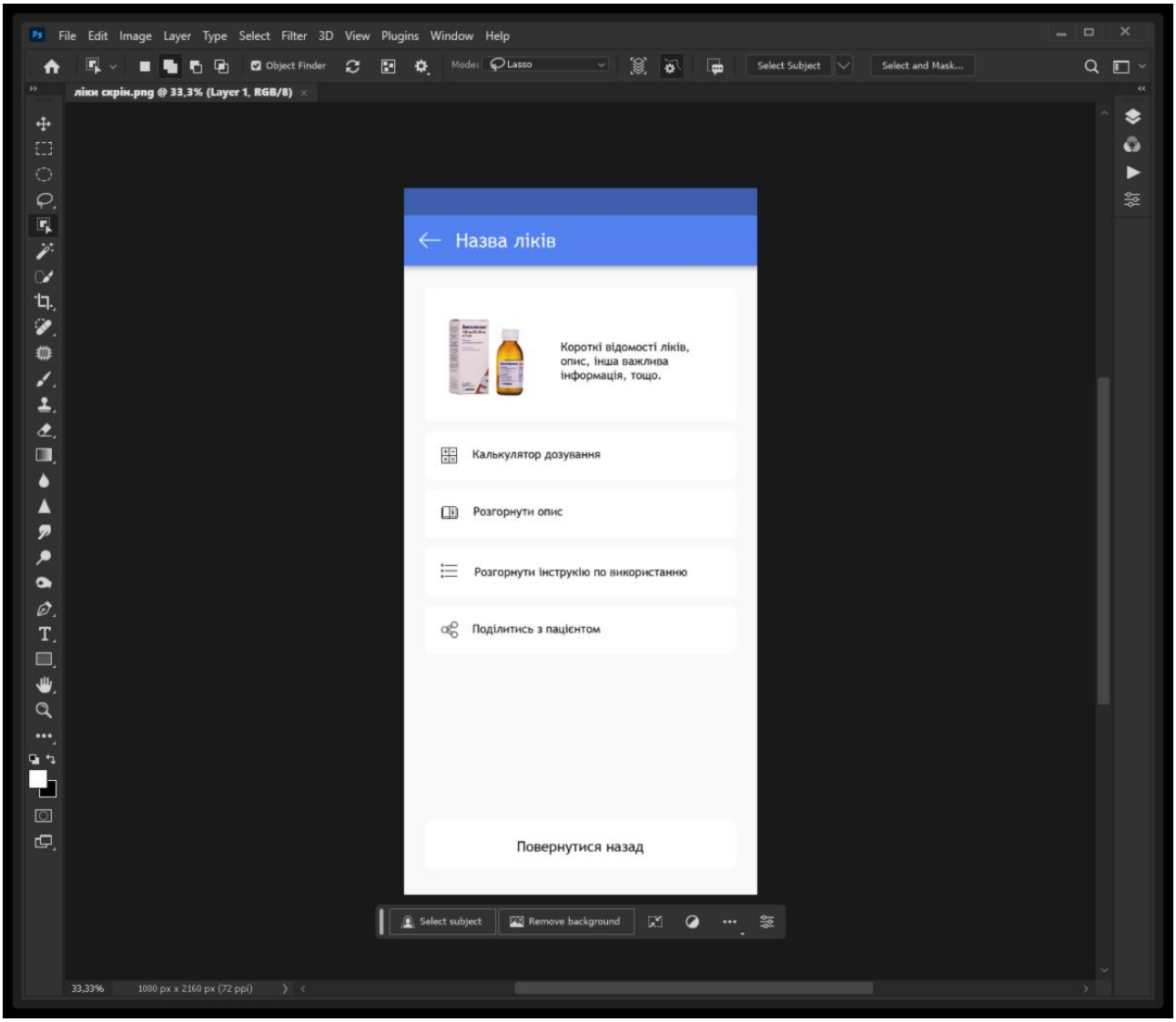

Рисунок 2.2 – Інтерфейс Adobe Photoshop с макетом одного з екранів розроблюваного застосунку

Але як і у будь-якої іншої програми, Adobe Photoshop має ряд своїх мінусів. Розглянемо їх детальніше.

Векторна обробка: у зв’язку з тим, що Photoshop базується на растровій графіці, це означає, що елементи інтерфейсу створюються в растровому форматі. Це може створити проблеми, коли потрібно збільшувати розмір елементів для різних пристроїв з різними роздільними здатностями.

Важкість для змін: у порівнянні з іншими інструментами, такими як Sketch або Figma, редагування дизайну в Photoshop може бути більш часо- та ресурсоємним процесом.

Відсутність прототипування: Photoshop не має вбудованих функцій для створення інтерактивних прототипів, тому для цього можуть знадобитись інші інструменти.

Ціна: нажаль, інструмент від Adobe хоч і має знижки для студентів, та все ж таки підтримує досить високі ціни.

Також досить гарною альтернативою програмному застосунку від Adobe є редактор Sketch від Bohemian Coding.

Sketch – це векторний графічний редактор, спеціально розроблений для дизайну інтерфейсів (див. рис. 2.3). Він є дуже популярним серед дизайнерів мобільних додатків завдяки своїй простоті використання, підтримці символів та можливостям швидкого створення компонентів.

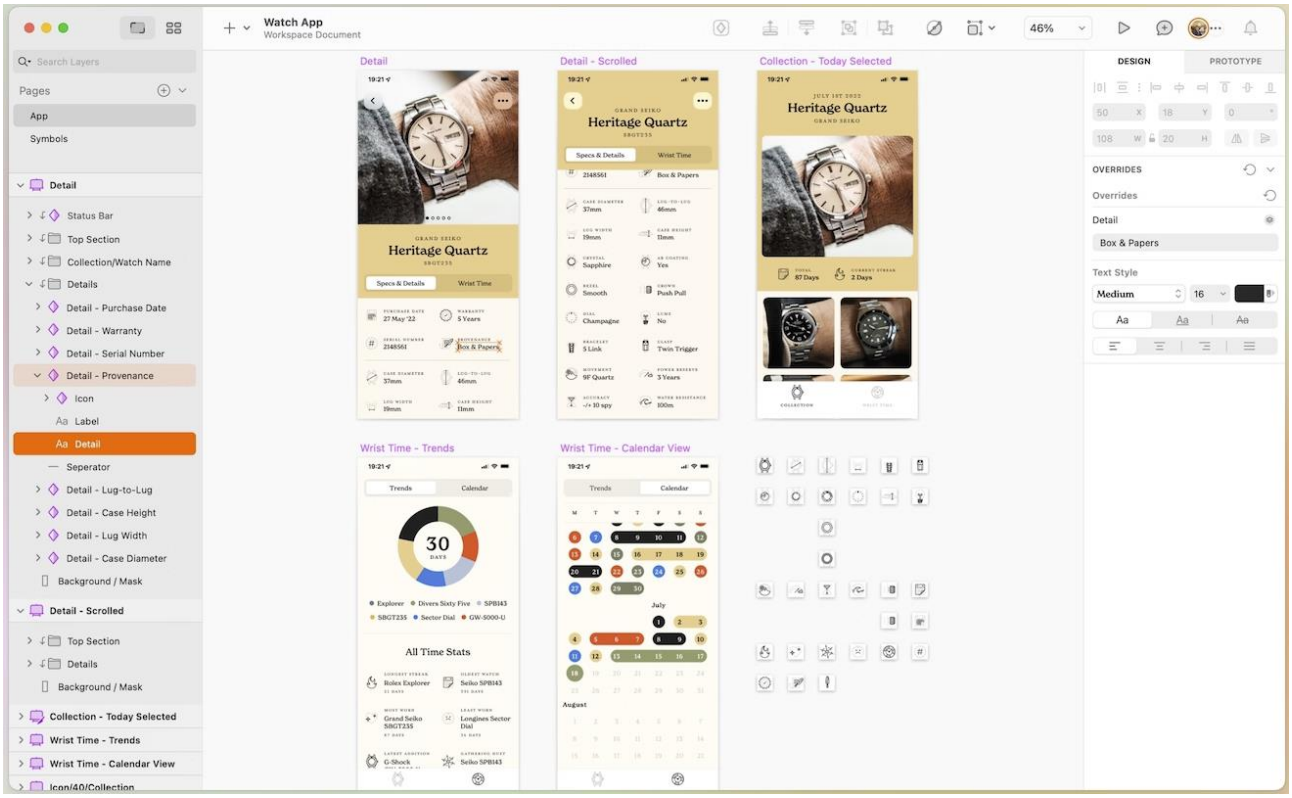

Рисунок 2.3 – Інтерфейс Sketch

Sketch спеціально розроблений для проєктування інтерфейсу, що робить його потужним інструментом для розробників мобільних додатків. Він надає широкий набір функцій і інструментів, спрямованих на спрощення процесу

проєктування. Також він базується на векторній графіці, що дозволяє створювати масштабовані елементи інтерфейсу, які легко адаптуються до різних роздільних здатностей екранів. Sketch має потужні функції символів та стилів, що дозволяють легко створювати і змінювати компоненти інтерфейсу. Це сприяє швидкому та послідовному розробленню дизайну. І звісно ж Sketch підтримує велику кількість сторонніх плагінів, які розширюють його функціональність і дозволяють автоматизувати деякі рутинні завдання, покращуючи продуктивність.

Навіть не зважаючи на достатню кількість позитивних сторін, все ж такі серйозні мінуси обійти стороною не вдасться, а саме:

- обмежена платформова підтримка: Sketch працює тільки на macOS, що обмежує його використання тими, хто працює на інших операційних системах, таких як Windows або Linux;
- залежність від сторонніх інтеграцій: хоча Sketch має багато плагінів, в деяких випадках користувачеві можуть знадобитись додаткові інтеграції з іншими інструментами для повного циклу розробки та спільної роботи над проєктом;
- відсутність вбудованих функцій прототипування: Sketch не має вбудованих інструментів для створення інтерактивних прототипів, тому для цього вам можуть знадобитись додаткові інструменти, які можна інтегрувати з Sketch;
- вартість: Sketch є комерційним програмним забезпеченням, тому використання його може потребувати придбання ліцензії. Для студентів це може бути досить значним фінансовим обмеженням.

Усупереч цим недолікам, Sketch є популярним інструментом серед дизайнерів мобільних додатків завдяки його спеціалізації на інтерфейсному дизайні та багатьом корисним функціям, які прискорюють процес розробки та забезпечують високу якість дизайну.

Останнім з доступних варіантів є такий досить потужний інструмент під назвою Figma від двох розробників Ділана Філда та Евана Воллеса.

Figma – це хмарний графічний редактор, який дозволяє створювати та спільно працювати над дизайном мобільних інтерфейсів (див. рис. 2.4). Він має вбудовані функції для прототипування та зручність спільної роботи в реальному часі.

Здатен працювати у двох форматах: у браузері користувача та як клієнтський застосунок на комп'ютері користувача. Додатково зберігає всі файли онлайн, з якими коли-небудь доводилося працювати користувачеві.

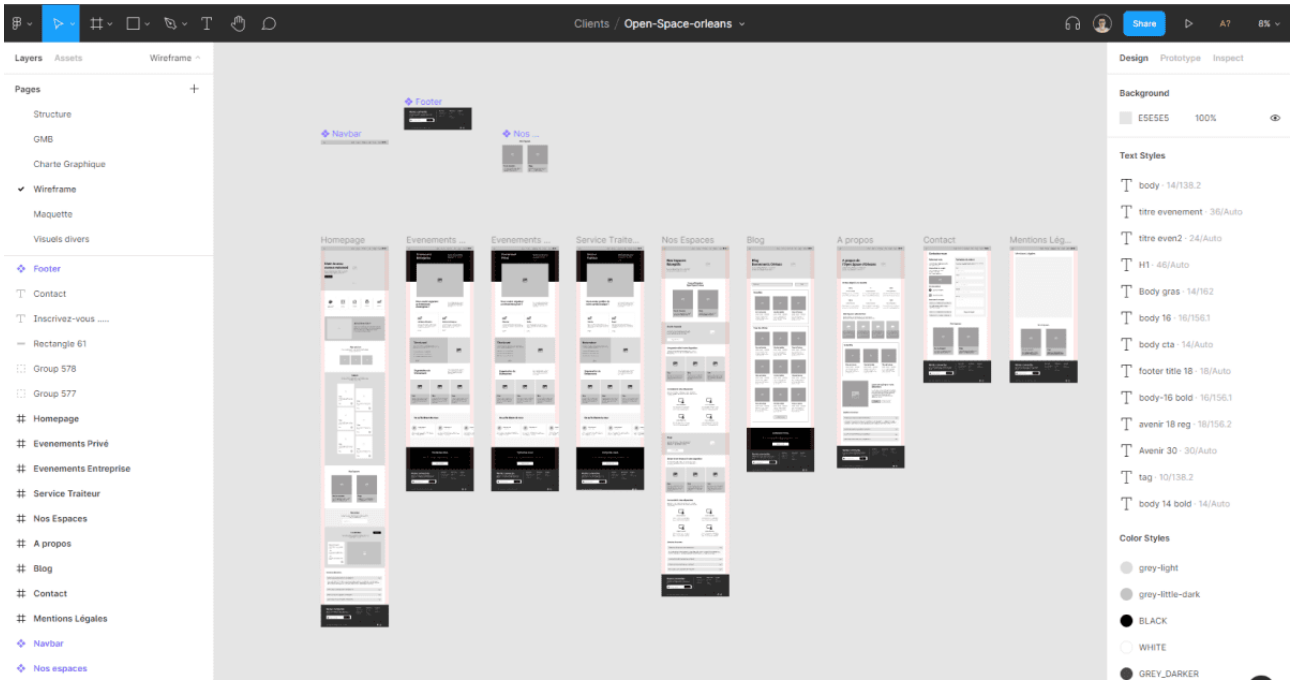

Рисунок 2.4 – Інтерфейс Figma

Figma є одночасно веб-платформою, що дозволяє багатьом користувачам одночасно працювати над одним проєктом. Користувачі можуть легко спілкуватися, коментувати, редагувати та спільно робити зміни зі своєю командою або клієнтами, що полегшує спільну роботу та збір зворотного зв'язку. Проєкти зберігаються в хмарі, що дозволяє отримати до них доступ з будь-якого пристрою. Користувачі можуть працювати над проєктом з різних місць та одночасно з іншими користувачами. Figma має вбудовані інструменти для створення інтерактивних прототипів. Користувач може створювати перехід між екранами, додавати анімацію та інтерактивні елементи, що дозволяє легко

демонструвати та тестувати функціональність розроблюваного додатку. Ну і на останок, Figma підтримує інтеграцію з іншими популярними інструментами дизайну та розробки, такими як Zeplin, InVision, Jira тощо. Це полегшує обмін даними та співпрацю з різними членами команди.

Нажаль, деколи плюси одного застосунку стають його ж мінусами, так сталося і з програмним застосунком Figma який має цілий ряд мінусів.

Залежність від інтернет-з’єднання: через те, що Figma є веб-платформою, їй потрібне постійне Інтернет-з’єднання для доступу до своїх проєктів та праці в них. Це може бути проблемою, якщо користувач часто працює без доступу до мережі або має обмежене Інтернет-підключення, наприклад мобільний зв’язок.

Обмежена можливість редагування зображень: у порівнянні з програмами, спеціалізованими на растровому редагуванні, такими як Photoshop, Figma нажаль має досить обмежені функції для редагування зображень.

Вартість: хоча Figma пропонує безкоштовний план з обмеженими функціями, деякі функції та додаткові можливості доступні за плату. Використання повної функціональності Figma може вимагати підписки на платний план. Але це все ж краще, аніж перші два варіанти, котрі взагалі не мають безкоштовних планів.

Усупереч цим недолікам, Figma залишається популярним інструментом і найкращим на мою думку для дизайну інтерфейсу завдяки його спрощеному процесу спільної роботи, інтегрованому прототипуванню та високій доступності з будь-якого пристрою з активним Інтернет-підключенням.

3 РОЗРОБКА ANDROID ЗАСТОСУНКУ

3.1 Характеристика потенційної аудиторії проєкту

Цільовою аудиторією проєкту є люди, які потенційно мають наступні характеристики:

– вік: дорослі від 18 років;

– стать: застосунок підійде для людей будь-якої статі;

– географічне розташування: Україна;

– освіта: медична;

– інтереси: фармацевтика, медицина;

– що хочуть від проєкту: зручне використання застосунку для пошуку, перегляду, та підбір дозування різних фармацевтичних виробів.

В цьому підрозділі була проведена характеристика потенційної аудиторії проєкту розробки Android застосунку в області фармацевтики. Визначено основні характеристики цільової аудиторії, такі як вік, географічне розташування, освіта, потреби та очікування користувачів. Це дозволило краще зрозуміти потреби цільової аудиторії та врахувати їх під час розробки застосунку.

3.2 Вибір засобів розробки

Серед можливих засобів розробки були обрані наступні програмні продукти: інтегроване середовище розробки Android Studio, фреймворк Flutter з мовою програмування Dart.

Обраним був саме Android Studio з декількох причин, однією з яких є те, що це інтегроване середовище розробки найкраще підходить для розробки

мобільних застосунків в першу чергу під операційну систему Android, тому що має в собі всі можливі інструменти, які можуть знадобитися при розробці мобільного застосунку. Також варто зазначити, що Android Studio розробляється самою компанією Google і є офіційним інструментом для розробки Android-застосунків.

Він постійно оновлюється та підтримується, що забезпечує доступ до останніх функцій та інструментів для розробки. Завдяки цьому, інтегроване середовище розробки працює в тісному співробітництві з Android-платформою, що дозволяє розробникам отримати повний доступ до всіх функцій та API, які надає Android.

Цей IDE має потужні інструменти для створення користувацького інтерфейсу, обробки даних, мережевого зв'язку, роботи з базами даних та багато іншого. До того ж, він надає розширені можливості для відлагодження мобільних застосунків. Він має вбудований відлагоджувальник, який дозволяє ставити точки зупинки, відстежувати значення змінних, аналізувати стек викликів та інше. Це допомагає при розробці ефективно виявляти та виправляти помилки.

Для тестування застосунків на різних пристроях з різними версіями Android, IDE Android Studio надає можливість використовувати вбудовані емулятори. Також можна підключати й реальні пристрої для відлагодження та тестування Android-застосунку.

Після вибору Android Studio як середовища розробки, було обрано фреймворк Flutter, який буде використовуватись при розробці Android застосунку разом з єдиною доступною для нього мовою програмування Dart.

Цей фреймворк було обрано у зв’язку з тим, що Flutter – це відкритий крос-платформовий фреймворк, який розроблений тією ж компанією Google, який дозволяє розробляти нативні мобільні застосунки для різних платформ, таких як: Android, iOS, Linux, macOS, Windows, Google Fuchsia та WEB [5]. Це означає, що можна створити одну базу коду для розробки застосунків як для

Android, так і для iOS. Це дозволяє значно зменшити час і зусилля, необхідні для розробки та підтримки додатків на різних платформах.

Завдяки гарнітурі готових до використання віджетів і багатофункціональному набору інструментів, Flutter пропонує швидкість розробки, що дозволяє зосередитися на створенні високоякісних застосунків з мінімальними затратами часу. Запуск та перевірка змін у реальному часі дозволяють швидше знаходити й усувати помилки.

Flutter має вбудовану систему візуального дизайну, яка дозволяє створювати вигляд, що відповідає дизайну матеріалів для Android та купі звичайних віджетів, які виглядають природно на платформі iOS. Це робить можливим створення додатків з гарним виглядом, які ідентичні на різних платформах.

Не дивлячись на те, що код пишеться не для кожної платформи окремо, не можна назвати застосунки розроблені за допомогою цього фреймворка повільнішими, адже Flutter використовує рушій Dart, який забезпечує високу продуктивність застосунків. Flutter використовує швидкий JIT (Just-in-Time) для швидкого розробки та гарантує швидку перевірку змін у режимі розробки, а також компілює код в швидкий AOT (Ahead-of-Time) для оптимальної продуктивності у фінальній версії додатку.

Загалом Flutter дозволяє створювати потужні та ефективні додатки для різних платформ з використанням одного коду, саме з цих причин він і був обраний.

У даному підрозділі був проведений аналіз різних засобів розробки для створення Android застосунку в області фармацевтики. Розглянуті були різні варіанти, такі як Flutter та нативна розробка за допомогою Java або Kotlin у Android Studio і Visual Studio Code. В результаті аналізу було обрано Flutter як оптимальний засіб розробки, оскільки він забезпечує швидку розробку за допомогою віджетів, переносимість на різні платформи та гарний інтерфейс користувача.

3.3 Розробка програмного застосунку

3.3.1 Ознайомлення і підготовка середовища розробки

Для того, щоб безпосередньо перейти до розробки Android застосунку, треба було підготувати все необхідне для цього, а саме:

1) завантажити та встановити Android Studio з офіційного сайту;
2) встановити Flutter SDK для того, щоб Android Studio “навчився” працювати з Flutter;
3) створити новий проєкт Flutter в Android Studio (див. рис. 3.1);
4) налаштувати емулятор або підключити фізичний пристрій у вкладці «AVD Manager»;
5) перевірити роботу емулятора запустивши тестовий проєкт, який надається Android Studio.

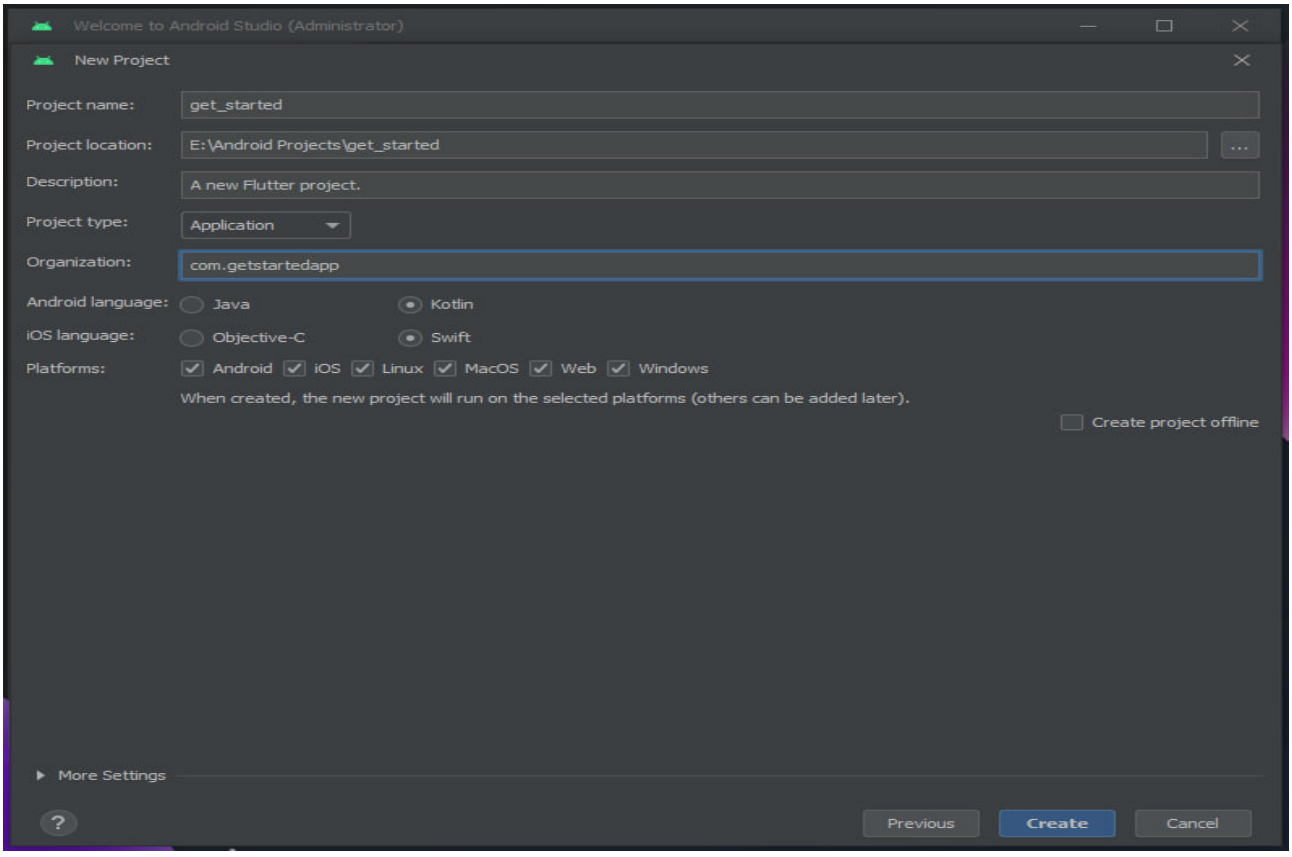

Рисунок 3.1 – Інтерфейс створення проєкту в Android Studio

При розробці з використанням фреймворку Flutter для створення інтерфейсу використовуються так звані віджети, які є основними будівельними блоками для створення користувацького інтерфейсу, на відміну від використання XML-розмітки при нативній розробці [12].

Вони представляють собою невеликі повторно використовувані елементи, які складаються разом, щоб утворити більш складні інтерфейси та функціонал. Flutter має багато вбудованих віджетів, а також дозволяє створювати власні віджети для задоволення унікальних потреб розробки.

Віджети у Flutter часто поділяють на Stateless і Stateful.

Stateless віджети називають такі, що не залежать від зовнішніх змін. Їх стан не може бути змінений після створення. Зазвичай такі віджети використовується для відображення статичного вмісту, такого як текст, зображення або кнопки.

Stateful віджет навпаки може змінювати свій стан під час виконання програми. Зазвичай використовується для створення динамічного вмісту, який може змінюватися відповідно до введення користувача або внутрішнього стану додатку.

Крім цього, Flutter надає можливість створювати власні віджети шляхом комбінування та композиції існуючих віджетів. Це дозволяє створювати користувацький інтерфейс та функціонал, який відповідає конкретним потребам вашого додатку.

Після створення проєкту, інтегроване середовище розробки вже пропонує код тестового мобільного застосунку під назвою “main.dart”, який складається з Stateful віджета та однієї головної частини під назвою Scaffold – одним з найважливіших та найчастіше використовуваних віджетів. Він представляє собою структуру або каркас для створення базового макетування та організації елементів інтерфейсу користувача в застосунку (див. рис. 3.2).

Scaffold надає так званий шаблон для розробки, на якому можна будувати сторінки та екрани застосунків [6]. Scaffold є потужним віджетом, який спрощує розробку макету та навігації в Flutter-застосунках.

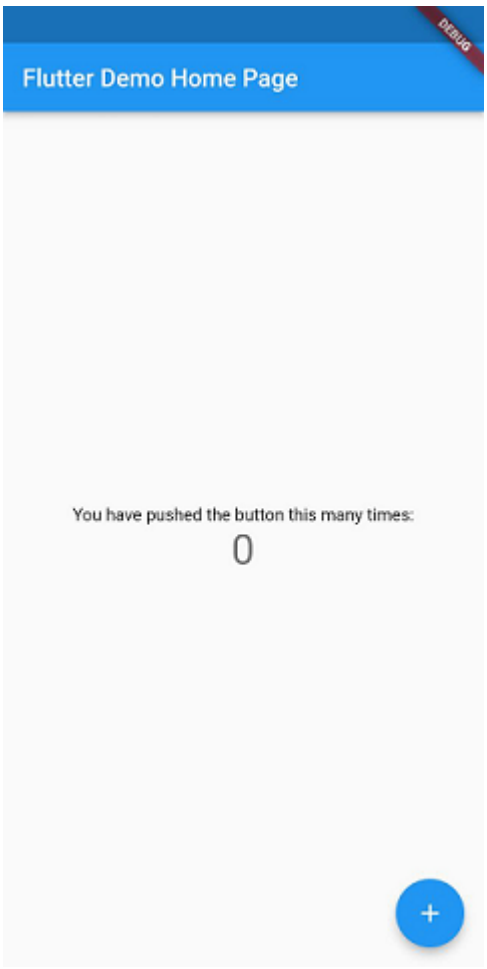

Рисунок 3.2 – Інтерфейс тестового застосунку

Основна мета Scaffold – забезпечити загальну структуру застосунку та надати основні компоненти, такі як:

– AppBar: панель заголовка, яка за замовчуванням встановлюється у верхній частині екрану і зазвичай окрім заголовку може містити в собі такі елементи, як кнопка пошуку, кнопка виклику бокового меню, та кнопку додаткових дій;
– Body: в перекладі з англійської означає тіло, що говорить про те, що це основна частина сторінки, яка може містити в собі різний зміст, такий як текст, зображення, списки, таблиці, різноманітні форми та багато іншого;
– BottomNavigationBar: цей компонент дозволяє створити нижню навігаційну панель з вкладками для зручного перемикання між різними екранами і може містити не тільки іконки, а й текст під ними;

- Drawer: дозволяє створити бокове меню, яке буде спливати з лівої частини екрану по натисканню спеціальної кнопки, що буде знаходитись у Appbar (це меню також можна використовувати разом з нижньою навігаційною панеллю);
- Snackbar: цей компонент використовується для показу повідомлень або сповіщень користувачу;
- FloatingActionButton: це спливаюча кнопка, яка за замовчуванням розташована в правому нижньому кутку екрана (ця кнопка може виконувати вказану розробником дію при натисканні на неї).

Тестовий застосунок з переліченого окрім самого Scaffold має такі компоненти, як: AppBar, Body та FloatingActionButton. В AppBar знаходиться тільки заголовок. Блок Body містить в собі два рядки тексту. А блок FloatingActionButton відповідає за кнопку, яка при натисканні виконує функцію «_incrementCounter», яка в свою чергу збільшує цифру в тексті блоку Body на одиницю та викликає функцію «setState» яка перемальовує екран, для того щоб оновити інформацію, яка була змінена у Body.

Для застосунку, який розробляється у рамках дипломного проєкту, нам цей тестовий код майже не знадобиться, тому цей код було видалено. В “main.dart” в проєкті буде лише перелік можливих маршрутів для навігації та стартовий екран, який буде відображатися при відкритті застосунку.

3.3.2 Розробка основних екранів

Для зручності та читабельності коду, при розробці кожний екран буде виведено в окремий файл формату “назва_екрану.dart”. Застосунок складається з основного екрану “main_screen_widget.dart” який являється Stateful та включає в себе віджет Scaffold, що в свою чергу має просту структуру, яка складається з:

- AppBar: містить в собі заголовок;

– Body: містить в собі IndexedStack віджет, для того щоб відображати три основних екрани, про які мова буде йти нижче;
– BottomNavigationBar: містить в собі нижнє навігаційне меню, яке складається з трьох кнопок, при натисканні на які буде змінюватись відображуваний екран.

Для того, щоб тримати у пам’яті три основні екрани, було використано саме IndexedStack, адже без нього при зміні екрану через нижнє навігаційне меню, екран кожен раз буде завантажуватись знов, що буде зайвий раз використовувати як ресурси пристрою, так і робити зайві запроси до серверу [7]. Також це додає зручності при використанні застосунку, адже якщо користувач щось буде шукати на одному екрані і випадково перейде на інший, то повернувшись знов екран не буде перемальовуватись і повернеться таким, яким він був до переходу на інший екран.

Основних екранів, які містяться в віджеті IndexedStack всього 3, а саме:

1) home_screen_widget.dart: містить у собі список з плитками, на яких повинні бути текст з назвою фармацевтичного виробу та його зображенням (при натисканні на плитку, повинна програватися анімація та відбуватися перехід на екран зазначеного фармацевтичного виробу);
2) search_screen_widget.dart: містить у собі рядок пошуку, та у разі введення до нього якогось тексту, виводити список із результатами пошуку;
3) about_screen_widget.dart: містить у собі основні відомості про програму, розробника та базові налаштування з можливістю звернутися до розробника з запитанням, тобто зворотній зв’язок для користувачів.

Для реалізації списку плиток було використано віджет «GridView», який більш оптимізований для великих списків і тримає у пам’яті тільки ті плитки, які вміщуються на екрані. До плиток було застосовано «decoration», який дозволяє додати тіні, потрібні закруглення та інші потрібні для дизайну засоби [8].

Заповнивши застосунок тестовими даними, було первинно протестовано роботу списку, анімацій і нижнього навігаційного меню (див. рис. 3.3).

Рисунок 3.3 – Інтерфейс екрану «Головна»

Наступним в черзі був екран пошуку, який складається з пошукового рядка та списку результатів пошуку (див. рис. 3.4). Окрім верстки було одразу реалізовано і функціонал пошуку, адже тестових даних для його функціонування уже вистачало. Список містить в собі назву, невелике зображення, та короткий опис фармацевтичного виробу. При натисканні користувача буде перенаправлено на екран того фармацевтичного виробу, який був вибраний.

Завдяки використанню IndexedStack на головному екрані, користувач може щось пошукати на вкладці пошуку, переключитись на іншу вкладку, а потім повернутись назад на цю і результати пошуку не будуть втрачені, а ресурси пристрою не будуть витрачатись на зайве перемальовування екранів [9].

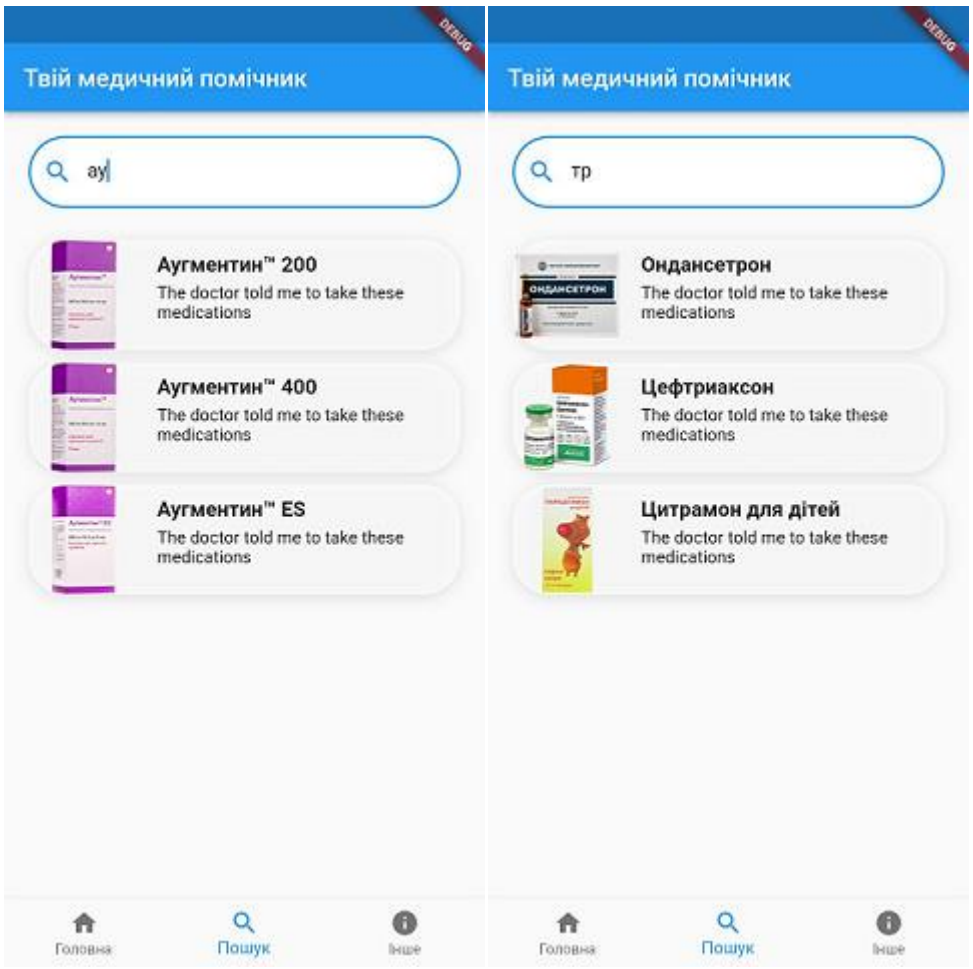

Рисунок 3.4 – Інтерфейс екрану «Пошук» на різних прикладах

Для зручності користувачів також було реалізовано автоматичне згортання клавіатури, якщо список вийшов занадто довгим і користувачу довелось прокручувати вниз по списку.

Сам пошук було реалізовано за допомогою використання проміжного масиву, який сортирує дані з основного масиву з препаратами та перекладає відповідаючі пошуковим запросам препарати в відфільтрований масив, а потім визиває «ListView.builder()» (див. рис. 3.5).

Останнім в черзі основних був екран, який має назву «Інше», який містить в собі короткі відомості про розробника, можливості зв’язку з розробником аби запропонувати покращення, а також декілька незначних налаштувань, таких як тема, мова застосунку та інші. Особливих складнощів з розробкою цього екрану не було, являє собою один з найпростіших екранів у застосунку (див. рис. 3.6).

```
void _searchMedications() {
  final query = _searchController.text;
  if (query.isNotEmpty) {
    _filteredMedications = _medications.where((Medication medication) {
      return medication.title.toLowerCase().contains(query.toLowerCase());
    }).toList();
  } else {
    _filteredMedications = [];
  }
  setState(() {});
}
```

Рисунок 3.5 – Програмна реалізація пошуку

Рисунок 3.6 – Інтерфейс екрану «Інше»

При натисканні на кнопку «Зв’язок з розробником» здійснюється перехід і автоматично передача пошти розробника та теми листа до застосунку електронної пошти, яке стоїть за замовчуванням в системі.

Щоб перенаправити користувача на електронну пошту з заданою темою у Flutter потрібно використати пакет «url_launcher». Пакет url_launcher є одним з найпоширеніших пакетів у Flutter для роботи з відкриттям URL-адрес. Він надає можливість відкривати URL-адреси в браузері або запускати поштові клієнти для надсилання електронної пошти. Пакет url_launcher також має підтримку відкриття URL-адрес у вбудованих додатках, виклику телефонного номера, відкриття карт у картографічних додатках та інші корисні функції, пов’язані з посиланнями.

Натискання на кнопку «Тема застосунку» (див. рис. 3.7) визиває діалогове вікно, в якому можна обрати одну з трьох тем:

– системна тема;
– світла тема;
– темна тема.

Для реалізації зміни тем знадобилися додаткові два пакети, а саме: Provider – для управління станом теми та Flutter_localizations – для отримання інформації про поточну тему системи.

Варто зазначити, що пакет «provider» є також одним з найпопулярніших пакетів у Flutter, але вже для управління станом додатка та забезпечення простого та ефективного способу передачі та оновлення даних між різними частинами додатка.

Основна ідея пакету provider полягає у використанні концепції “постачальника” (provider), який постачає дані з одного місця та забезпечує їх доступність для будь-яких віджетів, які їх використовують. Це дозволяє легко оновлювати дані та автоматично перерендерити віджети, які використовують ці дані.

Треба зазначити, що всі пакети, які були використані у проєкті заносяться розробником в файл “pubspec.yaml”, та виконується команда “flutter pub get”,

щоб отримати необхідні для функціонала пакети.

Пакет flutter_localizations є в свою чергу одним з важливих пакетів у Flutter, який дозволяє локалізувати додаток, тобто надавати переклади тексту та адаптувати його для різних мов та регіональних налаштувань.

Основна функція пакету flutter_localizations полягає у наданні локалізованих ресурсів, таких як рядки тексту, форматування дати та часу, числові формати і т.д. Завдяки цьому пакету розробник може створювати додатки, які автоматично адаптуються до мови, встановленої на пристрої користувача, або до мови, яку користувач обирає в додатку. Або в нашому випадку до теми, яка встановлена в системі.

```
class ThemeNotifier with ChangeNotifier {
  ThemeType _themeType = ThemeType.light;
  ThemeData get themeData {
    return _themeType == ThemeType.light
        ? ThemeData.light()
        : ThemeData.dark();
  }
  void toggleTheme() {
    _themeType = _themeType == ThemeType.light
        ? ThemeType.dark
        : ThemeType.light;
    notifyListeners();
  }
}
```

Рисунок 3.7 – Програмний код для зміни теми застосунку

Після натискання на кнопку «Мова застосунку» з’явиться діалогове вікно, в якому можна буде обрати мову, на якій будуть увесь текст у застосунку.

Після розробки основних екранів, створення пустих шаблонів, було

допрацьовано навігацію.

Навігація представляє собою маршрути, які зазвичай прописуються у основному файлі “main.dart”. Також було завершено роботу над плитками на першій сторінці застосунку (див. рис. 3.8).

```
Column(
            children: [
              SizedBox(
                height: 150,
                width: 150,
                child: Image(image: AssetImage(medications.imageName)),
              ),
              Padding(
                padding: const EdgeInsets.symmetric(horizontal: 5),
                child: Text(medications.title, maxLines: 1, overflow:
TextOverflow.ellipsis,),
              ),
            ],
          )
```

Рисунок 3.8 – Програмний код плиток першої сторінки

3.3.3 Розробка додаткових екранів

Наступними по плану розробки йдуть додаткові екрани, які втім не поступаються за важливістю, адже містять в собі усю надважливу інформацію стосовно фармацевтичного вибору. До цих екранів належать:

- екран з короткою інформацією, зображенням, та кнопками для переходу на інші екрани (див. рис. 3.9);
- екран з описом: на цьому екрані можна прочитати більш докладний опис

про обраний фармацевтичний виріб, дізнатися більше про виробника, тощо;

– екран з калькулятором: на цьому екрані можна буде порахувати дозування для особи, в залежності від вказаних факторів, котрих потребує калькулятор, частіше за все це вага особи;
– екран з інструкцією: екран на якому можна прочитати повністю інструкцію по застосуванню, склад, побочні ефекти та інші корисні і важливі пункти.

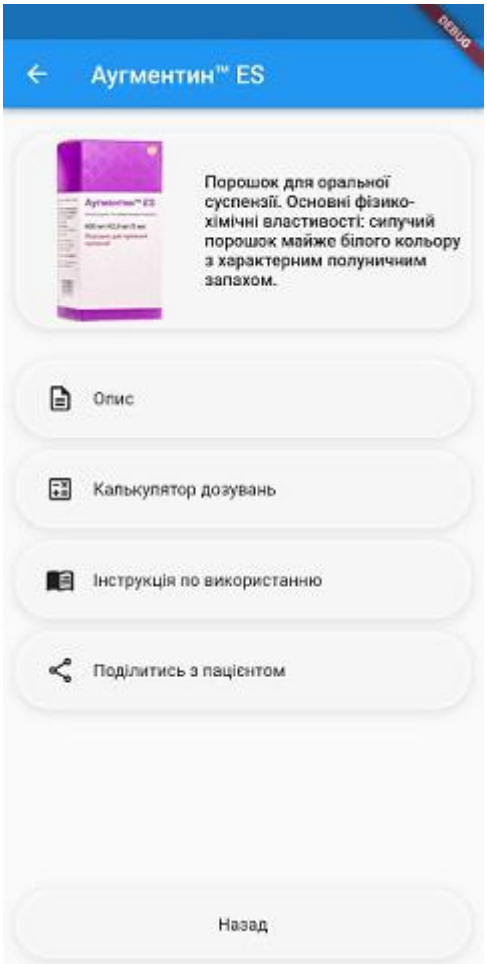

Рисунок 3.9 – Інтерфейс екрану з інформацією

При натисканні на кожну із карток відбувається перехід до відповідного екрана, окрім картки «Поділитись з пацієнтом», при натисканні на яку з'явиться діалогове вікно, з якого можна поділитися обраною інформацією з іншими застосунками, такими як електронна пошта, мессенджери, соц. мережі та ін.

Щоб реалізувати функцію «Поділитись з пацієнтом», було додатково використано пакет «Share», який є попередником більш сучасного «Share Plus», але для нашого проєкту вистачить і старого пакету.

Також для зручності користувачів, кнопка для переходу назад була продубльована знизу, адже якщо користувач не користується керуванням за допомогою жестів або пристрій не підтримує такий тип навігації, то деяким користувачам тягнутись до верхнього лівого кута екрану може бути незручно.

Анімація дотиків була реалізована за допомогою класу InkWell() та методу onTap() {} як в цьому екрані, так і усіх попередніх на послідуючих. Перехід на інші екрани відбувається за допомогою класу Navigator [11]. Він управляє стеком маршрутів (англ. routes) нашого застосунку. Коли користувач переходить з одного екрана на інший, то фактично додається новий маршрут до стеку або видаляється існуючий маршрут зі стеку. Основні методи Navigator:

- push: додає новий маршрут до стеку і переходить на новий екран;
- pop: видаляє останній маршрут зі стеку і повертається назад;
- pushReplacement: заміняє поточний маршрут новим маршрутом;
- popUntil: видаляє маршрути зі стеку до певного маршруту;

Наступним етапом було зробити екран, який буде містити в собі інформацію про опис того чи іншого фармацевтичного виробу, в залежності від того, який був вибраний на головному екрані (див. рис. 3.10).

У зв’язку з тим, що сторінка опису від сторінки з інструкцією майже нічим не відрізняється окрім тексту, яким вони будуть наповнені, то і код був використаний один і той самий, з тою лиш різницею, що при відкритті сторінки передавався різний текст, який буде відображатися на сторінці. Це дозволяє зекономити як ресурси пристрою, так і час для розробки, адже немає сенсу повторно розробляти те, що вже було розроблено.

Для реалізації гортання сторінки було використано звичайний SingleChildScrollView(), адже в нас всього лише звичайний текст, цього повинно вистачити.

Рисунок 3.10 – Інтерфейс екрану опису та в даному випадку інструкцій

Один з найголовніших екранів, це екран з калькулятором дозувань (див. рис. 3.11). На цьому екрані є лише декілька блоків, а саме:

– зображення і назва фармацевтичного виробу;
– текст з вагою особи, обирається за допомогою слайдера нижче;
– текст з результатами прорахування(в деяких випадках таких блоки може бути два);
– кнопка, аби повернутися назад.

За допомогою слайдера користувач зручно і швидко може обрати потрібну йому вагу, завдяки зображенню не загубиться на якій саме він сторінці, а за допомогою блоків з результатами одразу зрозуміє, яке саме дозування йому буде потрібно.

Для підрахування потрібного дозування використовується інструкція від

виробника та звичайні математичні функції, для кожного медикаменту окремі. Для деяких медикаментів може використовуватися декілька різних формул для різних випадків наприклад легкий або тяжкий стан захворілого.

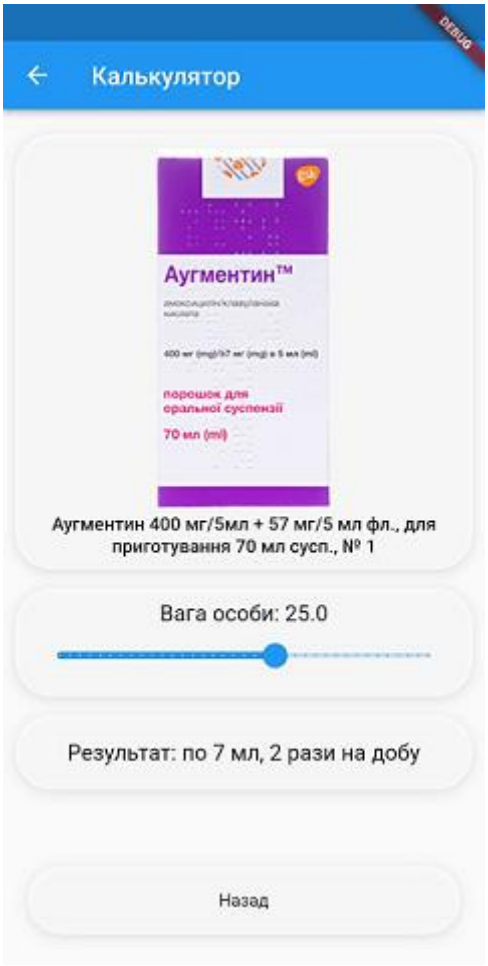

Рисунок 3.11 – Інтерфейс калькулятора дозувань

При перетягуванні повзунка активується функція «onChanged», в якій викликається «setState» задля того, аби перемалювати екран вже з значеннями, які ми отримуємо використовуючи самописну функцію «calculation» і в подальшому присвоюємо це значення в змінну, яка виводиться на екран у наступному за слайдером блоці (див. рис. 3.12).

Задля того, щоб повернутися на попередній екран, використовується метод «Navigator.pop(context)».

```
child: Column(
          children: [
            Text('Вага особи: $_currentSliderValue', style: TextStyle(fontSize: 18),),
            Slider(
              value: _currentSliderValue,
              min: 4,
              max: 40,
              divisions: 36,
              label: _currentSliderValue.round().toString(),
              onChanged: (double value) {
                setState(() {
                  _result = _calculation(value).toStringAsFixed(1);
                  _currentSliderValue = value;
                });
              },
            ),
          ],
        ),
```

Рисунок 3.12 – Код слайдера від калькулятора

Якщо узагальнити, то можна сказати, що у даному підрозділі була проведена безпосередня розробка програмного застосунку з використанням Flutter. Були використані сучасні практики розробки та рекомендації для покращення продуктивності та якості застосунку.

Розроблений застосунок має такі функції, такі як перегляд списку ліків, перегляд скороченого і детального опису ліків, калькулятор дозувань для ліків, перегляд інструкції, можливість поділитись переглянутою інформацією, пошук ліків по застосунку, та можливість зв’язатись з розробником напряму.

Результатом цієї розробки є готовий Android застосунок в області фармацевтики, який може забезпечити зручний доступ до фармацевтичних

послуг для користувачів на платформі Android і не тільки, адже фреймворк Flutter дозволяє займатися розробкою одної кодової бази для декількох платформ, таких як Android, iOS, Linux, macOS, Windows, Google Fuchsia, та WEB.

ВИСНОВКИ

У результаті теоретичної частини було проведено дослідження та розробка Android застосунку в області фармацевтики. Під час аналізу предметної області було виявлено потенціал мобільних додатків для поліпшення доступності та ефективності фармацевтичних послуг. На основі цього аналізу було визначено об'єкт дослідження та сформульовано мету роботи і виявлено, що для розробки Android застосунку найбільш підходящими методами розробки є використання програмних засобів Android Studio, Dart та Flutter.

В рамках дипломної роботи було розроблено технічне завдання для програмного застосунку з використанням Flutter, в якому описали основні функціональні вимоги, вимоги до безпеки, надійності та складу і параметрів технічних засобів.

Також було спроєктовано діаграму використання за допомогою застосунку diagrams.net, а інтерфейс майбутнього застосунку з використанням программного застосунку Adobe Photoshop. В результаті чого було розроблено мобільний Android застосунок, який містить в собі більше 6 універсальних екранів, пошуковик медикаментів, калькулятор дозування медикаментів, переглядач змісту опису та інструкцій обраного медикаменту, шеринг обраного змісту, базові налаштування зовнішнього вигляду, бажаної мови, та зв’язок з розробником за допомогою електронної пошти.

В результаті розробки було з'ясовано, що розробка Android застосунку в області фармацевтики має великий потенціал для поліпшення доступності та ефективності фармацевтичних послуг. Мобільний додаток може забезпечити користувачам зручний доступ до інформації про ліки, інструкцій, дозувань та іншою корисної інформації як для пацієнтів, так і для лікарів.

Розробка такого застосунку з використанням Flutter та врахування всіх особливостей фармацевтичної галузі є актуальним завданням, яке може

принести значний вплив на сферу фармації та допомогти користувачам забезпечити своє здоров'я та благополуччя.

Загальна робота над розробкою Android застосунку в області фармацевтики привела до створення функціонального та корисного інструменту, який може покращити життя людей та допомогти їм зберігати здоров'я. Результати дослідження та розробки можуть бути використані як основа для подальших покращень та розширення функціональності застосунку. Дана робота вносить свій внесок у розвиток мобільних технологій та використання їх у сфері фармацевтики.

Отже, робота демонструє значимість мобільних додатків у фармацевтичній галузі та надає конкретні рекомендації щодо розробки Android застосунку. Цей додаток може стати цінним інструментом для покращення фармацевтичних послуг та сприяти поліпшенню здоров'я та добробуту користувачів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Herbert S. Java. The Complete Reference. McGraw-Hill. URL: https://www.oreilly.com/library/view/java-the-complete/9781260463422/ (дата звернення: 14.03.2023).
2. Bryan S., Brian G., Kristin M., Chris S. Android Programming: The Big Nerd Ranch Guide, 5th Edition. Addison-Wesley Professional. URL: https://www.oreilly.com/library/view/android-programming-the/9780137645794/ (дата звернення: 17.03.2023).
3. Maximilian S. Learn Flutter and Dart to Build iOS and Android Apps. Packt Publishing. URL: https://www.oreilly.com/library/view/learn-flutter-and/9781789951998/ (дата звернення: 20.03.2023).
4. Dart documentation. URL: https://dart.dev/guides (дата звернення: 22.03.2023).
5. Flutter documentation. URL: https://docs.flutter.dev (дата звернення: 25.03.2023).
6. Flutter – Material Design 3. URL: https://m3.material.io/develop/flutter (дата звернення: 28.03.2023).
7. Paulo D. The Complete Flutter and Dart App Development Course. Packt Publishing. URL: https://www.oreilly.com/library/view/the-complete-flutter/9781800563322/ (дата звернення: 10.04.2023).
8. Eric W. Flutter in Action. Manning Publications. URL: https://www.oreilly.com/library/view/flutter-in-action/9781617296147/ (дата звернення: 13.04.2023).
9. What’s The Difference Between Web Apps, Native Apps, And Hybrid Apps. URL: https://aws.amazon.com/compare/the-difference-between-web-apps-native-apps-and-hybrid-apps/ (дата звернення: 15.04.2023).
10. Theresa N. Mobile Design Pattern Gallery: UI Patterns for Smartphone Apps. O'Reilly Media. URL: https://www.amazon.com/Mobile-Design-Pattern-

Gallery-Smartphone/dp/1449363636 (дата звернення: 18.04.2023).

11. Gilad B. The Dart Programming Language. Addison-Wesley Professional. URL: https://www.oreilly.com/library/view/the-dart-programming/9780133429961/ (дата звернення: 22.04.2023).
12. Richard R. Flutter and Dart Cookbook. O'Reilly Media. URL: https://www.oreilly.com/library/view/flutter-and-dart/9781098119508/ (дата звернення: 25.04.2022).