МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

**КОСТЕНКО РОМАН ОЛЕКСАНДРОВИЧ**

Допускається до захисту:
в.о. завідувача кафедри
інформаційних технологій
канд. техн. наук, доцент
_____ О. В. Зелінська
«____» _____ 20__ р.

# МОБІЛЬНИЙ ДОДАТОК ДЛЯ КЕРУВАННЯ ПРОЦЕСОМ ВОЛОНТЕРСЬКОЇ ДОПОМОГИ ВЕТЕРАНАМ

Спеціальність 122 Комп'ютерні науки

**Кваліфікаційна (бакалаврська) робота**

Керівник:
Поремський Ю.В., к. т. н
кафедри інформаційних
технологій

_____
(Підпис)

Оцінка _____/_____/_____
(бали/за шкалою ЄКТС/за національною шкалою)
Голова ЕК: _____
(Підпис)

Вінниця – 2025

## АНОТАЦІЯ

**Костенко Р. О.** Мобільний додаток для керування процесом волонтерської допомоги ветеранам. Спеціальність 122 “Комп’ютерні науки”, Освітня програма “Комп’ютерні технології обробки даних (Data Science)”. Донецький національний університет імені Василя Стуса, Вінниця, 2025.

У кваліфікаційній роботі представлено розробку мобільного Android-додатку Elpis, призначеного для оптимізації процесу волонтерської допомоги ветеранам. Додаток дозволяє ветеранам швидко створювати запити на допомогу з можливістю прикріплення фотографій, вибору категорії проблеми та передачі геолокаційних даних. Запропоноване рішення сприяє посиленню адресної допомоги, підвищенню ефективності волонтерських ініціатив та соціальній підтримці ветеранів.

**Ключові слова:** мобільний додаток, волонтерська допомога, ветерани, Android, Firebase, соціальна підтримка, геолокація.

71 с., 16 рис., 5 табл., 42 джерела.

## ABSTRACT

**Kostenko R.O.** A Mobile Application for Managing the Process of Volunteer Assistance to Veterans. Specialty 122 “Computer Science”, Educational Program “Computer Data Processing Technologies (Data Science)”. Vasyl Stus Donetsk National University, Vinnytsia, 2025.

This qualification work presents the development of an Android mobile application named Elpis, designed to optimize the process of volunteer assistance to veterans. The application enables veterans to quickly create help requests with the ability to attach photos, select a category of the issue, and share geolocation data. The proposed solution enhances the efficiency of targeted aid, improves the effectiveness of volunteer initiatives, and strengthens social support for veterans.

**Keywords:** mobile application, volunteer assistance, veterans, Android, Firebase, social support, geolocation.

71 p., 16 fig., 5 tab., 42 sources.

**ЗМІСТ**

ВСТУП ..... 4
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ РІШЕНЬ ..... 6
1.1 Проблема соціальної адаптації ветеранів та роль волонтерства ..... 6
1.2 Аналіз існуючих цифрових рішень у сфері волонтерської допомоги ..... 9
1.3 Організація процесу волонтерської допомоги та його цифровізація ..... 13
1.4 Специфіка розробки мобільних додатків соціального спрямування ..... 16
1.5 Технологічні платформи для розробки Android-додатку ..... 19
1.6 Формування вимог до мобільного додатку ..... 23
Висновоки до першого розділу ..... 25
РОЗДІЛ 2. РОЗРОБКА КОНЦЕПЦІЇ ТА АЛГОРИТМІВ ДОДАТКУ ..... 26
2.1 Постановка задачі та загальна концепція ..... 26
2.2 Модель користувацької взаємодії ..... 28
2.3 Архітектура додатку ..... 31
2.4 Алгоритми роботи системи ..... 35
2.5 Моделювання даних ..... 41
2.6 Безпека та автентифікація ..... 45
Висновоки до другого розділу ..... 47
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ ..... 48
3.1 Структура та компоненти програмного продукту ..... 48
3.2 Реалізація ключових функцій ..... 50
3.3 Інтерфейс користувача ..... 54
3.4 Тестування програмного продукту ..... 62
3.5 Аналіз результатів реалізації ..... 64
Висновоки до третього розділу ..... 66
ВИСНОВКИ ..... 68
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ..... 70

# ВСТУП

Сучасні виклики, пов’язані з військовими подіями, що відбуваються в Україні, обумовили зростання потреби у підтримці ветеранів, які повертаються до мирного життя після участі у бойових діях. Надання своєчасної та адресної волонтерської допомоги цій категорії населення є важливою складовою соціальної адаптації та інтеграції. Однак, існуюча система координації волонтерської допомоги часто є фрагментарною та неефективною, особливо на локальному рівні, де відсутні централізовані механізми організації таких ініціатив.

В умовах широкого розповсюдження мобільних технологій виникає можливість оптимізувати цей процес за допомогою спеціалізованих мобільних додатків. Такі рішення здатні забезпечити зручну платформу для взаємодії між тими, хто потребує допомоги, та тими, хто може її надати, скорочуючи час реагування, підвищуючи точність відповідності потреб і ресурсів та покращуючи загальну ефективність комунікації. Попри значний розвиток цифрових сервісів, у відкритому доступі майже відсутні комплексні рішення, спеціально орієнтовані на підтримку ветеранів через волонтерську мережу.

У рамках цієї роботи основна увага зосереджується на створенні Android-додатку, який забезпечить зручну взаємодію між ветеранами, що потребують допомоги, та волонтерами, які готові її надати. Перший етап дослідження полягає у вивченні існуючих мобільних рішень у сфері волонтерської допомоги, аналізу переваг та недоліків подібних додатків, визначенню особливостей організації запитів на допомогу, способів комунікації та зберігання даних.

Другий етап передбачатиме проектування архітектури майбутнього додатку, визначення ролей користувачів та інтеграцію з геолокаційними сервісами. У процесі реалізації функціональності буде створено можливості для створення запитів, їх відображення на карті, прийняття запитів волонтерами.

Результатом роботи стане створений мобільний додаток, що демонструватиме потенціал цифрових технологій у сфері соціальної допомоги. Таке рішення може бути корисним як для громадських ініціатив, так і для

організацій, що опікуються ветеранами. Дане дослідження також матиме теоретичне значення для подальших розвідок у галузі мобільної розробки та цифровізації соціальних сервісів.

# РОЗДІЛ 1
# АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ РІШЕНЬ

## 1.1 Проблема соціальної адаптації ветеранів та роль волонтерства

Соціальна адаптація ветеранів військової служби до цивільного життя є складним багатогранним процесом, що охоплює психологічні, соціальні, економічні та медичні аспекти. Особливої актуальності ця проблематика набуває в сучасних умовах, коли значна кількість військовослужбовців повертається до мирного життя після участі у бойових діях.

Дослідження свідчать, що понад 60% ветеранів відчувають труднощі з адаптацією протягом перших двох років після демобілізації. Основними викликами є:

- Психологічні аспекти адаптації характеризуються високим рівнем посттравматичних стресових розладів (ПТСР), депресії та тривожності. Згідно з дослідженням Національного інституту психічного здоров'я, близько 20-30% ветеранів, які брали участь у бойових діях, страждають на ПТСР різного ступеня тяжкості [1]. Симптоми ПТСР суттєво ускладнюють процес адаптації та негативно впливають на якість життя ветеранів.
- Соціальна дезадаптація проявляється у складнощах встановлення контактів з цивільним оточенням, відчуженості та ізоляції. Дослідження демонструє, що 47% ветеранів відчувають труднощі у відновленні соціальних зв'язків та інтеграції в громаду [2]. Ця проблема поглиблюється через відсутність розуміння з боку оточення та недостатню інформованість суспільства щодо специфіки ветеранського досвіду.
- Працевлаштування і професійна реалізація становлять окрему групу проблем. Статистичні дані свідчать, що рівень безробіття серед ветеранів у 1,5-2 рази вищий, ніж серед загального населення [3].

Причинами цього є невідповідність військових навичок вимогам цивільного ринку праці, проблеми зі здоров'ям, а також упереджене ставлення роботодавців.

- Медичні проблеми, зокрема реабілітація після поранень, травм та контузій, часто потребують довготривалого лікування та допомоги. За даними Міністерства у справах ветеранів, близько 35% учасників бойових дій потребують тривалої медичної реабілітації різного спрямування.

Дослідження процесів адаптації ветеранів до цивільного життя здійснюється з використанням різних методологічних підходів. Особливу увагу науковці приділяють вивченню факторів успішної реінтеграції. Аналіз наукових праць виявив, що успішна адаптація ветеранів корелює з такими чинниками, як:

1. Наявність соціальної підтримки з боку родини, друзів та громади.
2. Доступність комплексних реабілітаційних програм.
3. Можливості для професійної перепідготовки та працевлаштування.
4. Психологічний супровід на всіх етапах реінтеграції.

Міждисциплінарні дослідження вказують на необхідність комплексного підходу до вирішення проблем адаптації ветеранів. Зокрема, модель інтегрованої підтримки передбачає поєднання психологічної, медичної, соціальної та економічної допомоги для забезпечення всебічної реінтеграції.

Волонтерство відіграє ключову роль у процесі реінтеграції ветеранів, заповнюючи існуючі прогалини в державних програмах підтримки. Аналіз сучасних практик свідчить про різноманітність напрямків волонтерської діяльності:

- Психосоціальний супровід реалізується через програми групової та індивідуальної психологічної підтримки, телефони довіри, групи взаємодопомоги. Статистика демонструє, що учасники таких програм на 40% швидше адаптуються до цивільного життя порівняно з тими, хто не отримує подібної підтримки.

- Сприяння працевлаштуванню включає професійну орієнтацію, навчання новим професіям, консультування щодо створення власного бізнесу. Успішні приклади таких ініціатив демонструють, що рівень працевлаштування ветеранів може зростати до 70% за умови комплексного підходу [4].
- Медична реабілітація та оздоровлення передбачають залучення волонтерів-медиків, психологів, фізіотерапевтів. Волонтерські організації часто забезпечують доступ до реабілітаційних послуг, недоступних у державних закладах.
- Правова підтримка ветеранів здійснюється через надання безкоштовних юридичних консультацій, допомогу в оформленні документів, захист прав ветеранів у різних інстанціях.

Аналіз діяльності волонтерських організацій, показав, що залучення ветеранів до волонтерських програм на 63% підвищує їх шанси на успішну соціальну адаптацію. Найбільш ефективними моделями волонтерської допомоги ветеранам є:

- Модель «ветеран-ветерану» ґрунтується на залученні колишніх військовослужбовців до волонтерської діяльності. Цей підхід має особливу цінність, оскільки базується на спільному досвіді та розумінні специфічних проблем ветеранів.
- Інтегрована модель передбачає координацію зусиль державних інституцій, громадських організацій та волонтерів. Така модель дозволяє забезпечити комплексний підхід до реінтеграції ветеранів.
- Громадоцентрична модель орієнтована на активне залучення місцевих громад до процесу адаптації ветеранів. Ця модель сприяє формуванню підтримуючого середовища на місцевому рівні.

Дослідження ефективності різних моделей волонтерської допомоги свідчить, що найкращі результати досягаються при поєднанні різних підходів з урахуванням індивідуальних потреб ветеранів.

## 1.2 Аналіз існуючих цифрових рішень у сфері волонтерської допомоги

В умовах активного розвитку інформаційних технологій та зростаючої потреби в ефективній координації волонтерської діяльності відбувається стрімке впровадження цифрових рішень у сферу соціальної допомоги. Особливо гостро постає питання застосування технологічних інновацій для підтримки ветеранів, які потребують систематизованої та оперативної допомоги. На сучасному етапі існує низка цифрових платформ і рішень, спрямованих на оптимізацію волонтерської допомоги різним категоріям населення, зокрема ветеранам.

Аналіз наявних на ринку рішень дозволяє класифікувати їх за кількома ключовими критеріями:

1. За типом технологічного рішення:
    - Веб-платформи.
    - Мобільні додатки.
    - Чат-боти.
    - Інтегровані системи (комбінація веб-платформи та мобільного додатка).
2. За функціональним призначенням:
    - Координація волонтерської діяльності.
    - Інформаційна підтримка ветеранів.
    - Психологічна допомога та реабілітація.
    - Пошук спеціалізованих послуг.
3. За цільовою аудиторією:
    - Універсальні рішення для різних категорій населення.
    - Спеціалізовані рішення для ветеранів.
    - Платформи для волонтерів.
    - Системи, орієнтовані на взаємодію між волонтерами та бенефіціарами.

В умовах сучасних викликів, з якими стикається українське суспільство, волонтерська діяльність набуває особливого значення. Розглянемо декілька ключових платформ та додатків, які сприяють розвитку волонтерства в Україні:

1. Волонтерська Платформа [5] є комплексним веб-рішенням, яке об'єднує волонтерів та оптимізує процеси надання допомоги. Вона забезпечує координацію волонтерських ініціатив на національному рівні. Основні функціональні можливості:
    - Реєстрація волонтерів та організацій.
    - Публікація запитів на допомогу.
    - Комунікація між учасниками процесу.
    - Відстеження виконання волонтерських завдань.
    - Формування звітності щодо наданої допомоги.

    Щодо переваг та недоліків описаного рішення, можна навести наступну таблицю порівняння 1.1:

    Таблиця 1.1 Порівняння переваг та недоліків «Волонтерської Платформи»

| Переваги | Недоліки |
|---|---|
| Розгалужена мережа волонтерів по всій території України | Обмежена функціональність мобільної версії |
| Прозора система верифікації потреб | Недостатня спеціалізація для роботи саме з ветеранами |
| Конфіденційність даних | Бюрократизація багатьох процесів |
| Інтуїтивно зрозумілий інтерфейс | Відсутність інтеграції з державними сервісами підтримки ветеранів |

2. Волонтер.орг [6] є інформаційно-координаційним ресурсом, що об'єднує волонтерські ініціативи та оптимізує їх діяльність, надаючи можливість систематизації процесів волонтерської допомоги. Основні функціональні можливості:
    - Формування бази даних волонтерів та бенефіціарів.
    - Аналітика результатів волонтерської діяльності.
    - Категоризація потреб та запитів.

- Відстеження виконання волонтерських завдань.
- Координація логістичних процесів.

В контексті переваг та недоліків даного рішення, можна скласти наступну таблицю для порівняння 1.2:

Таблиця 1.2 Порівняння переваг та недоліків «Волонтер.орг»

| Переваги | Недоліки |
|---|---|
| Розвинена система категоризації послуг | Недостатньо розвинена мобільна версія |
| Можливість формування волонтерських спільнот за напрямками | Складність інтерфейсу для користувачів старшого віку |
| Інтеграція з соціальними мережами для швидкого поширення інформації | Відсутність спеціалізованих інструментів для роботи з ветеранами |
| Статистичний аналіз наданої допомоги з візуалізацією результатів | Недостатня захищеність персональних даних користувачів |

Зарубіжні цифрові рішення в сфері волонтерської допомоги ветеранам демонструють значний рівень технологічної зрілості та комплексного підходу, що дозволяє ефективно вирішувати різноманітні проблеми, з якими стикаються колишні військовослужбовці. Ці рішення не лише забезпечують доступ до необхідних ресурсів, але й сприяють створенню спільнот, де ветерани можуть отримувати підтримку та обмінюватися досвідом. Варто розглянути кілька провідних прикладів для порівняльного аналізу:

1. Veterans Affairs Mobile [7] є діючим комплексом мобільних додатків, розроблених Департаментом у справах ветеранів США, представляє собою інтегровану екосистему цифрових рішень для різних аспектів підтримки ветеранів. Основні функціональні можливості:
    - Управління медичними призначеннями.
    - Комунікація між учасниками процесу.
    - Доступ до персональної медичної інформації.
    - Психологічна підтримка та моніторинг.
    - Координація соціальних послуг.

Відносно переваг та недоліків цього рішення, можна навести таблицю для порівняння 1.3:

Таблиця 1.3 Порівняння переваг та недоліків «Veterans Affairs Mobile»

| Переваги | Недоліки |
|---|---|
| Висока інтеграція з державними системами підтримки | Складність адаптації до українських реалій |
| Комплексний підхід до різних аспектів допомоги | Орієнтація на специфічну систему підтримки США |
| Функціональність для віддаленого моніторингу стану здоров'я | Обмеження на певні види підтримки в залежності від категорії ветеранської служби |

2. VolunteerMatch [8] є глобальною платформою, що об'єднує волонтерів та організації, які потребують допомоги. Хоча платформа не спеціалізується виключно на підтримці ветеранів, вона демонструє ефективні механізми координації волонтерської діяльності. Основні функціональні можливості:
   - Пошук волонтерських можливостей за категоріями та місцем розташування.
   - Управління волонтерськими проєктами.
   - Відстеження волонтерської активності.
   - Аналітика результатів волонтерської діяльності.

   Щодо переваг та недоліків описаного рішення, можна навести наступну таблицю порівняння 1.4:

   Таблиця 1.4 Порівняння переваг та недоліків «VolunteerMatch»

| Переваги | Недоліки |
|---|---|
| Масштабна база даних волонтерів та організацій | Недостатня локалізація для українського ринку |
| Розвинені алгоритми пошуку відповідностей | Відсутність спеціалізації на потребах ветеранів |
| Гнучка система фільтрів та категоризації | Обмежені можливості для адаптації під специфічні потреби |

Аналіз вищезазначених цифрових рішень дозволяє виявити певні закономірності та прогалини в існуючому технологічному ландшафті волонтерської допомоги ветеранам. Основні виявлені проблеми:

- Фрагментованість рішень – платформи охоплюють лише окремі аспекти підтримки, ускладнюючи отримання комплексної допомоги.
- Недостатня інтеграція – рішення працюють ізольовано, без взаємодії між собою та державними сервісами.
- Обмежена мобільність – переважання веб-платформ зі слабкою мобільною підтримкою знижує доступність.
- Складність управління волонтерським процесом – бракує ефективних інструментів для управління, моніторингу та звітності.
- Недостатня персоналізація – рішення не враховують індивідуальні потреби ветеранів.

Таким чином, аналіз існуючих цифрових рішень у сфері волонтерської допомоги ветеранам виявляє значні можливості для оптимізації процесів через розробку спеціалізованого мобільного додатка, який враховуватиме виявлені недоліки та реалізовуватиме перспективні напрямки розвитку.

### 1.3 Організація процесу волонтерської допомоги та його цифровізація

Дослідження волонтерської допомоги ветеранам виявляє складну структуру, що потребує координації й оптимізації, а аналіз підходів до її організації та цифровізації є основою для створення мобільного додатку. Процес надання волонтерської допомоги ветеранам характеризується визначеною послідовністю етапів, що забезпечують його системність та результативність:

1. Ідентифікація потреб ветеранів. На цьому етапі відбувається виявлення конкретних потреб ветеранів шляхом проведення опитувань, інтерв'ю, аналізу запитів та звернень.
2. Планування волонтерської допомоги. Цей етап передбачає розробку програм та заходів, спрямованих на задоволення виявлених потреб. За

даними Міжнародного інституту волонтерства, ефективне планування дозволяє на 40% підвищити результативність волонтерських ініціатив [9].

3. Рекрутинг та підготовка волонтерів. На цьому етапі здійснюється залучення добровольців, оцінка їхніх компетенцій та проведення необхідного навчання. Якісна підготовка волонтерів є критичним фактором успіху програм допомоги ветеранам.
4. Координація волонтерської діяльності. Цей етап включає розподіл завдань між волонтерами, встановлення комунікаційних каналів, моніторинг виконання завдань. Дослідження виявило, що недостатня координація є однією з основних причин неефективності волонтерських ініціатив [10].
5. Надання безпосередньої допомоги. На цьому етапі відбувається практична реалізація запланованих заходів та безпосередня взаємодія волонтерів з ветеранами.
6. Оцінка результатів та зворотний зв'язок. Заключний етап передбачає аналіз досягнутих результатів, отримання відгуків від ветеранів, коригування подальших дій.

Варто зазначити, що організація процесу волонтерської допомоги характеризується циклічністю та взаємозалежністю всіх етапів. У випадку порушення послідовності або неналежна реалізація будь-якого з етапів призводить до значного зниження ефективності всього процесу.

Аналіз практичного досвіду організації волонтерської допомоги ветеранам дозволяє виділити кілька основних моделей координації, кожна з яких має свої переваги та обмеження:

- Централізована модель координації характеризується наявністю єдиного координаційного центру, що здійснює планування, розподіл завдань та контроль. В загальному, централізована модель демонструє найвищу ефективність при реалізації масштабних програм допомоги.
- Децентралізована модель передбачає розподіл координаційних функцій між кількома центрами або групами волонтерів. Ця модель забезпечує

вищу адаптивність до локальних умов та потреб, але може страждати від недостатньої узгодженості дій.

- Гібридна модель поєднує елементи централізованої та децентралізованої моделей, забезпечуючи баланс між контролем та гнучкістю. Гібридна модель демонструє найвищу ефективність у довгострокових програмах підтримки ветеранів.
- Мережева модель базується на горизонтальних зв'язках між волонтерами та організаціями, без чіткої ієрархії. Така модель сприяє інноваційності та швидкому обміну інформацією, проте може страждати від недостатньої структурованості. Аналіз практик волонтерських організацій свідчить, що мережева модель є найбільш адаптивною до змінних умов.

Ефективність різних моделей координації суттєво залежить від масштабу волонтерської ініціативи, типу надаваної допомоги та локального контексту, адже оптимальний вибір моделі координації може підвищити ефективність волонтерської допомоги на 30-45%.

Цифровізація процесу волонтерської допомоги представляє значний потенціал для підвищення її ефективності та доступності. Аналіз сучасних тенденцій та досліджень дозволяє виділити такі напрямки цифрової трансформації:

- Автоматизація процесів планування та координації. Дослідження демонструють, що використання спеціалізованого програмного забезпечення для координації волонтерської діяльності підвищує ефективність виконання завдань на 42%.
- Створення інтегрованих баз даних. Розробка цифрових систем для збору, зберігання та аналізу інформації про потреби ветеранів та можливості волонтерів сприяє більш точному визначенню пріоритетів та розподілу ресурсів.
- Впровадження онлайн-платформ для комунікації. Використання цифрових комунікаційних платформ збільшує охоплення цільової

аудиторії на 58% та підвищує рівень задоволеності наданою допомогою.

- Розробка мобільних додатків. Мобільні технології відкривають нові можливості для організації волонтерської допомоги, забезпечуючи оперативність, доступність та персоналізацію. Аналіз досвіду використання мобільних додатків у волонтерській сфері свідчить, що їх впровадження дозволяє скоротити час реагування на запити ветеранів до 70%.
- Аналітика даних та прогнозування потреб. використання предиктивної аналітики підвищує точність визначення потреб на 45%.
- Інтеграція з соціальними мережами та месенджерами. Забезпечення взаємодії цифрових інструментів волонтерської допомоги з популярними каналами комунікації підвищує їх доступність та зручність використання.

Варто зазначити, що цифрова трансформація процесу волонтерської допомоги стикається з рядом викликів, таких як нерівний доступ до цифрових технологій, проблеми захисту персональних даних, необхідність навчання користувачів, адже близько 30% ветеранів мають обмежений доступ до цифрових технологій або недостатні навички їх використання.

## 1.4 Специфіка розробки мобільних додатків соціального спрямування

Розробка мобільних додатків соціального спрямування характеризується низкою особливостей, що відрізняють їх від комерційних рішень та вимагають специфічного підходу. Аналіз даної проблематики дозволяє виявити ключові методологічні принципи та практичні аспекти, що мають бути враховані при створенні мобільного додатку для керування процесом волонтерської допомоги ветеранам.

Дослідження методологічних підходів до розробки соціально орієнтованих додатків виявляє їх суттєву відмінність від традиційних комерційних проектів.

Згідно з дослідженням [11], такі додатки характеризуються специфічною методологією, що базується на наступних принципах:

- Орієнтація на соціальний вплив. Первинна мета – досягнення позитивних соціальних змін, а не фінансовий прибуток. За даними, проекти з чіткою соціальною орієнтацією демонструють на 43% вищий рівень адаптації серед цільової аудиторії.
- Інклюзивний дизайн-підхід. Залучення різних категорій потенційних користувачів на всіх етапах проектування підвищує рівень задоволеності на 37%.
- Гнучкі ітеративні методології. Дослідження [12] свідчить, що використання гнучких методологій (Agile, Lean) у соціальних проектах підвищує їх успішність на 32%.
- Людиноцентричний підхід до проектування. Цей підхід передбачає фокусування на потребах, обмеженнях та контексті користувачів. Додатки, розроблені з використанням людиноцентричного підходу, демонструють на 47% вищу користувацьку активність.
- Кооперативна розробка. Співпраця з кінцевими користувачами на етапі розробки підвищує ефективність готового рішення на 40%.

Методологічні підходи до розробки соціально орієнтованих додатків також характеризуються специфічним життєвим циклом проекту. Такі проекти мають розширену фазу дослідження потреб користувачів та більш тривалий період тестування з цільовою аудиторією.

Розробка мобільного додатку для допомоги ветеранам передбачає створення інтерфейсів для різних категорій користувачів із специфічними потребами:

- Адаптивність інтерфейсу до потреб користувачів з різним рівнем цифрової грамотності. Дослідження демонструють, що серед ветеранів рівень цифрової грамотності значно варіюється, при цьому 37% мають низький рівень володіння мобільними технологіями. Адаптивний

інтерфейс, що пропонує різні режими складності, підвищує зручність використання додатку для користувачів різних категорій.

- Мінімізація когнітивного навантаження. Ветерани з ПТСР демонструють підвищену чутливість до інформаційного перевантаження, тому інтерфейс, що характеризується простотою та логічністю структури, мінімізує когнітивне навантаження та підвищує зручність використання.
- Емоційно-нейтральний дизайн. Певні елементи дизайну можуть провокувати негативні реакції у ветеранів з психологічними травмами.
- Диференційовані інтерфейси для різних ролей користувачів. Аналіз існуючих рішень демонструє необхідність розробки специфічних інтерфейсів для різних категорій користувачів (ветерани, волонтери, координатори, адміністратори). Рольова диференціація інтерфейсів підвищує ефективність використання додатку на 35%.

Практичне застосування цих принципів при проектуванні інтерфейсів мобільного додатку для керування процесом волонтерської допомоги ветеранам дозволяє забезпечити його доступність та ефективність для всіх категорій користувачів.

Питання безпеки даних та конфіденційності набувають особливої значущості у контексті розробки мобільних додатків для вразливих соціальних груп, зокрема ветеранів. Аналіз наукових досліджень та міжнародних стандартів дозволяє виділити наступні ключові аспекти:

1. Захист персональних даних та медичної інформації. Мобільні додатки для ветеранів часто містять чутливу інформацію, що потребує підвищеного рівня захисту. 78% ветеранів виражають занепокоєння щодо безпеки їхніх персональних даних у цифрових системах. Впровадження багаторівневих систем шифрування та анонімізації даних є критично важливим для забезпечення конфіденційності.
2. Диференційований доступ до інформації. Система управління правами доступу повинна забезпечувати чітке розмежування можливостей різних

категорій користувачів. 32% витоків даних у соціальних додатках пов'язані з неправильною конфігурацією прав доступу.

3. Прозорість політики конфіденційності. Користувачі мають бути чітко поінформовані про те, які дані збираються, як вони використовуються та з ким можуть бути поділені.
4. Принцип мінімізації даних. Згідно з цим принципом, додаток повинен збирати лише ті дані, які безпосередньо необхідні для його функціонування.
5. Безпека комунікаційних каналів. Захист даних під час їх передачі між клієнтською та серверною частинами додатку є критично важливим. За даними дослідження [13], 49% витоків даних у додатках соціального спрямування пов'язані з незахищеними комунікаційними каналами.
6. Відповідність міжнародним стандартам та нормативним вимогам. Розробка мобільного додатку для ветеранів повинна враховувати вимоги таких регуляторних документів, як GDPR [14], HIPAA [15] та інші локальні нормативні акти.

Імплементація комплексного підходу до забезпечення безпеки даних та конфіденційності є необхідною умовою для створення надійного та довіреного мобільного додатку для керування процесом волонтерської допомоги ветеранам.

### 1.5 Технологічні платформи для розробки Android-додатку

Створення мобільного додатку для керування процесом волонтерської допомоги ветеранам потребує обґрунтованого вибору технологічних платформ, що забезпечуватимуть необхідну функціональність, продуктивність та надійність рішення. При здійсненні аналізу сучасних технологій та інструментів розробки Android-додатків, можемо визначити оптимальний стек технологій для реалізації проекту з урахуванням його специфіки.

Вибір мови програмування є фундаментальним рішенням, що визначає подальші можливості розробки та підтримки мобільного додатку. Сучасна

розробка Android-додатків передбачає використання однієї з кількох основних мов програмування, кожна з яких має свої переваги та обмеження:

- Kotlin демонструє значні переваги у контексті розробки соціально орієнтованих додатків. За даними дослідження Android`s Kotlin Approach (2024) [16], 70% професійних розробників Android використовують Kotlin як основну мову програмування.
- Java залишається поширеною мовою для Android-розробки, що має велику спільноту підтримки та перевірену часом стабільність. Проте, згідно з дослідженням, розробка на Java потребує на 27% більше часу порівняно з Kotlin для реалізації тотожного функціоналу.
- Dart з Flutter представляє крос-платформову альтернативу, що дозволяє розробляти додатки одночасно для Android та iOS. Проте, нативні Kotlin-додатки демонструють на 15-20% кращу продуктивність та більш природну інтеграцію з системними компонентами Android.

Архітектурний підхід визначає структурну організацію компонентів додатку, впливаючи на його надійність, масштабованість та зручність супроводу. Аналіз сучасних архітектурних парадигм дозволяє виокремити кілька релевантних підходів:

- MVVM (Model-View-ViewModel) [17] забезпечує чітке розділення логіки представлення, бізнес-логіки та інтерфейсу користувача. Інтеграція MVVM з компонентами Android Architecture Components дозволяє ефективно управляти життєвим циклом компонентів додатку.
- MVP (Model-View-Presenter) [18] демонструє високий рівень розділення відповідальності, проте потребує на 22% більше шаблонного коду порівняно з MVVM.
- MVI (Model-View-Intent) [19] представляє односпрямований потік даних, що забезпечує передбачуваність стану додатку. За даними, MVI зменшує кількість помилок, пов'язаних з управлінням станом, на 41%, проте збільшує складність коду на 18% порівняно з MVVM.

- Clean Architecture [20] забезпечує організацію коду у вигляді концентричних шарів з чітко визначеними залежностями, що спрямовані від зовнішніх шарів до внутрішніх.
- Single Activity Architecture [21] передбачає використання однієї активності з множиною фрагментів, що оптимізує управління життєвим циклом компонентів та покращує навігацію.

Оптимальним архітектурним рішенням для розробки додатку керування волонтерською допомогою ветеранам видається комбінація MVVM та Clean Architecture з використанням Single Activity підходу [22]. Це забезпечить чітке розділення відповідальності між компонентами, високу здатність до тестування коду та гнучкість при подальшому розвитку додатку.

Сучасна розробка Android-додатків спирається на широкий спектр фреймворків та бібліотек, що дозволяють оптимізувати процес розробки та забезпечити високу якість кінцевого продукту. На основі аналізу функціональних вимог до додатку керування процесом волонтерської допомоги ветеранам, визначено наступні ключові компоненти технологічного стеку:

1. Jetpack представляє набір бібліотек, що забезпечують дотримання найкращих практик розробки Android-додатків. Найбільш релевантними компонентами Jetpack [23] для розроблюваного додатку є:
    - LiveData – компонент для реалізації шаблону спостерігача з урахуванням життєвого циклу.
    - Room – абстракція над SQLite, що забезпечує безпечний доступ до бази даних та переваги компіляційної перевірки SQL-запитів.
    - Navigation Component – бібліотека для управління навігацією у додатку та підвищує зручність навігації для користувачів на 28%.
2. Kotlin Coroutines [24] надає можливість асинхронного програмування в неблокуючому стилі, що є критично важливим для додатків з інтенсивним мережевим обміном.
3. Hilt [25] – бібліотека для впровадження залежностей, що спрощує архітектуру додатку та підвищує тестування коду. Використання Hilt

зменшує кількість шаблонного коду на 62% порівняно з ручною реалізацією впровадження залежностей.

4. Retrofit [26] – типобезпечний HTTP-клієнт для Android та Java, що спрощує взаємодію з RESTful API. Retrofit забезпечує на 35% меншу кількість помилок при роботі з мережевими запитами порівняно з використанням нативних компонентів.

Для ефективної координації волонтерської допомоги критично важливим є застосування геолокаційних сервісів, що дозволяють оптимізувати логістику та забезпечити адресну підтримку ветеранів. Аналіз існуючих рішень дозволяє виділити кілька релевантних технологій:

- Google Maps Platform надає широкий спектр геолокаційних сервісів, включаючи відображення карт, побудову маршрутів, геокодування та інші функції.
- Mapbox представляє гнучку альтернативу з розширеними можливостями кастомізації карт та оптимізованою продуктивністю на мобільних пристроях.
- HERE Maps пропонує розширені можливості для офлайн-використання карт, що особливо важливо для роботи у регіонах з обмеженим доступом до інтернету.

З огляду на специфіку додатку, оптимальним є використання Mapbox API, який поєднує продуктивність, точність і гнучку кастомізацію. Його інтеграція забезпечить ключові функції – відображення розташування ветеранів і волонтерів та геокодування адрес.

Забезпечення надійної синхронізації даних, автентифікації користувачів та обміну повідомленнями у режимі реального часу потребує використання сучасних хмарних та бекенд-рішень. Аналіз доступних технологій дозволяє визначити наступні компоненти технологічного стеку:

- Firebase Realtime Database забезпечує синхронізацію даних у режимі реального часу, що є критично важливим для координації волонтерської діяльності.

- Firebase Authentication надає готові механізми автентифікації користувачів з підтримкою різних провайдерів (електронна пошта, телефон, соціальні мережі). Firebase Authentication зменшує ризики порушення безпеки на 47% порівняно з власними реалізаціями механізмів автентифікації.
- Firebase Cloud Storage забезпечує можливість зберігання та обміну файлами, що необхідно для функціонування додатку.
- Firebase Cloud Functions дозволяє реалізувати серверну логіку без необхідності утримання власної інфраструктури.
- Firebase Cloud Messaging забезпечує механізм доставки push-повідомлень, що є критично важливим для оперативного інформування користувачів.

Використання екосистеми Firebase [27] як основного бекенд-рішення для додатку керування волонтерською допомогою ветеранам забезпечує оптимальний баланс між функціональністю, надійністю та вартістю для проектів даного типу.

### 1.6 Формування вимог до мобільного додатку

На основі аналізу предметної області визначено ключові вимоги до мобільного додатку для координації волонтерської допомоги ветеранам. Цей етап критично важливий для технічної реалізації та відповідності продукту потребам користувачів.

Функціональні вимоги описують конкретні можливості та функції, які рекомендовано виконувати мобільному додатку:

- Реєстрація та автентифікація користувачів: підтримка входу через пошту, номер телефона, соціальні мережі; розмежування ролей (ветерани, волонтери); редагування профілю.
- Управління запитами допомоги: створення, редагування, категоризація (медична, матеріальна); вкладення файлів; сповіщення про зміни.

- Система пошуку та фільтрації: пошук за ролями, навичками, геолокацією.
- Система координації та управління задачами: відстеження прогресу виконання, підбір волонтерів за критеріями.
- Інтеграція з зовнішніми ресурсами: взаємодія з державними базами, карта об'єктів допомоги.

Нефункціональні вимоги визначають якісні характеристики системи, які забезпечують ефективне виконання функціональних вимог:

- Продуктивність: швидкий відгук, стабільна робота при слабкому інтернеті, оптимізація ресурсів.
- Надійність та стабільність: мінімум збоїв, автоматичне збереження, підтримка різних версій Android.
- Безпека та захист даних: шифрування, захист доступу, відповідність законодавству.
- Масштабованість: підтримка зростання кількості користувачів та даних.
- Зручність використання: простий інтерфейс, адаптивність, доступність.
- Підтримка та оновлення: автоматичне оновлення, зворотний зв’язок, діагностика.

Базуючись на рекомендованому стеку технологій, можна сформулювати такі технічні вимоги до розробки:

1. Платформа та мова програмування: Android, мова Kotlin, API 21 (Android 5.0 Lollipop) і вище.
2. Архітектурні патерни: MVVM, Clean Architecture, Single Activity, компонентність.
3. Технології бекенду та сховища даних: Firebase (Realtime DB, Auth, Storage, Functions, Cloud Messaging), SharedPreferences.
4. Асинхронна обробка: Kotlin Coroutines, LiveData.
5. Геолокаційні сервіси: Mapbox API, визначення місцезнаходження.
6. Впровадження залежностей: Hilt, структурована DI-архітектура.
7. Інтерфейс користувача: XML, Material Design.

Таким чином, сформована система вимог створює надійну основу для подальшої розробки мобільного додатку, який відповідатиме практичним потребам волонтерів та ветеранів, забезпечуючи ефективну координацію процесів надання допомоги.

**Висновки до першого розділу**

У першому розділі проведено всебічний аналіз предметної області та існуючих рішень у сфері волонтерської допомоги ветеранам за допомогою мобільних технологій. Дослідження охопило ключові аспекти соціальної адаптації ветеранів, сучасні цифрові рішення та можливості для створення спеціалізованого додатку.

Встановлено, що проблема реінтеграції ветеранів залишається актуальною і потребує комплексного підходу із залученням волонтерів. Існуючі механізми підтримки часто неефективні та потребують удосконалення через цифрові інструменти.

Аналіз наявних платформ показав, що спеціалізованих рішень для ветеранів обмаль, а існуючі мають обмежений функціонал і не враховують специфіку потреб. Дослідження організації волонтерської допомоги дозволило визначити процеси, які потребують цифровізації: реєстрація потреб, пошук волонтерів, координація, моніторинг та звітність.

Вивчено принципи проєктування соціальних мобільних додатків: інклюзивність, доступність, безпека та орієнтація на користувача. Розглянуто оптимальний технологічний стек інші сучасні інструменти розробки. Результати аналізу створюють міцну основу для розробки мобільного додатку, що ефективно відповідатиме реальним потребам ветеранів і волонтерів.

# РОЗДІЛ 2
# РОЗРОБКА КОНЦЕПЦІЇ ТА АЛГОРИТМІВ ДОДАТКУ

## 2.1 Постановка задачі та загальна концепція

Розробка мобільного додатку для керування процесом волонтерської допомоги ветеранам обумовлена необхідністю створення інтерактивної платформи, яка дозволить оптимізувати комунікацію між ветеранами та волонтерами, забезпечити швидку та адресну допомогу в режимі реального часу. Дане дослідження спрямоване на вирішення актуальної соціальної проблеми через впровадження сучасних технологій мобільної розробки.

Основним завданням є розробка функціонального додатку для платформи Android, який забезпечить ефективну систему взаємодії між ветеранами та волонтерами. Реалізація даного завдання передбачає дотримання наступних функціональних вимог:

- Формування двосторонньої моделі взаємодії, що забезпечує реєстрацію та авторизацію двох категорій користувачів: ветеранів, які потребують допомоги, та волонтерів, які готові її надати.
- Інтеграція технологій геолокації для ефективного пошуку та підбору волонтерів, які знаходяться найближче до ветеранів, що потребують допомоги.
- Створення системи публікації та відгуку на запити про допомогу з можливістю визначення категорій типу допомоги.
- Забезпечення захисту персональних даних та конфіденційності користувачів.

Функціональна модель розроблюваного додатку являє собою структуру взаємопов'язаних модулів, кожен з яких відповідає за реалізацію певного аспекту роботи системи. Основними компонентами функціональної моделі є:

- Модуль автентифікації та реєстрації – забезпечує процес створення облікових записів та авторизації користувачів з розмежуванням ролей "Ветеран" та "Волонтер".

- Модуль створення та керування запитами про допомогу – дозволяє ветеранам формулювати потреби в допомозі та категоризувати їх.
- Система пошуку та підбору – забезпечує автоматичний підбір відповідних запитів для волонтерів на основі їх місцезнаходження та обраних фільтрів категорій.
- Модуль геолокації – відображає на інтерактивній карті розташування запитів про допомогу та активних волонтерів.
- Модуль налаштування користувача – надає можливість налаштування та редагування наявних персональних даних.
- Модуль сповіщень – забезпечує інформування користувачів про зміну стану запиту допомоги в режимі реального часу.

Процес взаємодії користувачів з розроблюваним додатком побудований на принципах інтуїтивності та мінімізації кількості дій для досягнення результату. Основний сценарій використання додатку включає наступні етапи:

1. Реєстрація та авторизація. Користувач обирає роль (ветеран або волонтер). Він проходить процедуру реєстрації з верифікацією особи та здійснює вхід до системи, використовуючи створені облікові дані.
2. Для ветеранів. Користувач створює запит про допомогу, зазначаючи категорію та опис проблеми. Він має можливість прикріпити фото до запиту, переглядати статус запиту в режимі реального часу.
3. Для волонтерів. Користувач переглядає доступні запити про допомогу на карті, фільтрують запити за категоріями, приймають запити до виконання та комунікують з ветеранами.
4. Спільні функції. Користувачі можуть налаштовувати профіль, керувати сповіщеннями, а також отримувати доступ до інформаційних ресурсів та контактів служб підтримки.

Взаємодія користувача з додатком базується на принципах Single Activity Architecture, що забезпечує плавність переходів між екранами та збереження стану додатку. Навігаційна модель додатку враховує особливості кожної ролі, надаючи доступ до відповідних функцій та інструментів.

В результаті розробки мобільного додатку для керування процесом волонтерської допомоги ветеранам очікується створення стабільної, безпечної та зручної платформи, яка забезпечить:

- Спрощення процесу пошуку та надання волонтерської допомоги ветеранам.
- Підвищення ефективності розподілу волонтерських ресурсів.
- Скорочення часу реагування на запити про допомогу.
- Створення стійкої спільноти взаємодопомоги.

Додаток розробляється з урахуванням можливості подальшого масштабування та розширення функціональності, а також адаптації для використання в інших сферах соціальної допомоги.

## 2.2 Модель користувацької взаємодії

Проектування ефективної моделі користувацької взаємодії становить невід'ємну частину розробки мобільного додатку для керування процесом волонтерської допомоги ветеранам. Дана модель визначає ролі користувачів, сценарії використання, логіку поведінки системи та забезпечує фундамент для створення інтуїтивного інтерфейсу.

У розроблюваному додатку ідентифіковано дві основні ролі користувачів з відмінними потребами та функціональними можливостями:

- "Ветеран" – особа, яка потребує волонтерської допомоги, формує запити через додаток, відстежує їх статус та комунікує з волонтерами.
- "Волонтер" – особа, яка надає допомогу ветеранам, здійснює пошук запитів, визначає свою доступність та встановлює комунікацію з ветеранами щодо деталей виконання завдань.

Адміністративні функції реалізуються через веб-інтерфейс Firebase, що знаходиться поза межами розроблюваного мобільного рішення, тому окрема роль адміністратора системи в рамках додатку не передбачена.

Формалізація функціональних вимог та створення моделі поведінки системи здійснюється через розробку комплексу сценаріїв використання, що охоплюють основні аспекти взаємодії користувачів з додатком. На рисунку 2.1 можна побачити UML-діаграму варіантів використання сценаріїв експертної системи:


Рисунок 2.1 – UML-діаграма використання сценаріїв експертної системи

Загальні сценарії використання включають процеси реєстрації у системі, авторизації користувачів та управління профілем. При реєстрації користувач обирає роль, вводить базову інформацію, проходить верифікацію електронної пошти та заповнює розширену інформацію. Під час авторизації система перевіряє введені дані та надає доступ до функцій відповідно до ролі користувача. Управління профілем дозволяє користувачу змінювати особисту інформацію та налаштування додатку.

Для ветеранів розроблено специфічні сценарії використання, такі як створення запиту про допомогу, відстеження статусу запиту та можливість зв’язатись з волонтером. При створенні запиту ветеран заповнює форму з назвою, описом та категорією проблеми та може додати фотографії проблеми.

Відстеження статусу дозволяє ветерану переглядати поточний стан запиту та дані волонтера, що відгукнувся.

Волонтери мають доступ до сценаріїв пошуку запитів про допомогу, прийняття запиту до виконання та зв'язку з ветераном. Пошук запитів здійснюється через інтерактивну карту з можливістю застосування фільтрів. Прийняття запиту передбачає перегляд деталей, підтвердження готовності надати допомогу та зміну статусу запиту. Комунікація з ветераном забезпечує узгодження деталей та обмін важливою інформацією.

Для моделювання взаємодії між компонентами системи та візуалізації послідовності обміну даними розроблено діаграму послідовності. На рисунку 2.2 можна побачити комунікацію між клієнтською частиною додатку та сервісами Firebase при виконанні ключових операцій:

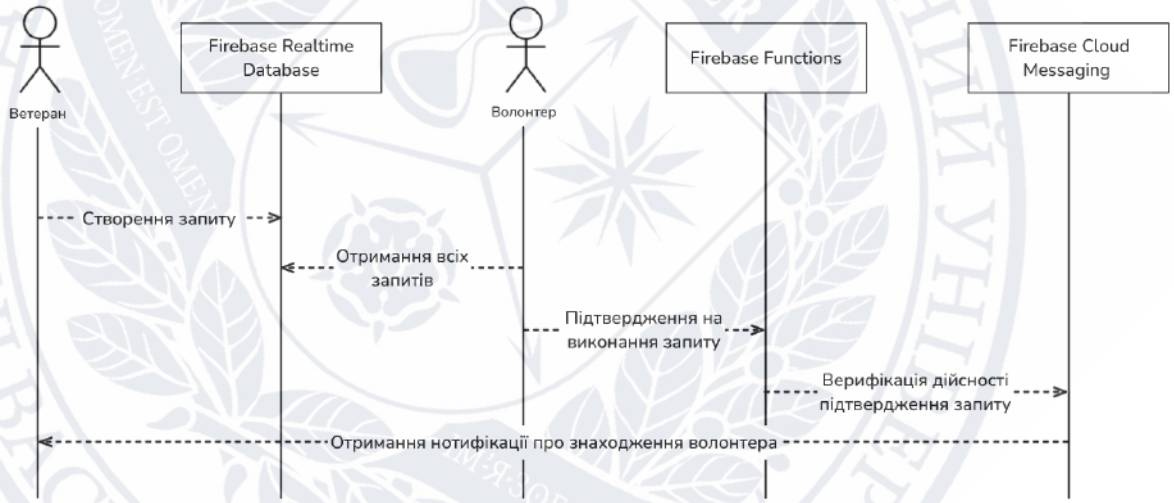


Рисунок 2.2 – UML-діаграма комунікації клієнтської частинами додатку та сервісами Firebase

Діаграма послідовності для процесу створення та прийняття запиту демонструє взаємодію між інтерфейсом користувача, локальною базою даних, сервісами Firebase та системою сповіщень. Вона відображає потоки даних від моменту формування запиту ветераном до отримання повідомлення про прийняття запиту волонтером, включаючи проміжні етапи збереження даних, валідації та сповіщення.

Запропонована модель користувацької взаємодії спрямована на забезпечення максимальної доступності та ефективності процесу надання волонтерської допомоги ветеранам. Чітке розмежування ролей забезпечує спеціалізований інтерфейс та функціональність для кожної категорії користувачів. Інтуїтивність навігації мінімізує кількість дій для досягнення результату завдяки логічній структурі екранів та переходів.

Таким чином, модель користувацької взаємодії створює фундамент для подальшої розробки інтерфейсу та функціональних компонентів додатку, визначаючи структуру взаємодії між різними категоріями користувачів та формуючи базис для алгоритмічної реалізації основних бізнес-процесів системи.

## 2.3 Архітектура додатку

Архітектура програмного забезпечення є одним із ключових аспектів розробки мобільного додатку для керування процесом волонтерської допомоги ветеранам. Правильно спроектована архітектура забезпечує надійність, розширюваність та зручність обслуговування додатку в довгостроковій перспективі. У контексті даної розробки обрано комбінований підхід, що поєднує принципи MVVM та Clean Architecture з використанням Single Activity Architecture. На рисунку 2.3 можна побачити схему архітектури додатку, яка відображає взаємозв'язки між основними шарами та їх компонентами:

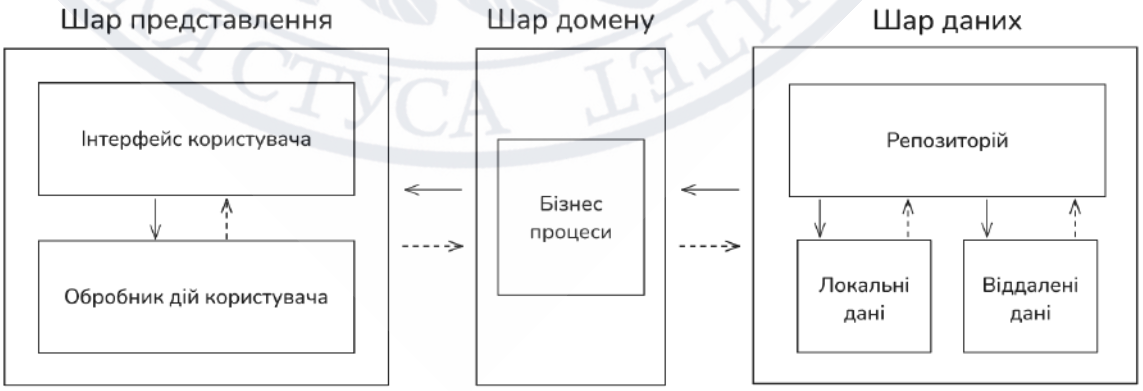


Рисунок 2.3 – Схема архітектури додатку

Розроблена архітектура додатку базується на принципах розділення відповідальності та інверсії залежностей, що дозволяє досягти високого рівня модульності, тестованості та масштабованості системи. Архітектура складається з трьох основних шарів:

- Шар представлення (Presentation Layer) [28] відповідає за відображення даних та взаємодію з користувачем. Цей шар включає всі компоненти користувацького інтерфейсу та відповідає за правильне відображення даних, а також за перехоплення та обробку користувацьких дій.
- Шар домену (Domain Layer) [29] містить бізнес-логіку та правила застосування. Тут зосереджені всі основні бізнес-процеси додатку, такі як обробка запитів на допомогу, співставлення волонтерів із ветеранами, валідація даних тощо. Також цей шар є незалежним від зовнішніх технологій та фреймворків.
- Шар даних (Data Layer) [30] забезпечує доступ до даних з різних джерел. Він відповідає за взаємодію з локальними та віддаленими сховищами даних, такими як Firebase Realtime Database, Firebase Auth, Firebase Storage, а також локальне сховище на пристрої.

Кожен із зазначених шарів має свої специфічні компоненти, які виконують визначені функції та взаємодіють між собою згідно з принципами Clean Architecture. Важливо зазначити, що залежності спрямовані від зовнішніх шарів до внутрішніх, забезпечуючи незалежність доменного шару від конкретних реалізацій інтерфейсу та сховищ даних.

Шар представлення реалізовано з використанням шаблону MVVM, що складається з декількох важливих компонентів:

- View: представлений фрагментами та XML-макетами, відповідає за відображення інтерфейсу та перенаправлення взаємодій до ViewModel. Фрагменти не містять бізнес-логіки, а лише відображають дані.
- ViewModel: сполучна ланка між View та бізнес-логікою, обробляє події від інтерфейсу та надає дані через LiveData. Зберігає стан при зміні конфігурації пристрою.

- Односпрямований потік даних: View реагує на зміни у LiveData, а ViewModel отримує події від View та взаємодіє з Use Cases, що спрощує налагодження.
- Single Activity Architecture: організує навігацію через єдину активність, що керує фрагментами, покращуючи продуктивність та зменшуючи споживання ресурсів.

Шар домену є центральним елементом архітектури та містить ключові компоненти, що визначають основну функціональність додатку:

- Use Cases (Interactors): класи, що інкапсулюють бізнес-процеси, відповідаючи за конкретні операції, такі як "Отримання списку запитів" або "Створення нового запиту". Вони взаємодіють з репозиторіями для виконання операцій з даними та реалізують бізнес-правила, забезпечуючи незалежність від технологій інтерфейсу.
- Entities (сутності): бізнес-моделі, що представляють основні об'єкти, такі як "Запит на допомогу". Вони містять дані та базову логіку, не залежать від зовнішніх фреймворків і є ядром доменної моделі.
- Repository Interfaces: визначають методи доступу до даних для Use Cases, абстрагуючи деталі зберігання. Це відповідає принципу інверсії залежностей, де високорівневі модулі залежать від абстракцій.

Шар домену є незалежним від конкретних технологій та фреймворків, що робить його ізольованим від змін у зовнішніх шарах та спрощує тестування. Ця незалежність забезпечує гнучкість при розробці та можливість заміни технологій без зміни бізнес-логіки додатку.

Шар даних відповідає за забезпечення доступу до різних джерел даних і містить декілька важливих компонентів:

- Repositories Implementation: класи, що реалізують інтерфейси репозиторіїв, координуючи роботу з джерелами даних. Вони забезпечують логіку отримання, збереження та оновлення даних, вибираючи джерело в залежності від стану додатку, включаючи офлайн-режим з використанням кешованих даних.

- Data Sources: компоненти, що взаємодіють з конкретними джерелами даних. Наприклад, Firebase Realtime Database для зберігання основних даних, Firebase Auth для автентифікації, Firebase Storage для зберігання файлів, а локальне сховище (SharedPreferences) для налаштувань та кешування.
- DTOs (Data Transfer Objects): об'єкти для передачі даних між шаром даних та зовнішніми сервісами, що відображають структуру даних API або бази даних і використовуються для серіалізації та десеріалізації.
- Mappers: класи для перетворення між DTOs та доменними моделями, ізолюючи доменний шар від деталей реалізації сховищ даних і формату даних.

Для реалізації інверсії залежностей та забезпечення слабкого зв'язування між компонентами використовується Hilt – бібліотека для ін'єкції залежностей, побудована на основі Dagger [31]. Використання Hilt дозволяє зменшити обсяг шаблонного коду для створення та управління залежностями, а також покращує тестованість додатку, оскільки дозволяє легко підміняти реальні імплементації компонентів на тестові заглушки.

Враховуючи, що додаток значною мірою покладається на сервіси Firebase, архітектура передбачає спеціальні компоненти для взаємодії з Firebase:

- Firebase Auth Service: керує автентифікацією користувачів, забезпечуючи реєстрацію, вхід та відновлення паролю, а також зберігає інформацію про авторизованого користувача та його роль.
- Firebase Database Service: забезпечує доступ до Realtime Database, відповідаючи за читання, запис та підписку на зміни даних в реальному часі, що важливо для відстеження статусу запитів на допомогу.
- Firebase Storage Service: дозволяє завантажувати та отримувати фотографії та інші файли, пов'язані із запитами на допомогу або профілями користувачів.

- Firebase Cloud Messaging Service: обробляє пуш-повідомлення, забезпечуючи отримання сповіщень про нові запити, зміни статусу та інші важливі події.
- Firebase Functions Service: взаємодіє з хмарними функціями для виконання серверної логіки, такої як розсилка повідомлень та інтеграція з зовнішніми сервісами.

Ці сервіси інкапсулюють взаємодію з відповідними API Firebase та надають більш високорівневий інтерфейс для використання у репозиторіях. Такий підхід спрощує заміну або модифікацію конкретних реалізацій без зміни загальної архітектури додатку. Кожен сервіс також забезпечує кешування даних та обробку помилок, що покращує стабільність та продуктивність додатку.

Для обробки асинхронних операцій, таких як мережеві запити та обчислення, використовуються Kotlin Coroutines, що дозволяє уникнути блокування головного потоку і забезпечує реактивний підхід до обробки даних. Такий підхід забезпечує чітке розділення відповідальності та можливість обробки помилок на кожному етапі. Використання Kotlin Coroutines забезпечує більш читабельний та підтримуваний код порівняно з традиційними підходами, такими як Callbacks або RxJava, спрощуючи роботу з асинхронними операціями та покращуючи загальну продуктивність додатку.

### 2.4 Алгоритми роботи системи

Проектування алгоритмів функціонування мобільного додатку для керування процесом волонтерської допомоги ветеранам є найбільш відповідальним моментом у контексті розробки додатку. Особлива увага приділяється ключовим функціям, які забезпечують ефективну взаємодію між користувачами системи: пошуку волонтера, управлінню станами запиту та системі сповіщень про знайденого волонтера.

Алгоритм пошуку волонтера є однією з основних функціональних складових системи, що забезпечує встановлення зв'язку між ветераном, який

потребує допомоги та потенційним волонтером. Цей процес є однонаправленим, тобто ініціюється виключно ветераном через створення запиту на допомогу. На рисунку 2.4 можна побачити блок-схему алгоритму пошуку волонтера. На етапі проектування визначено наступну структуру алгоритму пошуку волонтера:

1. Спочатку ветеран ініціює запит через інтерфейс додатку, заповнюючи обов'язкові параметри, такі як назва, опис і категорія допомоги, а також може додати мультимедійні матеріали, зокрема фото проблеми. Система автоматично визначає геолокацію ветерана.
2. Далі відбувається процес валідації, який включає перевірку заповненості всіх обов'язкових полів, коректності прикріплених мультимедійних файлів та наявності координат місцезнаходження. У разі виявлення помилок користувач отримує інформативне повідомлення.
3. Після цього конструюється модель запиту, що включає генерацію унікального ідентифікатора та формування структурованого об'єкта даних з атрибутами, такими як ідентифікатор запиту, дані про ветерана, опис, категорія допомоги, геолокаційні координати, час створення, початковий статус "Очікування" та атрибут для ідентифікатора волонтера.
4. Наступним етапом є збереження даних, що передбачає передачу моделі запиту до Firebase Realtime Database та завантаження мультимедійних матеріалів до Firebase Storage, а також формування зв'язків між записами в базі даних і медіафайлами.
5. Система фільтрації та відображення автоматично оновлює списки запитів у волонтерській частині додатку, реалізуючи фільтрацію, яка показує лише запити в статусі "Очікування", та сортує їх за часовою міткою оновлення для відображення найновіших запитів.

Принциповою особливістю даного алгоритму є використання реактивного підходу до оновлення даних. На відміну від класичних систем із циклічним опитуванням сервера, запроектована система використовує Firebase Realtime Database, що забезпечує обробку повідомлень про зміни даних. Це дозволяє:

- Знизити мережевий трафік через відсутність зайвих запитів до сервера.

- Миттєво відображати нові запити на допомогу в інтерфейсі волонтерів.
- Зменшити споживання енергії мобільних пристроїв.
- Забезпечити актуальність даних у всіх користувачів системи в реальному часі.

Алгоритм проектується з орієнтацією на патерн "Спостерігача" (Observer) [32], що дозволяє автоматично відображати зміни в базі даних на клієнтських пристроях. Це значно спрощує архітектуру додатку, оскільки не потребує розробки додаткових механізмів синхронізації даних.

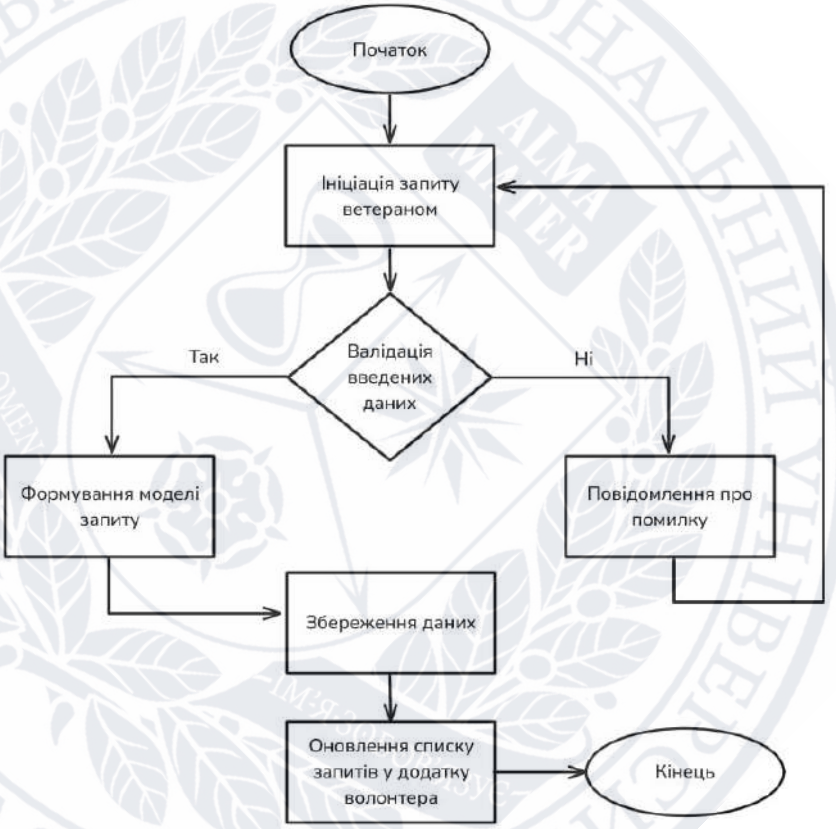


Рисунок 2.4 – Блок-схема алгоритму пошуку волонтера

Керування станами запиту є ключовим механізмом, який визначає логіку та послідовність усіх етапів процесу надання волонтерської допомоги – від моменту створення запиту до його повного виконання. Такий підхід дозволяє системі чітко відслідковувати прогрес кожного звернення, запобігаючи плутанині та дублюванню дій. Перехід між станами відбувається автоматично та відображає актуальний статус кожного запиту в режимі реального часу. Така структурованість забезпечує прозорість процесу, покращує комунікацію між

сторонами та дозволяє ефективніше управляти ресурсами. Основні стани запиту в системі можна пояснити та навести у вигляді таблиці 2.1:

Таблиця 2.1 Основні стани запиту додатку

| Стан | Опис |
|---|---|
| Очікування (Pending) | Початковий стан запиту після його створення |
| Активний (Active) | Стан запиту після прийняття його волонтером |
| Завершений (Completed) | Стан успішно виконаного запиту |
| Скасований (Canceled) | Стан скасованого запиту з будь-яких причин |

При проектуванні алгоритму керування станами запиту використовується патерн "Кінцевий автомат" (Finite State Machine) [33], що забезпечує структуроване управління життєвим циклом запиту. На рисунку 2.5 можна побачити схему, яка відображає послідовність переходів станів запиту в алгоритмі. Алгоритм передбачає кілька ключових транзакцій:

1. По-перше, транзакція створення запиту встановлює початковий стан "Очікування", фіксує часову мітку створення та не має призначеного волонтера. Далі, у транзакції активації запиту відбувається перехід до стану "Активний", записується ідентифікатор волонтера, фіксується час прийняття запиту, а також надається доступ до контактних даних ветерана.
2. У разі деактивації запиту, що відбувається через відмову волонтера, система повертається зі стану "Активний" до "Очікування", очищаючи дані про призначеного волонтера та повертаючи запит до пулу доступних для інших волонтерів.
3. Транзакція завершення запиту передбачає перехід до стану "Завершений", запис часової мітки завершення, видалення запиту з активних списків волонтерів та архівування його для статистичних цілей.
4. Окрім того, реалізується транзакція скасування запиту, що дозволяє перейти до стану "Скасований" з будь-якого іншого стану та повідомляючи всіх зацікавлених сторін.

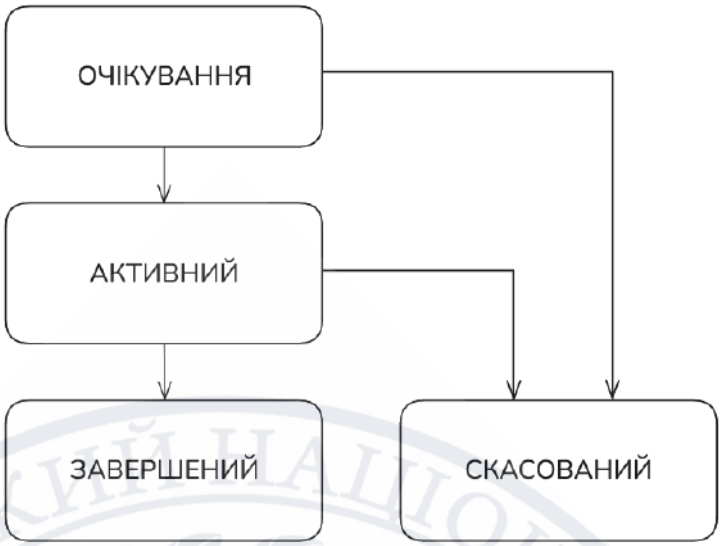


Рисунок 2.5 – Схема переходів станів запиту в алгоритмі зміни станів

Для забезпечення атомарності операцій і запобігання колізіям даних при одночасному доступі, алгоритм використовує транзакційні можливості Firebase Realtime Database, де кожна зміна стану реалізується як єдина транзакція, що гарантує цілісність даних і передбачуваність поведінки системи.

Алгоритм системи сповіщень є критичним компонентом додатку, що забезпечує своєчасне інформування користувачів про важливі події в режимі реального часу. На рисунку 2.6 можна побачити схему, яка відображає послідовність взаємодії компонентів системи під час виконання алгоритму сповіщення.

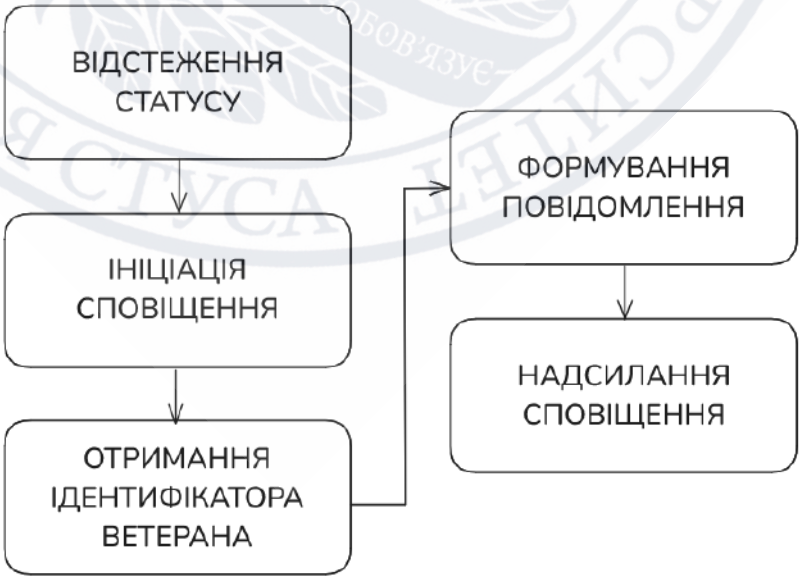


Рисунок 2.6 – Схема етапів алгоритму сповіщення

При проектуванні алгоритму системи сповіщень особлива увага приділяється розподіленню відповідальності між серверною та клієнтською частинами. Архітектура алгоритму передбачає:

- Алгоритм включає серверну частину, що реалізується через Firebase Cloud Functions, яка відповідає за моніторинг змін стану запитів у реальному часі, відбір подій для нотифікації користувачів, визначення цільового отримувача повідомлення, формування структурованого об'єкта сповіщення та передачу його до сервісу Firebase Cloud Messaging.
- Клієнтська частина, представлена Android-додатком, займається реєстрацією пристрою в системі Firebase Cloud Messaging, збереженням та оновленням унікального FCM-токена, обробкою вхідних повідомлень та їх відображенням для користувача, а також коректною обробкою повідомлень у різних станах додатку (активний, фоновий, закритий).

Проектування алгоритму також враховує потребу в мінімізації навантаження на пристрої користувачів, оскільки логіка формування та відправки сповіщень повністю делегується серверній частині. Це дозволяє зменшити енергоспоживання мобільних пристроїв, забезпечити надійність доставки повідомлень навіть при тимчасовій відсутності мережі та гарантувати отримання сповіщень, коли додаток неактивний. Саме обраний підхід до реалізації системи сповіщень має ряд переваг, а саме:

- Забезпечення асинхронної обробки подій без додаткового навантаження на клієнтський додаток.
- Гарантована доставка повідомлень навіть при неактивному стані додатку на пристрої користувача.
- Мінімальне використання ресурсів мобільного пристрою завдяки виконанню логіки на стороні сервера.
- Масштабованість рішення, що дозволяє обробляти велику кількість запитів одночасно.

У контексті загальної архітектури додатку, спроектовані алгоритми забезпечують ефективну взаємодію компонентів системи та надійне функціонування ключових бізнес-процесів. Використання Firebase як бекенд-платформи з функціями реального часу сприяє оптимізації обміну даними між користувачами системи та забезпечує високу доступність сервісу.

### 2.5 Моделювання даних

Ефективна організація та керування даними є ключовим аспектом розробки мобільного додатку для волонтерської допомоги ветеранам. При розробці мобільного додатку було прийнято рішення використовувати комбінацію технологій зберігання даних, що забезпечують оптимальну продуктивність, масштабованість та зручність використання:

1. Firebase Realtime Database використовується для зберігання основних даних додатку, що забезпечує їх миттєву синхронізацію між користувачами. Ця технологія має ряд важливих переваг:
    - Синхронізація даних у реальному часі, що дозволяє миттєво оновлювати інформацію про запити та користувачів без додаткового програмування механізмів синхронізації.
    - Високий рівень масштабованості та надійності, що підтверджується використанням цієї технології у великих проектах з мільйонами користувачів.
    - NoSQL-структура [34], що забезпечує гнучкість при зміні схеми даних без необхідності складних міграцій та оновлень бази даних.
    - Автоматичне резервне копіювання та відновлення даних, що забезпечує додатковий рівень надійності.
2. Firebase Storage призначено для зберігання файлів, таких як фотографії користувачів і зображення, прикріплені до запитів. Ця технологія характеризується наступними перевагами:

- Оптимізоване зберігання бінарних даних великого розміру, що зменшує навантаження на основну базу даних.
- Можливість контролю доступу до файлів на основі ролей користувачів, що забезпечує належний рівень захисту персональних даних.
- Автоматичне масштабування ресурсів відповідно до потреб додатку, що гарантує стабільну роботу при зростанні кількості користувачів.
- Вбудовані механізми оптимізації передачі даних через мережу, що зменшує використання інтернет-трафіку.

3. SharedPreferences [35] використовується для зберігання легких локальних налаштувань користувача, що забезпечує швидкий доступ до часто використовуваних даних. Головні переваги цієї технології:
   - Швидкий доступ до часто використовуваних даних без необхідності мережевих запитів.
   - Мінімальне використання ресурсів пристрою, що особливо важливо для мобільних додатків.
   - Зручність для зберігання простих користувацьких персоналізацій та налаштувань.

Порівняльний аналіз різних технологій зберігання даних показав, що обрана комбінація забезпечує оптимальний баланс між продуктивністю, надійністю та зручністю розробки. Так, наприклад, використання локальної SQLite бази даних було б менш ефективним для забезпечення синхронізації даних між різними пристроями, а використання традиційної серверної реляційної бази даних вимагало б значно більших витрат на розробку та підтримку серверної інфраструктури.

Структура бази даних Firebase Realtime Database розроблена відповідно до функціональних вимог додатку та специфіки волонтерської взаємодії. Оскільки Firebase Realtime Database використовує NoSQL-підхід, було застосовано принцип денормалізації даних для оптимізації швидкості доступу та зменшення кількості запитів до бази даних. На рисунку 2.7 можна побачити схему, яка

відображає зв'язки між сутностями в додатку. База даних організована у вигляді JSON-дерева з двома основними вузлами: requests та users.

Вузол users містить вичерпну інформацію про зареєстрованих користувачів додатку. Ключем кожного запису є унікальний ідентифікатор користувача (userId), який автоматично генерується Firebase Authentication при реєстрації, що забезпечує унікальність кожного запису та зручність доступу до інформації про конкретного користувача. Структура даних користувача розділена на логічні блоки для кращої організації та гнучкості. Такий підхід до структурування даних користувачів забезпечує ряд переваг:

- Ефективний доступ до інформації про користувача за його ідентифікатором.
- Можливість легко розширювати модель даних без необхідності зміни існуючої структури.
- Логічне групування пов'язаних даних, що спрощує управління та розуміння структури.
- Оптимізація кількості запитів до бази даних при отриманні інформації про користувача.
- Можливість встановлення детальних правил безпеки на рівні кожного поля.

У майбутньому, за необхідності, структура даних користувача може бути розширена додатковими блоками, такими як preferences (для зберігання налаштувань користувача), statistics (для зберігання статистичних даних про активність користувача) тощо, без порушення існуючої логіки роботи з даними.

Вузол requests містить детальну інформацію про запити на допомогу від ветеранів. Ключем кожного запису є унікальний ідентифікатор запиту (requestId), який генерується автоматично при створенні запиту. Структура даних запиту включає всю необхідну інформацію для забезпечення взаємодії між ветераном та волонтером. Описана структура даних запиту забезпечує:

- Збереження повної інформації про запит та його стан.
- Зручний доступ до контактної інформації обох сторін взаємодії.

- Геолокаційні дані для відображення запитів на карті.
- Можливість фільтрації запитів за різними критеріями.

Денормалізація даних у цій моделі, а саме зберігання даних про ветерана та волонтера безпосередньо в запиті – зменшує кількість звернень до бази, що підвищує продуктивність додатку, особливо на мобільних пристроях.

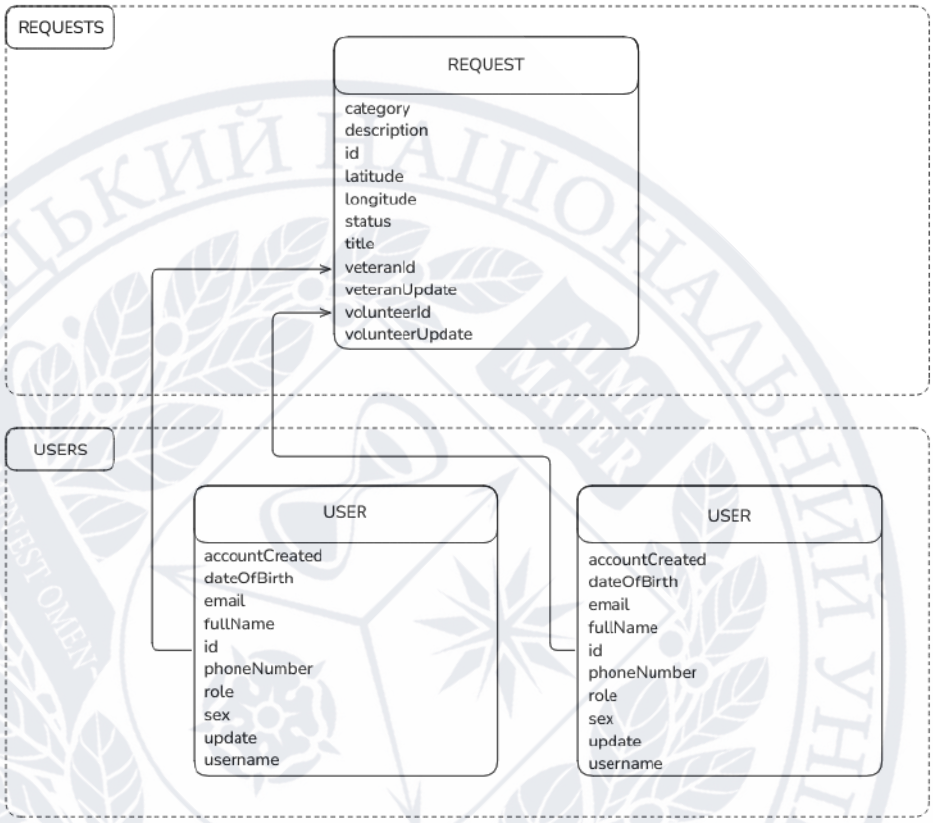


Рисунок 2.7 – Схема відношень між основними сутностями в Realtime Database

Firebase Storage використовується для зберігання файлів, пов'язаних із користувачами та запитами. Цей сервіс оптимізований для роботи з бінарними даними великого розміру та забезпечує ефективний механізм управління доступом до файлів. Структура зберігання організована відповідно до логічної структури додатку та спрощує процес доступу до файлів:

- images/ – директорія для зберігання фотографій користувачів. Використання ідентифікатора користувача як імені файлу забезпечує унікальність та спрощує процес оновлення зображення профілю. При зміні фотографії користувача новий файл просто замінює старий, зберігаючи те саме ім'я файлу.

- photos/ – директорія для зберігання зображень, прикріплених до запитів. Аналогічно до фотографій користувачів, використання ідентифікатора запиту як імені файлу забезпечує простий та ефективний доступ до зображень, пов'язаних з конкретним запитом.

Такий підхід до зберігання файлів забезпечує зручну ідентифікацію та доступ за ідентифікаторами без зайвих запитів до бази даних, спрощує оновлення й видалення, оптимізує використання ресурсів та зменшує складність коду завдяки стандартизованому іменуванню.

Для зберігання локальних налаштувань та персоналізованих даних користувача використовується SharedPreferences – легковагий механізм зберігання примітивних типів даних у форматі "ключ-значення". Цей підхід забезпечує швидкий доступ до часто використовуваних даних без необхідності мережевих запитів та мінімізує використання ресурсів пристрою.

## 2.6 Безпека та автентифікація

У розробці мобільного додатку для керування процесом волонтерської допомоги ветеранам питання безпеки та автентифікації займають пріоритетне місце, оскільки система оперує чутливими персональними даними користувачів. Реалізація механізмів захисту ґрунтується на використанні сучасних компонентів технологічного стеку проекту, зокрема Firebase Auth та інших інструментів екосистеми Firebase.

Основним компонентом безпеки додатку є Firebase Authentication, що надає готові рішення для надійної автентифікації користувачів. У додатку реалізовано наступні механізми автентифікації:

- Автентифікація за допомогою електронної пошти та пароля – базовий механізм автентифікації, що передбачає реєстрацію користувача із застосуванням унікальної електронної адреси та надійного пароля.

- Верифікація електронної пошти – після реєстрації користувач отримує лист з посиланням для підтвердження вказаної адреси електронної пошти, що забезпечує перевірку автентичності наданих даних.

Firebase Realtime Database використовується як основне сховище даних додатку та забезпечує захист інформації за допомогою наступних механізмів:

1. Правила безпеки Firebase [36] – детально визначені правила доступу до даних, що дозволяють контролювати читання та запис інформації на рівні окремих вузлів бази даних:
   - Ветерани мають доступ тільки до власних запитів про допомогу.
   - Волонтери можуть переглядати лише запити, до яких вони призначені.
   - Координатори мають розширений доступ до керування запитами в межах своєї зони відповідальності.
2. Валідація даних – перевірка вхідних даних на відповідність визначеним схемам та обмеженням як на клієнтській, так і на серверній частині.
3. Шифрування даних при передачі [37] – усі комунікації між додатком та Firebase Realtime Database захищені за допомогою протоколу SSL/TLS, що унеможливлює перехоплення даних.
4. Резервне копіювання даних – регулярне створення резервних копій бази даних для швидкого відновлення інформації у випадку її пошкодження або втрати.

Для забезпечення безпеки передачі даних між клієнтською частиною додатку та серверами Firebase впроваджено:

- SSL/TLS-шифрування – всі комунікації відбуваються через захищений HTTPS-протокол із застосуванням сучасних алгоритмів шифрування.
- Certificate Pinning – фіксація сертифікатів серверів Firebase для запобігання атакам типу "людина посередині".

Впроваджені механізми дозволяють ефективно захистити персональні дані користувачів, забезпечити цілісність інформації та запобігти несанкціонованому доступу до системи. Важливо відзначити, що система безпеки розроблена з урахуванням специфіки цільової аудиторії додатку – ветеранів та волонтерів, для яких конфіденційність та захист персональних даних мають особливе значення.

**Висновки до другого розділу**

У другому розділі роботи розроблено концептуальні та алгоритмічні засади мобільного додатку для керування процесом волонтерської допомоги ветеранам. Результати виконаної роботи свідчать про комплексний підхід до проектування системи.

Розроблено чітку функціональну модель додатку та створено модель користувацької взаємодії з визначеними ролями та сценаріями використання, що забезпечує ефективну взаємодію між ветеранами та волонтерами та гарантує інтуїтивно зрозумілий інтерфейс.

Обрано архітектурне рішення на основі MVVM та принципів Clean Architecture з використанням Single Activity підходу, що забезпечує розділення логіки та оптимальне використання ресурсів мобільного пристрою. Розроблено алгоритми ключових функцій системи: пошуку волонтерів, керування станами запитів та пуш-нотифікація підтвердження запиту.

Спроектовано структуру даних з використанням Firebase Realtime Database, що забезпечує обмін даними в реальному часі, та створено комплексну систему безпеки та автентифікації на базі Firebase Auth для захисту персональних даних користувачів.

Запропоновані рішення формують основу для практичної реалізації додатку та відзначаються адаптивністю і гнучкістю, що дозволить у майбутньому розширювати функціональність системи відповідно до потреб користувачів. Ключовою перевагою підходу є орієнтація на користувача при збереженні технічної ефективності.

# РОЗДІЛ 3
# РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ

## 3.1 Структура та компоненти програмного продукту

Розроблений мобільний додаток для керування процесом волонтерської допомоги має чітку структуру, побудовану згідно з принципами Clean Architecture та MVVM, що забезпечує максимальну ефективність, масштабованість та легкість подальшої підтримки проекту. Структура додатку передбачає поділ на модулі, які виконують конкретні завдання та взаємодіють між собою за допомогою спеціально визначених інтерфейсів.

Розглянемо загальну структуру проекту, яка відображає архітектурний підхід, обраний для розробки додатку. У проекті реалізовано принцип Single Activity Architecture, де використовується одна основна активність, а зміна екранів відбувається за допомогою фрагментів. Структура проекту складається з наступних основних модулів:

1. base – базовий модуль, що містить компоненти, які використовуються в усіх інших модулях:
    - data – містить репозиторії та їх реалізації.
    - domain – містить бізнес-логіку та моделі даних.
    - presentation – містить UI-компоненти, спільні для всіх екранів.
    - utils – утилітарні класи та функції.
2. di – модуль для впровадження залежностей (Dependency Injection):
    - ElpisApplication – точка входу для ініціалізації додатку та DI-контейнера.
    - DataModule – надає залежності для роботи з даними (репозиторії, джерела даних).
    - FirebaseModule – налаштовує та надає залежності для сервісів Firebase.

- SharedPreferencesModule – працює з SharedPreferences для зберігання простих даних.
- UtilsModule – містить утиліти та допоміжні функції для різних частин додатку.

3. features – модуль, що містить функціональні компоненти додатку:
    - home – головний екран.
    - login – екран авторизації.
    - settings – екран налаштувань.
    - splash – екран ознайомчий.
    - utils – допоміжні компоненти для всіх функціональних модулів.

Модуль base є фундаментом усього додатку і містить базові компоненти, які використовуються іншими модулями. Цей модуль організований за принципами Clean Architecture та поділений на підмодулі data, domain, presentation та utils.

У підмодулі data зосереджені всі класи, відповідальні за роботу з даними. Тут знаходяться реалізації репозиторіїв, які визначені у підмодулі domain. Всі репозиторії побудовані за однаковим принципом і забезпечують доступ до різних джерел даних (Firebase Realtime Database, SharedPreferences тощо).

Підмодуль domain містить інтерфейси репозиторіїв та класи бізнес-логіки (use cases), а також моделі даних. Кожен use case реалізує конкретний сценарій використання додатку і має єдину відповідальність, що відповідає принципу Single Responsibility.

Підмодуль presentation містить загальні UI-компоненти, які використовуються в усіх екранах додатку, наприклад базові класи для фрагментів та активності, спільні адаптери для списків та компоненти користувацького інтерфейсу.

Підмодуль utils містить утилітарні класи та функції, які використовуються у всьому додатку, такі як розширення функцій Kotlin для спрощення роботи з різними типами даних, утиліти для роботи з датами і часом, функції для форматування тексту та допоміжні класи для роботи з файлами та мережею.

У всіх функціональних модулях додатку застосовується однакова структура, що забезпечує уніфікованість підходу до розробки та полегшує підтримку проекту.

Модуль di відповідає за впровадження залежностей у проекті з використанням бібліотеки Hilt. Всі модулі Hilt побудовані за єдиним принципом: вони надають необхідні залежності для різних компонентів додатку. Такий підхід дозволяє централізовано керувати залежностями та забезпечує їх одноразову ініціалізацію, що оптимізує використання ресурсів.

Модуль features містить усі функціональні компоненти додатку, кожен з яких організований за принципами Clean Architecture та MVVM. Усі функціональні модулі мають однакову структуру:

- data – містить реалізації репозиторіїв та джерела даних.
- domain – містить інтерфейси репозиторіїв та класи бізнес-логіки.
- presentation – містить фрагменти, адаптери та viewmodel-класи.

Така уніфікована структура забезпечує консистентність коду та полегшує навігацію по проекту.

В підсумку було створено чітку та логічну структуру, що відповідає принципам Clean Architecture та MVVM. Усі модулі та компоненти побудовані за єдиними принципами, що забезпечує уніфікованість коду та полегшує його підтримку.

### 3.2 Реалізація ключових функцій

Розглянемо практичну імплементацію основних функціональних можливостей мобільного додатку на основі раніше спроектованих алгоритмів. Згідно проектування алгоритмів у попередніх розділах, було реалізовано три ключові функціональні блоки додатку:

1. Система пошуку волонтера.
2. Механізм керування станами запиту.
3. Система сповіщень про знайденого волонтера.

Реалізація ключових функцій спирається на обраний технологічний стек та інші компоненти, які забезпечують ефективне функціонування системи відповідно до встановлених вимог. Розглянемо детально процес імплементації кожної з функцій.

Функціональність пошуку волонтера є центральним елементом взаємодії між ветеранами та потенційними помічниками. Процес починається з того, що ветеран створює запит через спеціальний інтерфейс – окремий фрагмент із формою, яка містить поля для введення назви, опису та вибору категорії допомоги (медична, транспортна, фізична, інше). Категорії обираються через зручний BottomSheet-діалог. Для зручності автоматично визначається місцезнаходження за допомогою геолокації, тож не потрібно вводити адресу вручну. Також можна прикріпити фото проблеми з галереї.

Перед відправкою система проводить ретельну валідацію всіх полів запиту. Перевіряється наявність назви, опису та категорії, а також коректність вибраного зображення. У випадку виявлення порожніх полів користувачу демонструється діалогове вікно з повідомленням про необхідність заповнення всіх обов'язкових полів. Такий підхід до валідації забезпечує високу якість даних, що надходять до системи.

Після успішної валідації система ініціює багатоетапний процес створення запиту у хмарній базі даних. Спочатку генерується унікальний ключ для ідентифікації запиту в Firebase Realtime Database. Система отримує повну інформацію про користувача-ветерана, включаючи його ідентифікатор, ім'я, електронну пошту та контактний телефон. На основі цих даних формується структурована модель запиту, яка включає всі введені користувачем поля, а також системні поля: час створення, статус запиту (початково встановлюється як "Очікування"), координати місцезнаходження та ідентифікатор ветерана.

Сформований об'єкт запиту зберігається в базі даних за визначеним шляхом, а фотографія завантажується до Firebase Storage з унікальним іменем, що включає ідентифікатор запиту. Це забезпечує однозначну прив'язку зображення до відповідного запиту та спрощує подальший доступ до нього.

Після збереження в базі даних запит автоматично стає видимим для всіх зареєстрованих волонтерів. Інтерфейс волонтера реалізує інтелектуальну фільтрацію запитів, відображаючи лише ті, що мають статус "Очікування" (очікують на прийняття) або "Активний" (якщо волонтер вже прийняв запит). Для покращення користувацького досвіду запити сортуються за часом оновлення, що дозволяє волонтерам бачити найновіші запити першими.

Завдяки Firebase Realtime Database нові запити миттєво синхронізуються на всіх пристроях, що дозволяє волонтерам швидко реагувати на потреби ветеранів.

Система керування станами запиту є важливим компонентом логіки взаємодії між ветеранами та волонтерами, забезпечуючи послідовний перебіг процесу надання допомоги – від створення запиту до його завершення. Її основою є патерн "Кінцевий автомат", який дозволяє контролювати допустимі переходи між станами. За реалізацію відповідає спеціалізований репозиторій, що містить чотири основні методи для зміни статусу запиту відповідно до поточної ситуації.

Коли волонтер погоджується допомогти, викликається метод активації, який змінює статус запиту з "Очікування" на "Активний" та зберігає дані про волонтера – його ім’я, контактну інформацію та час прийняття запиту. Це дозволяє ветерану бачити, хто саме надає допомогу. Якщо волонтер з будь-яких причин відмовляється, використовується метод деактивації: запит повертається до стану "Очікування", а всі дані про волонтера очищуються, відкриваючи можливість для інших допомогти.

Після успішного виконання допомоги ветеран переводить запит у стан "Завершений", фіксується час виконання, і така інформація зберігається в історії, що слугує майбутнім підґрунтям для оцінки ефективності системи. У випадку, якщо потреба в допомозі відпадає, ветеран може скасувати запит, після чого той змінює статус на "Скасований" і зникає з поля зору волонтерів.

Усі операції виконуються з гарантією цілісності даних завдяки використанню методу updateChildren [38], який дозволяє змінювати декілька

полів одночасно в рамках єдиної транзакції. Це важливо в умовах паралельного доступу. Окрім зміни статусу, кожна операція повертає результат у вигляді Callback-функції, яка дозволяє клієнтському інтерфейсу реагувати на результат – повідомляти користувача про успішну зміну або помилку.

Ключовим аспектом реалізації є використання Firebase Realtime Database, що дозволяє синхронізувати зміни між усіма учасниками системи в реальному часі. Завдяки цьому кожна зміна статусу миттєво відображається на екрані іншої сторони, забезпечуючи прозорість і своєчасність у комунікації між ветеранами та волонтерами.

Система сповіщень виконує важливу функцію інформування користувачів про ключові події в режимі реального часу. Її головне завдання – забезпечити своєчасне повідомлення ветеранів про те, що їх запит на допомогу був прийнятий, навіть коли додаток неактивний. Архітектурно вона складається з двох частин: серверної, побудованої на Firebase Cloud Functions, і клієнтської, реалізованої у вигляді Android-сервісу на базі FirebaseMessagingService.

Серверна логіка реагує на зміну поля "status" у базі даних. Коли статус запиту змінюється з "Очікування" на "Активний", функція автоматично запускається, зчитує ідентифікатор ветерана, знаходить відповідний FCM-токен (унікальний ідентифікатор його пристрою) й надсилає на нього повідомлення. Повідомлення містить текст з деталями про волонтера і додаткові параметри, зокрема ідентифікатор запиту та його новий стан, що дає змогу відкрити конкретний екран у додатку при натисканні на сповіщення.

На стороні клієнта за прийом повідомлень відповідає сервіс, який реагує на події Firebase Messaging через метод onMessageReceived. Він перевіряє наявність даних у повідомленні, витягує з них потрібну інформацію та формує відповідне сповіщення для користувача. Цей механізм працює навіть коли додаток закритий або перебуває у фоновому режимі, оскільки Firebase Cloud Messaging має змогу доставляти повідомлення напряму через системні служби Android. Щоб гарантувати безперервну роботу системи після перевстановлення додатку чи оновлення сервісів Google, реалізовано механізм автоматичного

оновлення FCM-токена [39]. Метод onNewToken зберігає нове значення токена в базі даних, що дозволяє підтримувати актуальність інформації для доставки повідомлень. Завдяки такому підходу система сповіщень залишається стабільною, надійною та невід'ємною частиною комунікації між ветеранами й волонтерами.

Усі ключові функції реалізовані з урахуванням можливості подальшого розширення функціональності та інтеграції з іншими системами. Використання сучасних технологій та архітектурних підходів забезпечує високу якість та надійність розробленого програмного продукту.

### 3.3 Інтерфейс користувача

Інтерфейс мобільного додатку є ключовим елементом взаємодії між користувачами та системою. Його проектування враховувало специфіку двох основних ролей – ветеранів та волонтерів. Особливу увагу приділено простоті, доступності та інтуїтивності UX-дизайну. Розглянемо детальніше основні екрани додатку та їх функціональні можливості для кожної ролі користувача.

Екрани адаптації – перші, з якими стикається користувач при запуску додатку. Вони демонструють візуальну ідентичність і виконують підготовчі дії перед завантаженням основного контенту. На рисунку 3.1 можна побачити екрани адаптації в логічній послідовності:

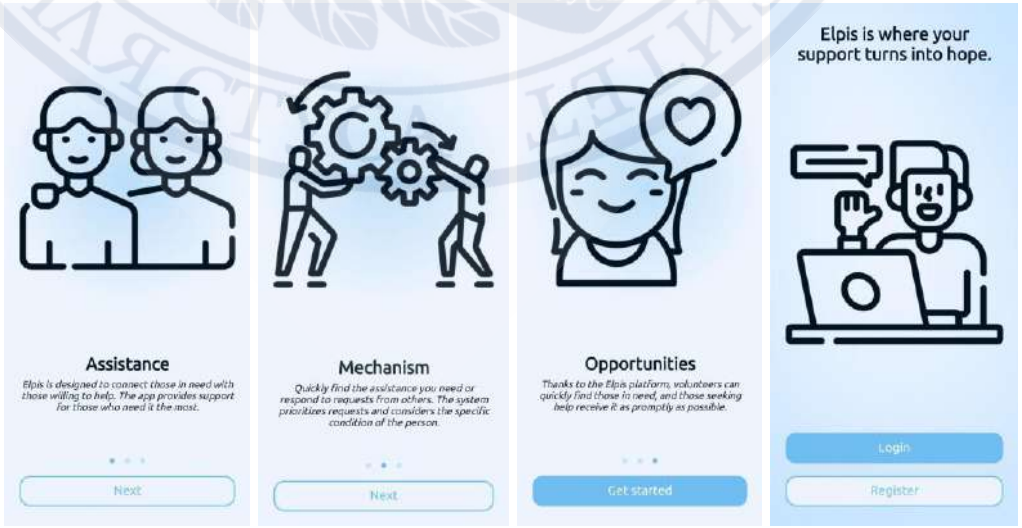


Рисунок 3.1 – Екрани адаптації

Екрани авторизації та реєстрації є важливими елементами інтерфейсу, що забезпечують безпечний доступ користувачів до системи. При розробці екранів адаптації було реалізовано механізм перевірки стану авторизації користувача за допомогою Firebase Auth, що визначає подальший маршрут навігації: на екран авторизації чи безпосередньо до головного екрану. Для забезпечення оптимального користувацького досвіду, процес реєстрації розділено на декілька етапів, що дозволяє користувачам поступово вводити необхідну інформацію.

Екран входу містить мінімальний набір елементів управління, необхідних для авторизації:

- Поля для введення електронної пошти та паролю.
- Кнопку входу.
- Кнопку з посиланням на екран реєстрації.

Для забезпечення надійності введених даних реалізовано валідацію полів із відображенням відповідних підказок у випадку введення некоректних даних. На рисунку 3.2 можна побачити екран авторизації в додаток:


Рисунок 3.2 – Екран авторизації

Процес реєстрації розділено на три основні етапи:

1. Базова інформація – введення електронної пошти, псевдоніму та паролю. На цьому етапі відбувається первинна реєстрація користувача у Firebase Auth.

2. Верифікація – підтвердження електронної пошти користувача. Для реалізації цього етапу використано функціонал Firebase Auth з надсиланням верифікаційного листа.
3. Розширена інформація – заповнення додаткових даних профілю: повне ім'я, дата народження, номер телефону, вибір ролі (ветеран або волонтер), стать та інші дані, необхідні для функціонування додатку.

Особливу увагу при розробці інтерфейсу реєстрації приділено процесу вибору ролі, оскільки це визначає подальший функціонал та інтерфейс додатку для конкретного користувача. На рисунку 3.3 можна побачити екрани реєстрації в логічній послідовності:

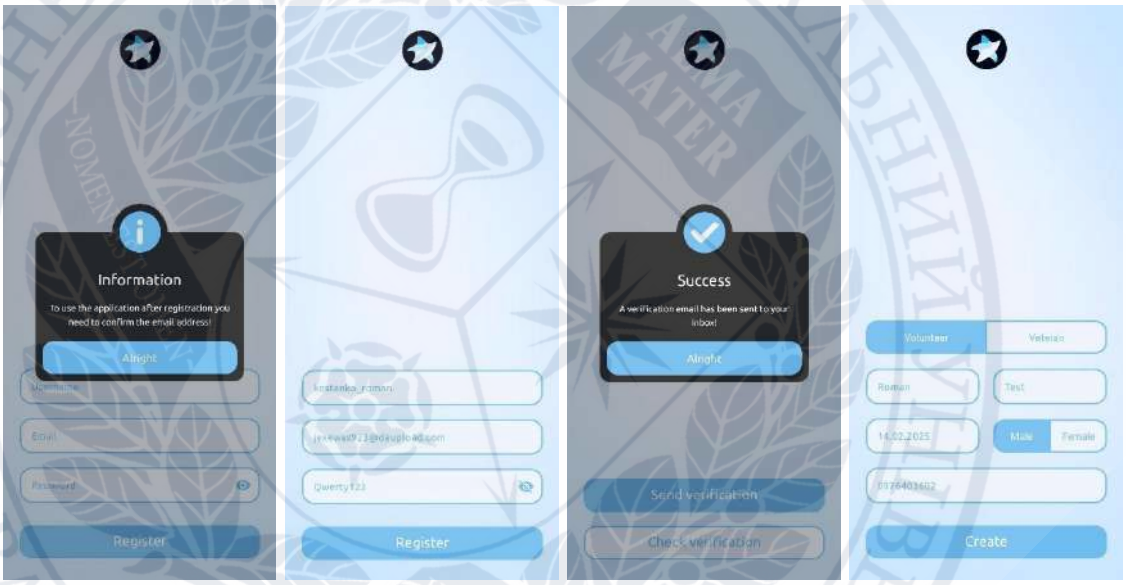


Рисунок 3.3 – Екрани реєстрації

Головний екран для користувачів з роллю "Ветеран" реалізовано у вигляді списку запитів про допомогу з можливістю перегляду їх статусів. Інтерфейс розроблено з урахуванням принципів матеріального дизайну та з фокусом на простоту взаємодії. Екран містить наступні основні елементи:

- Бокове меню з інформацією про користувача.
- Список запитів, розділених за статусами (активні, завершені, в очікуванні, скасовані).
- Кнопка створення нового запиту.

Для відображення списку запитів використано компонент RecyclerView з модифікованим адаптером, що забезпечує ефективне відображення та оновлення

даних. Кожен елемент списку містить основну інформацію про запит: заголовок, короткий опис, категорію, статус та дату створення. На рисунку 3.4 можна побачити екрани для користувачів з роллю "Ветеран" в логічній послідовності:

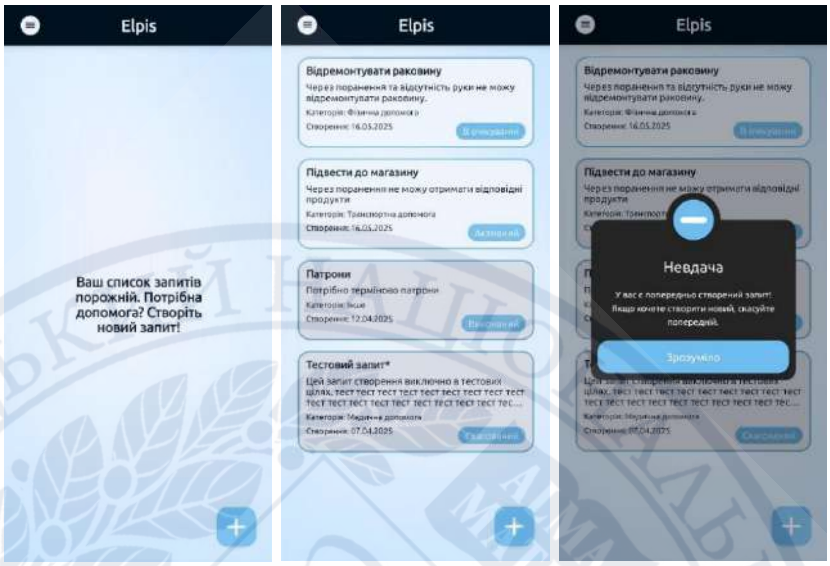


Рисунок 3.4 – Екрани для користувачів з роллю "Ветеран"

Екран створення запиту дозволяє ветеранам формувати нові запити про допомогу з детальним описом проблеми. На рисунку 3.5 можна побачити екран створення запиту:

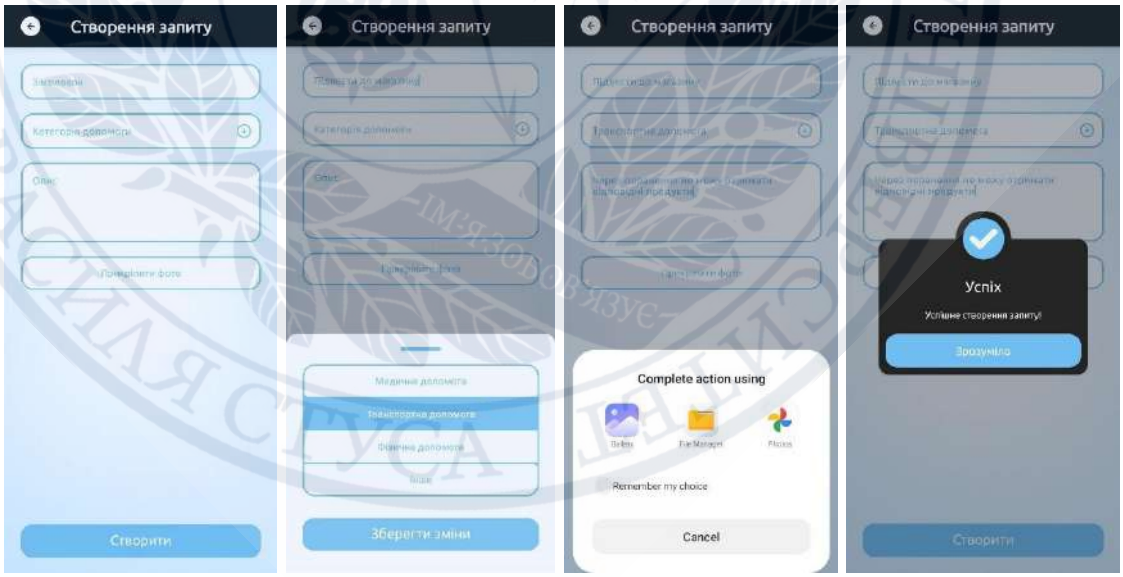


Рисунок 3.5 – Екран створення запиту

Для забезпечення зручності введення даних реалізовано валідацію полів із відображенням відповідних повідомлень про помилки. Інтерфейс містить форму з наступними полями:

- Заголовок запиту та детальний опис проблеми.

- Вибір категорії запиту.
- Можливість додавання фотографій проблеми.
- Кнопки збереження або скасування запиту.

Функціонал додавання фотографій реалізовано з використанням Firebase Storage, що дозволяє зберігати та отримувати медіа-контент.

Екран деталей запиту надає повну інформацію про конкретний запит та дозволяє ветерану відстежувати його статус. Інтерфейс містить:

- Детальну інформацію про запит (заголовок, опис, категорія).
- Статус запиту з візуальним індикатором.
- Інформацію про призначеного волонтера (якщо запит прийнято).
- Фотографії, додані до запиту.
- Кнопки для управління запитом (відміна, завершення).

На рисунку 3.6 можна побачити екран деталей запиту:

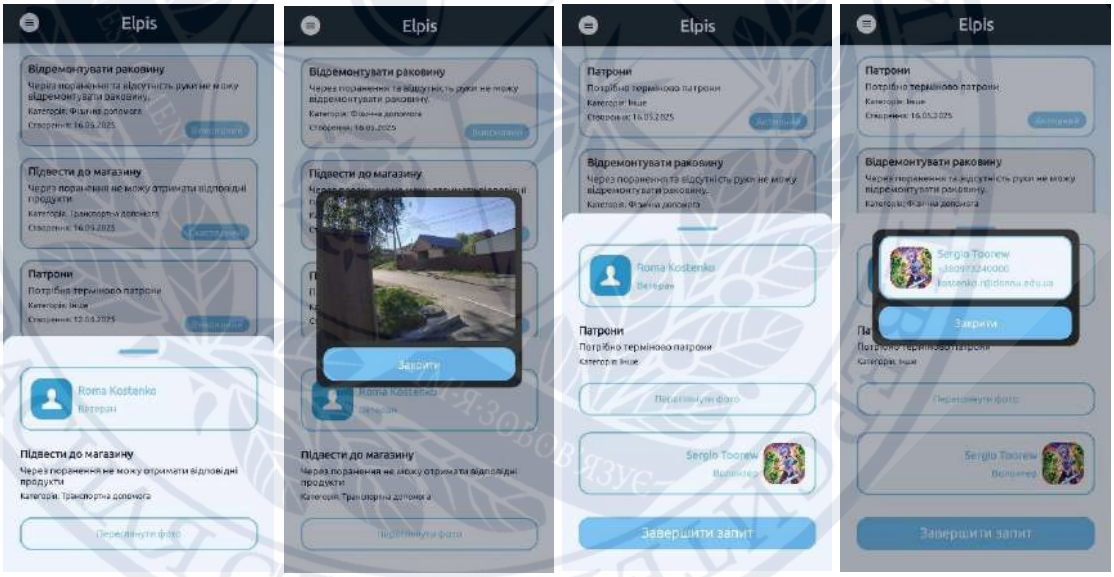


Рисунок 3.6 – Екран деталей запиту

Головний екран для користувачів з роллю "Волонтер" реалізовано у вигляді інтерактивної карти з відображенням запитів про допомогу. Такий підхід дозволяє волонтерам візуально оцінити розташування запитів та обрати ті, які знаходяться поблизу. Екран містить наступні основні елементи:

- Бокове меню з інформацією про користувача.
- Інтерактивну карту з маркерами запитів.

- Панель активних запитів.
- Індикатор поточного місцезнаходження волонтера.

Для забезпечення оптимальної продуктивності при відображенні карти та взаємодії з нею використано оптимізовані методи роботи з Mapbox API, зокрема, кластеризацію маркерів при віддаленні та детальне відображення при наближенні. При виборі конкретного запиту на карті волонтеру надається детальна інформація про нього та можливість прийняти запит для виконання. Інтерфейс містить:

- Детальну інформацію про запит (заголовок, опис, категорія).
- Інформацію про ветерана, який створив запит.
- Фотографії, додані до запиту.
- Кнопки для управління запитом (прийняти, відхилити).

На рисунку 3.7 можна побачити екран для користувачів з роллю "Волонтер" в логічній послідовності:

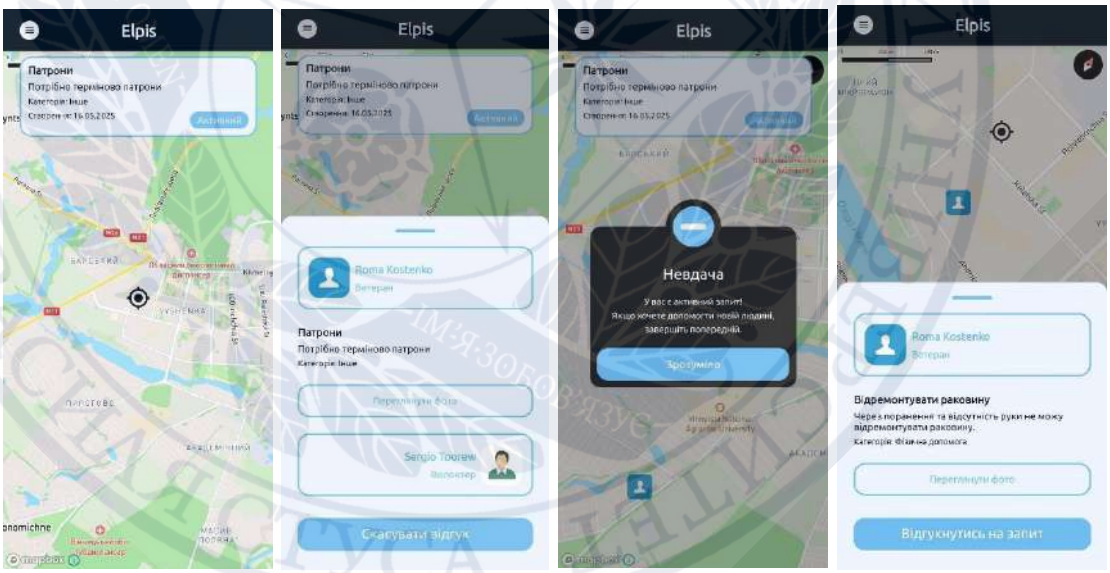


Рисунок 3.7 – Екран для користувачів з роллю "Волонтер"

Для забезпечення зручної навігації між різними розділами додатку реалізовано бокове меню (Drawer Navigation), доступне з будь-якого основного екрану додатку. Меню містить:

- Інформацію про поточного користувача (фото профілю, ім'я, роль).
- Доступ до головного екрану додатку.
- Доступ до налаштувань додатку.

- Доступ до інформаційного розділу.
- Інформацію про версію та код додатку.

Реалізація бокового меню виконана з використанням компонента DrawerLayout та NavigationView, інтегрованих в Single Activity Architecture. Це забезпечує узгоджену навігацію та збереження стану додатку при переході між різними екранами. На рисунку 3.8 можна побачити екран бокового меню:

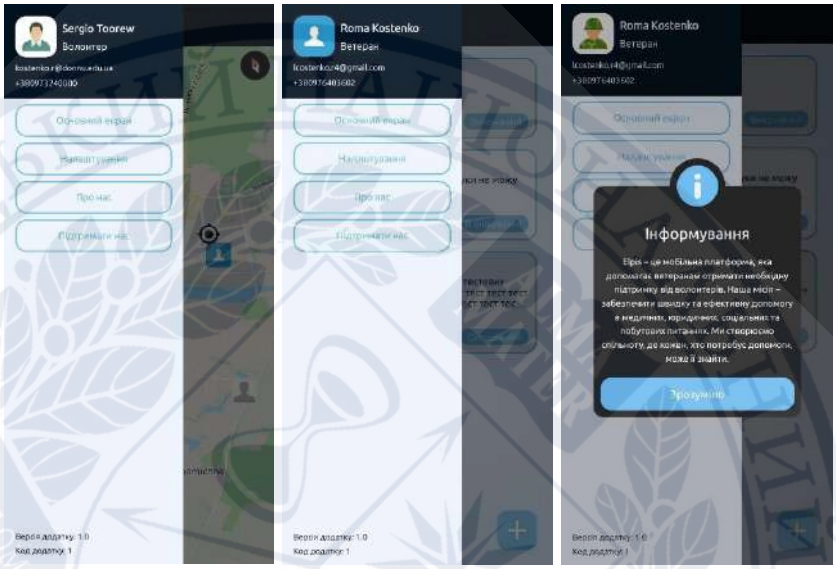


Рисунок 3.8 – Екран бокового меню

Екран налаштувань надає користувачам можливість конфігурувати додаток відповідно до своїх потреб. На рисунку 3.9 можна побачити екран налаштувань:

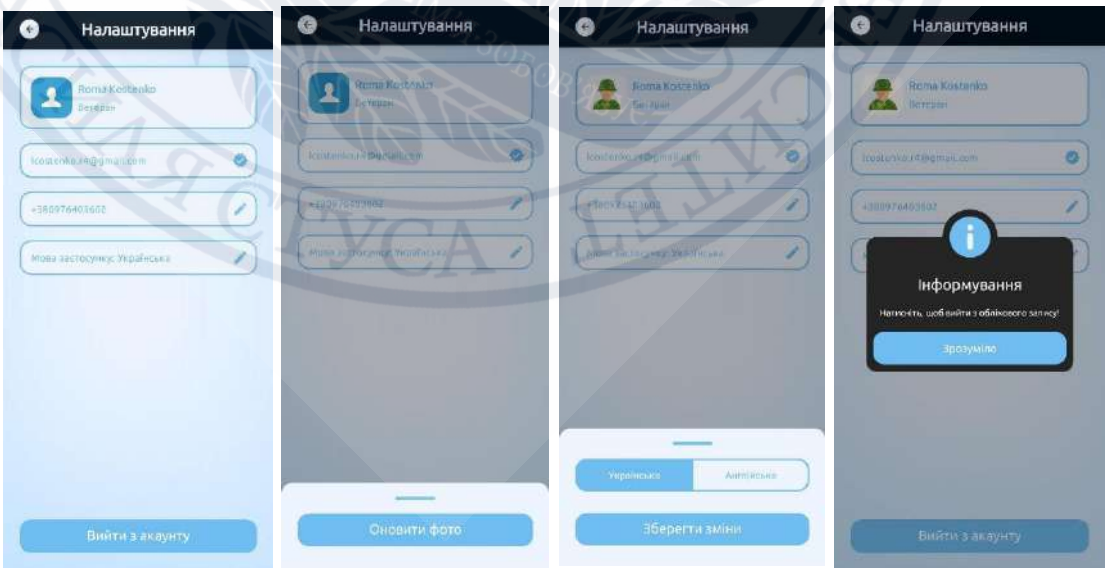


Рисунок 3.9 – Екран налаштувань

Для реалізації екрану налаштувань використано компонент PreferenceFragmentCompat, що забезпечує стандартний досвід взаємодії з налаштуваннями для користувачів Android. Інтерфейс містить наступні елементи:

- Управління профілем (зміна фото профілю, номера телефону).
- Перевірка верифікації електронної пошти.
- Вибір мови додатку (українська або англійська).
- Кнопка виходу з облікового запису.

Збереження налаштувань відбувається з використанням SharedPreferences, що дозволяє зберігати конфігурацію між сеансами роботи з додатком.

При розробці інтерфейсу користувача було дотримано наступних принципів UX-дизайну:

1. Простота – інтерфейс розроблено з мінімальною кількістю елементів, необхідних для виконання конкретних завдань. Використано чітку ієрархію елементів та інтуїтивно зрозумілі позначення.
2. Інтуїтивність – інтерфейс розроблено з урахуванням стандартних патернів взаємодії в Android, що дозволяє користувачам легко адаптуватися до додатку.
3. Відповідний зворотний зв'язок – реалізовано систему повідомлень та сповіщень, що інформують користувача про результати його дій. Використано візуальні індикатори для відображення стану процесів.
4. Ефективність – інтерфейс оптимізовано для мінімізації кількості дій, необхідних для виконання типових завдань. Реалізовано оптимальні шляхи взаємодії для кожної ролі користувача.

Розроблений інтерфейс реалізовано відповідно до визначеного технологічного стеку, де XML використовується для опису UI-компонентів, а MVVM архітектура забезпечує чітке розділення відображення даних від бізнес-логіки. Використання LiveData дозволяє реалізувати реактивне оновлення інтерфейсу без додаткового втручання користувача, що є особливо важливим для відображення статусів запитів та повідомлень у реальному часі.

### 3.4 Тестування програмного продукту

У процесі розробки мобільного додатку було проведено комплексне тестування з метою забезпечення високої якості та надійності програмного продукту. Тестування здійснювалося на різних рівнях, використовуючи сучасні методики та інструменти, що дозволило виявити та усунути потенційні проблеми ще на етапі розробки.

Інтеграційне тестування [40] дозволило перевірити взаємодію між різними компонентами системи та переконатися, що вони правильно працюють разом. Ключовим аспектом інтеграційного тестування була перевірка взаємодії додатку з сервісами Firebase. Особлива увага приділялася тестуванню процесів авторизації користувачів через Firebase Auth, включаючи реєстрацію нових користувачів та вхід в систему. Було перевірено правильність збереження та отримання даних з Firebase Realtime Database, а також роботу з Firebase Storage для завантаження та отримання файлів.

Важливим елементом інтеграційного тестування була перевірка функціональності Firebase Cloud Messaging для надсилання та отримання сповіщень. Цей аспект є критично важливим для забезпечення вчасного інформування користувачів про зміни статусу запитів на допомогу. Також було протестовано взаємодію між різними модулями додатку, зокрема передачу даних між екранами, обмін інформацією між репозиторіями та ViewModel.

Функціональне тестування [41] проводилося для перевірки відповідності розробленого додатку визначеним функціональним вимогам. Цей етап тестування був спрямований на перевірку правильності реалізації всіх користувацьких сценаріїв. Особлива увага приділялася тестуванню функціональності роботи з запитами на допомогу, включаючи створення нових запитів, перегляд існуючих запитів, а також оновлення та видалення запитів. Тести підтвердили відповідність цих функцій визначеним вимогам.

Важливим аспектом функціонального тестування була перевірка геолокаційних функцій додатку. Було підтверджено коректність визначення

поточного місцезнаходження користувача, правильність відображення запитів на карті та точність розрахунку відстані до запитів. Також було протестовано функціональність управління профілем користувача, включаючи оновлення персональних даних, зміну налаштувань додатку та зміну мови інтерфейсу. Всі ці функції продемонстрували повну відповідність визначеним вимогам.

Тестування інтерфейсу [42] користувача є критично важливим аспектом забезпечення якості мобільного додатку, оскільки саме через інтерфейс відбувається взаємодія користувача з системою. Даний етап тестування був спрямований на перевірку зручності та інтуїтивності інтерфейсу.

В рамках тестування відображення UI-елементів було перевірено коректність відображення всіх екранів на різних розмірах екранів, адаптивність дизайну та правильність відображення шрифтів, кольорів та інших візуальних елементів. Тести підтвердили, що інтерфейс додатку коректно відображається на пристроях з різними розмірами екранів та роздільною здатністю.

Значна увага приділялася тестуванню навігації в додатку. Було перевірено правильність переходів між екранами, функціонування навігаційного меню та роботу кнопки повернення. Тести показали інтуїтивність та логічність навігаційної структури додатку. Окремим блоком тестування була перевірка форм введення, включаючи валідацію полів форм, обробку некоректних даних та відображення повідомлень про помилки. Було підтверджено, що форми введення даних правильно обробляють різні типи даних та надають користувачу зрозумілі повідомлення про помилки. Важливим аспектом тестування інтерфейсу користувача була перевірка доступності додатку, включаючи контрастність кольорів, роботу з екранними читачами та розмір тексту та елементів керування. Тести показали, що додаток відповідає сучасним вимогам до доступності та може використовуватися людьми з різними потребами.

Останнім було проведено ручне тестування за заздалегідь визначеними сценаріями використання додатку. Цей підхід дозволив оцінити додаток з точки зору реального користувача та виявити проблеми, які могли бути пропущені. В рамках тестування сценарію було проведено одночасне створення великої

кількості запитів, перевірено швидкодію системи при масовому створенні запитів та роботу бази даних при високому навантаженні. Тестування показало, що система зберігає стабільність та продуктивність навіть при значному збільшенні кількості запитів.

Тестування підтвердило, що розроблений мобільний додаток повністю відповідає поставленій меті – забезпечення ефективної організації процесу взаємодії між ветеранами, що потребують допомоги, та волонтерами, які можуть цю допомогу надати.

### 3.5 Аналіз результатів реалізації

Після реалізації та тестування здійснюється всебічний аналіз результатів, пов'язаних із розробкою мобільного додатку, що сприяє управлінню процесом волонтерської допомоги ветеранам. Аналіз охоплює оцінку виконання запланованих функціональних вимог, відповідність отриманого продукту визначеним цілям, а також перспективи подальшого розширення функціональності системи.

У ході розробки було успішно реалізовано всі критично важливі функціональні компоненти системи, що відповідають концептуальним положенням, викладеним у попередніх розділах. Зокрема, було реалізовано:

- Повноцінну систему реєстрації та авторизації з розподілом ролей (ветеран/волонтер) на базі Firebase Auth, що забезпечує надійну верифікацію та автентифікацію користувачів.
- Функціонал створення, редагування та відстеження запитів про допомогу для ветеранів з можливістю прикріплення фотоматеріалів через інтеграцію Firebase Storage.
- Інтерактивну карту на основі Mapbox API для волонтерів із візуалізацією доступних запитів та можливістю фільтрації за категоріями.

- Механізм сповіщень на базі Firebase Cloud Messaging для оперативного інформування користувачів про зміни в статусі запитів.

Розроблений мобільний додаток повністю відповідає меті, сформульованій раніше – створення функціонального Android-додатку для ефективної організації процесу взаємодії між ветеранами та волонтерами. Аналіз відповідності реалізації конкретним цілям проекту демонструє наступні результати:

1. Спрощення процесу пошуку та надання волонтерської допомоги ветеранам. Інтуїтивно зрозумілий інтерфейс та чітка система категоризації запитів зробили процес пошуку та надання допомоги максимально простим для обох категорій користувачів. Використання Mapbox API дозволило реалізувати геопросторовий підхід до організації допомоги.
2. Підвищення ефективності розподілу волонтерських ресурсів. Система запитів за категоріями та місцем розташування забезпечує оптимальний розподіл волонтерських ресурсів відповідно до компетенцій волонтерів та їх територіального розміщення.
3. Скорочення часу реагування на запити про допомогу. Інтеграція Firebase Realtime Database та Firebase Cloud Messaging забезпечила миттєве оновлення даних та оперативне інформування користувачів про зміни в системі, що значно скоротило час реагування на запити.

Результати реалізації мобільного додатку демонструють значний потенціал для подальшого розвитку та масштабування. На основі розробленої архітектури можливе впровадження таких додаткових функціональних можливостей:

- Розширення системи ролей із введенням категорій "координатор допомоги" та "представник організації", що дозволить структурувати волонтерську діяльність на інституційному рівні.
- Інтеграція з державними електронними системами для автоматичної верифікації статусу ветерана та волонтера.

- Впровадження аналітичного модуля для збору та аналізу статистичних даних щодо наданої допомоги з метою оптимізації розподілу ресурсів.
- Розробка алгоритмів на базі машинного навчання для прогнозування потреб у допомозі та автоматичного підбору відповідних волонтерів.
- Адаптація системи для використання в інших сферах соціальної допомоги, таких як підтримка літніх людей або внутрішньо переміщених осіб.

Таким чином, розроблений програмний продукт успішно вирішує поставлені завдання щодо організації ефективної взаємодії між ветеранами та волонтерами, а також має значний потенціал для подальшого розвитку та масштабування в інші сфери соціальної допомоги.

**Висновки до третього розділу**

У третьому розділі представлено процес реалізації мобільного додатку для керування процесом волонтерської допомоги ветеранам, результати якого демонструють успішне досягнення поставлених цілей дослідження.

В ході розробки програмного продукту було повністю реалізовано функціональні вимоги, визначені на етапі проектування. Використання Android (Kotlin) як основної технології розробки забезпечило створення нативного додатку з високою продуктивністю та оптимізованим використанням ресурсів мобільного пристрою. Архітектурний підхід MVVM у поєднанні з Single Activity Architecture та принципами Clean Architecture дозволив створити масштабований, легко підтримуваний та тестований код з чітким розподілом відповідальності між компонентами.

Інтеграція Firebase Realtime Database, Firebase Auth та Firebase Storage виявилася оптимальним рішенням для забезпечення безперебійної синхронізації даних між користувачами системи, надійної автентифікації та зберігання медіа файлів. Реалізація геолокаційних функцій за допомогою Mapbox API дозволила

точно відображати розташування користувачів, що є критично важливою функціональністю для координації волонтерської допомоги.

У процесі розробки було подолано низку технічних викликів, зокрема:

- Складнощі оптимізації запитів до Firebase Realtime Database при масштабуванні кількості користувачів, які вирішено шляхом оптимізації структури бази даних.
- Виклики, пов'язані з інтеграцією Firebase Cloud Messaging для push-повідомлень, які вирішено шляхом налаштування Firebase Functions для автоматичного запуску відправки сповіщень при оновленні даних.

Застосування залежності Hilt для впровадження залежностей значно покращило структуру коду та спростило тестування, а використання LiveData забезпечило реактивну взаємодію між шарами додатку, що дозволило оперативно відображати зміни у користувацькому інтерфейсі.

Результатом проведеної роботи став повнофункціональний мобільний додаток, який успішно вирішує проблему цифрового керування волонтерською допомогою ветеранам. Програмний продукт забезпечує зручний інтерфейс для реєстрації потреб ветеранів, пошуку відповідних волонтерів, координації дій та моніторингу статусу наданої допомоги. Всі компоненти системи працюють злагоджено та відповідають сучасним стандартам розробки Android-додатків.

Комплексна реалізація усіх запланованих функцій підтверджує доцільність обраного технологічного стеку та архітектурних рішень для розв'язання поставлених завдань у контексті організації та цифрового керування волонтерською допомогою ветеранам засобами мобільних технологій.

# ВИСНОВКИ

Виконана робота представляє функціональний мобільний Android додаток для керування процесом волонтерської допомоги ветеранам. Проведене дослідження та практична реалізація дозволили досягти поставленої мети та вирішити визначені завдання.

У першому розділі роботи проведено аналіз предметної області та існуючих рішень у сфері волонтерської допомоги ветеранам, що дозволило встановити актуальність проблеми та недостатність спеціалізованих цифрових рішень. Виявлено ключові процеси, які потребують цифровізації: реєстрація потреб ветеранів, пошук волонтерів та координація дій. Вивчено принципи проектування соціальних мобільних додатків: інклюзивність, доступність, безпека та орієнтація на користувача.

У другому розділі розроблено концептуальні та алгоритмічні засади мобільного додатку. Створено функціональну модель додатку та модель користувацької взаємодії з визначеними ролями та сценаріями використання. Запропоновано архітектурне рішення, спроектовано алгоритми ключових функцій системи та структуру даних з використанням зовнішніх сервісів.

У третьому розділі реалізовано повнофункціональний мобільний додаток для мобільної системи Android. Впроваджено нативний додаток з високою продуктивністю та оптимізованим використанням ресурсів. Інтегровано системи Firebase для забезпечення синхронізації даних, автентифікації та зберігання медіа файлів. Реалізовано геолокаційні функції за допомогою Mapbox API та впроваджено технології для управління асинхронними операціями. У процесі розробки подолано технічні виклики, зокрема оптимізацію запитів до бази даних при масштабуванні та інтеграцію системи push-повідомлень.

Результатом проведеної роботи є готовий до впровадження програмний продукт, який відповідає поставленій меті – ефективній організації процесу взаємодії між ветеранами та волонтерами. Розроблений додаток забезпечує зручний інтерфейс для реєстрації потреб ветеранів, ефективний пошук

відповідних волонтерів з урахуванням геолокації, координацію дій між учасниками процесу та моніторинг статусу наданої допомоги. Всі компоненти системи працюють злагоджено та відповідають сучасним стандартам розробки Android-додатків.

Практична значущість розробленого додатку полягає у створенні ефективного цифрового інструменту для вирішення актуальної соціальної проблеми – підтримки ветеранів у процесі реінтеграції до цивільного життя. Мобільний додаток сприятиме підвищенню доступності волонтерської допомоги та оптимізації координації волонтерських ресурсів. Систематизовано вимоги до подібних соціальних додатків та встановлено оптимальний технологічний стек для їх реалізації.

Запропоновані у роботі технологічні рішення та методологічні підходи можуть бути використані для розробки інших соціальних додатків та платформ, спрямованих на вирішення суспільно значущих проблем засобами мобільних технологій. Достовірність отриманих результатів підтверджується успішною реалізацією всіх запланованих функцій програмного продукту та його відповідністю визначеним на етапі проектування вимогам.

Комплексна реалізація усіх запланованих функцій підтверджує доцільність обраного технологічного стеку та архітектурних рішень для розв'язання поставлених завдань у контексті організації та цифрового керування волонтерською допомогою ветеранам засобами мобільних технологій. Уточнено принципи побудови мобільних сервісів для волонтерської діяльності в умовах сучасних соціальних викликів.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Горбунова В. В. Національний інститут психічного здоров'я. Посттравматичний стресовий розлад серед ветеранів: статистичний звіт. 2023. С. 3-5.
2. Грішнова О. А., Лисенко В. Д. Реінтеграція учасників бойових дій у цивільне життя: проблеми зайнятості та шляхи їх розв'язання. 2024. С. 7-9.
3. Яценко Л. Д. Проблеми працевлаштування ветеранів війни: ризики для держави та суспільства. 2023. С. 1-4.
4. Методичні рекомендації для роботодавців з організації працевлаштування та розвитку трудових відносин з ветеранами війни. Київ: Міністерство у справах ветеранів України. 2025. 57 с.
5. Волонтерська Платформа. URL: https://platforma.volunteer.country/ (дата звернення: 15.05.2025).
6. Волонтер.орг. URL: https://volonter.org/ (дата звернення: 15.05.2025).
7. Veterans Affairs Mobile. URL: https://mobile.va.gov/ (дата звернення: 15.05.2025).
8. VolunteerMatch. URL: https://www.volunteermatch.org/ (дата звернення: 15.05.2025).
9. Лях Т.Л., Бондаренко З.П., Журавель Т.В., Спіріна Т.П. Менеджмент волонтерських груп від А до Я: навч. посіб. Київ: Київський університет імені Бориса Грінченка. 2012. 288 с.
10. McCurley S., Lynch R. Volunteer Management: Mobilizing All the Resources of the Community. 3th Edition. 2010. 434 с.
11. Gassmann O., Frankenberger K., Csik M. The Business Model Navigator: 55 Models That Will Revolutionise Your Business. 1st Edition. 2014. 400 с.
12. Sutherland J., Sutherland J.J. Scrum: The Art of Doing Twice the Work in Half the Time. 2014. 384 с.
13. McGraw G. Software Security: Building Security In. 1st Edition. 2006. 448 с.
14. GDPR. URL: https://gdpr-info.eu/ (дата звернення: 16.05.2025).
15. HIPAA. URL: https://www.hhs.gov/hipaa/for-professionals/privacy/laws-regulations/index.html (дата звернення: 16.05.2025).
16. Sills B., Gardner B., Marsicano K., Stewart C. Android Programming: The Big Nerd Ranch Guide. 5th Edition. 2022. 688 с.
17. Carter T. Android Development with Kotlin: How to Build Modern Mobile Applications: A Hands-On Guide to Jetpack, MVVM Architecture, and Android Studio. 2025. 158 с.
18. Chesterfield G. Kotlin for Android Development: Modern Mobile Programming for Android Applications. 2025. 242 с.
19. Hellman E. Android Programming: Pushing the Limits. 2013. 432 с.
20. Martin R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. 2017. 432 с.
21. Mednieks Z., Dornin L., Meike G.B., Nakamura M., Barnert W. Programming Android: Java Programming for the New Generation of Mobile Devices. 3rd Edition. 2016. 650 с.

22. Dominguez A. O., Cheng Y. Advanced Android App Architecture: Real-world app architecture in Kotlin. 1st Edition. 2019. 273 с.
23. Künneth T. Android UI Development with Jetpack Compose: Bring declarative and native UIs to life quickly and easily on Android using Jetpack Compose. 1st Edition. 2022. 248 с.
24. Tigcal J. Simplifying Android Development with Coroutines and Flows: Learn how to use Kotlin coroutines and the flow API to handle data streams asynchronously in your Android app. 2022. 164 с.
25. Seemann M., van Deursen S. Dependency Injection Principles, Practices, and Patterns. 1st Edition. 2019. 552 с.
26. Forrester A., Boudjnah E., Dumbravan A., Tigcal J. How to Build Android Apps with Kotlin: A hands-on guide to developing, testing, and publishing your first apps with Android. 1st Edition. 2021. 794 с.
27. Smyth N. Firebase Essentials: Android Edition. 1st Edition. 2017. 534 с.
28. App Architecture: Presentation layer. URL: https://proandroiddev.com/app-architecture-presentation-layer-0d704ede2564 (дата звернення: 17.05.2025).
29. App Architecture: Domain layer. URL: https://proandroiddev.com/app-architecture-domain-layer-b9f6aa839e33 (дата звернення: 17.05.2025).
30. App Architecture: Data layer. URL: https://proandroiddev.com/app-architecture-data-layer-8d681e8f8a6d (дата звернення: 17.05.2025).
31. Massimo C. Dagger by Tutorials: Dependency Injection on Android with Dagger & Hilt. 1st Edition. 2021. 544 с.
32. Shalloway A., Trott J. Design Patterns Explained: A New Perspective on Object Oriented Design. 2nd Edition. 2004. 468 с.
33. Freeman E., Robson E. Head First Design Patterns: Building Extensible and Maintainable Object-Oriented Software. 2nd Edition. 2021. 669 с.
34. Houssem Y. Firebase Cookbook: Over 70 recipes to help you create real-time web and mobile applications with Firebase. 1st Edition. 2017. 290 с.
35. Bailey J., Djermanović D., Dominguez A. O., Kamal F., Subhrajyoti S., Wangereka H. Saving Data on Android: Learn Jetpack Data Store, Room, Firebase & SQLite with Kotlin. 2nd Edition. 2021. 344 с.
36. Firebase DB Security. URL: https://firebase.google.com/docs/database/security (дата звернення: 17.05.2025).
37. Understand Privacy and Security in Firebase. URL: https://firebase.google.com/support/privacy (дата звернення: 17.05.2025).
38. Moroney L. The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform. 1st Edition. 2017. 288 с.
39. Class for receiving messages from FCM – FirebaseMessagingService. URL: https://firebase.google.com/docs/reference/kotlin/com/google/firebase/messaging/FirebaseMessagingService (дата звернення: 17.05.2025).
40. Jorgensen P. C., DeVries B. Software Testing: A Craftsman’s Approach. 5th Edition. 2022. 528 с.
41. Gayathri M. Full Stack Testing: A Practical Guide for Delivering High Quality Software. 1st Edition. 2022. 405 с.
42. Travis D., Hodgson P. Think Like a UX Researcher. 2nd Edition. 2023. 352 с.