Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Програмна система для організації збору коштів. Мобільний застосунок. Підсистема роботи з користувачами та комунікаціями
(тема)

Виконав:
здобувач 4 року навчання
групи ПЗПІ-21-1

Кірілл БЛУВБАНД
(Власне ім'я, ПРІЗВИЩЕ)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Програмна інженерія
(повна назва освітньої програми)

Керівник ст.викл. кафедри ПІ Віталій ЛЯПОТА
(посада, Власне ім'я, ПРІЗВИЩЕ)

Допускається до захисту
Зав. кафедри _____ Кирило СМЕЛЯКОВ
(підпис) (Власне ім'я, ПРІЗВИЩЕ)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти перший (бакалаврський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми Освітньо-професійна
Освітня програма Програмна Інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Блувбанд Кіріллу Ігоровичу
(прізвище, ім’я, по батькові)

1. Тема роботи Програмна система для організації збору коштів. Мобільний застосунок. Підсистема роботи з користувачами та комунікаціями
Затверджена наказом по університету від 19.05.2025 р. № 397 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 19.06.2025 р.
3. Вихідні дані до роботи розробити мобільний застосунок для проведення благодійних зборів коштів (фандрейзингу), методи розробки, Kotlin і React
4. Перелік питань, що потрібно опрацювати в роботі: Вступ, аналіз предметної галузі, формування вимог до програмної системи, архітектура та проектування програмного забезпечення, опис прийнятих програмних рішень, тестування програмного забезпечення, висновки, додатки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	08.04.2025	*виконано*
2	Створення специфікації ПЗ	12.04.2025	*виконано*
3	Проектування ПЗ	15.04.2025	*виконано*
4	Розробка ПЗ	16.04.2025	*виконано*
5	Тестування ПЗ	28.04.2025	*виконано*
6	Оформлення пояснювальної записки	30.04.2025	*виконано*
7	Підготовка презентації та доповіді	10.05.2025	*виконано*
8	Попередній захист	17.06.2025	*виконано*
9	Нормоконтроль, рецензування	17.06.2025	*виконано*
10	Здача роботи у електронний архів	18.06.2025	*виконано*
11	Допуск до захисту у зав. кафедри	18.06.2025	*виконано*

Дата видачі завдання « 07» «квітня» 2025 р.

Здобувач _____ Кірілл БЛУВБАНД
(підпис)

Керівник роботи _____ ст.викл. Віталій ЛЯПОТА
(підпис) (посада, Власне ім’я, ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи бакалавра, 57 стор., 14 рис., 1 табл., 12 джерел, 3 додатків.

GDPR/PSD2, PUSH-НОТИФІКАЦІЇ, ZERO TRUST, МОБІЛЬНИЙ ФАНДРЕЙЗИНГ, МОБІЛЬНІ ПЛАТЕЖІ.

Метою роботи є розробка мобільного застосунку для проведення благодійних зборів коштів (фандрейзингу), з використанням сучасних підходів до взаємодії з користувачами, технологій захисту персональних даних та мобільних платіжних рішень.

Методи дослідження: аналіз моделей взаємодії користувачів у мобільних додатках, проектування UX/UI-дизайну, побудова архітектури застосунку на основі мікросервісів, створення UML-діаграм для визначення структури компонентів, реалізація безпечної автентифікації користувачів, а також інтеграція з платіжними шлюзами та push-сповіщеннями.

У результаті дослідження розроблено мобільний застосунок із простим та інтуїтивним інтерфейсом для створення та підтримки благодійних ініціатив, забезпечено систему двосторонньої комунікації з користувачами, реалізовано безпечну обробку платежів і персональних даних, а також спроєктовано архітектуру, що підтримує масштабованість та високу продуктивність у пікові моменти навантаження.

ABSTRACT

GDPR/PSD2, PUSH NOTIFICATIONS, ZERO TRUST, MOBILE FUNDRAISING, MOBILE PAYMENTS.

The purpose of this work is to develop a mobile application for conducting charitable fundraising, using modern approaches to user interaction, personal data protection technologies, and mobile payment solutions.

Research methods include analyzing user interaction models in mobile applications, designing UX/UI, building the application architecture based on microservices, creating UML diagrams to define component structure, implementing secure user authentication, as well as integrating payment gateways and push notifications.

As a result of the study, a mobile application with a simple and intuitive interface for creating and supporting charitable initiatives was developed; a system for two-way communication with users was ensured; secure processing of payments and personal data was implemented; and an architecture supporting scalability and high performance during peak load times was designed.

ЗМІСТ

ВСТУП

Мобільні фандрейзингові системи – невід'ємна складова сучасної цифрової економіки, оскільки саме смартфон нині виступає «першим екраном» для спонтанної благодійної взаємодії. За даними галузевих оглядів, більше шістдесяти відсотків регулярних пожертв здійснюється через мобільні застосунки; тенденція до зростання посилюється завдяки мінімізації тертя під час платежу, інтеграції біометричної автентифікації та персоналізованих push-повідомлень. Утім, поряд із зручністю постають виклики інформаційної безпеки та комплаєнсу: вимоги GDPR, PSD2 і KYC поступово ускладнюють архітектуру сервісів, змушуючи розробників поєднувати швидкість UX із суворими протоколами захисту персональних даних.

Актуальність курсової роботи обумовлена необхідністю створення технічного рішення, яке здатне одночасно забезпечити: по-перше, скорочення користувацького сценарію до двох-трьох дотиків; по-друге, горизонтальне масштабування у разі пікового навантаження; по-третє, криптографічно стійке зберігання мінімального набору PII без втрати аналітичної цінності. Мета дослідження полягає в розробці модульної архітектури бекенду та мобільного клієнта, що підтримує швидке ініціювання зборів, прозору звітність і автоматизовану перевірку ініціатив, а також у формалізації вимог до захисту даних у контексті мобільного фандрейзингу.

Об'єктом дослідження є процес мобільного збору благодійних коштів, предметом – методи забезпечення безпечної й високопродуктивної взаємодії користувача з фандрейз-платформою. Наукова новизна полягає у запропонованій схемі поєднання псевдонімізації даних та zero-knowledge-доказів для підтвердження платоспроможності, що дозволяє мінімізувати зберігання чутливої інформації. Практичне значення роботи вбачається у створенні прототипу API та мобільного застосунку, придатного для впровадження в благодійних організаціях, фінтех-стартапах і громадських проектах, які прагнуть скоротити витрати на запуск власних сервісів збору коштів і водночас гарантувати довіру донорів.

Для досягнення поставленої мети використано методи аналізу предметної області, моделювання мікросервісної архітектури, розробки REST-інтерфейсів, а також криптографічні підходи до шифрування даних і протоколи аутентифікації на базі токенів. Валідація результатів здійснювалася за допомогою сценаріїв навантажувального тестування та експертної оцінки відповідності нормативним вимогам. Отримані висновки можуть бути інтегровані в існуючі мобільні платформи або слугувати базою для подальших досліджень у сфері безпечних цифрових фінансових сервісів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Особливості мобільних фандрейзингових систем та комунікацій у цифрових продуктах

Після початку повномасштабного вторгнення Росії культура донатів в Україні значно посилилася – соціальні мережі й медіа дедалі частіше популяризують благодійність, а лідери думок мотивують не залишатися осторонь. Згідно зі звітом Philanthropy in Ukraine, після вторгнення слово «донат» стало надбанням суспільства, а його значення поширилося на різні форми допомоги [1]. Разом із цим відбувається «цифровізація» збору коштів – дедалі більше громадян користуються смартфонами та електронними платежами, що створює сприятливе підґрунтя для мобільних фандрейзингових сервісів.

Статистика підтверджує стрімкий ріст онлайн-пожертв: у світі третина всіх донатів у 2023 році надходила з мобільних пристроїв. Водночас в Україні обсяги благодійних платежів у застосунках зростають ще швидше. Зокрема, сервіс Monobank за 2023 рік отримав від українців близько 720 млн USD пожертв (утричі більше, ніж у 2022, та в середньому 349 грн за донат) [2]. Загалом у 2023 році громадяни України передали на благодійні цілі понад 27,4 млрд грн. Згідно з даними Biz.Censor, за неповний 2024 рік ця сума досягла вже ~40 млрд грн. Отже, мобільні та онлайн-канали дедалі більше витісняють офлайн-збори, відкриваючи нові можливості для платформи донатів.

Системи цифрового фандрейзингу бувають різних типів. Існують як класичні краудфандингові платформи, так і мобільні/банківські додатки з благодійними функціями. Однак наразі саме традиційні канали – колективні скриньки та особисті волонтерські збори – залишаються найпопулярнішими: близько 29% українців роблять внески у стаціонарні скриньки в магазинах і супермаркетах, ще 11% – жертвують через волонтерів на публічних заходах [3]. Натомість лише незначна частина громадян користувалася краудфандингом. З технічної точки зору найпоширенішими є мобільні застосунки банків: наприклад, у Monobank – функція «банки», а в Приват24 – сервіс «конверти», де користувач може створити збір із

описом, ціллю і оновленням прогресу. Також дедалі частіше застосовуються QR-донати і смс-пожертви, а платіжні шлюзи використовуються благодійними фондами для прийому онлайн-платежів.

Попри технічний прогрес, цифрова благодійність має низку суттєвих викликів. Головна з них – низька довіра: багато донорів остерігаються, що їхні кошти підуть не за призначенням, а небагато хто перевіряє прозорість фондів. Як наголошується у дослідженні з UX-дизайну, відсутність чіткої інформації про те, куди потрапляють гроші, й нечітка подача мети проекту миттєво знижують довіру користувача [4]. Аналогічно, будь-яка «фрикція» у процесі пожертви – зайві кроки, заплутана навігація або неадаптованість під мобільні – часто призводить до відмови від донату. Ще один психологічний бар'єр – перевантаження вибором. Класичні спостереження показують, що коли донору пропонують забагато параметрів (різні суми чи категорії пожертв, багато опцій у формі), він просто «заморожується» й відмовляється діяти [5]. Тому зайва складність форм може нашкодити конверсії.

Інтерфейс додатку Donate24 показовий: він щоденно пропонує користувачам актуальні збори на потреби ЗСУ, дозволяючи відфільтрувати кампанії за темою, легко перейти до донату чи скопіювати реквізити з одним

Такі рішення посилюють ефект миттєвої дії – якщо користувач бачить кнопку «Пожертвувати зараз» на початку заклику, це значно підвищує відгук. Також мобільні застосунки використовують push-повідомлення для рекрутингу донорів і мотивації «дати зараз»: вони спонукатимуть зробити внесок своєчасно та нагадують «давати часто». Не менш важливий аспект – соціальний резонанс: коли люди діляться у мережах інформацією про власний донат, це може генерувати нові залучення (соціальне підтвердження) кліком (див. рис 1.1).

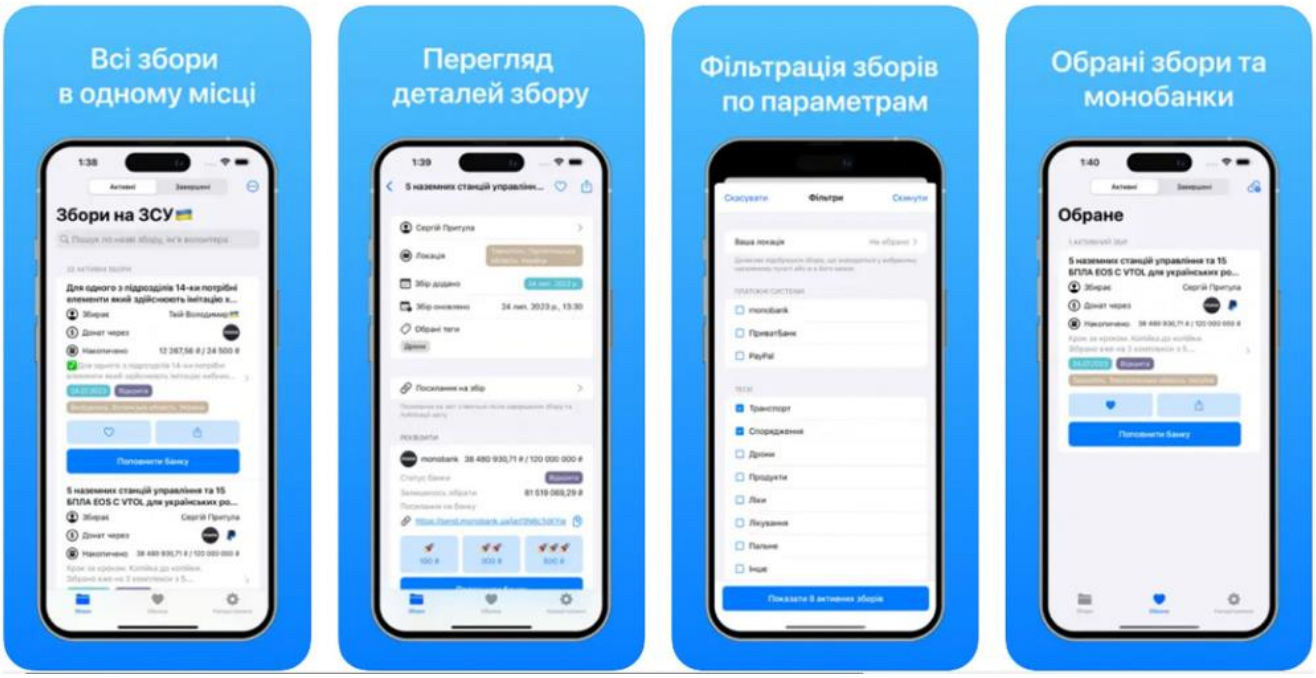

Рисунок 1.1 – Donate24 (за даними [6])

Найкращі практики у комунікації та технічному вирішенні формули руху донорів зосереджені на простоті та прозорості. Наприклад, показ реального прогресу кампанії у вигляді progress bar та конкретних числових статистик значно підвищують довіру користувача. Важливо також мінімізувати кількість полів у формі донату та забезпечити «UX у 2 кліки»: досвід показує, що перенесення головного заклику до дії на початок (наприклад, кнопка «Donate» у перших рядках листа) різко збільшує конверсію. Після завершення транзакції корисно одразу відображати екран-підтвердження з візуальною анімацією або коротким повідомленням-подякою, щоб донор відчув «ефект завершеності» (feedback) і розумів, що його пожертва зарахована. Автоматизовані листи чи підтвердження з докладними даними про витрати також допомагають утримати довіру і мотивують повторні внески.

Прикладами вдалих рішень є як українські розробки, так і світові сервіси. Зазначимо, що функція «банок» у мобільному додатку Monobank зібрала понад 1 млрд євро від початку війни, а минулого року через неї пройшло 43,68 млрд грн донатів [7] (Monobank оприлюднив ці дані). Подібний інструмент – «Конверти» у Приват24 – дозволяє групувати збори та прозоро показувати збірникам опис і

прогрес. Ще один приклад – сайт VolunteeringUkraine, який публічно відображає всі пожертви на фронтовий проєкт Front Line Kit, що надходять через платіжний шлюз WayForPay; так донори бачать повну прозорість усіх транзакцій. Міжнародні платформи як Patreon чи Donorbox стають стандартними для творчих і навчальних проєктів, а українські рішення активно запозичують ці підходи. Разом вони ілюструють прагнення зробити цифрову благодійність зручною та прозорою, що важливо для подальшого зростання фондів і залучення довіри суспільства.

1.2 Виявлення та вирішення проблем

Однією з ключових проблем у сфері мобільного фандрейзингу є перевантаженість та складність інтерфейсів мобільних додатків та платформ, що використовуються для збору коштів. В умовах цифрової економіки, коли увага користувача є вкрай короткотривалою, інтуїтивність та простота інтерфейсів мають надзвичайне значення для успіху благодійних кампаній. Багато платформ досі страждають від надмірної інформаційної завантаженості та незрозумілих процесів, які потребують зайвих кліків, тривалого очікування або зайвих дій, таких як заповнення форм чи підтвердження через численні кроки. Дослідження свідчать, що чим складнішим є процес пожертвування, тим вищою є ймовірність, що донор покине платформу, так і не завершивши транзакцію. Це суттєво знижує ефективність збору коштів і може завдати шкоди репутації самої організації, яка намагається залучити пожертви. Тому вкрай важливою є оптимізація користувацького досвіду, що передбачає скорочення кількості кроків, автоматичне заповнення інформації та інші рішення, які допомагають донору легко і швидко завершити свою транзакцію.

Ще одним важливим викликом є проблема недовіри до благодійних ініціатив та організацій, особливо в умовах активізації цифрових інструментів. В Україні цей аспект особливо актуальний, оскільки кількість благодійних платформ та ініціатив постійно зростає, а інформації про ефективність використання коштів часто бракує. Недовіра виникає насамперед через побоювання щодо потенційного шахрайства чи неправомірного використання коштів, оскільки користувачам складно перевірити,

чи дійсно кошти будуть витрачені за заявленим призначенням. Дослідження показують, що саме це є ключовою перепоною для донорів: вони бояться бути ошуканими або не бачать прямого результату своїх пожертв. Прозорість і чітке звітування про витрачені кошти, регулярні оновлення стану кампаній, а також публічна верифікація благодійних ініціатив могли б допомогти у подоланні цієї проблеми. Тим не менш, забезпечення такої прозорості часто вимагає значних ресурсів і спеціалізованих технологічних рішень.

Також серед ключових проблем цифрової благодійності варто зазначити регуляторні виклики. В Україні та в багатьох інших країнах законодавче поле щодо нових мобільних та цифрових інструментів збору пожертв ще перебуває у стадії формування. Особливо складним питанням є статус криптовалют, які дедалі частіше використовуються у сфері благодійності. Відсутність чіткого законодавчого регулювання криптовалютних транзакцій породжує численні складності для організацій, які приймають такі пожертви. Багато організацій змушені шукати способи обходу нормативних бар'єрів, що може призводити до ускладнень та правової невизначеності. Це не тільки створює додаткове навантаження для організацій, але й може відлякувати потенційних донорів, які прагнуть уникнути ризиків, пов'язаних із юридичними аспектами криптодонатів.

Наступною важливою проблемою є верифікація та ідентифікація організаторів збору коштів. З огляду на зростаючу кількість мобільних платформ і кампаній, існує реальна загроза появи шахрайських ініціатив, які намагаються використати благодійність для незаконного збагачення. Відсутність належних процедур верифікації та ідентифікації ініціаторів кампаній може сприяти поширенню таких шахрайських схем. Для подолання цього виклику необхідно запроваджувати системи, що дозволяють швидко і надійно перевіряти легітимність кампаній. Проте такі системи часто потребують значних ресурсів та додаткових кроків з боку самих організаторів, що може створити додаткові перешкоди для добросовісних волонтерів чи невеликих ініціатив.

Ще однією важливою проблемою є безпека даних користувачів мобільних платформ. Оскільки більшість пожертв здійснюється за допомогою мобільних

пристроїв, користувачі передають платформам значні обсяги персональної інформації, включаючи платіжні дані та інші чутливі відомості. Витоки такої інформації можуть мати серйозні наслідки не лише для донорів, але й для репутації самої платформи. В умовах зростаючої кіберзагрози організаціям необхідно приділяти особливу увагу стандартам кібербезпеки, таким як шифрування переданих даних, застосування багатофакторної автентифікації та використання надійних платіжних шлюзів. Недотримання цих стандартів може призвести до витоку інформації, фінансових втрат для користувачів та значних репутаційних збитків для організації.

Загалом, мобільний фандрейзинг в Україні та світі має чимало викликів, пов'язаних як із технічними аспектами, так і з довірою користувачів, правовими питаннями та безпекою. Вирішення цих проблем потребує комплексного підходу: оптимізації користувацьких інтерфейсів, підвищення прозорості благодійних кампаній, створення належних регуляторних умов, впровадження ефективних механізмів верифікації та забезпечення високих стандартів безпеки даних. Саме ці напрями мають стати ключовими у розвитку цифрової благодійності, яка здатна значно посилити соціальний вплив громадських ініціатив і зробити процес збору коштів доступнішим і безпечнішим для всіх учасників процесу.

1.3 Постановка задачі

У ході даної роботи нами буде досліджено предметну область мобільного фандрейзингу як перспективного напрямку цифрових комунікацій у сфері благодійності. Основна увага буде приділена аналізу технологічних, організаційних та регуляторних викликів, пов’язаних із розробкою мобільного застосунку для збору коштів.

Метою є виявлення вимог до системи, яка забезпечить прозору, зручну та безпечну взаємодію між донорами й організаторами зборів, а також створення архітектури програмного забезпечення, здатного підтримувати високі навантаження, швидкі транзакції та адаптивний інтерфейс для різних платформ.

Реалізація проєкту складається з наступних етапів:

- аналіз предметної області та поточних викликів мобільного фандрейзингу;
- формування функціональних та нефункціональних вимог до системи;
- розробка архітектури мобільного застосунку та проєктування його компонентів;
- створення інтерфейсу з урахуванням принципів доступності та адаптивності;
- розробка прототипу з базовим функціоналом автентифікації, перегляду зборів, пожертв та профілю;
- тестування, оцінка ефективності запропонованих рішень та аналіз результатів.

Ця робота спрямована на побудову прототипу мобільного додатку, що інтегрує сучасні підходи до UX/UI-дизайну, платіжної безпеки, персоналізації комунікацій та верифікації користувачів, що дозволить вирішити проблеми недовіри, складності інтерфейсів, шахрайства та нормативної невизначеності.

2 ФОРМУВАННЯ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

Виходячи з аналізу предметної галузі, визначено ключові особливості, які повинна враховувати мобільна фандрейзингова система. Додаток має максимально спрощувати процес донату, створюючи інтуїтивно зрозумілий інтерфейс і мінімізуючи кількість дій, необхідних для здійснення пожертви. Наприклад, застосунок Donate24 дозволяє користувачам повноцінно переглядати доступні збори, легко поповнювати рахунки одним дотиком, копіювати банківські реквізити для переказу, ділитися посиланнями на кампанії й стежити за статусом зборів [8]. Така функціональність демонструє, що користувачі повинні мати змогу швидко знайти релевантну кампанію, одним рухом зробити внесок та за необхідності поділитися інформацією зі спільнотою. Щоб знизити когнітивне навантаження, інтерфейс має бути адаптований під різні платформи – екран телефону чи планшету – і заохочувати застосування мобільних платежів, соціальних входів чи сканування QR-кодів для авторизації. Усе це дозволяє досягнути мети «мінімізації тертя» – тобто максимальної простоти і швидкості дій користувача на кожному етапі.

Одночасно система має враховувати, що фандрейзинг орієнтований на побудову довіри між донорами та тими, хто збирає кошти. Це реалізується через двосторонню комунікацію: донори повинні мати канал для зворотного зв’язку, отримувати оновлення про цілі та результати кампаній, а благодійники – надсилати інформаційні повідомлення про використання коштів. Двостороння комунікація, за якою відкритість інформації і зворотний зв’язок рівнозначно важливі, сприяє формуванню довіри raisely.com. Функціонально це означає наявність внутрішньо-додаткового чату або форуму запитань і відповідей, можливість залишити коментар чи відгук, а також регулярні push-повідомлення з оновленнями. Важливо врахувати, що увімкнені пуш-повідомлення значно підвищують активність користувачів: за даними маркетингових досліджень, користувачі, що дозволили надсилання сповіщень, у середньому демонструють на 53 % більше активних сеансів на місяць порівняно з тими, хто їх вимкнув [9]. Тому система повинна

делікатно запитувати згоду на отримання push-повідомлень і давати змогу легко їх відключати.

Функціональні вимоги мають охоплювати повний цикл благодійної кампанії. По-перше, система повинна підтримувати різні ролі користувачів: донорів, волонтерів чи представників організацій, які створюють збір, а також модераторів і адміністраторів платформи. Наприклад, роль «донор» може мати лише право долучатися до існуючих кампаній та отримувати персоналізовані повідомлення, а роль «організатор» – створювати нові збори, описувати їх мету і слідкувати за ходом зборів. Для контролю за безпекою та прозорістю введення кампаній передбачається ідентифікація волонтера чи благодійної організації: платформа може вимагати верифікацію особи та підтвердження деталей проекту ще до публікації збору [10]. Це дозволяє запобігати шахрайству, адже всі кампанії проходять модерацію та перевірку легітимності. Така функціональність модерації може містити блокування підозрілих кампаній, звітність користувачів про неналежний контент та алгоритми оцінки ризиків транзакцій. У випадку виявлення шахрайства платформа має мати механізми повернення пожертв і повідомлення служб безпеки.

По-друге, ключовою функцією є власне процес пожертвування. Система повинна інтегруватися з провайдерами платежів і банківськими сервісами. Сучасні мобільні платформи пропонують низку способів: оплата карткою, через мобільний гаманець, банківські перекази, а також API відкритого банкінгу. Для українських користувачів можна реалізувати інтеграцію з локальними платіжними системами – наприклад, через QR-коди monobank чи ПриватБанку, а для міжнародних – через шлюзи Visa/Mastercard, PayPal або аналогічні сервіси. Важливою вимогою є безперебійна обробка платежів і підтримка PSD2/SCA: у Європі директива PSD2 вимагає сильну двофакторну аутентифікацію під час онлайн-платежів, тобто принаймні два фактори з категорій: знання, володіння або біометрія. Система має передбачати співпрацю з платіжними провайдерами, які реалізують ці норми та дозволяти донорам проходити процес оплати з мінімальним ризиком. Одночасно має забезпечуватися зворотний зв’язок про успішність платежу: надсилання

електронних листів чи пуш-повідомлень з підтвердженням або квитанцією про пожертвування та подальшими інструкціями.

Крім того, функціонально необхідні механізми персоналізації та комунікації. Наприклад, система може автоматично формувати і відправляти доповіді донорам про те, куди було спрямовано їх кошти, у вигляді push-нотифікацій або e-mail. При цьому користувачі мають мати змогу обирати канали зв'язку і налаштовувати їх. Вимогою також є підтримка webhooks і API для зовнішніх інтеграцій: наприклад, при кожній успішній пожертві система може надсилати вебхук у зовнішню CRM, служби аналітики чи бухгалтерські сервіси, що дозволяє синхронізувати дані про донорів, суми пожертв тощо. Також можуть бути реалізовані інтеграції із соціальними мережами для швидкого розповсюдження інформації про кампанію. Ці функціональні можливості мають працювати чітко та стабільно, особливо під час пікових навантажень, таких як масові кампанії або екстрені збори.

Важливо, щоб усі функції інтерфейсу відповідали принципам UX/UI-дизайну для благодійних платформ. Інтерфейс має бути формально стриманим і водночас дружнім до користувача: зрозуміла навігація між категоріями зборів, виразні заклики до дії для пожертв та підписки, адаптивний дизайн під різні екрани. Різні мобільні платформи повинні підтримувати єдиний стиль та стандартні елементи взаємодії. Необхідно передбачити локалізацію інтерфейсу на різні мови і налаштування валют для міжнародних донорів. Усе це задля того, щоб користувачі відчували зручність і впевненість незалежно від регіону. При розробці UX/UI слід враховувати принципи Accessibility, забезпечивши адаптацію контенту під різні допоміжні технології.

Нефункціональні вимоги охоплюють архітектуру, продуктивність та безпеку системи. По-перше, архітектура повинна бути масштабованою та відмовостійкою. Оскільки мобільний додаток працюватиме з великою кількістю паралельних користувачів і транзакцій, необхідно використовувати принципи розподілених систем. Практичними підходами можуть бути мікросервіси або серверлесс-функції, кластеризація серверів та горизонтальне масштабування. Достатній резерв продуктивності забезпечується використанням балансувальників навантаження,

кешування даних та CDN для статичних ресурсів. Архітектура повинна допускати автоматичне горизонтальне масштабування у хмарі, що гарантує високу доступність. Надійна інфраструктура складається з кількох зон чи дата-центрів, де в разі збою однієї локації трафік перенаправляється в резервні кластери. Слід встановити вимоги до SLA, наприклад, 99,9 % доступності сервісу.

По-друге, до нефункціональних належать вимоги до продуктивності й стабільності. Система має витримувати пікові навантаження: тисячі одночасних платежів і запитів на перегляд. Для цього необхідно проводити навантажувальне тестування й аналіз «вузьких місць» на ранніх етапах. Робочі навантаження можуть корисно вирівнюватися через черги повідомлень для обробки повідомлень між сервісами. У разі виникнення збоїв чи помилок повинна спрацювати система моніторингу та сповіщень, щоб швидко відновити роботу, коли щось піде не так. Регулярне резервне копіювання даних є обов'язковим – наприклад, з повним бекапом раз на добу і інкрементальними знімками через кожні декілька годин. Резервні копії мають зберігатися мінімум у двох незалежних локаціях для гарантії цілісності інформації.

Третій блок нефункціональних вимог – безпека. Платформа працює з чутливими персональними даними донорів та фінансовими транзакціями, тому необхідно прийняти підхід Zero Trust. Це означає, що жоден компонент системи не має повністю довіряти іншому автоматично: доступ до ресурсів дозволяється лише після ретельної ідентифікації та авторизації користувача або сервісу nccoe.nist.gov. У практичному сенсі застосовується багаторівнева аутентифікація користувачів, розмежування прав доступу за ролями, захищений канал зв'язку для всіх API-викликів і шифрування даних у стані спокою. Оскільки додаток мобільний, особливу увагу варто приділяти захисту від зломів пристроїв: використання безпечного сховища ключів, валідація цілісності клієнтської частини, застосування політик MDM/EMM при потребі. Для серйозних функцій може вимагатися повторна двофакторна аутентифікація.

Також необхідно забезпечити кібербезпеку інфраструктури: регулярне оновлення серверного ПЗ, виявлення та виправлення вразливостей, сегментація

мережі. Система аудиту логів повинна фіксувати критичні події, щоб у разі інцидентів можна було провести розслідування. Усі ці заходи відповідають принципам Zero Trust – жодна операція не вважається безпечною за замовчуванням [10].

Регуляторні вимоги диктуються законодавством про захист персональних даних, фінансові транзакції та благодійну діяльність. Якщо платформа працює з європейськими користувачами, потрібно суворо виконувати норми GDPR. Це передбачає мінімізацію збору даних – наприклад, просити в користувачів тільки необхідний мінімум інформації і пояснювати їх призначення. Система повинна отримувати явну та інформовану згоду донорів на обробку їхніх персональних даних і забезпечувати можливість доступу користувачів до своїх даних відповідно до статей GDPR. Не менш важливою є політика приватності з чітким описом, для чого використовується інформація, які є права користувачів і як довго зберігаються дані. У технічному плані це означає обов'язкове шифрування персональних даних у базах, доступ до них лише за необхідності та регулярні перевірки на відповідність стандартам. Водночас благодійні платформи мають дотримуватись і локальних нормативів – наприклад, Закону України «Про захист персональних даних» і вимог НБУ щодо проведення платіжних операцій.

Щодо платіжних сервісів, крім PSD2/SCA в ЄС, повинні дотримуватися й інші фінансові регламенти. Наприклад, при роботі з кредитними картками необхідно виконувати стандарти PCI DSS для збереження безпеки платіжної інформації. Платіжні провайдери зазвичай сертифіковані за цим стандартом, але система має забезпечити, щоб токени карток ніколи не зберігались у відкритому вигляді, а всі платежі проходили через перевірені шлюзи. Оскільки благодійні перекази можуть бути підозрілими, важливо врахувати вимоги AML/KYC: для великих пожертв може бути передбачено автоматичне розслідування або інформування банківської установи про аномальні транзакції.

Не можна забувати і про GDPR/PSD2 з точки зору взаємодії з користувачем. Зокрема, оскільки додаток активно використовує push-сповіщення, треба врахувати, що отримання згоди на нотифікації також регламентується загальними

принципами згоди користувача. Крім того, якщо застосунок передбачає обмін інформацією між Україною та ЄС чи іншими країнами, потрібно забезпечити законність таких трансферів даних.

У підсумку, вимоги до системи мають бути всеосяжними та взаємопов’язаними. Функціональні сценарії, нефункціональні очікування і регуляторні норми повинні органічно поєднуватись. Наприклад, підтримка високого навантаження та резервування даних гарантує, що платформа працюватиме навіть під час масових мобілізаційних кампаній; безпечна нуль-довірча архітектура зі шифруванням і верифікацією дозволяє будувати довіру в очах користувачів; а уважність до UX/UI – зручні форми, адаптивність та мультимовність – сприятиме мінімізації тертя у користувачів різних культур і регіонів. Інтеграція push-повідомлень, контролю контенту та зовнішніх вебхуків доповнює цей комплекс, забезпечуючи своєчасну комунікацію, захист від шахраїв та гнучкість у взаємодії з іншими сервісами. Лише такий комплексний, аргументований підхід до формування вимог дозволить створити мобільну фандрейзингову систему, яка буде надійною, масштабованою і зручною для української та міжнародної аудиторії.

3 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 UML проектування ПЗ

Для формування прозорої картини архітектури мобільного клієнта системи організації збору коштів було розроблено UML-діаграму, що фіксує склад і взаємодію ключових компонентів застосунку. Така візуалізація на ранньому етапі дозволила окреслити основні сутності, їхні обов’язки та зв’язки з іншими підсистемами, що є визначальним для командної розробки й подальшої підтримки проєкту.

Діаграма демонструє логіку мобільної частини в контексті всієї системи: від модулів керування ініціативами та профілем користувача до історії донатів, підписок і комунікаційних подій (email- та push-сповіщення).Завдяки цьому забезпечується узгодженість між реалізацією функціоналу в межах мобільної частини системи та вимогами, закладеними у загальне технічне завдання. UML-діаграма наведене у рисунку 3.1.

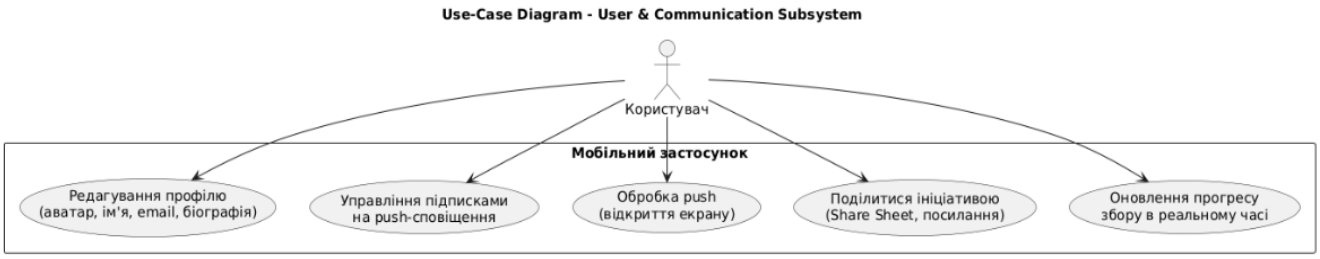

Рисунок 3.1 – UML-діаграма мобільної частини (рисунок виконано самостійно)

Нижче розглянуто кожен елемент більш детально.

У представленій діаграмі актором виступає користувач мобільного застосунку, а всі сценарії взаємодії згруповані всередині єдиного програмного контексту «Мобільний застосунок».

Сценарій «Редагування профілю» охоплює будь-які дії, пов’язані зі зміною персональних даних: від оновлення аватара до корекції імені, електронної пошти й короткої біографічної довідки; його виконання безпосередньо модифікує локальний стан інтерфейсу і ініціює синхронізацію з віддаленим сервером. Сценарій «Управління підписками на push-сповіщення» надає змогу користувачеві

регулювати, які категорії повідомлень він отримуватиме, після чого зміни одразу передаються в систему доставляння й стають базою для подальшої фільтрації push-трафіку. «Обробка push» описує реакцію клієнтського додатка на вже отримане повідомлення: розпаковування його вмісту, визначення цільового екрану й детермінований перехід до відповідного фрагмента інтерфейсу.

Функціональний блок «Поділитися ініціативою» охоплює виклик нативного діалогу поширення, автоматичну генерацію короткого посилання та передавання метаданих, що забезпечує швидку соціальну дистрибуцію контенту за межі застосунку. Нарешті, «Оновлення прогресу збору в реальному часі» представляє підписку клієнтської частини на потік подій WebSocket або Server-Sent Events; ці події асинхронно змінюють відображення прогрес-бару і забезпечують користувачеві відчуття безперервної участі.

Усі сценарії реалізовані як незалежні випадки використання, однак вони поділяють спільну ресурсну базу (мережевий шар, локальний кеш, сервіс push-нотифікацій) і формують цілісний цикл персоналізованої взаємодії користувача з благодійними ініціативами.

3.2 Проєктування архітектури ПЗ

Вибір Kotlin як основної мови реалізації мобільного застосунку обумовлений його здатністю поєднувати сумісність із перевіреною екосистемою Java та водночас пропонувати сучасні мовні конструкції, які підвищують продуктивність розробки й якість кінцевого коду. Kotlin завдяки офіційній підтримці з боку Google інтегрується в усі інструменти Android-стека, забезпечуючи повну підтримку Jetpack Compose, що дає змогу описувати інтерфейс декларативно й істотно скорочує обсяги шаблонного коду. Мова пропонує вбудовану null-безпеку, яка мінімізує ризик критичних помилок часу виконання, водночас її корутини забезпечують легковаге та читабельне керування асинхронними операціями без верстви callback-інверсій, що позитивно позначається на реактивності інтерфейсу й енергоспоживанні пристрою.

Завдяки повній бінарній сумісності з Java використання Kotlin не накладає обмежень на повторне застосування наявних бібліотек або поступовий рефакторинг спадкових модулів, а можливість мультиплатформової компіляції відкриває перспективу спільної логіки між Android, iOS та веб-клієнтами, що стратегічно полегшує подальшу еволюцію проєкту.

Крім того, компактний синтаксис, підтримка функціональних парадигм і розвинена система розширень підвищують читабельність і тестованість коду, знижуючи загальні витрати на підтримку й забезпечуючи вищу швидкість впровадження змін, що є критичним для динамічних фандрейзингових сервісів, де час виходу нових функцій безпосередньо впливає на залучення користувачів і конкурентоспроможність продукту.

Kotlin і React посіли ключові позиції у сучасному стеку розробки завдяки спільному прагненню мінімізувати шаблонний код, підвищити виразність програмного полотна та забезпечити високий рівень масштабованості проєктів. Kotlin – як мова, що зародилася в екосистемі JVM, але з часом набула мультиплатформових можливостей пропонує розробникам синтаксис, у якому функціональні та об’єктно-орієнтовані парадигми не конфліктують, а взаємодоповнюють одна одну. Його типова система з вбудованою null-безпекою, лямбда-виразами й корутинами дозволяє створювати асинхронні процеси без перевантаження кодової бази callback-конструкціями, забезпечуючи при цьому безпеку пам’яті та високий ступінь читабельності.

React, у свою чергу, змінив правила гри у фронтенд-розробці, запровадивши декларативну модель побудови інтерфейсу, де стан компонента виступає єдиним джерелом правди, а віртуальний DOM гарантує мінімальні витрати на перерендер.

Поєднання Kotlin і React стало можливим завдяки Kotlin/JS, що компілює Kotlin-код у JavaScript і забезпечує типову сумісність із React-API. У такій конфігурації розробники зберігають переваги строгої типової системи та корутин навіть у браузерному середовищі, використовуючи DSL-підхід для оголошення компонентів. Це особливо цінно для масштабних SPA-застосунків, де необхідна жорстка перевірка типів і повторне використання бізнес-логіки на різних

платформах; один і той самий модуль Kotlin Multiplatform може обслуговувати мобільний клієнт на Android (Kotlin/JVM або Compose), iOS-додаток (Kotlin/Native) та веб-інтерфейс (Kotlin/JS + React), скорочуючи витрати на підтримку й синхронізацію функціоналу. У додаток до згаданих переваг Kotlin забезпечує повну бінарну сумісність із Java, що дає змогу інтегрувати перевірені бібліотеки для мережевої взаємодії, тестування чи диференційованої логіки, тоді як React завдяки розгалуженій екосистемі npm надає безліч UI-фреймворків, інструментів стан-менеджменту і готових шаблонів для серверного рендерингу. Синергія цих технологій проявляється у всіх шарах розробки: на рівні продуктивності команд, які можуть використовувати єдині підходи до написання чистих, модульних компонентів; на рівні інфраструктури CI/CD, де артефакти компілюються, тестуються та деплоїться уніфіковано; й на рівні кінцевого користувача, що отримує швидкий, чутливий інтерфейс із комплексною бізнес-логікою, перевіреною одноразово у спільному коді. Таким чином, Kotlin та React у будь-якій їхній конфігурації, чи то через Kotlin/JS, чи через класичну інтеграцію REST- або GraphQL-шарів – утворюють конкурентоспроможний фундамент для створення повноцінних кросплатформових систем, у яких сувора типова безпека й декларативна побудова інтерфейсу поєднуються, щоб гарантувати високу якість коду, прискорений цикл розробки й масштабованість у довгостроковій перспективі.

На рисунку 3.2 наведено приклад стандартної спрощеної архітектури подібного мобільного клієнту.

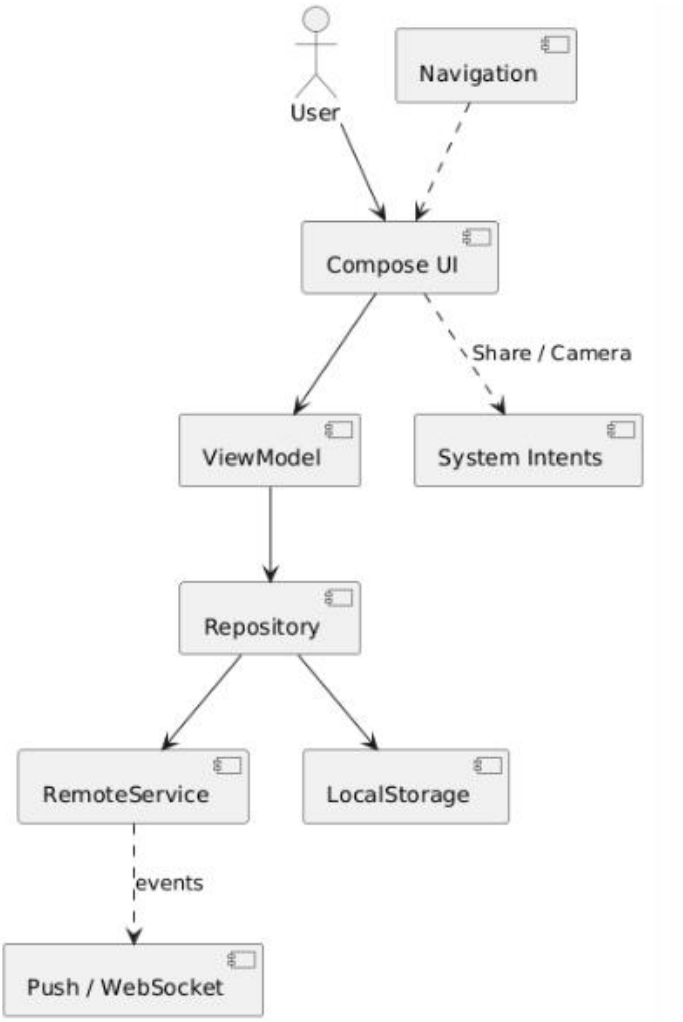

Рисунок 3.2 – Архітектура мобільного клієнту (рисунок виконано самостійно)

Схема відображає типовий потік даних усередині клієнтського Android-застосунку, реалізованого на Kotlin із використанням Jetpack Compose. Користувач взаємодіє з інтерфейсом, сформованим у шарі Compose UI; навігаційний компонент забезпечує переходи між екранами, але не містить бізнес-логіки, залишаючись допоміжним механізмом маршрутизації. Усі запити, що походять із UI, прямують до ViewModel, яка акумулює стан екрана, оркеструє асинхронні операції та ізолює представлення від деталей отримання даних.

ViewModel делегує роботу з джерелами інформації репозиторію, який виступає єдиною точкою доступу й абстрагує доменну логіку від конкретних реалізацій зберігання. Репозиторій звертається до RemoteService для мережевих операцій – сюди входять REST-запити, WebSocket-канали та система push-сповіщень, що надсилає повернені події назад у UI. Для офлайнової стійкості і

кешування ті самі дані синхронізуються в LocalStorage, представлений Room або DataStore. Нарешті, сам інтерфейс може ініціювати системні інтенти, наприклад, відкривати камеру чи викликати стандартний Share Sheet, не залучаючи при цьому проміжні шари.

Така конфігурація забезпечує чітке розділення відповідальностей: UI залишається тонким, ViewModel керує станом, репозиторій інкапсулює дані, а віддалені та локальні сервіси легко підмінюються без впливу на верхні рівні.

3.3 Проектування UI/UX дизайну

У межах підсистеми мобільного застосунку, що відповідає за управління профілем користувача, сповіщення та функціональні доповнення, розроблено інтерфейси, які забезпечують інтуїтивну та комфортну взаємодію з ключовими елементами системи. Основною метою дизайну є збереження балансу між простотою, швидкістю доступу до важливих функцій та відповідністю сучасним принципам мобільної UX-архітектури.

UI/UX рішення зосереджені на забезпеченні персоналізації взаємодії (через профіль), контекстної залученості (через push-сповіщення) та поширення ініціатив (через систему «Поділитися»). Також реалізовано підтримку динамічного оновлення прогресу зборів у режимі реального часу, що посилює ефект живої участі для користувача. Усі екрани дотримуються єдиної стилістики застосунку, з урахуванням принципів accessibility, адаптивності до різних розмірів екранів та локалізації. За ца відповідають єкрани профілю користувача, авторизації та автентифікації та редагування профілю.

Сторінка профілю користувача (User Profile Screen). Відображає базову інформацію про користувача, історію його активності та надає доступ до налаштувань облікового запису.

Сторінка профілю користувача:

- відображення аватара, імені, email та короткої біографії;
- відображення суми донатів та кількості підтриманих ініціатив;

- перехід до редагування профілю;
- перегляд історії донатів;
- доступ до налаштувань сповіщень;
- функція «поділитися ініціативою»;
- вихід із облікового запису. Сторінка редагування профілю:
- завантаження або зміна аватара з галереї/камери;
- редагування імені, email, короткої біографії;
- валідація введених даних;
- збереження змін.

Сторінка авторизації та автентифікації:

- вхід за email і паролем;
- реєстрація нового користувача;
- вхід через google oauth;
- верифікація email через код;
- автоматичне перенаправлення до профілю після входу.

Сторінка налаштувань сповіщень:

- увімкнення/вимкнення сповіщень про нові збори
- увімкнення/вимкнення сповіщень про досягнення цілей зборів
- обробка пушів і перенаправлення до відповідного екрану
- тестування отримання сповіщення Діалог «Поділитися ініціативою»:
- виклик системного діалогу share sheet;
- формування короткого опису;
- генерація лінку на конкретну ініціативу.

Компонент реального часу:

- підключення до websocket або eventstream;
- миттєве оновлення прогрес-бару збору;
- динамічне відображення нових донатів або змін статусу.

Загалом було розроблено екрани для перегляду та редагування персональних даних, зміни аватара, а також впроваджено можливість авторизації та верифікації користувача.

Окрему увагу було приділено налаштуванням push-сповіщень із підтримкою переходу до відповідного екрану застосунку при їх відкритті. Крім того, було реалізовано функціонал поширення ініціатив за допомогою системного діалогу

«Поділитися», що дозволяє швидко передавати інформацію про збір через зовнішні сервіси.

Також було впроваджено компонент оновлення в реальному часі, який забезпечує динамічне відображення прогресу збору коштів на сторінці ініціативи. Усі функціональні частини підсистеми було розроблено відповідно до загального дизайну застосунку, з урахуванням адаптивності, зручності використання та вимог мобільної UX-архітектури.

Нижче на рисунку 3.3 неведено приклад інтерфейсу типового сповіщення.

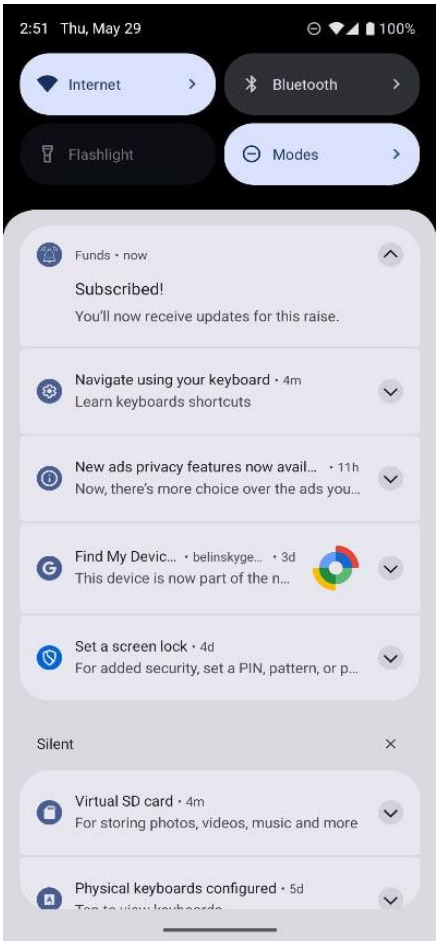

Рисунок 3.3 – Інтерфейс сповіщень (рисунок виконано самостійно)

На екрані відображено розгорнуту «шторку» повідомлень Android-пристрою. У верхній частині розміщено панель швидких налаштувань із плитками Internet, Bluetooth, Flashlight та Modes, які дозволяють миттєво вмикати чи вимикати ключові системні функції. Під панеллю розташована стрічка активних сповіщень: першим іде повідомлення від застосунку Funds з позначкою «now», що інформує користувача про успішну підписку на оновлення конкретного збору («Subscribed! You'll now receive updates for this raise») і тим самим підтверджує активацію push-каналу. Далі подано стандартні системні сповіщення Google-сервісів, згруповані за каналами та відсортовані за часом надходження.

Компоновка ієрархічно розділяє взаємодії: важливі повідомлення виділені окремим блоком із розгортуваними деталями, тоді як фонові сповіщення не відволікають користувача, залишаючись доступними для перегляду за потреби.

Так користувачу будть надходити повідомлення по підписці та сборам.

Нмжче на рисунку 3.4 наведено приклад функції «Поділитись».

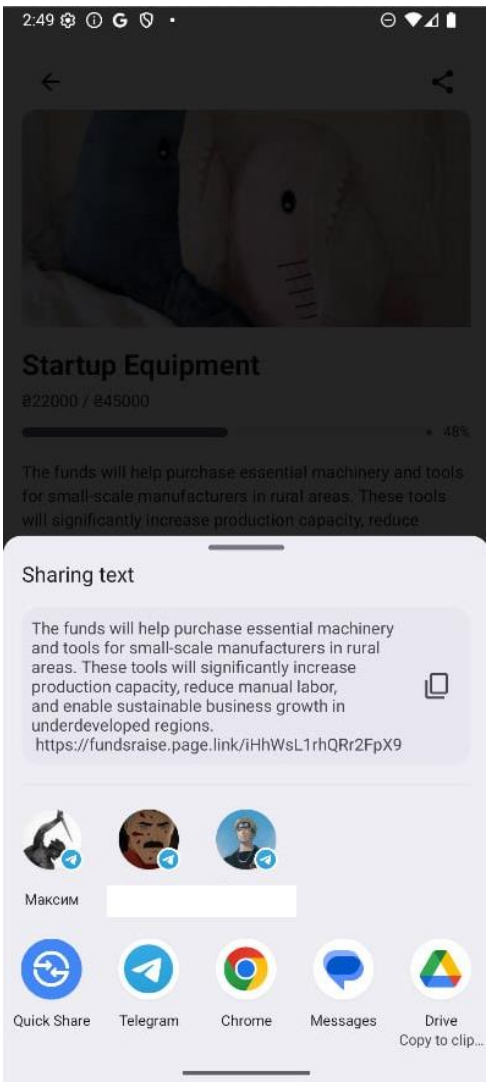

Рисунок 3.4 – Інтерфейс «Поділитись» (рисунок виконано самостійно)

На екрані представлено реалізацію нативної функції «Поділитися ініціативою», що є однією з ключових вимог технічного завдання мобільного застосунку для збору коштів. Після натискання піктограми «share» на сторінці детальної інформації про кампанію (у верхньому тлі напівпрозоро видно картку ініціативи «Startup Equipment» із прогрес-баром 48 %), система відкриває стандартний Android-діалог Share Sheet. Поле «Sharing text» уже заповнене автозгенерованим повідомленням: коротким, але змістовним описом цілі збору та унікальним посиланням на сторінку кампанії в домені fundsraise.page.

Користувач може одразу скопіювати цей текст або надіслати його через запропоновані канали. У верхньому рядку діалога відображаються «пріоритетні адресати» – контакти, з якими відправник найчастіше ділиться даними; нижче

розміщено основні додатки для поширення: Quick Share, Telegram, Chrome, Messages, Drive тощо.

Такий підхід у контексті нашого застосунку виконує дві критичні функції: по-перше, мінімізує бар’єр для вірусного розповсюдження ініціативи, оскільки користувачеві не потрібно самостійно формулювати повідомлення чи шукати посилання; по-друге, забезпечує узгодженість контенту, гарантуючи, що потенційні донори отримають чітке пояснення мети збору й прямий доступ до сторінки внеску з будь-якого підтримуваного месенджера або сервісу.

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Файлова структура та діаграма діяльності

Далі для ефективної організації коду, забезпечення масштабованості та зручності подальшої підтримки мобільного застосунку, важливим етапом є проєктування архітектури програмного забезпечення. Саме архітектура визначає взаємозв’язки між компонентами, розподіл відповідальностей та підхід до інтеграції функціональних модулів.

У цьому контексті доцільно розглянути файлову структуру застосунку як один із ключових аспектів архітектурної організації. Чітко визначена структура папок та файлів дозволяє уникнути дублювання логіки, спрощує навігацію в кодовій базі та полегшує командну роботу. Крім того, вона відображає обрані архітектурні принципи (наприклад, розділення за доменами або за типами компонентів) і забезпечує основу для масштабування функціоналу в майбутньому.

Файлова система проєкту FUNDS організована за типовою для Android-модуля схемою Gradle, де кореневі службові каталоги (.gradle та .idea) зберігають метадані збірки і конфігурацій середовища, тоді як усі артефакти компіляції та вихідний код ізосереджено в модулі app.

Усередині app знаходиться папка build, що формується інструментами Gradle під час збірки; вона містить підкаталоги generated й intermediates з автоматично згенерованими ресурсами (наприклад, res-значення, PNG-спрайти, Google Services JSON) та проміжними класами, які створюються плагінами Android і Kotlin перед остаточною лінковкою (див. рис. 4.1).

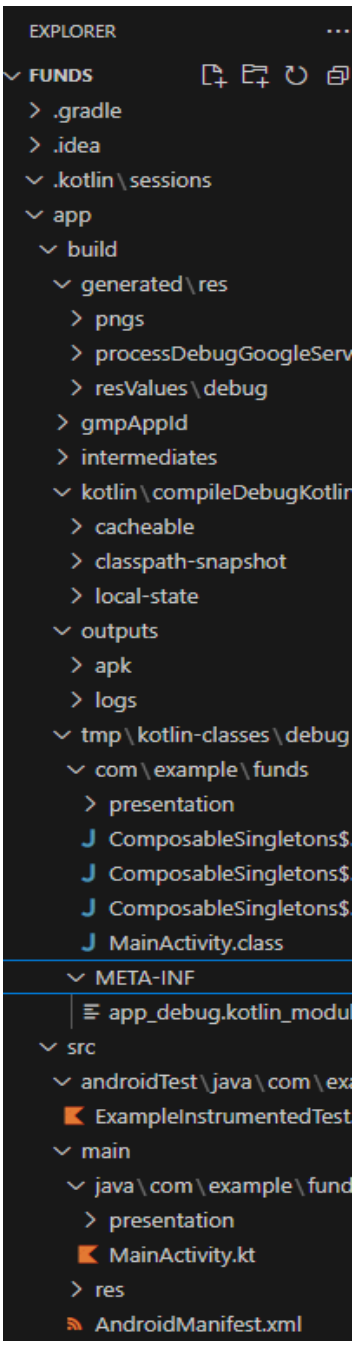

Рисунок 4.1 – Файлова структура (рисунок виконано самостійно)

Підкаталог kotlin у intermediates зберігає кешовані результати компіляції корутин і snapshot-дані, що прискорюють інкрементальні збірки. Каталог outputs розбито на apk і logs: перший містить готові пакети, підписані для дебагу чи релізу, другий – журнали збірки і тестів. Тимчасова директорія tmp/kotlin-classes/debug зберігає байт-код .class, що завантажується під час запуску відладочного APK; тут же видно класи, що згенерував Jetpack Compose (наприклад ComposableSingletons$…), а також компільований MainActivity.class. У структурі

src традиційно виділено два основних джерельних набори: androidTest, де розташовано інструментальні тести для пристроїв (підпростір com.example.funds), та main, що містить виробничий код. Усередині src/main/java/com/example/funds видно підпакет presentation, який зосереджує екрани та компоненти Jetpack Compose, а файл MainActivity.kt слугує точкою входу застосунку. Папка res у тому ж наборі main містить статичні ресурси (макети, стилі, зображення), а AndroidManifest.xml описує метадані пакета.

Така ієрархія відділяє машиногенеровані й кешовані артефакти від ручного коду, забезпечує чисту збірку через просте видалення каталогу build та дозволяє Gradle чітко ізолювати результати кожного підпроєкту, що спрощує як командну розробку, так і автоматичні CI-процеси.

Далі варто розглянути діаграму діяльності за наступним функціоналом:

- особистий кабінет, налаштування профілю користувача;
- перелік історії донатів з датою, сумою та назвою ініціативи;
- можливість переглянути деталі кожного донату;
- редагування аватара (завантаження/кадр із камери);
- зміна ім’я, email, короткої біографії.

Push Notifications:

- підписка/відписка на пуш-сповіщення про;
- нові збори у підписаних ініціативах;
- досягнення цілей зборів.

Обробка отриманих пушів у додатку (відкриття релевантного екрану)

Нативна функція «Поділитися ініціативою»:

- виклик стандартного діалогу share sheet;
- передача лінку на збір (також короткий опис).

Нижче на рисунку 4.2 наведена діаграма діяльності.

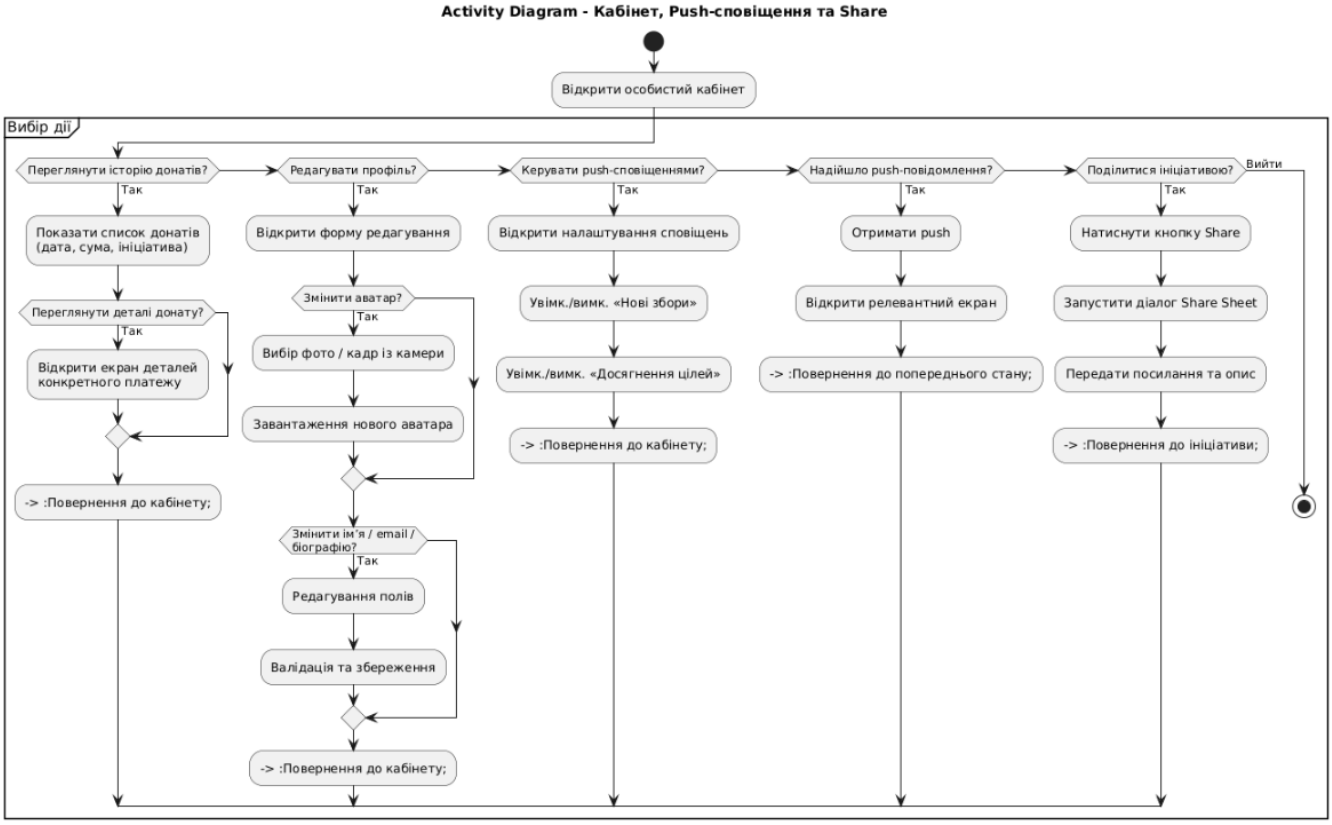

Рисунок 4.2 – Діаграма діяльності (рисунок виконано самостійно)

Діаграма діяльності відтворює повний цикл взаємодії користувача всередині особистого кабінету благодійного застосунку та пов’язаних із ним сервісних функцій. Після входу користувач опиняється у центральному вузлі, звідки може вибрати одну з кількох гілок: відкрити список власних донатів і, за потреби, переглянути деталі конкретного платежу; перейти до форми редагування профілю, де підтримується як зміна аватара через камеру чи галерею, так і корекція персональних полів із обов’язковою валідацією; керувати push-підписками, увімкнувши або вимкнувши сповіщення про нові збори чи досягнення цілей; відреагувати на отримане push-повідомлення, що автоматично переадресує його на релевантний екран ініціативи; або скористатися нативною функцією поширення, натиснувши кнопку Share, яка викликає системний діалог і передає попередньо сформований опис із посиланням на збір.

Після завершення кожної дії користувач повертається до початкового стану кабінету, підтримуючи безперервність навігації без втрати контексту. Таким чином діаграма фіксує, як підсистема профілю, сповіщень і соціального поширення

інтегрується в єдину логіку та забезпечує інтуїтивний доступ до найважливіших сценаріїв взаємодії зі зборовими ініціативами.

4.2 Цікаві методи та алгоритми

Далі розглянуто код шерингу сторінки ініціативи:

```
un shareRaise(context: Context, raise: Raise) {
    val dynamicLink = Firebase.dynamicLinks.dynamicLink {
        link = "https://fundsraise.com/raise?id=${raise.id}".toUri()
        domainUriPrefix = "https://fundsraise.page.link"
        androidParameters { }
    }
    val dynamicLinkUri = dynamicLink.uri
    Firebase.dynamicLinks.createDynamicLink()
      .setLink("https://fundsraise.com/raise?id=${raise.id}".toUri())
        .setDomainUriPrefix("https://fundsraise.page.link")

setAndroidParameters(DynamicLink.AndroidParameters.Builder().build())
        .buildShortDynamicLink()
        .addOnSuccessListener { result ->
            val shortLink = result.shortLink
            val intent = Intent(Intent.ACTION_SEND).apply {
                type = "text/plain"
                putExtra(Intent.EXTRA_TEXT, "${raise.description} \n
$shortLink")
            }
            context.startActivity(Intent.createChooser(intent, "Share
via"))
        }
        .addOnFailureListener {
            Toast.makeText(context, "Failed to create link",
Toast.LENGTH_SHORT).show()
        }
}
```

Тут функція shareRaise інкапсулює весь робочий цикл створення й поширення динамічного посилання на конкретний збір (об’єкт Raise). Спочатку за допомогою Kotlin-DSL для Firebase Dynamic Links формується «довге» посилання:

в ньому основний URL сторінки ініціативи доповнюється параметром id, а також указується префікс власного домену fundsraise.page.link, що забезпечує коректний deep-linking усередині мобільного застосунку.

Одразу після цього викликається метод buildShortDynamicLink(), який асинхронно генерує коротку версію того ж лінка; це не лише зменшує обсяг передаваного тексту, а й підвищує клікабельність у зовнішніх каналах. У колбеку addOnSuccessListener утворюється інтент типу ACTION_SEND із MIME-типом text/plain: у поле EXTRA_TEXT підставляється опис ініціативи та сформоване коротке посилання, після чого система відкриває стандартний діалог вибору застосунку для поширення.

У випадку, якщо створення динамічного лінка завершується помилкою, користувач негайно отримує зворотний зв’язок через Toast із повідомленням «Failed to create link». Таким чином функція забезпечує повністю автономний, зручний для користувача механізм соціального поширення ініціатив без необхідності ручного копіювання URL або опису.

5 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У цьому розділі розглянуто декілька сценаріїв тест-кейсів для візуального розуміння процесу розробки та прогресу виконання кожного етапу відповідно технічному завданню. Далі у таблиці 5.1 наведено тестові сценарії.

Таблиця 5.1 – Тест-кейс №1 (таблиця виконана самостійно)

Інформація про тест-кейс			
Ідентифікатор тесту:	Тест-кейс №1		
Власник тесту:	Блувбанд Кірілл Ігорович		
Дата створення:	23.05.2025		
Мета тесту:	гарантувати, що особистий кабінет коректно відображає історію донатів та дозволяє редагувати профіль, а підписки / push-сповіщення, функція «Поділитися» й оновлення прогресу в реальному часі працюють безвідмовно.		
Особистий кабінет і редагування профілю			
№	Опис випадку	Очікуваний результат	Висновок
1	Відкрити вкладку Profile	Екран показує аватар, ім’я, email, коротку біо й лічильник сумарних донатів	Пройдено
2	Свайпнути «Donations history»	Таблиця із трьома останніми донатами: дата, сума, назва ініціативи	Пройдено
3	Тап по рядку донату #2	Відкривається діалог-деталь із id платежу, status = Succeeded та кнопкою «View initiative»	Пройдено
4	Натиснути Edit profile → зробити фото камерою та зберегти	Аватар оновився, Toast «Avatar updated», запис PATCH у логи	Пройдено
5	У формі змінити ім’я на порожнє → Save	Поле підсвічується червоним, кнопка Save неактивна («Name required»)	Пройдено
6	Ввести нову біографію (≤140 симв.) → Save	200 OK; біографія відображена на головному профілі; updatedAt > createdAt	Пройдено
7	Натиснути Logout і знову зайти	Після повторного входу зміни збереглися, аватар завантажився з кешу	Пройдено
8	Ввімкнути офлайн-режим та відкрити профіль	Кешована інформація показана; під заголовком banner «Offline»	Пройдено

Кінець таблиці 5.1.

№	Опис випадку	Очікуваний результат	Висновок
9	У профілі натиснути «Delete account» та підтвердити	Запит API → 204; застосунок переходить на екран Sign Up	Пройдено
10	Спроба логіну після видалення	401 InvalidCredentials	Пройдено
Push-підписки, Share-sheet і реальний час			
№	Опис випадку	Очікуваний результат	Висновок
1	Settings → Notifications увімкнути «New raises» і «Goal reached»	Тумблери активні, настройки збережені (204 No Content)	Пройдено
2	PATCH /settings/notifications {email=daily, push=instant}	Через ≤3 с приходить push «New raise: Help Animals»	Пройдено
3	Відкрити push	Додаток відкриває деталі ініціативи #190 (deep-link)	Пройдено
4	На деталях натиснути іконку Share	З’являється системний Share Sheet із прелімінарним текстом і коротким URL	Пройдено
5	Копіювати посилання й відкрити в браузері	Браузер редиректить на сторінку raise #190; мобільний клієнт ловить intent і показує ту ж кампанію	Пройдено
6	На деталях натиснути «Subscribe to progress»	WebSocket підключився (icon green), Toast «Real-time updates ON»	Пройдено
7	Через симулятор бекенду збільшити суму збору +20 ₴	Прогрес-бар анімується, сума підвищується без перезавантаження екрана	Пройдено
8	У налаштуваннях вимкнути push-канал «Goal reached»	Запис у БД оновлено (enabled=false); наступний тест-push цього типу не з’являється	Пройдено
9	Спроба підписки з вимкненими Google Play-services	Додаток показує діалог «Push unavailable», прапор залишається OFF	Пройдено
10	Отримати push, коли застосунок у killed-state	Notification приходить; після тапу запускається app → переходить на релевантний екран	Пройдено

ВИСНОВОК

У межах курсового дослідження було розроблено та детально описано мобільний фандрейзинговий застосунок, який дозволяє ефективно організовувати та керувати благодійними зборами за допомогою сучасних цифрових технологій. Проведений аналіз предметної галузі засвідчив, що мобільні фандрейзингові системи сьогодні є одним із найефективніших інструментів для швидкого та масового залучення коштів завдяки психології миттєвої реакції, технологічній простоті та мінімізації «тертя» під час здійснення пожертв. Реалізований застосунок враховує всі ці аспекти, надаючи користувачам зручні функції для здійснення донатів, комунікації та відстеження результатів своєї допомоги.

Під час формування вимог до системи акцент було зроблено на простоту взаємодії, швидкість виконання операцій та максимальну прозорість збору і використання коштів. Застосунок інтегрує різні платіжні рішення, включаючи банківські сервіси, мобільні гаманці та міжнародні шлюзи, забезпечуючи високу швидкість і надійність транзакцій. Також реалізовано механізми персоналізованих push-повідомлень, що значно підвищують активність та утримання користувачів у межах платформи.

Архітектурне проектування програмного забезпечення базується на використанні UML-діаграм, що дозволило чітко структурувати взаємодію компонентів мобільного додатку. Запропонована структура файлів орієнтована на модульність, що забезпечує легкість масштабування та підтримки у майбутньому. Особлива увага приділялася розробці UI/UX дизайну: усі інтерфейси мають мінімалістичний стиль, прості форми введення, зрозумілі елементи навігації та миттєвий зворотний зв’язок. Дизайн забезпечує інтуїтивне сприйняття користувачем усіх функцій програми, включаючи редагування профілю, авторизацію та керування сповіщеннями.

Прийняті програмні рішення спрямовані на створення максимально простого, але водночас безпечного середовища. Для цього використано принципи Zero Trust, багаторівневе шифрування даних, псевдонімізацію персональної

інформації та жорсткий контроль доступу за ролями користувачів. Усі ці заходи дозволяють забезпечити високий рівень безпеки без погіршення користувацького досвіду.

У результаті проведеного дослідження розроблено мобільну фандрейзингову систему, яка відповідає сучасним вимогам як у плані функціональності та продуктивності, так і щодо відповідності регуляторним стандартам GDPR, PSD2 та AML/KYC. Майбутні напрямки розвитку проєкту включають подальше розширення функціоналу, інтеграцію штучного інтелекту для покращення рекомендаційних алгоритмів, а також посилення міжнародного масштабування платформи. Отже, дана робота створює надійну основу для подальших досліджень та практичного застосування мобільних технологій у сфері цифрового фандрейзингу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Культура донатів в Україні: що потрібно знати [Електронний ресурс]. – Режим доступу: https://philanthropyinukraine.org/en/knowledge-hub/publication/kultura-donativ-vukraini-shcho-potribno-znaty (дата звернення: 11.05.2025).

2. Ukrainians triple donations for army in 2023 through Monobank [Електронний ресурс]. – Режим доступу: https://euromaidanpress.com/2023/12/30/ukrainians-triple-donations-for-army-in2023-through-monobank-number-2-ukrainian-bankingapp/#:~:text=The%20statistics%20further%20reveal%20that,to%20349%20UAH%20in%202023 (дата звернення: 11.05.2025).

3. Annex 7 USAID/ENGAGE Individual Giving in Ukraine [Електронний ресурс]. – Режим доступу: https://api.home.ednannia.ua/upload/kch/25/05/08/Individual_giving.pdf#:~:text=Te%20most%20convenient%20way%20to,for%20Ukrainian%20fundraising%20capaigns%3A%20campaigns (дата звернення: 11.05.2025).

4. How Smart UX Can Deliver 126% More Donations for Nonprofits [Електронний ресурс]. – Режим доступу: https://www.designrush.com/news/how-smart-ux-can-deliver-126-more-donationsfor-nonprofits#:~:text=,or%20unclear%2C%20users%20abandon%20it (дата звернення: 11.05.2025).

5. Too Tired To Choose [Електронний ресурс]. – Режим доступу: https://agitator.thedonorvoice.com/too-tired-tochoose/#:~:text=Going%20back%20to%20the%20reply,unrelated%20to%20the20immediate%20ask (дата звернення: 11.05.2025).

6. З початку повномасштабної війни через сервіси monobank громадяни задонатили 1 млрд євро [Електронний ресурс]. – Режим доступу: https://ms.detector.media/internet/post/34188/2024-02-12-z-pochatku-povnomasshtabnoi-viyny-cherez-servisy-monobank-gromadyany-zadonatyly-1-

mlrdievro/#:~:text=благодійні%20онлайн,повідомив%20співвласник%20monobnk%20Олег%20Гороховський (дата звернення: 11.05.2025).

7. Donate24 – AppStore: https://apps.apple.com/us/app/donate24-%D0%B7%D0%B1%D0%BE%D1%80%D0%B8-%D0%BF%D1%96%D0%B4%D1%82%D1%80%D0%B8%D0%BC%D0%BA%D0%B0/id6450211519 - Дата звернення: 18.06.2025

8. Push Notifications: Not Just for Corporate Marketers [Електронний ресурс]. – Режим доступу: https://www.charitydynamics.com/push-notifications-not-just-for-corporate-marketers/ (дата звернення: 11.05.2025).

9. The Role of Fundraising Platforms to Protect Donors and Nonprofits [Електронний ресурс]. – Режим доступу: https://betterworld.org/blog/nonprofits/the-role-of-fundraising-platforms-to-protect-donors-and-nonprofts/ (дата звернення: 11.05.2025).

10. PSD2 & Charities: What It Means for Your Organisation [Електронний ресурс]. – Режим доступу: https://blog.iraiser.com/psd2-charities-what-it-means-for-your-organisation (дата звернення: 11.05.2025).

11. Zero Trust Applied to the Mobile World (NIST NCCoE) [Електронний ресурс]. – Режим доступу: https://www.nccoe.nist.gov/news-insights/zero-trust-applied-mobile-world (дата звернення: 11.05.2025).

12. Github репозиторій проекту [Електронний ресурс]. — Режим доступу: https://github.com/NureBluvbandKirill/2025_B_PI_-PZPI-21-1_Bluvband_K_I