# Додаток В. Специфікація програмного продукту

**СПЕЦИФІКАЦІЯ ПЗ**

До системи програмної підтримки волонтерської діяльності

Виконали:

Сорокін Вадим Сергійович

Крикуха Максим Едуардович

Голя Станіслав Володимирович

1. Вступ

**1.1 Огляд продукту**

Ціль цього документа полягає у створенні програмної системи для підтримки волонтерської діяльності, яка дозволяє волонтерам та отримувачам допомоги ефективно координувати процеси збору коштів та матеріальних ресурсів, обліку закупівель, керування потребами та формування звітності. Система забезпечить прозору взаємодію між усіма учасниками та надійне зберігання даних.

**1.2 Межі**

Система програмної підтримки волонтерської діяльності – це комплексна платформа, яка дозволяє користувачам (волонтерам, отримувачам допомоги, адміністраторам) ефективно організовувати та управляти волонтерськими ініціативами.

Зареєстровані користувачі можуть переглядати, створювати, редагувати та керувати зборами коштів та матеріальних ресурсів, а також формувати та відслідковувати запити на допомогу (потреби). Система дозволяє додавати підтверджуючі документи до закупівель для забезпечення прозорості.

Волонтери можуть бачити інформацію про створені збори, їхній прогрес, а також переглядати детальні звіти. Отримувачі допомоги можуть створювати свої потреби та відстежувати їхній статус, а також залишати відгуки про отриману допомогу.

Система забезпечує централізовану авторизацію користувачів з різними ролями (волонтер, отримувач, адміністратор), а також надає зручний інтерфейс для взаємодії через веб-застосунок та мобільний застосунок. Інтеграція із зовнішніми платіжними сервісами (наприклад, Monobank) дозволить відображати прогрес зборів.

**1.3 Означення та абревіатури**

Таблиця 1 – Означення

|  |  |
| --- | --- |
| **Термін** | **Означення** |
| Збір | Організована кампанія зі збору коштів або матеріальних цінностей для певної мети. |
| Потреба | Конкретний запит на допомогу (перелік необхідних предметів або послуг) від отримувача. |
| Волонтер | Зареєстрований користувач, який організовує збори та керує процесом надання допомоги. |
| Звіт | Документ, що підтверджує використання зібраних коштів або матеріалів, часто включає чеки, фото та висновки. |
| Front-end/Клієнт | Графічний інтерфейс, який відображається користувачеві (веб-застосунок, мобільний застосунок) для перегляду даних та взаємодії з ними. |
| Серверна частина/АПІ | Інтерфейс взаємодії між клієнтом і сервером, який відповідає за бізнес-логіку, обробку запитів, забезпечення захисту даних та взаємодію з базою даних. |

**2. Загальний опис**

**2.1 Перспективи продукту**

Програмна система підтримки волонтерської діяльності складається з таких частин: веб-застосунок для користувачів, мобільний застосунок для користувачів, централізована серверна частина (back-end) та база даних.

Основна база даних міститься в СУБД PostgreSQL, яка містить у собі дані користувачів, зборів, потреб, звітів та інших сутностей системи. Користувач не може напряму комунікувати з сервером бази даних, але взаємодіє з системою через клієнтські застосунки, які формують запити до серверної частини.

При потребі даних клієнт (веб-або мобільний застосунок) формує запит і посилає його на API серверної частини. Серверна частина інтерпретує запит, перевіряє права доступу та валідність даних. Якщо запит може бути виконаний, серверна частина формує та надсилає запит до бази даних. База даних повертає дані або повідомлення про помилку, після чого серверна частина обробляє цю інформацію (наприклад, трансформує дані, застосовує бізнес-логіку) і надсилає відповідь клієнту. Тільки клієнт відповідає за надання даних у графічному представленні.

Серверна частина може додавати або змінювати дані під час обміну інформацією. API має бути прихована від кінцевих користувачів, але доступна для клієнтських частин системи та, за потреби, для інтеграції із зовнішніми сервісами (наприклад, банківськими API).

Зображення та інші файли, пов'язані зі зборами чи звітами (наприклад, фото чеків), зберігаються у зовнішньому файловому сховищі, що забезпечує масштабованість та ефективне зберігання великих обсягів медіаданих.

Система передбачає також можливість інтеграції з API зовнішніх платіжних сервісів (наприклад, Monobank Jars) для відображення прогресу фінансових зборів без безпосередньої обробки фінансових транзакцій самою системою.

**2.2 Функції продукту**

За допомогою веб- та мобільного застосунків зареєстровані користувачі зможуть створювати та редагувати збори: Волонтери можуть ініціювати нові кампанії зі збору коштів або матеріалів, вказуючи назву, опис, мету, необхідну суму (для фінансових зборів) або перелік предметів, а також додавати зображення та посилання на банківські рахунки.

Отримувачі допомоги можуть створювати деталізовані запити (потреби), вказуючи необхідні предмети, їх кількість, категорію та пріоритетність. Волонтери можуть переглядати ці потреби та брати їх у роботу.

Волонтери можуть надсилати звіти про використання зібраних ресурсів, додаючи фото чеків та документів, що підвищує прозорість для донорів.

Користувачі можуть відстежувати прогрес фінансових зборів (через інтеграцію з Monobank API) та бачити загальний статус виконання матеріальних потреб.

Волонтери та отримувачі допомоги мають особистий кабінет для перегляду та редагування персональних даних, а також історії своєї активності.

Отримувачі допомоги можуть залишати відгуки та оцінки про роботу волонтерів, що сприяє формуванню системи рейтингів.

Адміністратори зможуть керувати користувацькими профілями, контролювати збори та звіти, а також налаштовувати загальні параметри системи.

**2.3 Характеристики користувачів**

У системі існують 3 типи зареєстрованих користувачів та незареєстровані відвідувачі.

Незареєстрованим користувачам доступні лише функції перегляду публічної інформації про збори без можливості взаємодії.

Зареєстрованим користувачам волонтерам можна створювати, редагувати та переглядати власні збори, керувати пов'язаними потребами, надсилати звіти, переглядати свій профіль та статистику власної діяльності, а також переглядати загальний список потреб від отримувачів.

Зареєстрованим користувачам отримувачам допомоги можна створювати, переглядати та редагувати власні потреби, отримувати допомогу, підтверджувати її отримання та залишати відгуки/оцінки про волонтерів. Мають доступ до свого профілю.

Адміністратори мають повний контроль над системою: можуть керувати профілями користувачів (блокування/видалення), змінювати статуси зборів та потреб, переглядати всі звіти та статистику, а також керувати загальними налаштуваннями платформи.

**2.4 Загальні обмеження**

Веб-застосунок обмежений можливостями веб-браузера користувача. Дизайн враховує адаптивність, але можуть виникати незначні візуальні відмінності на різних пристроях та браузерах.

Мобільний застосунок обмежений характеристиками мобільного пристрою користувача (розмір екрану, апаратні можливості). Розроблений для операційних систем Android та iOS.

Робота системи критично залежить від стабільного інтернет-з'єднання. Без зв’язку між клієнтом і сервером отримання та оновлення інформації неможливе.

Серверна частина обмежена характеристиками сервера та бази даних (PostgreSQL). При великому навантаженні можуть виникати затримки в обробці запитів, хоча архітектура передбачає можливості масштабування.

**2.5 Припущення й залежності**

Припускається, що більшість користувачів (волонтерів, отримувачів) матимуть стабільний доступ до мережі Інтернет для використання веб- та мобільного застосунків.

Припускається, що користувачі мають базові навички роботи з веб-додатками та мобільними застосунками та здатні самостійно взаємодіяти з інтерфейсом системи.

Робота системи значною мірою залежить від доступності та стабільності зовнішніх сервісів, таких як банківські API (наприклад, Monobank для відображення прогресу зборів), а також сервісів файлового сховища.

Наявність функціонального та стабільного серверного API є критичною залежністю для коректної роботи клієнтських застосунків.

Ефективність системи значною мірою залежить від активної участі волонтерів та отримувачів у наданні актуальних даних, оновленні інформації та взаємодії між собою.

Припускається, що більшість сучасних веб-браузерів (для веб-застосунку) та актуальних версій мобільних операційних систем (Android, iOS для мобільного застосунку) підтримують необхідні технології (HTML5, CSS3, JavaScript, Flutter/Dart) та забезпечують коректне відображення та функціональність.

1. **Конкретні вимоги**

**3.1 Вимоги до зовнішніх інтерфейсів**

**3.1.1 Інтерфейс користувача**

Веб-застосунок реалізований на React/TypeScript з використанням Material UI та Tailwind CSS. Забезпечує інтуїтивно зрозумілий та адаптивний дизайн для різних розмірів екранів. Включає форми для створення/редагування зборів та потреб, сторінки профілів, списки з активними зборами та потребами, інтерфейси для перегляду звітів та статистичних даних, функціонал пошуку та фільтрації. Підтримує багатомовність (українська, англійська).

Мобільний застосунок реалізований на Flutter/Dart. Забезпечує кросплатформеність (Android, iOS) з нативним виглядом та високою продуктивністю. Включає ті ж ключові функціональні екрани, що й веб-застосунок, з оптимізацією під мобільний досвід (навігація, сенсорне керування).

**3.1.2 Апаратний інтерфейс**

Для відображення інтерфейсу користувача та взаємодії з системою, пристрій (персональний комп'ютер, ноутбук, планшет або смартфон) має бути оснащений дисплеєм, клавіатурою (віртуальною чи фізичною) та вказівним пристроєм (миша, тачпад, сенсорний екран).

**3.1.3 Програмний інтерфейс**

Для веб-застосунку: користувач повинен використовувати сучасний веб-браузер, який підтримує стандарти HTML5, CSS3 та JavaScript. Рекомендовані браузери: Google Chrome, Mozilla Firefox, Safari, Microsoft Edge актуальних версій.

Для мобільного застосунку: користувач повинен використовувати мобільний пристрій під управлінням операційних систем Android (версії 7.0 або вище) або iOS (версії 12.0 або вище). Застосунок потребує доступу до мережевих з'єднань.

**3.1.4 Комунікаційний протокол**

Для доступу до системи необхідно стабільне підключення до мережі Інтернет. Вся комунікація між клієнтськими застосунками (веб, мобільний) та серверною частиною здійснюється за протоколом HTTPS для забезпечення безпеки та шифрування даних. Взаємодія з зовнішніми сервісами (наприклад, банківськими API) також здійснюється за захищеними протоколами, що відповідають їхнім вимогам.

**3.2 Функції продукту**

**3.2.1 Авторизація**

3.2.1.1 Вступ

Зареєстровані користувачі можуть увійти до системи, щоб отримати доступ до своїх функцій та даних.

3.2.1.2 Вхідні дані

Текст: Електронна пошта (до 50 символів), пароль (8-50 символів).

3.2.1.3 Обробка

Клієнтський застосунок надсилає запит на авторизацію (POST-запит) з електронною поштою та паролем на серверний API.

Серверна частина хешує отриманий пароль і порівнює його з хешованим паролем, збереженим для користувача з відповідною електронною поштою у базі даних. Якщо облікові дані збігаються, сервер генерує JWT-токен доступу, який містить інформацію про користувача та його роль, і повертає його клієнту. Клієнт зберігає токен локально (наприклад, у localStorage для вебу, SharedPreferences для мобільного) та перенаправляє користувача на відповідний дашборд (волонтера або отримувача).

3.2.1.4 Вихідні дані

Якщо вхід виконано успішно: користувач перенаправляється на головну сторінку/дашборд відповідно до своєї ролі, а система відображає повідомлення про успішну авторизацію (опціонально).

3.2.1.5 Обробка помилок

Якщо вхід не вдається: клієнт відображає повідомлення про помилку (наприклад, "Невірний email або пароль", "Помилка сервера") і користувач залишається на формі входу.

3.2.1.6 Обмеження

Неавторизовані/незареєстровані користувачі.

**3.2.2 Перегляд зборів**

3.2.2.1 Опис

Зареєстровані користувачі (волонтери, отримувачі) та незареєстровані відвідувачі можуть переглядати список активних зборів та деталі кожного з них.

3.2.2.2 Вхідні дані

Для перегляду списку зборів користувач переходить на головну сторінку або сторінку "Всі збори". Для перегляду деталей конкретного збору – натискає на картку збору у списку. Можливе використання фільтрів (за категорією, статусом) та пошуку (за назвою, описом).

3.2.2.3 Обробка

Клієнтський застосунок надсилає GET-запит на серверний API для отримання списку зборів (з параметрами фільтрації/пошуку, якщо вони вказані). Сервер повертає список зборів. Клієнт відображає отримані дані у вигляді карток.

При натисканні на картку збору, клієнт надсилає GET-запит на серверний API для отримання детальної інформації про конкретний збір (за ID). Сервер повертає деталі збору, включаючи прогрес (якщо це фінансовий збір, з інтеграцією Monobank API), список пов'язаних потреб та звітів. Клієнт відображає цю інформацію на окремому екрані.

3.2.2.4 Вихідні дані

Список зборів у вигляді карток на відповідних сторінках.

Детальна інформація про конкретний збір (назва, опис, зображення, прогрес, організатор, отримувач, пов'язані потреби/звіти) на окремому екрані.

3.2.2.5 Обробка помилок

Якщо зборів не знайдено (наприклад, за критеріями пошуку): відображається повідомлення "Зборів не знайдено".

Якщо сталася помилка під час запиту до сервера: відображається повідомлення про помилку.

3.2.2.6 Обмеження

Перегляд загального списку та деталей збору доступний усім користувачам (авторизованим та неавторизованим). Деякі деталі можуть бути приховані для неавторизованих користувачів (наприклад, повні контактні дані).

**3.2.3 Створення збору**

3.2.3.1 Опис

Волонтери можуть створювати нові збори коштів або матеріальних ресурсів.

3.2.3.2 Вхідні дані

Текст: Назва збору (2-100 символів), опис (до 1000 символів), посилання на Monobank Jar, дата завершення.

Файл: Зображення для збору.

Вибір: Отримувач допомоги (зі списку зареєстрованих), список пов'язаних потреб (зі списку існуючих потреб або можливість створити нові).

3.2.3.3 Обробка

Волонтер заповнює форму створення збору. Клієнтська частина виконує валідацію вхідних даних (наприклад, перевірка заповнення обов'язкових полів, формату посилання, діапазону сум).

Якщо дані валідні, клієнт надсилає POST-запит на серверний API. Запит включає текстові дані з форми та завантажуване зображення (як multipart/form-data).

Серверна частина перевіряє валідність даних, зберігає інформацію про збір у базі даних, завантажує зображення у файлове сховище та пов'язує його зі збором.

3.2.3.4 Вихідні дані

Якщо створення успішне: клієнт відображає повідомлення про успішне створення збору та перенаправляє користувача на сторінку деталей цього збору або на список власних зборів. Новий збір з'являється у списках.

3.2.3.5 Обробка помилок

Клієнт відображає помилки валідації (наприклад, "Поле 'Назва' обов'язкове", "Некоректне посилання на Monobank Jar") до відправки запиту на сервер.

Якщо помилка виникла на стороні сервера (наприклад, внутрішня помилка, помилка при завантаженні файлу): клієнт відображає повідомлення про помилку з сервера.

3.2.3.6 Обмеження

Доступно лише для авторизованих користувачів з роллю "Волонтер".

**3.2.4 Видалення набору опитувань**

3.2.4.1 Вступ

Користувачу доступна функція видалення групи опитувань, у випадку, коли вона стає йому непотрібна.

3.2.4.2 Вхідні дані

Для видалення групи опитувань користувачу потрібно перейти до сторінки конкретної групи опитувань та натиснути на кнопку видалення, потім виконати підтвердження, натиснувши на відповідну кнопку.

3.2.4.3 Обробка

Після натискання користувачем на кнопку підтвердження видалення, клієнт виконує запит на видалення групи опитувань на сервер, після чого отримує результат запиту.

3.2.4.4 Вихідні дані

Кнопка видалення буде знаходитися на екрані користувача на сторінці конкретної групи опитувань. Повідомлення про успішне видалення чи помилку буде відображено модальним вікном на екрані користувача.

3.2.4.5 Обробка помилок

У випадку успішного видалення буде відображено відповідне повідомлення, у випадку помилки буде відображена причина помилки.

3.2.4.6 Обмеження

Авторизовані користувачі. Створена група опитувань для видалення

**3.2.5 Редагування збору**

3.2.5.1 Опис

Волонтери можуть редагувати інформацію про власні збори, які вони створили.

3.2.5.2 Вхідні дані

Користувач переходить на сторінку деталей власного збору та натискає кнопку "Редагувати". Він може змінити: Назву збору, Опис. Також може завантажити нове зображення або відредагувати/видалити пов'язані потреби.

3.2.5.3 Обробка

Клієнтський застосунок відображає форму редагування, попередньо заповнену поточними даними збору. Волонтер вносить зміни та натискає кнопку "Зберегти".

Клієнт виконує валідацію вхідних даних. Якщо дані валідні, надсилає PUT-запит на серверний API з оновленими даними збору (за його ID). Якщо було завантажено нове зображення, воно надсилається окремо.

Серверна частина перевіряє права доступу користувача (чи є він організатором цього збору), валідує дані, оновлює інформацію про збір у базі даних та, за потреби, оновлює зображення у файловому сховищі.

3.2.5.4 Вихідні дані

Якщо редагування успішне: клієнт відображає повідомлення про успішне оновлення збору та відображає оновлену інформацію на сторінці деталей збору.

3.2.5.5 Обробка помилок

Клієнт відображає помилки валідації (аналогічно створенню збору).

Якщо помилка виникла на стороні сервера (наприклад, користувач не має прав на редагування, внутрішня помилка): клієнт відображає відповідне повідомлення про помилку.

3.2.5.6 Обмеження

Доступно лише для авторизованих користувачів з роллю "Волонтер", які є організаторами конкретного збору.

**3.2.6 Видалення збору**

3.2.6.1 Опис

Волонтери можуть видалити власні збори, які вони створили, якщо вони більше не актуальні або були створені помилково.

3.2.6.2 Вхідні дані

Користувач переходить на сторінку деталей власного збору, натискає кнопку "Видалити" та підтверджує дію у модальному вікні.

3.2.6.3 Обробка

Клієнтський застосунок надсилає DELETE-запит на серверний API, передаючи ID збору, який потрібно видалити.

Серверна частина перевіряє права доступу користувача. Якщо права підтверджені, інформація про збір та всі пов'язані з ним дані (потреби, звіти, посилання на файли) видаляються з бази даних та файлового сховища.

3.2.6.4 Вихідні дані

Якщо видалення успішне: клієнт відображає повідомлення про успішне видалення збору та перенаправляє користувача на список власних зборів. Збір зникає зі всіх списків у системі.

3.2.6.5 Обробка помилок

Якщо помилка виникла на стороні сервера (наприклад, користувач не має прав на видалення, збір не знайдено, внутрішня помилка): клієнт відображає відповідне повідомлення про помилку.

3.2.6.6 Обмеження

Доступно лише для авторизованих користувачів з роллю "Волонтер", які є організаторами конкретного збору.

**3.2.7 Створення потреби**

3.2.7.1 Опис

Отримувачі допомоги можуть створювати нові запити на допомогу (потреби), вказуючи перелік необхідних предметів. Волонтери також можуть створювати потреби від імені отримувачів.

3.2.7.2 Вхідні дані

Текст: Назва потреби (2-100 символів), опис (до 1000 символів), бажаний термін виконання (дедлайн, опціонально)

Вибір: Пріоритетність (наприклад, "Низький", "Високий").

Список: Предмети (назва предмета, кількість, категорія (наприклад, "Їжа", "Медицина", "Одяг", "Обладнання")).

3.2.7.3 Обробка

Користувач заповнює форму створення потреби. Клієнтська частина виконує валідацію вхідних даних

Якщо дані валідні, клієнт надсилає POST-запит на серверний API з усіма даними про потребу.

Серверна частина перевіряє валідність даних, зберігає інформацію про потребу та її предмети у базі даних, пов'язуючи її з користувачем, який створив запит.

3.2.7.4 Вихідні дані

Якщо створення успішне: клієнт відображає повідомлення про успішне створення потреби та перенаправляє користувача на сторінку списку його потреб або на деталі щойно створеної потреби. Нова потреба з'являється у відповідних списках.

3.2.7.5 Обробка помилок

Клієнт відображає помилки валідації (наприклад, "Поле 'Назва' обов'язкове", "Необхідно додати хоча б один предмет") до відправки запиту на сервер.

Якщо помилка виникла на стороні сервера: клієнт відображає повідомлення про помилку з сервера.

3.2.7.6 Обмеження

Доступно для авторизованих користувачів з роллю "Отримувач допомоги" або "Волонтер.

**3.2.8 Перегляд потреб**

3.2.8.1 Опис

Зареєстровані користувачі (волонтери, отримувачі) можуть переглядати список активних потреб. Отримувачі бачать свої потреби, волонтери – всі активні потреби.

3.2.8.2 Вхідні дані

Для перегляду списку потреб користувач переходить на відповідну сторінку ("Мої потреби" для отримувачів, "Всі потреби" для волонтерів). Можливе використання фільтрів (за категорією, пріоритетом, статусом) та пошуку. Для перегляду деталей конкретної потреби – натискає на картку потреби у списку.

3.2.8.3 Обробка

Клієнтський застосунок надсилає GET-запит на серверний API для отримання списку потреб (з параметрами фільтрації/пошуку, якщо вони вказані, та з урахуванням ролі користувача). Сервер повертає список потреб. Клієнт відображає отримані дані у вигляді карток.

При натисканні на картку потреби, клієнт надсилає GET-запит на серверний API для отримання детальної інформації про конкретну потребу (за ID). Сервер повертає деталі потреби, включаючи статус виконання кожного предмета та пов'язані збори. Клієнт відображає цю інформацію на окремому екрані.

3.2.8.4 Вихідні дані

Список потреб у вигляді карток на відповідних сторінках.  
Детальна інформація про конкретну потребу (назва, опис, пріоритет, дедлайн, список предметів із статусом виконання) на окремому екрані.

3.2.8.5 Обробка помилок

Якщо потреб не знайдено (наприклад, за критеріями пошуку або у користувача немає створених потреб): відображається повідомлення "Потреб не знайдено".

Якщо сталася помилка під час запиту до сервера: відображається повідомлення про помилку.

3.2.8.6 Обмеження

Доступно для авторизованих користувачів. Волонтери можуть бачити всі активні потреби, отримувачі – лише свої.

**3.2.9 Редагування потреби**

3.2.9.1 Опис

Отримувачі допомоги (або волонтери, які створили потребу від їх імені) можуть редагувати власні потреби.

3.2.9.2 Вхідні дані

Користувач переходить на сторінку деталей власної потреби та натискає кнопку "Редагувати". Може змінити: Назву потреби, Опис, Дедлайн, Пріоритетність. Також може додати, відредагувати або видалити предмети зі списку потреб.

3.2.9.3 Обробка

Клієнтський застосунок відображає форму редагування, попередньо заповнену поточними даними потреби. Користувач вносить зміни та натискає кнопку "Зберегти".

Клієнт виконує валідацію вхідних даних. Якщо дані валідні, надсилає PUT-запит на серверний API з оновленими даними потреби (за її ID).

Серверна частина перевіряє права доступу користувача, валідує дані, оновлює інформацію про потребу та її предмети у базі даних.

3.2.9.4 Вихідні дані

Якщо редагування успішне: клієнт відображає повідомлення про успішне оновлення потреби та відображає оновлену інформацію на сторінці деталей потреби.

3.2.9.5 Обробка помилок

Клієнт відображає помилки валідації (аналогічно створенню потреби).

Якщо помилка виникла на стороні сервера: клієнт відображає відповідне повідомлення про помилку.

3.2.9.6 Обмеження

Доступно лише для авторизованих користувачів з роллю "Отримувач допомоги" (або "Волонтер", якщо він створив цю потребу).

**3.2.10 Управління профілем користувача**

3.2.10.1 Опис

Кожен авторизований користувач (волонтер, отримувач) має доступ до свого особистого профілю для перегляду та редагування особистих даних.

3.2.10.2 Вхідні дані

Користувач переходить на сторінку "Профіль".

Для перегляду: ніяких вхідних даних.

Для редагування: Текст: Ім’я, Прізвище, Телефон, Електронна пошта (для зміни пошти може знадобитися додаткове підтвердження). Файл: Зображення профілю (аватар).

3.2.10.3 Обробка

При завантаженні сторінки профілю клієнтський застосунок надсилає GET-запит на серверний API для отримання інформації про поточного авторизованого користувача. Сервер повертає дані профілю. Клієнт відображає ці дані.

При редагуванні: користувач заповнює форму редагування. Клієнт виконує валідацію та надсилає PUT-запит на серверний API з оновленими даними. Серверна частина оновлює профіль у базі даних та, за потреби, зображення у файловому сховищі.

3.2.10.4 Вихідні дані

Сторінка профілю з актуальною інформацією (ім'я, прізвище, email, телефон, роль, історія зборів/потреб, рейтинг для волонтерів).

Повідомлення про успішне оновлення профілю.

3.2.10.5 Обробка помилок

Якщо помилка при завантаженні даних профілю: відображається повідомлення про помилку.

Якщо помилка при редагуванні (валідація, конфлікт з існуючими даними, помилка сервера): відображається відповідне повідомлення.

3.2.10.6 Обмеження

Доступно лише для авторизованих користувачів.

**3.2.11 Оцінювання зборів**

3.2.11.1 Опис

Отримувачі допомоги можуть залишати оцінки для зборів, які були створені для задоволення їхніх потреб.

3.2.11.2 Вхідні дані

Користувач (отримувач) переходить на сторінку деталей збору, який було створено для нього. Натискає кнопку "Оцінити збір". Вибирає значення оцінки (наприклад, від 1 до 5 зірок).

3.2.11.3 Обробка

Клієнтський застосунок відображає модальне вікно для оцінки. Користувач обирає оцінку та натискає "Зберегти".

Клієнт надсилає POST/PUT-запит на серверний API з ID збору та значенням оцінки.

Серверна частина перевіряє, чи користувач є отримувачем допомоги для цього збору, зберігає або оновлює оцінку у базі даних та, за потреби, перераховує середній рейтинг волонтера.

3.2.11.4 Вихідні дані

Повідомлення про успішне збереження оцінки. Оновлений рейтинг збору/волонтера відображається на відповідних сторінках.

3.2.11.5 Обробка помилок

Якщо користувач не має права оцінювати цей збір: відображається повідомлення "Ви не можете оцінити цей збір".

Якщо помилка на стороні сервера: відображається повідомлення про помилку.

3.2.11.6 Обмеження

Доступно лише для авторизованих користувачів з роллю "Отримувач допомоги", для зборів, які стосуються їхніх потреб.

**3.2.12 Завантаження звітів**

3.2.12.1 Опис

Волонтери можуть завантажувати звіти (наприклад, фото чеків, документи) про виконані закупівлі або надану допомогу, прив'язуючи їх до конкретного збору.

3.2.12.2 Вхідні дані

Волонтер переходить на сторінку деталей свого збору. Натискає кнопку "Додати звіт". Обирає файли (фото, PDF) для завантаження.

3.2.12.3 Обробка

Клієнтський застосунок дозволяє обрати один або кілька файлів. Клієнт виконує валідацію розміру та типу файлів.

Якщо файли валідні, клієнт надсилає POST-запит (multipart/form-data) на серверний API, передаючи файли та ID збору.

Серверна частина перевіряє права доступу, завантажує файли у файлове сховище, зберігає інформацію про звіт (посилання на файли, опис) у базі даних та пов'язує його зі збором.

3.2.12.4 Вихідні дані

Повідомлення про успішне завантаження звіту. Звіт відображається на сторінці деталей збору.

3.2.12.5 Обробка помилок

Клієнт відображає помилки валідації файлів (розмір, тип).

Якщо помилка на стороні сервера (наприклад, невірні права, помилка сховища): відображається повідомлення про помилку.

3.2.12.6 Обмеження

Доступно лише для авторизованих користувачів з роллю "Волонтер", для власних зборів.

**3.2.13 Перегляд статистики та прогресу зборів**

3.2.13.1 Опис

Користувачі можуть переглядати прогрес фінансових зборів (через інтеграцію з Monobank API) та загальний статус виконання матеріальних потреб.

3.2.13.2 Вхідні дані

Перехід на сторінку збору або дашборд.

3.2.13.3 Обробка

При завантаженні сторінки збору: клієнт надсилає запит на серверний API для отримання інформації про Monobank Jar ID. Використовуючи цей ID, клієнт надсилає запит до Monobank API для отримання поточного прогресу "банки". Отримані дані відображаються у вигляді прогрес-бару або відсотків.

3.2.13.4 Вихідні дані

Візуалізація прогресу збору (відсотки, прогрес-бар), статус виконання потреб.

3.2.13.5 Обробка помилок

Якщо Monobank API недоступний або повертає помилку: відображається повідомлення про недоступність даних прогресу.

3.2.13.6 Обмеження

Доступно для всіх користувачів на сторінках зборів.

**3.2.14 Пошук та фільтрація**

3.2.14.1 Опис

Користувачі можуть ефективно знаходити необхідні збори або потреби за допомогою пошуку за ключовими словами та фільтрації за різними критеріями.

3.2.14.2 Вхідні дані

Текст: Пошуковий запит (ключові слова).

3.2.14.3 Обробка

Користувач вводить пошуковий запит або обирає фільтри. Клієнтська частина надсилає GET-запит на серверний API, передаючи пошукові параметри та фільтри

Серверна частина виконує пошук та фільтрацію даних у базі даних та повертає відфільтрований список зборів/потреб.

Клієнт відображає оновлений список результатів у реальному часі.

3.2.14.4 Вихідні дані

Оновлений список зборів або потреб, що відповідають критеріям пошуку та фільтрації.

3.2.14.5 Обробка помилок

Якщо результатів не знайдено: відображається повідомлення "За вашим запитом нічого не знайдено".

Якщо помилка на стороні сервера: відображається повідомлення про помилку.

3.2.14.6 Обмеження

Доступно для всіх користувачів

**3.3 Класи/Об’єкти**

**3.3.1 Specific**

3.3.1.1 Атрибути

* ID – TEXT (PRIMARY KEY)
* Name – TEXT (NOT NULL)
* Description – TEXТ

**3.3.2 Volunteer**

3.3.2.1 Атрибути

* ID – TEXT (PRIMARY KEY)
* Email – TEXT
* PasswordHash – TEXT
* ProfilePic – TEXT (URL зображення профілю)
* Phone – TEXT
* Name – TEXT
* Surname – TEXT
* Age – TEXT
* Specific – TEXT (FOREIGN KEY до Specific.ID)
* Available – BOOLEAN (зі значенням за замовчуванням true)

**3.3.3 Report**

3.3.3.1 Атрибути

* ID – TEXT (PRIMARY KEY)
* Rating – INTEGER
* FinalConclution – TEXT

**3.3.4 Fund**

3.3.4.1 Атрибути

* ID – TEXT (PRIMARY KEY)
* Name – TEXT
* Description – TEXT
* MonoJarUrl – TEXT (посилання на банку Monobank, опціонально)
* Report – TEXT (FOREIGN KEY до Report.ID, опціонально)
* Volunteer – TEXT (FOREIGN KEY до Volunteer.ID)
* Status – TEXT (можливі значення: 'Active', 'Completed', 'Cancelled')
* Picture – TEXT (URL зображення збору)
* LongJarID – TEXT (ID Monobank Jar, за замовчуванням '')

**3.3.5 Requirement**

3.3.5.1 Атрибути

* ID – TEXT (PRIMARY KEY)
* Deadline – DATE
* Name – TEXT (NOT NULL)
* Priority – TEXT (можливі значення: 'Default', 'High')
* Fund – TEXT (FOREIGN KEY до Fund.ID, пов'язаний збір, якщо потреба є частиною збору)
* Description – TEXT
* Recipient – TEXT (FOREIGN KEY до Recipient.ID)

**3.3.6 Item**

3.3.6.1 Атрибути

* ID – TEXT (PRIMARY KEY)
* Name – TEXT (NOT NULL)
* Count – INTEGER (NOT NULL)
* Requirement – TEXT (FOREIGN KEY до Requirement.ID)
* Category – TEXT (можливі значення: 'Food', 'Medicine', 'Equipment', 'Other')
* ReservedBy – TEXT (FOREIGN KEY до Fund.ID, якщо предмет резервується певним збором'')

**3.3.7 Recipient**

3.3.7.1 Атрибути

* ID – TEXT (PRIMARY KEY)
* Name – TEXT (NOT NULL)
* Email – TEXT
* PasswordHash – TEXT
* ProfilePic – TEXT (URL зображення профілю)

**3.4 Нефункціональні вимоги**

**3.4.1 Продуктивність**

Час завантаження будь-якої сторінки/екрану (без врахування першого запуску застосунку та першого завантаження великих списків) не повинен перевищувати 3 секунди.

Час відгуку на дії користувача (натискання кнопки, перехід між полями форми) не повинен перевищувати 0.5 секунди.

Завантаження списків (зборів, потреб) повинно відбуватися з механізмами пагінації та/або лінивого завантаження (lazy loading) для оптимізації продуктивності при великих обсягах даних.

Оптимізація роботи з мережею: використання кешування даних та мінімізація кількості запитів до сервера;

**3.4.2 Надійність**

Система повинна коректно обробляти всі можливі помилки (мережеві збої, помилки валідації, помилки сервера) та надавати користувачеві зрозумілі повідомлення.

Імовірність непередбаченого завершення роботи (крашу) клієнтського застосунку не повинна перевищувати 0.01% на тиждень при нормальному використанні.

Збереження цілісності даних у базі даних при будь-яких операціях (CRUD).

Система повинна забезпечувати відмовостійкість серверної частини та бази даних (наприклад, резервне копіювання, кластерні рішення.

**3.4.3 Доступність**

Система повинна бути доступною 24/7, за винятком планових технічних робіт, про які користувачі попереджаються заздалегідь.

Веб-застосунок має бути доступним з будь-якого сучасного веб-браузера.

Мобільний застосунок має бути доступним для завантаження та використання на основних платформах (Android, iOS) через офіційні магазини застосунків.

**3.4.4 Безпека**

Всі конфіденційні дані (паролі) повинні передаватися та зберігатися в хешованому вигляді (наприклад, bcrypt).

Передача даних між клієнтом і сервером здійснюється виключно через HTTPS.

Токени автентифікації (JWT) повинні зберігатися безпечно на пристрої користувача (наприклад, Secure Storage для мобільних застосунків, HttpOnly Cookies для веб-застосунків) та мати обмежений термін дії.

Рзмежування прав доступу на серверній частині (ACL) повинно чітко контролювати доступ до функцій та даних відповідно до ролі користувача.

Захист від SQL-ін'єкцій та інших поширених веб-атак (XSS, CSRF).

Регулярне оновлення бібліотек та залежностей для усунення відомих вразливостей

**3.4.5 Придатність до обслуговування**

Код має бути добре структурований, модульний та коментований.

Архітектура системи повинна дозволяти легке додавання нового функціоналу та внесення змін без значного впливу на існуючі компоненти.

Наявність автоматизованих тестів (юніт-тестів, інтеграційних тестів) для ключових компонентів та функцій.

Наявність системи логування та моніторингу для відстеження роботи застосунку та виявлення проблем.

Документація API (наприклад, через Swagger/OpenAPI) повинна бути актуальною та повною.

**3.4.6 Портативність**

Веб-застосунок: повністю функціональний та адаптивний на різних розмірах екранів, включаючи мобільні браузери.

Мобільний застосунок: розроблений на Flutter для забезпечення кросплатформеності на Android та iOS з однієї кодової бази, мінімізуючи зусилля для підтримки двох окремих версій.

**3.5 Зворотні вимоги**

Система не зберігає платіжні дані користувачів (номери карток, CVV). Всі фінансові транзакції здійснюються через зовнішні сервіси (наприклад, Monobank, LiqPay), і система отримує лише статус оплати або прогрес збору через їхні API.

Система не є засобом для офіційних благодійних фондів та не веде офіційну бухгалтерську звітність, а слугує інструментом для координації та прозорості в межах волонтерських ініціатив.

**3.6 Обмеження дизайну**

Загальний стиль: Мінімалістичний, функціональний, з акцентом на чистоту та читабельність.

Кольорова гама: переважно світлі тони для фону. Акцентні кольори використовуються для кнопок, статусів та важливих елементів.

* основний колір інтерфейсу: #2196F3 (синій) – для панелей (хедер, нижня навігація), кнопок, акцентів.
* колір фону/тексту на світлому: #FFFFFF (білий).
* колір тексту на темному: #000000 (чорний).

Типографіка:

* заголовки: Шрифт Impact.
* підзаголовки та основний текст. Шрифт Century Gothic.

Інтерфейс:

* усі користувацькі інтерфейси повинні бути перекладені українською та англійською мовами.
* використання карток із закругленими кутами та тінями для візуального групування інформації.
* стандартні патерни навігації для мобільних застосунків.
* адаптивний дизайн: елементи UI мають коректно відображатися та взаємодіяти на екранах різного розміру та орієнтації

.

**3.7 Вимоги до логічної бази даних**

Будемо використовувати PostgreSQL як основну реляційну базу даних для зберігання всіх ключових даних системи (профілі користувачів, збори, потреби, предмети, звіти). Вибір PostgreSQL обґрунтований його надійністю, підтримкою ACID-властивостей, масштабованістю, розвиненим інструментарієм та можливістю ефективної роботи з транзакціями.

Для зберігання великих обсягів медіафайлів (зображень, документів) буде використано зовнішнє файлове сховище, сумісне з S3 (Simple Storage Service), що дозволить ефективно масштабувати обсяг даних та не перевантажувати основну базу даних великими бінарними об'єктами. У базі даних будуть зберігатися лише URL-посилання на ці файли.