<Програмна система для порятунку та реабілітації диких тварин з використанням геолокаційних сервісів та QR-кодів>

Software Requirements Specification

<Version 1.0>

<01.04.2025>

<Vira Kalenyk>

Frontend Developer

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 31.03.2025 | Version 1 | Vira Kalenyk | First Revision |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Document Approval**

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  | Oleksandr Oliynyk |  | 02.04.2025 |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[1. INTRODUCTION……………………………………………………………………………………..64](#_Toc200813661)

[1.2 Scope ………………………………………………………………………………………………62](#_Toc200813657)

[1.3 Definitions, Acronyms, and Abbreviations …………………………………………………...63](#_Toc200813658)

[1.4 References ……………………………………………………………………………………….. 63](#_Toc200813659)

[1.5 Overview …………………………………………………………………………………………..64](#_Toc200813660)

[2. General Description………………………………………………………………...64](#_Toc200813661)

[2.1 Product Perspective ……………………………………………………………………………...64](#_Toc200813662)

[2.2 Product Functions ………………………………………………………………………………..65](#_Toc200813663)

[2.3 User Characteristics …………………………………………………………………………....65](#_Toc200813664)

[2.4 Operating Environment ………………………………………………………………………….65](#_Toc200813665)

[2.5 Design and Implementation Constraints ……………………………………………………....66](#_Toc200813666)

[2.6 User Documentation ……………………………………………………………………………..66](#_Toc200813667)

[2.7 Assumptions and Dependencies ………………………………………………………………....66](#_Toc200813668)

[3. Specific Requirements ……………………………………………………………...66](#_Toc200813669)

[3.1 External Interface Requirements ………………………………………………………….….66](#_Toc200813670)

[*3.1.1 User Interfaces ……………………………………………………………………………………….….66*](#_Toc200813671)

[*3.1.2 Hardware Interfaces ………………………………………………………………………………….…66*](#_Toc200813672)

[*3.1.3 Software Interfaces ……………………………………………………………………………………...67*](#_Toc200813673)

[*3.1.4 Communications Interfaces ………………………………………………………………………….…67*](#_Toc200813674)

[3.2 Functional Requirements …………………………………………………………………….…67](#_Toc200813675)

[*3.2.1 Додавання інформації про знайдену тварину ………………………………………………….…67*](#_Toc200813676)

[3.2.1.1 Вступ ……………………………………………………………………………………...67](#_Toc200813677)

[3.2.1.2 Вхідні дані ………………………………………………………………………………..67](#_Toc200813678)

[3.2.1.3 Обробка ………………………………………………………………………………......67](#_Toc200813679)

[3.2.1.4 Вихідні дані ……………………………………………………………………………...68](#_Toc200813680)

[3.2.1.5 Обробка помилок ………………………………………………………………………..68](#_Toc200813681)

[*3.2.2 Перегляд і редагування картки тварини ………………………………………………………....68*](#_Toc200813682)

[3.2.2.1 Вступ ……………………………………………………………………………………..68](#_Toc200813683)

[3.2.2.2 Вхідні дані ……………………………………………………………………………….68](#_Toc200813684)

[3.2.2.3 Обробка …………………………………………………...……………………………..68](#_Toc200813685)

[3.2.2.4 Вихідні дані ……………………………………………………………………………..68](#_Toc200813686)

[3.2.2.5 Обробка помилок ……………………………………………………………………….68](#_Toc200813687)

[*3.2.3 Інтерактивна карта з тваринами та клініками ………………………………………………69*](#_Toc200813688)

[3.2.3.1 Вступ …………………………………………………………………………………….69](#_Toc200813689)

[3.2.3.2 Вхідні дані ………………………………………………………………………………69](#_Toc200813690)

[3.2.3.3 Обробка …………………………………………………………………………………69](#_Toc200813691)

[3.2.3.4 Вихідні дані ……………………………………………………………………………..69](#_Toc200813692)

[3.2.3.5 Обробка помилок ……………………………………………………………………….69](#_Toc200813693)

[*3.2.4 Реєстрація та авторизація користувачів ………………………………………………………69*](#_Toc200813694)

[3.2.4.1 Вступ …………………………………………………………………………………….69](#_Toc200813695)

[3.2.4.2 Вхідні дані ……………………………………………………………………………….69](#_Toc200813696)

[3.2.4.3 Обробка …………………………………………………………………………………..69](#_Toc200813697)

[3.2.4.4 Вихідні дані ……………………………………………………………………………...69](#_Toc200813698)

[3.2.4.5 Обробка помилок ………………………………………………………………………..70](#_Toc200813699)

[*3.2.5 Перегляд історії лікування тварини ………………………………………………………………70*](#_Toc200813700)

[3.2.5.1 Вступ ……………………………………………………………………………………..70](#_Toc200813701)

[3.2.5.2 Вхідні дані ……………………………………………………………………………….70](#_Toc200813702)

[3.2.5.3 Обробка …………………………………………………………………………….…….70](#_Toc200813703)

[3.2.5.4 Вихідні дані ………………………………………………………………………….…..70](#_Toc200813704)

[3.2.5.5 Обробка помилок …………………………………………………………………….….70](#_Toc200813705)

[*3.2.6 Панель адміністратора ………………………………………………………………………….….70*](#_Toc200813706)

[3.2.6.1 Вступ ………………………………………………………………………………….….70](#_Toc200813707)

[3.2.6.2 Вхідні дані ………………………………………………………………………….……70](#_Toc200813708)

[3.2.6.3 Обробка …………………………………………………………………………….……70](#_Toc200813709)

[3.2.6.4 Вихідні дані ………………………………………………………………………….…..70](#_Toc200813710)

[3.2.6.5 Обробка помилок …………………………………………………………………….….71](#_Toc200813711)

[*3.2.7 Донати/Платежі ……………………………………………………………………………………..71*](#_Toc200813712)

[3.2.7.1 Вступ ………………………………………………………………………………….….71](#_Toc200813713)

[3.2.7.2 Вхідні дані ……………………………………………………………………………….71](#_Toc200813714)

[3.2.7.3 Обробка ………………………………………………………………………………….71](#_Toc200813715)

[3.2.7.4 Вихідні дані ……………………………………………………………………………..71](#_Toc200813716)

[3.2.7.5 Обробка помилок ……………………………………………………………………….71](#_Toc200813717)

[*3.2.8 Сторінка “Допомога” / FAQ ……………………………………………………………………….71*](#_Toc200813718)

[3.2.8.1 Вступ …………………………………………………………………………………….71](#_Toc200813719)

[3.2.8.2 Вхідні дані ………………………………………………………………………………71](#_Toc200813720)

[3.2.8.3 Обробка ……………………………………………………………………………….…71](#_Toc200813721)

[3.2.8.4 Вихідні дані ………………………………………………………………………….….71](#_Toc200813722)

[*3.2.9 Push-сповіщення ………………………………………………………………………………….…..72*](#_Toc200813723)

[3.2.9.1 Вступ ……………………………………………………………………………………72](#_Toc200813724)

[3.2.9.2 Вхідні дані ………………………………………………………………………………72](#_Toc200813725)

[3.2.9.3 Обробка …………………………………………………………………………………72](#_Toc200813726)

[*3.2.10 Фільтрація, сортування тварин ……………...…………………………………………………72*](#_Toc200813727)

[3.2.10.1 Вступ …………………………………………………………………………………..72](#_Toc200813728)

[3.2.10.2 Вхідні дані …………………………………………………………………………….72](#_Toc200813729)

[3.2.10.3 Обробка ……………………………………………………………………………….72](#_Toc200813730)

[*3.2.11 QR-сканування для доступу до тварини ………………………………………………………72*](#_Toc200813731)

[3.2.11.1 Вступ …………………………………………………………………………………..72](#_Toc200813732)

[3.2.11.2 Вхідні дані …………………………………………………………………………….72](#_Toc200813733)

[3.2.11.3 Обробка ……………………………………………………………………………….72](#_Toc200813734)

[3.3 Use Cases ……………………………………………………………………………………….73](#_Toc200813735)

[*3.3.1 Use Case #1 – Додавання інформації про знайдену тварину ……………………………….73*](#_Toc200813736)

[*3.3.2 Use Case #2 – Перегляд і оновлення інформації про тварину ……………………………...73*](#_Toc200813737)

[*3.3.3 Use Case #3 – Перегляд інтерактивної карти ……………………………….…………………..74*](#_Toc200813738)

[*3.3.4 Use Case #4 – Надсилання фінансової допомоги ……………………………….………………..74*](#_Toc200813739)

[*3.3.5 Use Case #5 – Реєстрація нового користувача …………………………………….…………….74*](#_Toc200813740)

[*3.3.6 Use Case #6 – Отримання сповіщень про зміни …………………………………………………75*](#_Toc200813741)

[*3.3.7 Use Case #7 – Перегляд інформації про тварину через QR-код ………………………………75*](#_Toc200813742)

[3.4 Classes / Objects …………………………………………………………………………………76](#_Toc200813743)

[*3.4.1 Class / Object: Animal ………………………………………………………………………………….76*](#_Toc200813744)

[3.4.1.1 Attributes: …………………………………………………………………………………76](#_Toc200813745)

[3.4.1.2 Functions: …………………………………………………………………………………76](#_Toc200813746)

[Reference: ………………………………………………………………………………….……...76](#_Toc200813747)

[*3.4.2 Class / Object: User …………………………………………………………………………….………76*](#_Toc200813748)

[3.4.2.1 Attributes: ………………………………………………………………………….………76](#_Toc200813749)

[3.4.2.2 Functions: ………………………………………………………………………….………77](#_Toc200813750)

[Reference: …………………………………………………………………………………………77](#_Toc200813751)

[*3.4.3 Class / Object: Map ……………………………………………………………………………………..77*](#_Toc200813752)

[3.4.3.1 Attributes: ………………………………………………………………………………….77](#_Toc200813753)

[3.4.3.2 Functions: ………………………………………………………………………………….77](#_Toc200813754)

[Reference: ………………………………………………………………………………….………77](#_Toc200813755)

[*3.4.4 Class / Object: Donation ……………………………………………………………………….……….77*](#_Toc200813756)

[3.4.4.1 Attributes: ………………………………………………………………………….………77](#_Toc200813757)

[3.4.4.2 Functions: ………………………………………………………………………….………78](#_Toc200813758)

[Reference: …………………………………………………………………………………………78](#_Toc200813759)

[*3.4.5 Class / Object: Notification …………………………………………………………………………….78*](#_Toc200813760)

[3.4.5.1 Attributes …………………………………………………………………………………..78](#_Toc200813761)

[3.4.5.2 Functions …………………………………………………………………………………..78](#_Toc200813762)

[Reference: …………………………………………………………………………………………78](#_Toc200813763)

[*3.4.6 Class / Object: MedicalRecord ………………………………………………………………………...78*](#_Toc200813764)

[3.4.6.1 Attributes …………………………………………………………………………………..78](#_Toc200813765)

[3.4.6.2 Functions …………………………………………………………………………………..79](#_Toc200813766)

[Reference: …………………………………………………………………………………………79](#_Toc200813767)

[3.5 Non-Functional Requirements …………………………………………………………………79](#_Toc200813768)

[*3.5.1 Performance ……………………………………………………………………………………….……..79*](#_Toc200813769)

[*3.5.2 Reliability ………………………………………………………………………………………….……..79*](#_Toc200813770)

[*3.5.3 Availability ………………………………………………………………………………………….……79*](#_Toc200813771)

[*3.5.4 Security ……………………………………………………………………………………………….…..79*](#_Toc200813772)

[*3.5.5 Maintainability ………………………………………………………………………………………….80*](#_Toc200813773)

[*3.5.6 Portability ……………………………………………………………………………………………….80*](#_Toc200813774)

[3.6 Inverse Requirements …………………………………………………………………………..80](#_Toc200813775)

[3.7 Design Constraints ……………………………………………………………………………..80](#_Toc200813776)

[3.8 Logical Database Requirements ………………………………………………………………80](#_Toc200813777)

[3.9 Other Requirements …………………………………………………………………………….81](#_Toc200813778)

[4. Change Management Process ………………………..……………………...…...81](#_Toc200813779)

**1. Introduction**

Цей документ визначає вимоги до програмної системи для порятунку та реабілітації диких тварин з використанням геолокаційних сервісів та QR-кодів. Документ призначений для розробників проєкту, щоб учасники мали чітке уявлення про функціональність системи. Він слугує технічною основою для розробки, тестування та впровадження продукту.

**1.1 Purpose**

Метою документа є формалізація функціональних та нефункціональних вимог до розробки веб-застосунку, що координує порятунок диких тварин, їхнє лікування та реабілітацію. Система повинна забезпечувати можливість фіксації інформації про знайдених тварин, передачу цієї інформації відповідним службам, відстеження процесу лікування та надання доступу до історії тварини через QR-код.

**1.2 Scope**

1) Назва продукту: Програмна система для порятунку та реабілітації диких тварин.

2) Функціональні можливості:

* Реєстрація, авторизація та відновлення доступу.
* Додавання повідомлень про тварин, що потребують допомоги.
* Перегляд карти з розташуванням тварин, клінік та центрів.
* Адмін-панель для керування користувачами, тваринами, історіями лікування.
* Сканування QR-кодів для отримання даних про тварину.
* Звітність про врятованих тварин.

3) Обмеження:

* Система не керує внутрішніми процесами ветеринарних клінік або реабілітаційних центрів.
* Не виконує автоматичну діагностику стану тварин.

4) Цілі проєкту:

* Створення єдиної цифрової платформи для фіксації випадків виявлення тварин.
* Полегшення комунікації між волонтерами, ветеринарами та центрами.
* Забезпечення оперативного реагування на випадки через інтерактивну карту.
* Забезпечення швидкого доступу до історії лікування тварини за допомогою QR-кодів.
* Підвищення обізнаності громадян та зручності повідомлення про випадки.

5) Призначення програми:

* Система орієнтована на зоозахисні організації, ветеринарів, реабілітаційні центри та активних громадян, які мають змогу долучитися до процесу порятунку диких тварин.
* Продукт має бути доступним як через вебінтерфейс (десктоп/мобільні браузери), так і у вигляді мобільного застосунку (у перспективі).

**1.3 Definitions, Acronyms, and Abbreviations**

*Таблиця 1 – Терміни та абревіатури*

|  |  |
| --- | --- |
| **Термін / Абревіатура** | **Розшифрування / Пояснення** |
| *SRS* | *Software Requirements Specification — Специфікація вимог до програмного забезпечення* |
| *SPA* | *Single Page Application — односторінковий вебзастосунок* |
| *API* | *Application Programming Interface — програмний інтерфейс прикладного програмування* |
| *UI* | *User Interface — користувацький інтерфейс* |
| *UX* | *User Experience — досвід взаємодії користувача із застосунком* |
| *CRUD* | *Create, Read, Update, Delete — основні операції з даними* |
| *REST API* | *Представлення переданих ресурсів через HTTP-запити у форматі JSON* |
| *HTTPS* | *HyperText Transfer Protocol Secure — захищений протокол передачі даних* |
| *QR-код* | *Quick Response Code — код швидкого доступу для ідентифікації тварини* |
| *Docker* | *Платформа контейнеризації, що використовується для розгортання серверної частини* |
| *Redis* | *Система кешування та управління чергами повідомлень у серверній частині* |

**1.4 References**

1. *Angular Documentation —* [*https://angular.io/docs*](https://angular.io/docs)
2. *JSON Specification —* [*https://www.json.org/json-en.html*](https://www.json.org/json-en.html)
3. *Docker Documentation —* [*https://docs.docker.com/*](https://docs.docker.com/)
4. *Redis Documentation —* [*https://redis.io/documentation*](https://redis.io/documentation)

**1.5 Overview**

*Цей документ специфікації програмних вимог (SRS) містить повний опис функціональних і нефункціональних вимог до розроблюваного веб-застосунку — програмної системи для порятунку та реабілітації диких тварин з використанням геолокаційних сервісів та QR-кодів.*

*Документ структуровано наступним чином:*

* *Розділ 1 містить загальний опис системи, включаючи її мету, сферу застосування, цільову аудиторію, аналіз предметної області та аналіз існуючих аналогів.*
* *Розділ 2 — загальний опис, що розкриває контекст розробки, особливості продукту, користувачів і середовище експлуатації.*
* *Розділ 3 детально описує функціональні та нефункціональні вимоги, інтерфейси, логіку обробки, а також варіанти використання (use cases).*
* *Розділ 4 — моделі аналізу, зокрема діаграми потоків даних, діаграми переходів станів та інші візуальні моделі для ілюстрації логіки роботи системи.*
* *Розділ 5 — описує процес управління змінами у вимогах.*
* *Додатки містять додаткові матеріали, приклади, нотатки з обговорень тощо.*

**2. General Description**

**2.1 Product Perspective**

*Програмна система для порятунку та реабілітації диких тварин є частиною екосистеми цифрових рішень у сфері охорони довкілля та зоозахисту. Сервіс створюється як незалежна платформа, що забезпечує взаємодію між звичайними користувачами, волонтерами, ветеринарами та адміністраторами реабілітаційних центрів.*

*Система надає можливість повідомляти про постраждалих тварин, координувати їхній порятунок, моніторити процес лікування та реабілітації, а також формувати єдину базу з усією історією кожної тварини. Вона також використовує геолокаційні сервіси для точного визначення місця знаходження тварин і підтримує QR-коди для швидкої ідентифікації.*

*Існують схожі продукти, але жоден не поєднує одночасно картографічну інтеграцію, історію лікування, QR-ідентифікацію та гнучкий доступ для різних типів користувачів. Дана система націлена на покращення ефективності координації, підвищення обізнаності населення та оперативне реагування на ситуації, коли тварини потребують допомоги.*

**2.2 Product Functions**

*Основні можливості фронтенд-застосунку:*

1. *Реєстрація та вхід (через email, Google або Facebook, відновлення пароля)*
2. *Керування профілем (перегляд і редагування даних користувача, перемикання між світлою/темною темами)*
3. *Додавання тварин (форма з фото, описом, координатами, завантаження на сервер через API)*
4. *Карта з маркерами (відображення тварин, клінік, реабілітаційних центрів)*
5. *Історія тварини (перегляд подій, статусів, коментарів)*
6. *Панель адміністратора (управління користувачами, тваринами, звітністю, захищений доступ)*
7. *Система сповіщень (Toast/Alert повідомлення для подій (оновлення, успішна дія, помилка))*
8. *Підтримка пожертв (інтеграція платіжного модуля)*
9. *Фільтрація та пошук тварин (за видом, статусом, датою, розташуванням)*

**2.3 User Characteristics**

*1) Незареєстровані користувачі (перегляд відкритої інформації, створення звернення про тварину)*

*2) Зареєстровані користувачі (повідомлення про тварин, пожертвування, перегляд статусу тварин)*

*3) Волонтери та ветеринари (редагування даних про тварин, коментарі, додавання історії лікування, взаємодія через систему повідомлень)*

*4) Локальні адміністратори (керування персоналом у межах центру/клініки, перевірка історій, документація)*

*5) Глобальний адміністратор (повний контроль над системою, управління доступом, перевірка змін)*

**2.4 Operating Environment**

*Клієнтська частина застосунку функціонуватиме у сучасних веббраузерах, зокрема Google Chrome, Mozilla Firefox та Microsoft Edge. Завдяки адаптивному дизайну інтерфейс коректно відображатиметься на смартфонах, планшетах і десктопах. Фронтенд взаємодіє із серверною частиною через REST API, реалізоване на основі технологій Python, MariaDB, Redis та Docker. Для забезпечення захищеного обміну даними використовується протокол HTTPS. Архітектура Single Page Application (SPA) дозволяє здійснювати швидке перемикання між сторінками без повного перезавантаження, що забезпечує плавну та зручну взаємодію з інтерфейсом.*

**2.5 Design and Implementation Constraints**

*Фронтенд реалізується виключно з використанням Angular версії 15 або вище. Всі API-запити надсилаються через протокол HTTP(S) у форматі JSON. При необхідності, дані кешуються на рівні сервісів Angular для підвищення продуктивності. Передбачено інтеграцію з картографічними API та платіжними шлюзами.*

**2.6 User Documentation**

*До застосунку буде надано короткий текстовий посібник з прикладами використання в форматах PDF або HTML, відеоінструкцію, що демонструє основні дії користувача, розміщену на YouTube або локальному сервері, а також розділ «Питання та відповіді» (FAQ), доступний на сторінці «Допомога».*

**2.7 Assumptions and Dependencies**

*Передбачається, що користувачі мають стабільне з’єднання з Інтернетом та користуються підтримуваними браузерами. Також очікується, що користувачі володіють базовими навичками роботи з веб-застосунками. Робота клієнтської частини залежить від коректного функціонування REST API серверної частини, а також сторонніх сервісів, зокрема картографічних, автентифікаційних і платіжних.*

**3. Specific Requirements**

**3.1 External Interface Requirements**

**3.1.1 User Interfaces**

* *UI-1: Інтерфейс користувача повинен бути доступний як на десктопних, так і на мобільних пристроях з адаптивним дизайном.*
* *UI-2: Інтерфейс додавання тварини повинен містити зручну форму для введення даних: вид, фото, стан, геолокація, коментарі.*
* *UI-3: Панель волонтера/ветеринара має дозволяти перегляд і редагування картки тварини, додавання історії лікування.*
* *UI-4: Панель адміністратора повинна містити можливості керування користувачами, клініками, правами доступу та статистикою.*

**3.1.2 Hardware Interfaces**

* *HW-1: Застосунок має бути сумісним із сучасними пристроями під керуванням Windows, macOS, Android, iOS.*
* *HW-2: Не вимагається використання спеціалізованого апаратного забезпечення.*

**3.1.3 Software Interfaces**

* *SW-1: Клієнтська частина повинна використовувати REST API для взаємодії з бекендом.*
* *SW-2: Повинна бути реалізована інтеграція з картографічними сервісами (Google Maps / OpenStreetMap).*
* *SW-3: Повинна бути реалізована інтеграція з платіжними сервісами.*

**3.1.4 Communications Interfaces**

* *COM-1: Всі запити до API мають виконуватись через HTTPS.*
* *COM-2: Система повинна підтримувати надсилання email-сповіщень користувачам (через SMTP або сторонні сервіси).*

**3.2 Functional Requirements**

**3.2.1 Додавання інформації про знайдену тварину**

**3.2.1.1 Вступ**

*Ця функція дозволяє користувачу (навіть без реєстрації) повідомити про знайдену дику тварину через веб-форму. У фронтенді реалізована як окремий компонент, що включає введення даних, валідацію, завантаження фото та передачу даних на сервер через API. Метою є забезпечення швидкої фіксації випадку та сповіщення відповідних волонтерів і ветеринарів.*

**3.2.1.2 Вхідні дані**

* *IN-1: Введення текстових даних (вид тварини, місце знахідки, стан тварини).*
* *IN-2: Вибір або введення геолокації (через карту або автоматично).*
* *IN-3: Завантаження одного або декількох фото.*
* *IN-4: Натискання кнопки "Надіслати".*

**3.2.1.3 Обробка**

* *PR-1: Валідація обов’язкових полів (вид, місце, стан).*
* *PR-2: Перевірка типу/розміру зображень перед відправкою.*
* *PR-3: Формування JSON-запиту та надсилання до REST API.*
* *PR-4: Очікування відповіді від API з повідомленням про успіх або помилку.*

**3.2.1.4 Вихідні дані**

* *OUT-1: Повідомлення користувачу про успішне додавання.*
* *OUT-2: Автоматичне перенаправлення на іншу сторінку або відображення форми для нової заявки.*
* *OUT-3: Toast-сповіщення про статус (успіх/помилка).*

**3.2.1.5 Обробка помилок**

* *ER-1: Якщо обов’язкові поля не заповнені — підсвітка помилок під полями.*
* *ER-2: Якщо файл фото має невірний формат/розмір — попередження біля завантаження.*
* *ER-3: Якщо сервер не відповідає — виведення повідомлення “Неможливо зв’язатися із сервером”.*
* *ER-4: Якщо сервер повертає помилку валідації — відображення відповідного повідомлення користувачу.*

**3.2.2 Перегляд і редагування картки тварини**

**3.2.2.1 Вступ**

*Ця функція дозволяє зареєстрованим волонтерам та ветеринарам переглядати повну інформацію про тварину, а також редагувати дані (статус, медичну історію, фото).*

**3.2.2.2 Вхідні дані**

* *IN-1: ID тварини, що обрана для перегляду.*
* *IN-2: Нові дані картки тварини (статус, опис, документи, фото).*
* *IN-3: Авторизаційні дані користувача.*

**3.2.2.3 Обробка**

* *PR-1: Отримання детальної інформації про тварину через API.*
* *PR-2: Перевірка прав доступу.*
* *PR-3: Надсилання змінених даних до сервера.*

**3.2.2.4 Вихідні дані**

* *OUT-1: Візуальна картка тварини з усією інформацією.*
* *OUT-2: Сповіщення про успішне оновлення даних.*

**3.2.2.5 Обробка помилок**

* *ER-1: Відсутність доступу — виведення повідомлення.*
* *ER-2: Некоректні або порожні поля — валідація клієнтської частини.*

**3.2.3 Інтерактивна карта з тваринами та клініками**

**3.2.3.1 Вступ**

*Функція дозволяє користувачам переглядати тварин, клініки, волонтерські групи на карті, отримувати маршрут до об’єкта.*

**3.2.3.2 Вхідні дані**

* *IN-1: Позиція користувача (за дозволом браузера).*
* *IN-2: Фільтри (тип тварини, статус, тип об’єкта на карті).*

**3.2.3.3 Обробка**

* *PR-1: Отримання геоданих з API сервера.*
* *PR-2: Завантаження мапи (Google Maps/OpenStreetMap).*
* *PR-3: Побудова маршрутів (за потреби).*

**3.2.3.4 Вихідні дані**

* *OUT-1: Інтерактивна мапа з маркерами.*
* *OUT-2: Детальна інформація про об’єкт при натисканні на маркер.*

**3.2.3.5 Обробка помилок**

* *ER-1: Відсутність дозволу на геолокацію — відображення повідомлення.*
* *ER-2: Неможливо завантажити карту — сповіщення про технічну помилку.*

**3.2.4 Реєстрація та авторизація користувачів**

**3.2.4.1 Вступ***Забезпечує вхід, реєстрацію, відновлення паролю та вхід через Google/OAuth.*

**3.2.4.2 Вхідні дані**

* *IN-1: Email/пароль або OAuth-токен.*
* *IN-2: Дані реєстраційної форми (ім’я, email, роль).*

**3.2.4.3 Обробка**

* *PR-1: Валідація введених даних.*
* *PR-2: Взаємодія з API для аутентифікації.*
* *PR-3: Отримання токену доступу та збереження у браузері.*

**3.2.4.4 Вихідні дані**

* *OUT-1: Перенаправлення на головну сторінку після входу.*
* *OUT-2: Повідомлення про помилки входу.*

**3.2.4.5 Обробка помилок**

* *ER-1: Невірні дані — повідомлення про помилку.*
* *ER-2: Відсутність відповіді від сервера — вивід загального повідомлення.*

**3.2.5 Перегляд історії лікування тварини**

**3.2.5.1 Вступ***Дозволяє переглядати всі записи лікування конкретної тварини.*

**3.2.5.2 Вхідні дані**

* *IN-1: ID тварини.*

**3.2.5.3 Обробка**

* *PR-1: Отримання історії з API.*
* *PR-2: Форматування записів у хронологічному порядку.*

**3.2.5.4 Вихідні дані**

* *OUT-1: Таблиця/список з лікувальними записами та датами.*

**3.2.5.5 Обробка помилок**

* *ER-1: Якщо історію не знайдено — повідомлення користувачу.*

**3.2.6 Панель адміністратора**

**3.2.6.1 Вступ**

*Функція для глобальних і локальних адміністраторів керування користувачами, клініками, тваринами.*

**3.2.6.2 Вхідні дані**

* *IN-1: Авторизація адміністратора.*
* *IN-2: CRUD-запити (додати, редагувати, видалити об’єкти).*

**3.2.6.3 Обробка**

* *PR-1: Перевірка прав доступу.*
* *PR-2: Отримання списків через API.*
* *PR-3: Відправка змін на сервер.*

**3.2.6.4 Вихідні дані**

* *OUT-1: Інтерфейс зі списками користувачів, тварин, закладів.*

**3.2.6.5 Обробка помилок**

* *ER-1: Немає прав доступу — редирект.*
* *ER-2: Серверна помилка — сповіщення.*

**3.2.7 Донати/Платежі**

**3.2.7.1 Вступ**

*Надає можливість пожертвувати кошти через вбудовану платіжну систему.*

**3.2.7.2 Вхідні дані**

* *IN-1: Сума, реквізити платника.*
* *IN-2: Вибраний одержувач (центр/волонтер).*

**3.2.7.3 Обробка**

* *PR-1: Завантаження модуля платежів (iframe/SDK).*
* *PR-2: Передача даних до платіжного шлюзу.*

**3.2.7.4 Вихідні дані**

* *OUT-1: Повідомлення про успішну оплату.*

**3.2.7.5 Обробка помилок**

* *ER-1: Помилка при оплаті — повідомлення та пропозиція повторити.*

**3.2.8 Сторінка “Допомога” / FAQ**

**3.2.8.1 Вступ**

*Інформаційний розділ із часто заданими питаннями та інструкціями.*

**3.2.8.2 Вхідні дані**

* *IN-1: Запит на сторінку.*

**3.2.8.3 Обробка**

* *PR-1: Завантаження статичного контенту або з API.*

**3.2.8.4 Вихідні дані**

* *OUT-1: Вивід тексту, посилань, відеоінструкцій.*

**3.2.9 Push-сповіщення**

**3.2.9.1 Вступ**

*Система сповіщень у вигляді toast або alert, що інформує про події.*

**3.2.9.2 Вхідні дані**

* *IN-1: Дані про подію з WebSocket або API.*

**3.2.9.3 Обробка**

* *PR-1: Перевірка типу події.*
* *PR-2: Відображення toast повідомлення Angular-сервісом.*

**3.2.10 Пошук, фільтрація, сортування тварин**

**3.2.10.1 Вступ**

*Реалізація зручного пошуку за ключовими параметрами тварин.*

**3.2.10.2 Вхідні дані**

* *IN-1: Ключові слова, фільтри (статус, вид, дата).*

**3.2.10.3 Обробка**

* *PR-1: Валідація введення.*
* *PR-2: Запит до API, застосування фільтрів локально/на сервері.*

**3.2.11 QR-сканування для доступу до тварини**

**3.2.11.1 Вступ**

*Сканування QR-коду відкриває картку тварини для перегляду історії.*

**3.2.11.2 Вхідні дані**

* *IN-1: QR-код із ID тварини.*

**3.2.11.3 Обробка**

* *PR-1: Сканування через WebRTC/модуль Angular.*
* *PR-2: Переадресація на картку тварини.*

**3.3 Use Cases**

**3.3.1 Use Case #1 – Додавання інформації про знайдену тварину**

*Актори: Незареєстрований або зареєстрований користувач  
Опис: Користувач знаходить поранену тварину та хоче повідомити про це через веб-застосунок.  
Попередні умови: Користувач має доступ до інтернету та пристрою з веб-браузером.  
Основний сценарій:*

1. *Користувач відкриває форму додавання тварини.*
2. *Заповнює обов’язкові поля: вид тварини, місце виявлення, стан.*
3. *Завантажує фото.*
4. *Натискає кнопку «Надіслати».*
5. *Система перевіряє введені дані.*
6. *Якщо все коректно – зберігає інформацію та показує повідомлення про успіх.*
7. *Дані відображаються на карті та стають доступні ветеринарам/волонтерам.*

*Альтернативні сценарії:*

* *Якщо обов’язкові поля не заповнені — система виводить повідомлення про помилку.*
* *Якщо завантажене фото має некоректний формат — користувачу пропонується змінити файл.*

**3.3.2 Use Case #2 – Перегляд і оновлення інформації про тварину**

*Актори: Волонтер або ветеринар  
Опис: Авторизований користувач переглядає картку тварини та оновлює дані (статус, історію лікування).*

*Попередні умови: Користувач має відповідні права доступу.  
Основний сценарій:*

1. *Користувач авторизується в системі.*
2. *Відкриває сторінку обраної тварини.*
3. *Натискає «Редагувати».*
4. *Вносить зміни: оновлює статус, додає медичний запис або фото.*
5. *Зберігає зміни.*
6. *Система перевіряє права та зберігає нові дані.*

*Альтернативні сценарії:*

* *Якщо користувач не має прав — система блокує редагування.*
* *Якщо дані некоректні — виводиться повідомлення про помилку.*

**3.3.3 Use Case #3 – Перегляд інтерактивної карти**

*Актори: Усі користувачі  
Опис: Користувач переглядає карту з місцями перебування тварин і найближчими клініками/центрами.  
Основний сценарій:*

1. *Користувач відкриває розділ «Карта».*
2. *Система відображає поточне розташування користувача (якщо дозволено).*
3. *Користувач бачить позначки на карті:*

* *Ветеринарні клініки*
* *Центри реабілітації*

1. *Можна натиснути на будь-яку позначку для перегляду детальної інформації.*

*Альтернативні сценарії:*

* *Якщо геолокація недоступна — карта відкривається із загальним масштабом.*

**3.3.4 Use Case #4 – Надсилання фінансової допомоги**

*Актори: Зареєстрований користувач  
Опис: Користувач робить пожертву на лікування тварини або підтримку центру.  
Основний сценарій:*

* 1. *Користувач переходить на сторінку тварини або центру.*
  2. *Натискає «Підтримати».*
  3. *Обирає суму та платіжний метод.*
  4. *Переходить до платіжного інтерфейсу.*
  5. *Після успішної оплати отримує підтвердження.*

*Альтернативні сценарії:*

* *При помилці під час оплати — користувачу пропонується повторити спробу або змінити метод.*

**3.3.5 Use Case #5 – Реєстрація нового користувача**

*Актори: Новий користувач  
Опис: Користувач створює обліковий запис для взаємодії з системою (як волонтер, ветеринар або звичайний користувач).  
Попередні умови: Користувач має доступ до форми реєстрації.  
Основний сценарій:*

1. *Користувач відкриває сторінку реєстрації.*
2. *Обирає тип акаунта (звичайний, волонтер, ветеринар).*
3. *Вводить необхідні дані: ім’я, електронну пошту, пароль тощо.*
4. *Підтверджує дані та погоджується з умовами.*
5. *Система перевіряє унікальність пошти та створює обліковий запис.*
6. *Користувач отримує лист для підтвердження реєстрації.*

*Альтернативні сценарії:*

* *Якщо пошта вже використовується — система виводить помилку.*
* *Якщо поля не заповнені або некоректні — з’являється повідомлення з інструкціями.*

**3.3.6 Use Case #6 – Отримання сповіщень про зміни**

*Актори: Зареєстровані користувачі, які підписані на оновлення по тваринах або установах  
Опис: Користувач отримує push- або toast-сповіщення в застосунку щодо змін (нові тварини, зміни статусу).  
Основний сценарій:*

1. *Користувач авторизується в системі.*
2. *Переглядає головну сторінку або інші розділи.*
3. *У разі події (оновлення статусу тварини, нова заявка, повідомлення) – система автоматично відображає сповіщення.*
4. *Користувач може натиснути на сповіщення для переходу до відповідного розділу.*

*Альтернативні сценарії:*

* *Якщо користувач не підписаний — сповіщення не надходять.*
* *Якщо відключено сповіщення в браузері — система запропонує їх активувати.*

**3.3.7 Use Case #7 – Перегляд інформації про тварину через QR-код**

*Актори: Будь-який користувач (зареєстрований або ні)  
Опис: Користувач сканує QR-код тварини, щоб переглянути її стан, історію лікування та додану інформацію.  
Основний сценарій:*

1. *Користувач сканує QR-код (через камеру смартфона або окрему сторінку).*
2. *Система розпізнає код та завантажує дані тварини.*
3. *Користувач бачить короткий опис, фото, історію, статус реабілітації.*

*Альтернативні сценарії:*

* *Якщо код некоректний або тварина не знайдена — система повідомляє про помилку.*
* *Якщо тварина має обмеження на публічний доступ — показується лише базова інформація.*

**3.4 Classes / Objects**

**3.4.1 Class / Object: Animal**

**3.4.1.1 Attributes:**

* *id: унікальний ідентифікатор*
* *species: вид тварини*
* *location: координати місця знаходження*
* *status: статус (наприклад: «потребує допомоги», «у реабілітації», «випущена»)*
* *photoUrl: посилання на фото тварини*
* *healthInfo: короткий опис стану*
* *medicalHistory: список медичних записів*
* *assignedUsers: користувачі, які доглядають за твариною*

**3.4.1.2 Functions:**

* *getAnimalDetails(): отримати повну інформацію про тварину*
* *updateStatus(): змінити статус тварини*
* *addMedicalRecord(): додати медичний запис*
* *assignUser(): призначити відповідального*
* *generateQRCode(): згенерувати QR-код тварини*

**Reference:**

* *Functional requirements: 3.2.1, 3.2.2*
* *Use Cases: 3.3.1, 3.3.2, 3.3.7, 3.3.8*

**3.4.2 Class / Object: User**

**3.4.2.1 Attributes:**

* *id: унікальний ідентифікатор користувача*
* *fullName: ім’я та прізвище*
* *email: електронна пошта*
* *role: роль у системі (звичайний, волонтер, ветеринар, адміністратор)*
* *photoUrl: аватар*
* *assignedAnimals: список тварин, за якими закріплено користувача*

**3.4.2.2 Functions:**

* *register(): реєстрація користувача*
* *login(): авторизація*
* *updateProfile(): редагування профілю*
* *sendMessage(toUserId): відправити повідомлення*
* *viewAssignedAnimals(): перегляд тварин, за яких відповідає*

**Reference:**

* *Functional requirements: 3.2.6*
* *Use Cases: 3.3.4, 3.3.5, 3.3.10*

**3.4.3 Class / Object: Map**

**3.4.3.1 Attributes:**

* *centerCoordinates: координати центра карти*
* *animalMarkers: маркери розташування тварин*
* *clinicMarkers: маркери клінік*
* *filterSettings: поточні фільтри відображення*

**3.4.3.2 Functions:**

* *loadMap(): ініціалізувати карту*
* *addMarker(type, location): додати маркер (тварина/клініка)*
* *filterMarkers(type): фільтрація відображення*
* *navigateTo(location): побудова маршруту*

**Reference:**

* *Functional requirements: 3.2.3, 3.2.4*
* *Use Cases: 3.3.3, 3.3.6*

**3.4.4 Class / Object: Donation**

**3.4.4.1 Attributes:**

* *id: ідентифікатор транзакції*
* *amount: сума пожертви*
* *recipientId: одержувач (волонтер/центр)*
* *donorId: користувач, що зробив внесок*
* *date: дата транзакції*

**3.4.4.2 Functions:**

* *initiateDonation(): ініціалізація пожертви*
* *processPayment(): інтеграція з платіжним сервісом*
* *getHistory(): перегляд історії транзакцій*

**Reference:**

* *Functional requirements: 3.2.7*
* *Use Cases: 3.3.9*

**3.4.5 Class / Object: Notification**

**3.4.5.1 Attributes**

* *id: унікальний ідентифікатор сповіщення*
* *type: тип сповіщення (оновлення статусу тварини, нова заявка, підтвердження донату тощо)*
* *message: текст повідомлення*
* *userId: ID користувача, для якого призначене сповіщення*
* *isRead: булеве значення, що вказує, чи було прочитано повідомлення*
* *timestamp: дата та час створення*

**3.4.5.2 Functions**

* *markAsRead(): відмітити сповіщення як прочитане*
* *displayToUser(): вивести сповіщення у UI через toast / alert компонент*

**Reference:**

* *Use Cases 3.3.2, 3.3.5*
* *Functional Requirements REQ-12, REQ-13, REQ-19*

**3.4.6 Class / Object: MedicalRecord**

**3.4.6.1 Attributes**

* *id: унікальний ідентифікатор запису*
* *animalId: ID тварини, до якої належить запис*
* *createdBy: ID ветеринара, що створив запис*
* *description: опис процедури чи лікування*
* *files: список прикріплених медичних документів або фото*
* *date: дата створення запису*

**3.4.6.2 Functions**

* *createRecord(): створити новий запис*
* *editRecord(): редагувати існуючий запис*
* *attachFile(): прикріпити документ або зображення до запису*
* *viewHistory(): отримати повну історію записів по тварині*

**Reference:**

* *Use Case 3.3.3*
* *Functional Requirements REQ-7, REQ-8, REQ-10*

**3.5 Non-Functional Requirements**

**3.5.1 Performance**

* *Інтерфейс повинен завантажуватись не довше 1 секунди на сучасних пристроях при нормальному з’єднанні.*
* *95% запитів до API повинні оброблятися менш ніж за 2 секунди.*
* *Перемикання між розділами SPA повинно відбуватись без перезавантаження сторінки за < 500 мс.*

**3.5.2 Reliability**

* *Система повинна мати рівень успішного виконання функцій не нижче 99% протягом місяця.*
* *Всі помилки, що виникають на стороні клієнта, мають логуватись та виводитись у вигляді повідомлення користувачу.*

**3.5.3 Availability**

* *Фронтенд має бути доступним користувачам 24/7.*
* *Максимально допустимий час простою – не більше 1 хвилини на добу.*
* *Застосунок повинен автоматично відновлювати з’єднання при втраті доступу до сервера.*

**3.5.4 Security**

* *Всі запити до API мають виконуватись через HTTPS.*
* *Доступ до окремих функцій обмежено відповідно до ролі користувача.*
* *Дані сесії зберігаються безпечно (JWT або інший механізм авторизації).*
* *Валідація форм здійснюється як на фронтенді, так і на бекенді.*

**3.5.5 Maintainability**

* *Архітектура клієнтської частини має бути модульною з чітким розділенням компонентів.*
* *Використання Angular best practices (сервіси, RxJS, маршрути, DI).*
* *Код повинен мати коментарі до всіх складних функцій та бути легко розширюваним.*

**3.5.6 Portability**

* *Застосунок має підтримуватись у таких браузерах: Chrome, Firefox, Edge (останні 2 версії).*
* *Підтримка пристроїв: смартфони, планшети, десктопи.*
* *Код повинен бути адаптований до розгортання як у локальному середовищі, так і на хмарних платформах (через CI/CD).*

**3.6 Inverse Requirements**

* *Інтерфейс не повинен бути доступним без автентифікації для дій, що вимагають реєстрації (редагування тварин, перегляд медичних записів).*
* *Система не повинна зберігати чутливі дані у відкритому вигляді в локальному сховищі браузера.*

**3.7 Design Constraints**

* *Фреймворк: Angular (версія 15+)*
* *Використання REST API (JSON формат).*
* *Сторонні бібліотеки повинні бути ліцензовані для вільного/відкритого використання.*
* *Дизайн має відповідати WCAG 2.1 для базової доступності (контрастність, навігація з клавіатури).*
* *Контейнеризація у Docker передбачена для тестового середовища.*

**3.8 Logical Database Requirements**

* *Вся інформація про тварин, користувачів, сповіщення, транзакції – зберігається у базі даних.*
* *Формати даних мають бути уніфіковані (наприклад, дата у форматі ISO 8601).*
* *Дані повинні мати індексацію для швидкого доступу (особливо – тварини з певним статусом, локацією тощо).*
* *Має бути реалізовано логування змін до медичних записів тварин.*
* *Історія змін не повинна видалятись, лише оновлюватись з позначкою часу.*

**3.9 Other Requirements**

* *Повинна бути реалізована підтримка темної та світлої теми.*
* *Інтерфейс має підтримувати українську та англійську мови.*
* *Після кожного релізу фронтенду має проводитись smoke-тестування основних функцій.*
* *Повинна бути реалізована можливість збору статистики (наприклад, кількість збережених тварин, активні користувачі) – з подальшою візуалізацією на адмін-панелі (опціонально).*

**4.** **Change Management Process**

*Процес управління змінами в SRS-документі передбачає фіксацію, оцінку та затвердження будь-яких змін до вимог, що впливають на функціональність, дизайн або архітектуру фронтенд-застосунку.*

1. *Зміни можуть бути запропоновані будь-яким членом команди. Ініціатор надсилає опис пропонованої зміни, її обґрунтування та потенційний вплив на систему.*
2. *Команда розробки оцінює, як запропонована зміна вплине на поточну реалізацію, строки розробки та узгоджені функціональні вимоги. Якщо зміна зачіпає інші частини проєкту (наприклад, API або логіку бекенду), проводиться спільна координація.*
3. *Команда ухвалює рішення про включення зміни. У випадку великих змін – питання погоджується з викладачем.*
4. *У випадку затвердження зміни, відповідні розділи SRS оновлюються з обов’язковим маркуванням зміни (наприклад, через контроль версій або збереження історії редагувань).*
5. *Після внесення змін усі отримують сповіщення про оновлення вимог.*