

УДК 004.42:004.5]:636.09

**ПРОЄКТУВАННЯ АРХІТЕКТУРИ КЛІЄНТСЬКОЇ ЧАСТИНИ  
СИСТЕМИ МОНІТОРИНГУ СТАНУ ЗДОРОВ'Я  
ДОМАШНІХ ТВАРИН**

Зозуля В.А.

e-mail: viktoriia.zozulia@nure.ua

Науковий керівник – к.т.н., проф. Ситніков Д.Е.

Харківський національний університет радіоелектроніки, каф. КМІТ  
м. Харків, Україна

This paper explores the architectural and technological approaches to designing the client-side interface for a pet health monitoring system. The primary focus is on creating a responsive, highly interactive dashboard that allows pet owners to track daily behavioral changes, dietary habits, and rehabilitation progress, which is critical for animals recovering from street conditions or illnesses. To achieve optimal performance and a seamless developer experience, the frontend architecture is built using the React library combined with the Tailwind CSS framework. Furthermore, the Bun runtime environment is utilized as the primary toolchain, significantly accelerating module resolution, bundling, and testing processes.

У сучасній ветеринарній практиці та повсякденному догляді за домашніми тваринами дедалі більшої ваги набуває безперервний моніторинг їхнього стану. Особливо критичним цей процес є в періоди реабілітації тварин, врятованих з вулиці, або під час лікування хронічних захворювань. Найменші відхилення у поведінкових патернах, зміна харчових звичок чи рівня активності можуть свідчити про необхідність медичного втручання.

Отже, актуальною задачею є розробка спеціалізованих інформаційних систем, здатних агрегувати ці дані. Створення таких систем дозволяє перейти від реактивної моделі («лікуємо, коли вже стало погано») до проактивного моніторингу, що критично для тварин, які не можуть поскаржитися на біль.

Метою даної роботи є дослідження та обґрунтування вибору технологічного стеку для проєктування клієнтської частини (frontend) вебдодатка, орієнтованого на постійний моніторинг стану здоров'я домашніх тварин.

Специфіка предметної області вимагає від інтерфейсу користувача (UI) високої інтерактивності та здатності працювати з динамічними даними в режимі реального часу з можливістю швидкої реакції на небезпечні стани. Для реалізації цих вимог було обрано бібліотеку React [1]. Її компонентний підхід дозволяє ізолювати логіку окремих модулів системи: наприклад, модуль відстеження ваги, модуль

графіка прийому медикаментів та модуль фіксації поведінкових змін функціонують незалежно, що спрощує подальше масштабування системи.

Рівень представлення за принципом атомарного дизайну. Базові елементи (кнопки, поля вводу, картки) інкапсулюються у перевикористовувані компоненти. Особлива увага приділяється модулю візуалізації даних – інтерактивним графікам зміни ваги та температури. Для візуального оформлення та забезпечення адаптивності інтерфейсу доцільним є використання фреймворку Tailwind CSS [2]. Використання Tailwind у цьому контексті дозволяє легко адаптувати графічні елементи під різні розміри екранів, реалізуючи підхід Mobile-First. Це є критичною вимогою, адже власники тварин найчастіше взаємодіють із системою через смартфони під час прогулянок або візитів до ветеринара.

Розроблена архітектура передбачає поділ клієнтського додатка на рівні: рівень представлення (UI компоненти), рівень управління станом (state management) та рівень взаємодії з API серверної частини [3]. Рівень управління станом відіграє критичну роль у забезпеченні консистентності даних. Оскільки додаток оперує медичними показниками, що постійно оновлюються, доцільним є розділення глобального та асинхронного станів. Глобальний стан (дані профілю користувача, налаштування теми) керується за допомогою React Context API, що зменшує надмірне передавання пропсів (prop-drilling). Для управління асинхронним станом (взаємодія з сервером) архітектура передбачає використання спеціалізованих бібліотек. Це дозволяє реалізувати механізми кешування, фонового оновлення даних та оптимістичного оновлення інтерфейсу (optimistic UI), що робить додаток стійким до короткочасних втрат інтернет-з'єднання.

Ключовим інфраструктурним рішенням у процесі розробки є відмова від традиційних менеджерів пакетів на користь середовища виконання Bun [4]. Використання Bun як універсального інструментарію (bundler, test runner та package manager) дозволяє кардинально оптимізувати процес розробки. Завдяки реалізації на мові Zig та глибокій оптимізації роботи з файловою системою, Bun забезпечує миттєве встановлення залежностей та швидко гарячу підміну модулів (HMR). Додатковою перевагою обраного середовища Bun є вбудована підтримка швидкого тестування (bun test). Для систем моніторингу здоров'я надійність коду є пріоритетом. Bun дозволяє без додаткового налаштування стороннього інструментарію виконувати модульні тести для React-компонентів та функцій розрахунку статистичних медичних показників із високою швидкістю. Це значно полегшує інтеграцію клієнтського додатка в сучасні конвеєри безперервної інтеграції та розгортання (CI/CD). Загальну схему розробленої архітектури та взаємодію її рівнів наведено на рисунку 1.



Рисунок 1 – Блок-схема архітектури клієнтської частини додатку

Запропонований технологічний стек (React + Tailwind CSS + Bun) забезпечує створення надійного, швидкого та зручного інструменту для власників тварин. Застосування компонентної архітектури дозволяє в майбутньому легко інтегрувати нові функціональні модулі, наприклад, аналітику на базі алгоритмів машинного навчання для прогнозування можливих загострень захворювань.

#### ***Список використаних джерел:***

1. React: The library for web and native user interfaces // Meta Platforms, Inc. URL: <https://react.dev/> (дата звернення: 06.03.2026).
2. Tailwind CSS: Rapidly build modern websites without ever leaving your HTML // Tailwind Labs. URL: <https://tailwindcss.com/> (дата звернення: 06.03.2026).
3. Боков О. Г. Аналіз архітектури сучасних ВЕБ-додатків // Automation and Development of Electronic Devices (ADED'2022) : collection of Students' Scientific Paper. Kharkiv : Kind of Kharkiv National University of Radio Electronics, 2022. Part 2. P. 127–129.
4. Bun is a fast JavaScript all-in-one toolkit // Oven. URL: <https://bun.sh/> (дата звернення: 03.03.2026).