УДК 004.4
DOI: https://doi.org/ 10.17721/3041-2323.2024.304-311

Антон РЕТЮНСЬКИХ, студ.
ORCID ID: 0009-0004-8449-3718
e-mail: arretiunskykh@gmail.com
Київський національний університет
імені Тараса Шевченка, Київ, Україна

Юлія ЖИХАРЄВА, канд. фіз.-мат. наук, асист.
ORCID ID: 0009-0008-6876-9605
e-mail: y.zhykharieva@knu.ua
Київський національний університет
імені Тараса Шевченка, Київ, Україна

МОБІЛЬНИЙ ЗАСТОСУНОК ДЛЯ ПОШУКУ ДОМАШНІХ ТВАРИН

Роботу присвячено спрощенню й оптимізації процесу пошуку домашніх тварин за допомогою мобільного застосунку. Досліджено ринок застосунків для пошуку домашніх тварин та основні потреби, які задовольняють застосунки такого типу. Здійснено аналіз можливих технологій для створення застосунку. Спроєктовано архітектуру, створено кросплатформний застосунок для пошуку домашніх тварин з урахуванням вимог і сучасних підходів.

Ключові слова: ***Dart, Flutter, Firebase, мобільний застосунок, пошук домашніх тварин.***

Вступ

Поточна ситуація в країні, на жаль, диктує невтішні сценарії виникнення ситуацій, за яких тварини, що мали домівки, втрачають їх. В умовах технологічного прогресу актуальним є питання підтримувати процес пошуку й усиновлення тварин за допомогою інформаційних технологій. З огляду на те, що мобільні пристрої є невід'ємною частиною повсякденного життя, доцільним є створення спеціалізованого мобільного застосунку для пошуку домашніх тварин. Такий застосунок не лише оптимізує процес пошуку для потенційних власників, але й може відіграти вирішальну роль у збереженні життя тварин, для яких у притулках не вистачає ресурсів.

Подібні мобільні застосунки для пошуку домашніх тварин настільки непопулярні, що важко знайти необхідний застосунок, не знаючи про нього. Тому створення застосунку для пошуку тварин, спростить процеси пошуку домашніх тварин для людей, що цього прагнуть, і також позитивно вплине на кількість тварин, що не мають власників.

Результати

У сучасних технологіях є багато архітектурних рішень для вибору програмних засобів для реалізації кросплатформних застосунків. У розробленні зазначеного застосунку проаналізовано кілька з них.

React Native – це стек технологій для кросплатформної розробки iOS і Android, створений командою інженерів Facebook, які раніше розробляли ReactJS. Він поєднує нативну розробку з розробкою інтерфейсу JS. Серед переваг React Native можна виділити можливість повторного використання компонентів, що економить багато часу. Віртуальний DOM покращує взаємодію з користувачем. DOM – це структура документа в XML у формі дерева. Основна проблема традиційного DOM полягає в тому, що навіть для невеликих змін потрібно оновити все дерево, що не є ергономічно ефективним. Серед недоліків можна виокремити застарілі бібліотеки, проблеми сумісності, зокрема і системні збої, з якими розробник може зіткнутися під час роботи.

Ionic – це фреймворк для розроблення кросплатформних мобільних застосунків, що базується на вебтехнологіях, таких як HTML, CSS та JavaScript. Він використовується для створення мобільних застосунків, які можуть працювати як на платформі Android, так і на iOS. Основні переваги Ionic включають простоту використання та широкі можливості для розроблення кросплатформних застосунків і надає широкий набір компонентів і інструментів, що допомагають у розробленні застосунків із красивим інтерфейсом користувача. Ще однією перевагою Ionic є його активна спільнота розробників, що дозволяє швидко знаходити рішення на форумах і знаходити корисні бібліотеки. Серед мінусів можна виділити обмежені можливості доступу до апаратного забезпечення пристрою порівняно з іншими фреймворками, що може призвести до того, що деякі складні функції, такі як робота з

датчиками або геолокація, можуть бути менш ефективними або недоступними. Також Ionic-застосунки використовують вебзапити для отримання даних, які можуть бути чутливішими до якості та швидкості мережі, що призводить до повільної або нестабільної роботи застосунку в умовах обмеженого інтернет-з'єднання.

Xamarin – це платформа з відкритим вихідним кодом, призначена для побудови сучасних продуктивних програм для iOS, Android та Windows з .NET. Зазначену платформу вважають одним із найкращих фреймворків для служб розроблення гібридних програм. Серед переваг можна виділити те, що Xamarin пропонує високу можливість повторного використання коду – до 90 %, а також високу можливість налаштування. Серед мінусів виділяють затримки в оновленнях і можливу несумісність з останніми версіями iOS і Android, незважаючи на досить велику команду розробників, а це є досить суттєвим.

Flutter – це відкритий фреймворк розроблення користувацького інтерфейсу (UI) від компанії Google. Він дозволяє створювати красиві та високоефективні кросплатформні застосунки для мобільних, веб- і настільних платформ із використанням однієї кодової бази (Armstrong, n. d.). Плюси Flutter включають однакову візуалізацію на всіх платформах. На відміну від React Native або Xamarin, які мають властивості, підтримувані лише деякими платформами, Flutter відтворює програму однаково незалежно від того, чи вона запущена на Android чи iOS (A Searchable List of Flutter Resources, n. d.). Функція Hot Reload дозволяє розробникам не чекати довго на перезавантаження програми, як у React Native і Xamarin Forms, але з тією різницею, що більшість змін інтерфейсу користувача застосовуються в режимі реального часу під час роботи над кодом (Flutter documentation, n. d.). Вбудована підтримка тем полегшує створення та перегляд інтерфейсу користувача. Створення тем у Flutter набагато простіше порівняно з нативним розробленням для Android, завдяки чому розробники можуть швидше та легше розробляти всі аспекти інтерфейсу користувача. Серед недоліків можна зазначити використання власної мови програмування Dart, розробленої Google (Flutter, & Dart DevTools, n. d.).

Технології для оптимізації пошуку у мобільному застосунку. Для швидкої побудови інтерфейсу обрано вебсервіс Figma, що дозволяє працювати з графічними зображення та спеціалізується на створенні інтерфейсів. Figma має зручний інтерфейс, зображений на рис. 1, що покращує досвід користувачів.

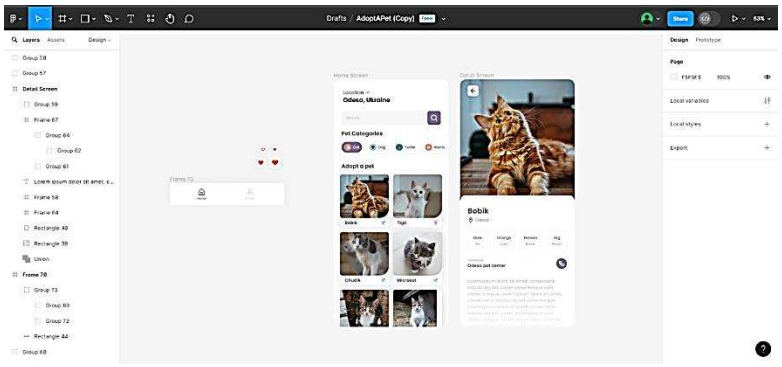

Рис. 1. *Інтерфейс сервісу Figma*

Застосунок складається з 5 основних екранів і нижньої панелі навігації. Дизайн розроблявся відповідно до сучасних трендів, а також рекомендацій від Apple для побудови інтерфейсів.

Під час розроблення архітектури було важливо дотримуватись основних принципів SOLID, що допомагає забезпечити гнучкість, розширюваність і підтримку коду.

Для реалізації застосунку для пошуку було обрано чисту архітектуру, оскільки вона підтримує тестування і забезпечує гнучкість у ході розвитку проєкту, чого не можна сказати про інші архітектури. Кодова база була поділена на чотири частини: домен, дані, презентацію та інтерфейс користувача. Домен взаємодіє із сервером і містить бізнес-логіку, дані відповідають за типи та поведінку моделей усередині застосунку, презентація керує взаємодією з моделями верхнього рівня, а інтерфейс користувача відображає моделі, отримані від презентації.

Для дотримання принципів SOLID, кожен рівень системи знає лише про абстракцію вищого рівня, але не про його конкретну реалізацію, що забезпечує незалежність компонентів і спрощує їхнє тестування. Для взаємодії з API використано сервіси й об'єк-

ти Data Transfer Object (DTO), які отримано з мережі. Для кожного сервісу створено абстрактний клас, який викликається на нижчому рівні, що дозволяє легко змінювати джерела даних або бібліотеки запитів у майбутньому.

На рівні презентації важливо обрати бібліотеку, що керуватиме станом усього застосунку. Для проєкту великого масштабу документація Flutter радить використовувати Riverpod або BloC. Загалом, обидві бібліотеки є чудовим вибором для проєкту та надають велику кількість інструментів, але кращою практикою для чистої архітектури є використання BloC, який розроблявся саме для цього. BloC працює за алгоритмом на рис. 2 (Flutter Reddit, n. d.).

Рис. 2. *Схема, що зображує принцип роботи BloC*

Як бачимо з рис. 2, бібліотека покращує зв'язок між рівнем даних і рівнем інтерфейсу.

Для створення застосунку було використано одночасно два API: власне та загальнодоступне. Для доступу до всесвітньої бази тварин, які потребують господарів, обрано Petfinder API. Вибір базовано на загальнодоступності системи та її популярності. За допомогою цього API можна додавати тварин або знаходити їх у будь-якому куточку світу. Для використання власного API база даних тварин була б досить невеликою на початку, що могло б негативно позначитися на ефективності пошуку.

Для авторизації використано інструмент Authentication, який дозволяє входити за допомогою електронної пошти та пароля, а також реєструватися через пошту Google. Після реєстрації кожен користувач отримує унікальний ідентифікатор у базі даних Firebase, за яким подальші дані про тварини будуть зберігатися (Google Assistant, n. d). Цей ідентифікатор записують у локальне сховище SharedPreference. Під час кожного входу до застосунку виконується перевірка цього ідентифікатора, і користувач автоматично переходить на потрібну сторінку залежно від результату цієї перевірки.

Для збереження даних використано такі методи як final users = FirebaseFirestore.instance.collection('users'); final userDoc = await users.doc(uid).get().

Загалом, база даних має досить просту структуру, адже фіксує лише збережені тварини кожного користувача, як вказано на рис. 3 (flutter/dev/docs at master·flutter/flutter, n. d.).

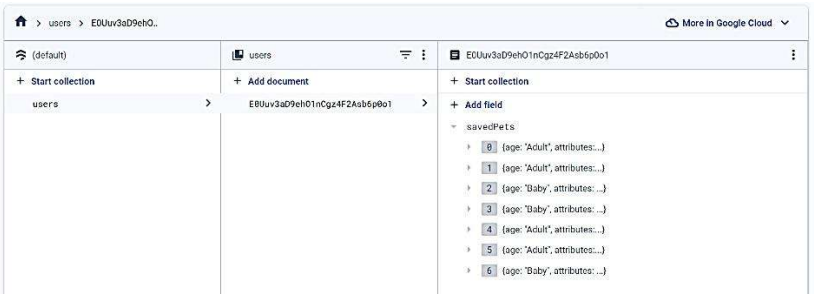

Рис. 3. *Структура бази даних*

Дискусія і висновки

У ході дослідження для створення кросплатформного застосунку застосовано фреймворк Flutter із його можливостями та деревом віджетів. Для створення інтерфейсу використано засоби Figma, за допомогою яких було розроблено макет із деталями, що легко конвертується у код. Для побудови інтерфейсу системами дизайну Material Design і Cupertino зручно керуватися для створення застосунку. Дотримання принципів SOLID та ООП для написання коду, дозволяє легко розвивати та підтримувати застосунок. Для серверної частини було обрано відкрите API та Firebase, за допомогою якого реалізується авторизація та збереження улюблених тварин.

Впровадження сучасних вебрішень для оптимізації бізнес-процесів стає визначальним елементом успішної стратегії розвитку компаній. Використання прогресивних вебзастосунків (PWA), хмарових сервісів, штучного інтелекту у вебзастосунках суттєво впливає на швидкість і якість адаптації бізнесу до змін ринку. Розумний вибір фреймворків, інтеграція API для роботи з зовнішніми сервісами, сучасні технології для оброблення запитів у режимі реального часу дозволяють створювати неймовірно гнучкі рішення для оптимізації найрізноманітніших бізнес-процесів.

У довгостроковій перспективі впровадження таких рішень дозволяє компаніям залишатися конкурентоспроможними та готовими до нових викликів, закладаючи основу для стабільного розвитку в умовах швидких технологічних змін.

Список використаних джерел

A Searchable List of Flutter Resources. (n. d.). *A Searchable List of Flutter Resources.* https://flutterx.com/ (Accessed: 11.09.2024).

Armstrong, B. (n. d.). *Flutter Institute.* https://flutter.institute/ (Accessed: 11.09.2024).

Flutter and Dart DevTools. (n. d.). *Flutter - Build apps for any screen.* https://flutter.dev/docs/development/tools/devtools (Accessed: 11.09.2024).

Flutter documentation. (n. d.). *Docs | Flutter.* https://docs.flutter.dev/ (Accessed: 12.09.2024).

Flutter Reddit. (n. d.). *Flutter Reddit.* https://www.reddit.com/r/FlutterDev/ (Accessed: 13.09.2024).

FlutterFire. (n. d.). *Firebase Flutter.* https://firebase.flutter.dev/ (Accessed: 13.09.2024).

Google Assistant. (n. d.). *Google Assistant, your own personal Google.* https://assistant.google.com (Accessed: 11.09.2024).

flutter/dev/docs at master·flutter/flutter. (n. d.). *GitHub.* https://github.com/flutter/-flutter/tree/master/dev/docs (Accessed: 12.09.2024).

References

A Searchable List of Flutter Resources. (n. d.). *A Searchable List of Flutter Resources.* https://flutterx.com/ (Accessed: 11.09.2024).

Armstrong, B. (n. d.). *Flutter Institute.* https://flutter.institute/ (Accessed: 11.09.2024).

Flutter and Dart DevTools. (n. d.). *Flutter - Build apps for any screen.* https://flutter.dev/docs/development/tools/devtools (Accessed: 11.09.2024).

Flutter documentation. (n. d.). *Docs | Flutter.* https://docs.flutter.dev/ (Accessed: 12.09.2024).

Flutter Reddit. (n. d.). *Flutter Reddit.* https://www.reddit.com/r/FlutterDev/ (Accessed: 13.09.2024).

FlutterFire. (n. d.). *Firebase Flutter.* https://firebase.flutter.dev/ (Accessed: 13.09.2024).

Google Assistant. (n. d.). *Google Assistant, your own personal Google.* https://assistant.google.com (Accessed: 11.09.2024).

flutter/dev/docs at master·flutter/flutter. (n. d.). *GitHub.* https://github.com/flutter/-flutter/tree/master/dev/docs (Accessed: 12.09.2024

Отримано редакцією журналу / Received: 17.09.24
Прорецензовано / Revised: 27.09.24
Схвалено до друку / Accepted: 01.10.24

Anton RETIUNSKYKH, Student
ORCID ID: 0009-0004-8449-3718
e-mail: arretiunskykh@gmail.com
Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

Yuliia ZHYKHARIEVA, PhD (Phys. & Math.), Assist.
ORCID ID: 0009-0008-6876-9605
e-mail: y.zhykharieva@knu.ua
Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

MOBILE APPLICATION FOR SEARCHING FOR PETS

The work is dedicated to simplifying and optimizing the process of finding pets through the development of a mobile application. The market of pet search applications and the main needs addressed by such applications have been researched. An analysis of potential technologies for creating the application was conducted. The architecture was designed, and a cross-platform application for pet search was developed, considering the requirements and modern approaches.

Keywords: ***Dart, Flutter, Firebase, mobile app, pet search.***

Автори заявляють про відсутність конфлікту інтересів. Спонсори не брали участі в розробленні дослідження; у зборі, аналізі чи інтерпретації даних; у написанні рукопису; в рішенні про публікацію результатів.

The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.