УДК 004.41 **DOI: 10.33042/2522-1809-2024-3-184-2-9**

І.В. Соловей, О.Г. Ворочек

Харківський національний університет радіоелектроніки, Україна

РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ ПОШУКУ ЗАГУБЛЕНИХ РЕЧЕЙ

Стаття описує розробку програмної системи пошуку загублених речей, яка складається із серверної, веб- та мобільної частин. У роботі досліджуються сучасні технології розробки та аргументується їх вибір, аналізується актуальність проблеми, наводяться сценарії використання проєкту у реальному світі.

Ключові слова: *програмна система, пошук речей, місцезнаходження, бюро знахідок.*

Постановка проблеми

У сучасному світі, де кожен з нас постійно в русі та зайнятий безліччю справ, дуже легко втратити особисті речі: від ключів та гаманців до електронних пристроїв. Знайти втрачену річ у великому місті може бути складно, а зворотний зв'язок з власником часто є майже неможливим без відповідної системи ідентифікації. Системи, які існують на ринку, зазвичай зосереджені на захисті речей до їх втрати, а не після. Це створює унікальну нішу для нових технологічних рішень, здатних забезпечити ефективність та безпеку процесу повернення втрачених речей.

Проблема втрати особистих речей особливо актуальна у великих містах, де шанси знайти загублену річ без зовнішньої допомоги надзвичайно малі. Пошуки втрачених предметів можуть виявитися часомісткими та малоефективними, особливо коли мова йде про громадські місця, як-от ресторани, кінотеатри або громадський транспорт. Оптимізація цього процесу за допомогою програмної системи могла б значно підвищити ефективність як звичайних перехожих, так і міських і комунальних служб у їх роботі зі знахідками.

Втрата важливих речей є загальною проблемою для багатьох людей. За даними від Chipolo, найчастіше губляться такі предмети, як мобільні телефони, ключі, гаманці, окуляри. Статистика вказує, що такі втрати відбуваються особливо часто під час вихідних, коли люди більш схильні до розсіяності [1].

Наявні рішення на ринку здебільшого зосереджені на превентивних заходах, наприклад, електронних брелоках або мобільних додатках, які сповіщають користувача про втрату речі в момент її відсутності. Проте ці заходи часто виявляються неефективними тоді, коли річ вже втрачена, а людина не встигла подбати про її безпеку заздалегідь. Зазвичай людина не готується до гірших сценаріїв та не припускає, що вона може десь загубити свою власність. Програмна система допомогла б вирішити цю проблему, надаючи засоби для миттєвого відновлення зв'язку з власником втраченої речі.

З іншого боку, процес повернення знайдених речей також має багато недоліків. Часто особа, яка знаходить загублену річ, не має засобів зв'язатися із власником. Навіть якщо на речі є якась інформація, не кожен готовий витратити свій час на пошук людини. Тому існує велика потреба в системі, яка б могла ефективно і безпечно з'єднувати людину, що знайшла річ, із її власником.

Це створює необхідність створення програмного рішення, яке б використовувало сучасні технології для ідентифікації та повернення втрачених речей. Така система могла б не тільки значно знизити витрати часу і зусиль на пошуки втрачених речей, але й зробити цей процес безпечнішим.

У сучасних мегаполісах існує значна потреба в ефективних системах пошуку втрачених речей не тільки для приватних осіб, але й для публічних організацій, комунальних підприємств, а також місць масового збору людей: ресторанів, кінотеатрів та бюро знахідок. Загублені особисті речі в таких місцях створюють додатковий тягар для адміністрації, яка має витрачати ресурси на зберігання та ідентифікацію власників. Наприклад, у США щороку втрачається та знаходиться понад 400 мільйонів предметів, причому більше 40 % втрат відбувається в громадських місцях, зокрема ресторанах, аеропортах та громадському транспорті. Оціночна вартість втрачених речей у США становить понад 5 мільярдів доларів на рік [2]. Ці дані підкреслюють значний обсяг втрат, який виникає у громадських місцях, та важливість розробки систем для інтеграції з ними.

Впровадження такої системи у міську інфраструктуру могло б об'єднати усе місто навколо єдиної платформи для пошуку втрачених речей, створюючи надійний інструмент, який би служив на благо всіх жителів та гостей міста.

Аналіз останніх досліджень і публікацій

Наукові дослідження розглядають технології для слідкування за втраченими речами, підкреслюючи,

як сучасні технологічні рішення можуть допомогти вирішити проблеми, пов'язані з втратою важливих предметів. Одне з досліджень, опубліковане в IOSR Journal of Business and Management [3], зосереджується на використанні смартфонів та інших пристроїв слідкування, які можуть значно знизити стрес та час, пов'язаний з пошуком втрачених речей, полегшуючи процес повернення зниклих предметів. Інше дослідження [4] представляє систему для слідкування за об'єктами в реальному часі з використанням технологій Інтернету речей, включно з GPS та Arduino Uno, що дозволяє ефективно відстежувати місцезнаходження втрачених предметів. Ці дослідження демонструють важливість використання інтегрованих технологічних рішень для створення надійних систем пошуку.

Для підвищення ефективності системи пошуку загублених речей необхідне широке охоплення, особливо серед користувачів мобільних пристроїв. За даними Exploding Topics [5], на кінець 2023 року, користувачі Android складали приблизно 28,99 % ринку смартфонів, тоді як користувачі iOS становили близько 70,29 %. З огляду на те, що переважна більшість населення використовує смартфони, розробка мобільного додатку є критично важливою для досягнення максимального охоплення.

Для розробки мобільного застосунку можна використовувати різні технології. Основними серед них є:

– Flutter. Використовується для розробки кросплатформних мобільних додатків. Код написаний на Dart та дозволяє компілювати його у нативний як для Android, так і для iOS, забезпечуючи високу продуктивність і гнучкість у підборі елементів дизайну [6];

– React Native. Ще одна популярна кросплатформна технологія, яка дозволяє розробляти мобільні додатки, використовуючи JavaScript та React. Вона забезпечує високий рівень повторного використання коду між платформами та доступ до широкого спектру JavaScript-бібліотек;

– Xamarin. Використовує C# для створення додатків для Android та iOS. Xamarin дозволяє розробникам використовувати велику частину коду для обох платформ, спрощуючи процес розробки та підтримки;

– Swift. Це основна мова програмування для розробки додатків iOS. Swift розроблений Apple для заміни Objective-C і забезпечує більшу безпеку типів, зручність і швидкість розробки;

– Kotlin. Це мова програмування від JetBrains, яка стала офіційною мовою для розробки Android-додатків. Kotlin вирізняється лаконічністю, безпекою та повною сумісністю з Java, що робить його відмінним вибором для розробки Android.

Для ефективної взаємодії між мобільними та веб-частинами системи необхідна потужна серверна складова. Вибір серверних рішень є ключовим для забезпечення надійності, безпеки та швидкості обробки даних. Ось декілька популярних технологій, які можуть бути використані для розробки серверної частини:

– Django. Потужний фреймворк для розробки веб-додатків на Python, який надає великий набір інструментів для швидкої розробки безпечних і обслуговуваних веб-сайтів;

– Node.js. Забезпечує високу продуктивність завдяки асинхронній обробці подій, що добре підходить для реалізації складних веб-додатків та API;

– Ruby on Rails. Фреймворк, який використовує принцип "Convention over Configuration", дозволяє розробникам швидко будувати додатки за допомогою великої кількості готових бібліотек і плагінів;

– ASP.NET Core. Потужний і гнучкий фреймворк від Microsoft для створення високопродуктивних веб-додатків та сервісів;

– Spring Boot. Дає змогу легко створювати стійкі, масштабовані веб-додатки і мікросервіси на базі Java.

Мета статті

Метою статті є представлення підходу до розробки програмної системи пошуку загублених речей, включно з детальним описом функціоналу, описом етапів розробки усіх компонентів – серверної, веб- та мобільної частини, їх взаємодії разом, пошуком необхідних програмних рішень.

Виклад основного матеріалу

Розробка програмної системи для пошуку втрачених речей в межах запропонованого підходу передбачає створення як веб-, так і мобільної версії. Має бути реалізований наступний функціонал:

– реєстрація нових користувачів. Ця функція дозволяє користувачам створювати особисті акаунти, які є необхідними для використання системи;

– реєстрація юридичних осіб / організацій. Можливість реєстрації для громадських місць, поліції, бюро знахідок тощо, що дозволяє їм використовувати систему для управління знахідками та загубленими речами;

– додавання речей без RFID-сканування. У веб-версії це можливо зробити вручну, вводячи ID, тоді як мобільна версія підтримує сканування;

– вибір типу речі та додавання фотографій. Користувачі можуть класифікувати додані речі та завантажувати зображення для кращої ідентифікації;

– перегляд доданих речей. Користувачі можуть переглядати всі речі, які вони зареєстрували в системі;

– маркування речі як втраченої. Користувачі можуть позначати речі, які вони втратили, щоб інші користувачі могли їх виявити;

– перегляд втрачених речей. Можливість переглядати всі втрачені речі в системі;

– відмітка знайденої речі. Знахідники можуть сканувати RFID або вручну вводити ID для маркування речі як знайденої;

– зазначення контактів та обмеження їх відображення. Користувачі можуть специфікувати, які контакти відображати залежно від типу речі;

– зв'язок зі знахідником. Система дає змогу знахіднику зв'язатися з власником;

– комунікація через популярні месенджери. Інтеграція з відомими месенджерами для спрощення комунікації.

Для функціонування веб- та мобільного застосунку потрібно надати їм доступ до даних. Одним із найбільш популярних та надійних способів реалізації цього є використання REST (Representational State Transfer) API [7], який дозволяє легко інтегрувати різні платформи та пристрої в єдину систему. REST API використовує стандартні HTTP-запити для отримання, зміни, видалення або додавання даних в базу даних, що робить його ідеальним для мобільних застосунків та веб-сайтів.

Для реалізації буде використовуватись Django [8] – сучасний веб-фреймворк для Python, який є чудовим вибором для розробки серверної частини, що використовує REST API. Для написання використовується мова програмування Python та програмне середовище PyCharm.

На рис. 1 наведена схема бази даних серверної частини програмної системи. Вона описує наступні таблиці: користувач застосунку (це може бути як звичайна людина, так і адміністратор закладу), предмет, який користувач може додати до системи, втрата (зареєстрована заявка про втрату) та, звичайно, таблиця з даними про контакти.

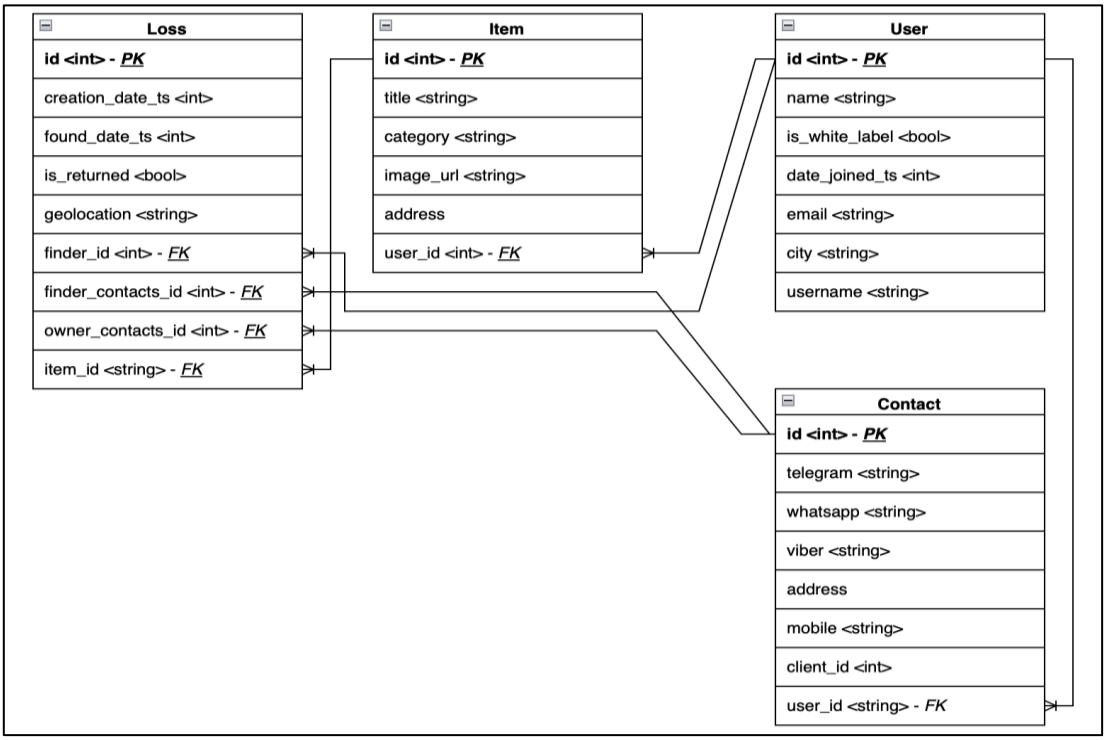

Рис. 1. Схема бази даних

Взаємодія з базою даних забезпечується кінцевими точками (або API-ендпоінтами). GET-запити, що відповідають за отримання даних, наступні:

– список користувачів. API-запит: '/api/v1/users/';

– інформація про користувача. API-запит: '/api/v1/user/{id}/';

– список усіх предметів користувача. API-запит: 'api/v1/user/items/{id}';

– список усіх предметів користувача, які зараз не є втраченими. API-запит: 'api/v1/user/items/{id}/available/';

– список загублених речей користувача. API-запит: '/api/v1/user/{id}/lost';

– список знайдених та повернутих речей користувача. API-запит: '/api/v1/users/{id}/found/returned/';

– список знайдених та не повернутих речей користувача. API-запит: '/api/v1/found/not_returned/';

– список предметів. API-запит: 'api/v1/items/';

– інформація про предмет. API-запит: '/api/v1/items/{id}';

– публічний список загублених речей. API-запит: '/api/v1/lost/;

– інформація про загублений предмет. API-запит: '/api/v1/lost/{id}';

– отримання часу, який залишився до повернення речі. API-запит: ‘/api/v1/lost/{id}/time’;

– отримання контактів користувача, який знайшов предмет. API-запит: ‘/api/v1/lost/{id}/finder/contact’;

– отримання контактів користувача, який загубив предмет. API-запит: ‘/api/v1/lost/{id}/owner/contact’.

POST-запити (відправка / редагування даних):

– реєстрація користувача. API-запит: ‘/api/v1/signup’;

– автентифікація. API-запит: ‘/api/v1/signin’;

– додати предмет. API-запит: ‘/api/v1/item/create’;

– додати загублений предмет. API-запит: ‘/api/v1/lost/create’;

– редагування користувача. API-запит: ‘/api/v1/user/{id}/’;

– редагувати предмет. API-запит: ‘/api/v1/item/{id}’;

– редагування загублених речей. API-запит: ‘/api/v1/lost/{id}’;

– редагування контактів користувача, який знайшов предмет. API-запит: ‘/api/v1/lost/{id}/finder/contact’;

– редагування контактів користувача, який загубив предмет. API-запит: ‘/api/v1/lost/{id}/owner/contact’.

DELETE-запити (видалення даних):

– видалення користувача. API-запит: ‘/api/v1/user/{id}/’;

– видалити предмет. API-запит: ‘/api/v1/item/{id}’;

– видалення загублених речей. API-запит: ‘/api/v1/lost/{id}’.

Ці ендпоінти формують основу API, яка забезпечує взаємодію між сервером та клієнтськими застосунками, дозволяючи користувачам реєструвати, відстежувати та управляти їх речами через веб- та мобільні інтерфейси.

Завдяки зчитуванню геолокації користувача під час заповнення анкети про втрату, користувачі, які знаходяться у радіусі ймовірного місця втрати, зможуть завантажити найближчі заявки. У базі даних MySQL 5.7 була представлена вбудована функція “ST_Distance_Sphere” для обчислення евклідової відстані між двома точками. Ця функція бере широту та довготу двох точок і повертає відстань між ними в метрах, після чого ми робимо порівняння з критичним значенням радіуса та визначаємо, чи потрапляє ця точка в коло з цим радіусом:

WHERE (ST_Distance_Sphere(point(longitude, latitude), point(longitude2, latitude2))) <= radius . (1)

Але тестування такого методу демонструє, що при значних обсягах рядків, які обробляються запитом, час обробки запиту помітно зростає.

Очевидно, що це зумовлює потребу у попередній фільтрації даних. Було вирішено фільтрувати за допомогою квадрата (рис. 2), в середину якого вписано коло критичного радіуса. Заявки про втрати, які потрапляють в квадрат, але не потрапляють в коло, будуть згодом фільтруватися за допомогою функції “ST_Distance_Sphere”, тобто левова частка фільтрації буде припадати на квадрат.

Рис. 2. Фільтрація координат за допомогою квадрата

Отже, для того, щоб зробити фільтрування за квадратом швидшим, ніж фільтрування за допомогою функції “ST_Distance_Sphere” потрібно:

1) обчислити координати вершин квадрата. Сторона квадрата дорівнює діаметру кола, тобто радіусу × 2;

2) обчислити координати квадрата. Відстань, яку ми шукаємо, вимірюється в кілометрах (радіус). Екваторіальна окружність Землі становить близько 40 075 км, коло – 360 градусів. Щоб повністю зрозуміти ці розрахунки, варто зазначити, що широта та довгота вимірюються в градусах:

– градус географічної широти становить 1/180 меридіана (широта – значення широти від −90 до 90);

– градус географічної довготи становить 1/360 екватора (довгота – значення довготи від −180 до 180).

```
const size = (Distance / 40075) * 360; // ступінь зсуву широти;
const radian = (latitude * Math.PI) / 180; // перетворення в радіан;
const longtitudeSize = size / Math.cos(radian); // перетворення в градуси зсуву довготи;
```
(2)

```
longitude − longtitudeSize // обчислення довготи Min (початкова довгота мінус зсув);
longitude + longtitudeSize // обчислення максимальної довготи (початкова довгота плюс зсув);
latitude − size // обчислення широти Min (початкова широта мінус зсув);
latitude + size // обчислення максимальної широти (оригінальна широта плюс зсув).
```
(3)

Таким чином, після обчислення координат вершин квадрата, в який вписано коло, можна застосувати проміжну фільтрацію разом з основною. Фільтрування за допомогою “WHERE between” швидше, ніж вбудоване “ST_Distance_Sphere”:

```
WHERE longitude between longitudeMin and longitudeMax
and latitude between latitudeMin and latitudeMax
and (ST_Distance_Sphere(point(longitude, latitude), point(longitude2, latitude2)) ) <= radius // radius in meters .
```
(4)

Для оптимізації додаємо індекс на стовпці «довгота» та «широта»:

```
CREATE INDEX coordinate ON table_name (latitude, longitude);
```
(5)

Результатом цієї оптимізації є успішна обробка 2 мільйонів записів, а час обробки зменшився у 3,46 раза.

Після створення серверної частини ми можемо перейти до веб-сайту. Для розробки інтерфейсу використаємо мову розмітки HTML зі стилями CSS, а для програмної частини – JavaScript. Оскільки сервіс може використовуватися з ПК старих поколінь у міських закладах, важливо зробити його максимально сумісним, саме тому було обрано перелічені технології. На рис. 3 наведено інтерфейс веб-частини.

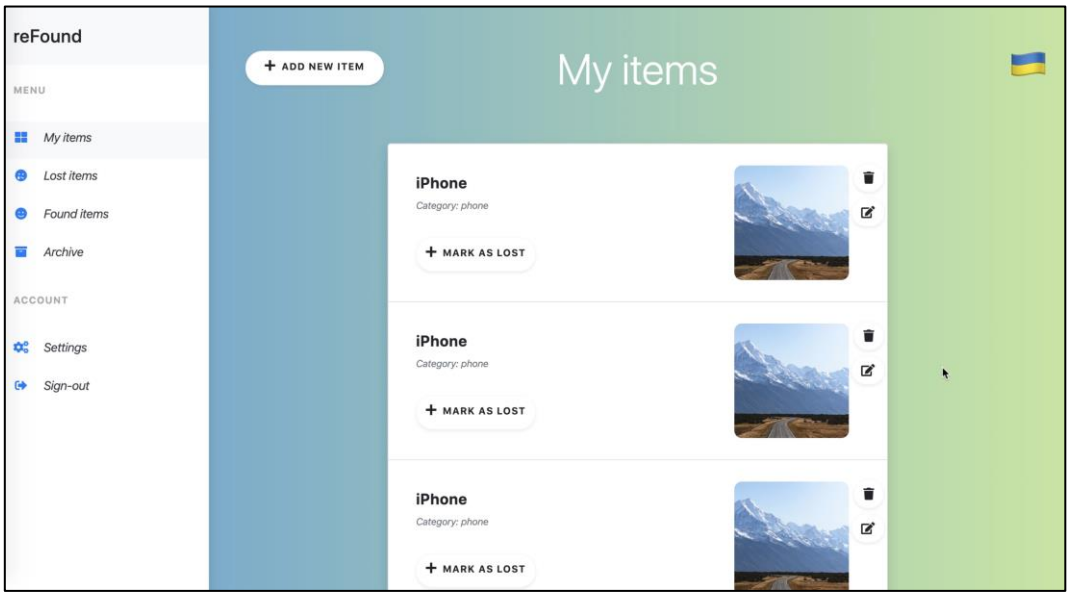

Рис. 3. Сторінка веб-частини «Мої речі»

На сайті доступні наступні розділи:

– мої речі. Відкрита сторінка, на ній відображаються речі, які є у користувача у наявності, тобто він їх не загубив, а захистив від втрати. Користувач може відмітити річ як загублену за допомогою відповідної кнопки;

– загублені речі. Відображає дошку оголошень із загубленими речами у найближчому до користувача радіусі. Логічно, що якщо річ була загублена у місті, розташованому за сотні кілометрів, то вона не буде відображена у списку, щоб підвищити вірогідність повернення речей, які були втрачені поруч. На цій сторінці будь-який користувач може відмітити річ як знайдену та ввести свої контактні дані;

– знайдені речі. Речі користувача, які знайшли інші люди, але передача ще не відбулась. На сторінці доступна кнопка, яка може завершити акт передачі речі, і тоді вона знову потрапить до списку «моїх

речей».

Як можна побачити із зображення, інтерфейс було інтернаціоналізовано, і він підтримує як англійську, так і українську мови.

Останнім етапом є розробка мобільного застосунку. Принцип роботи мобільного застосунку зазначено на рис. 4 за допомогою Use Case діаграми, або діаграми прецедентів, – графічного зображення, яке показує взаємодію користувачів (або «акторів») із системою, визначаючи різні способи, за допомогою яких користувачі можуть взаємодіяти з нею для досягнення конкретного цільового результату. Ця діаграма є частиною UML (Unified Modeling Language) [9], яка є стандартом для специфікації, візуалізації, розробки та документування об'єктно-орієнтованих програм.

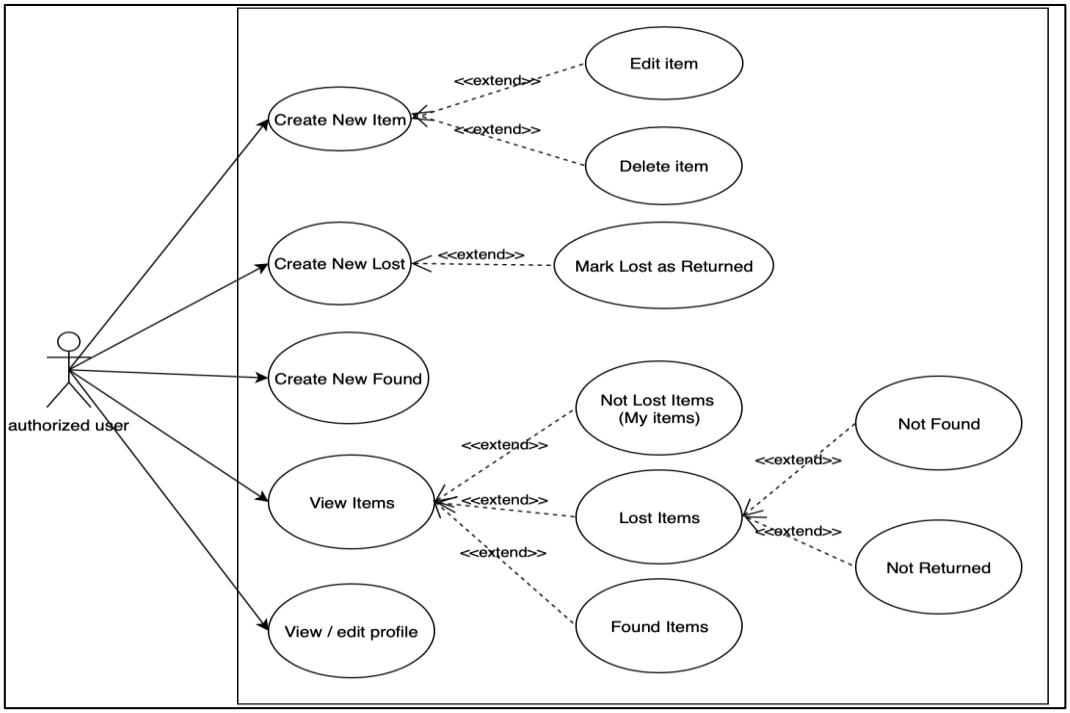

Рис. 4. Діаграма прецедентів

Як можна помітити, функціонал мобільного застосунку майже ідентичний із веб-частиною, але з одним виключенням – завдяки використанню технологій RFID та NFC сучасні смартфони здатні сканувати картки / мітки, прив'язані до системи, без необхідності підключення зовнішнього модуля. RFID є технологією, що дозволяє автоматично ідентифікувати об'єкти, тварин або людей за допомогою радіохвиль [10]. Технологія використовує так звані RFID-мітки або транспондери, які можна прикріпити до об'єкт або вбудувати в нього. Ці мітки містять антену, яка здатна приймати і передавати сигнал від зчитувального пристрою.

Розробка мобільних додатків передбачає використання Flutter – відкритого кросплатформного фреймворку, розробленого компанією Google, який дозволяє розробникам створювати нативні інтерфейси для мобільних додатків на iOS та Android з єдиної бази коду.

Однією з ключових особливостей Flutter є його здатність використовувати один і той же код для створення додатків, які не тільки виглядають і відчуваються як нативні на обох платформах, але й легко адаптуються до різних розмірів екранів і орієнтацій. На рис. 5 наведено підтвердження цієї тези: розроблений інтерфейс підтримує як маленький екран смартфона, так і великий екран планшета.

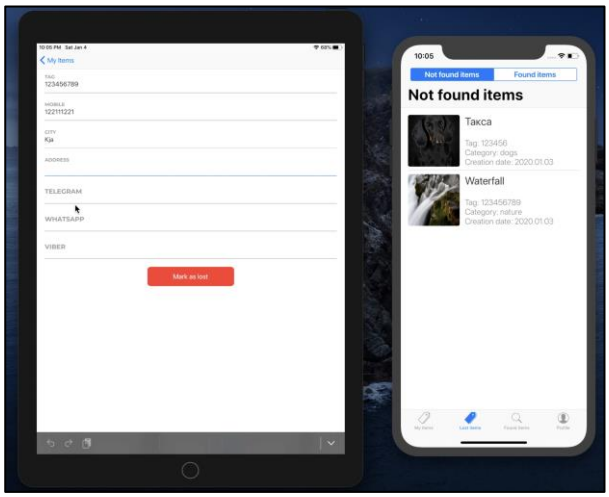

Рис. 5. Інтерфейс мобільного застосунку на пристроях різного розміру

Крім того, Flutter надає розробникам багатий набір віджетів, що дозволяє їм легко створювати персоналізовані користувацькі інтерфейси. Віджети в Flutter охоплюють широкий спектр елементів дизайну: від базових кнопок і перемикачів до складних анімацій та переходів. Ця універсальність і гнучкість роблять Flutter відмінним інструментом для розробки мобільних додатків, що потребують високої адаптивності до користувацьких пристроїв і платформ.

Висновки

У статті було представлено підхід до створення програмної системи пошуку загублених речей, розглянуто основні аспекти її реалізації. Було проаналізовано актуальність проблеми, досліджено відповідні джерела, наведено статистику. Програмна система складається із серверної частини, написаної на Python з використанням бібліотеки Django, вебсайту, написаного на JavaScript, HTML та CSS, та мобільного застосунку на Dart у парі з технологією Flutter. Система придатна для використання як звичайними користувачами, так і міськими організаціями. Завдяки доступності на різних платформах система дозволить максимізувати можливість повернення речей з її допомогою для будь-якої людини.

Література

1. *Lost And Missing Items: What We Lose The Most And Where We Lose It [Electronic resource] / Chipolo : website. – Trbovlje (Slovenia), 2013–2024. – Updated continuously. – Regime of access: https://chipolo.net/en/blogs/lost-and-missing-items-what-we-lose-the-most-and-where-we-lose-it, free (date of the application: 02.05.2024).*
2. *Lost and Found Statistics, Trends & Facts 2023 [Electronic resource] / Lostings : website. – New York (USA), 2017–2024. – Updated continuously. – Regime of access: https://www.lostings.com/lost-and-found-statistics/, free (date of the application: 02.05.2024).*
3. *How Does Matter Lost and Misplace Items Issue and Its Technological Solutions in 2015 – A Review Study / S. Ahmad, M. Ziaullah, L. Rauniyar, M. Su, Y. Zhang // IOSR Journal of Business and Management (IOSR-JBM). – 2015. – Vol. 17, Issue 4, Ver. I. – P. 79–84. – Regime of access: https://iosrjournals.org/iosr-jbm/papers/Vol17-issue4/Version-1/L017417984.pdf, free (date of the application: 02.05.2024).*
4. *Real-Time Tracking System for Object Tracking Using Internet of Things (IoT) / H. K. Sharma, T. Choudhury, A. Kandwal, A. Mor, P. Sharma, M. E. Ahmed, P. Ahlawat // Cyber Intelligence and Information Retrieval / ed. by J. M. R. S. Tavares, P. Dutta, S. Dutta, D. Samanta. – Singapore : Springer Nature, 2021. – P. 497–505. – (Lecture Notes in Networks and Systems (LNNS), Vol. 291). – DOI: 10.1007/978-981-16-4284-5_43.*
5. *iPhone vs Android User Stats (2024 Data) [Electronic resource] / Exploding Topics : website. – San Francisco, CA (USA), 2024. – Updated continuously. – Regime of access: https://explodingtopics.com/blog/iphone-android-users, free (date of the application: 02.05.2024).*
6. *Windmill E. Flutter in Action / E. Windmill. – Shelter Island, NY (USA) : Manning Publications, 2020. – 368 p. – Regime of access: https://www.manning.com/books/flutter-in-action, free (date of the application: 02.05.2024).*
7. *Biehl M. RESTful API Design / M. Biehl. – Scotts Valley, CA (USA) : CreateSpace Independent Publishing Platform, 2016. – 294 p. – (API-University Series, Vol. 3).*
8. *Forcier J. Python Web Development with Django / J. Forcier, P. Bissex, W. Chun. – Boston, MA (USA): Addison-Wesley Professional, 2008. – 408 p. – (Developer's Library).*
9. *Alhir S. S. Learning UML / S. S. Alhir. – Sebastopol, CA (USA) : O'Reilly Media, Inc., 2003. – 234 p. – Regime of access: https://www.oreilly.com/library/view/learning-uml/0596003447/, free (date of the application: 02.05.2024).*
10. *Hunt V. D. RFID – A Guide to Radio Frequency Identification / V. D. Hunt, A. Puglia, M. Puglia. – Hoboken, NJ (USA) : John Wiley & Sons, Inc., 2007. – 240 p. – Regime of access: https://onlinelibrary.wiley.com/doi/book/10.1002/0470112255, free (date of the application: 02.05.2024).*

References

1. Chipolo. (2020). *Lost And Missing Items: What We Lose The Most And Where We Lose It*. https://chipolo.net/en/blogs/lost-and-missing-items-what-we-lose-the-most-and-where-we-lose-it
2. Lostings. (2023). *Lost and Found Statistics, Trends & Facts 2023*. https://www.lostings.com/lost-and-found-statistics/
3. Ahmad, S., Ziaullah, M., Rauniyar, L., Su, M., & Zhang, Y. (2015). How Does Matter Lost and Misplace Items Issue and Its Technological Solutions in 2015 – A Review Study. *IOSR Journal of Business and Management (IOSR-JBM)*, *17*(4), 79–84. https://iosrjournals.org/iosr-jbm/papers/Vol17-issue4/Version-1/L017417984.pdf
4. Sharma, H. K., Choudhury, T., Kandwal, A., Mor, A., Sharma, P., Ahmed, M. E., & Ahlawat, P. (2021). Real-Time Tracking System for Object Tracking Using Internet of Things (IoT). In J. M. R. S. Tavares, P. Dutta, S. Dutta, & D. Samanta (Eds.), *Cyber Intelligence and Information Retrieval* (pp. 497–505). Springer Nature. https://doi.org/10.1007/978-981-16-4284-5_43
5. Howarth, J. (2024, June 14). *iPhone vs Android User Stats (2024 Data)*. Exploding Topics. https://explodingtopics.com/blog/iphone-android-users
6. Windmill, E. (2020). *Flutter in Action*. Manning Publications. https://www.manning.com/books/flutter-in-action
7. Biehl, M. (2016). *RESTful API Design*. CreateSpace Independent Publishing Platform.
8. Forcier, J., Bissex, P., & Chun, W. (2008). *Python Web Development with Django*. Addison-Wesley Professional.
9. Alhir, S. S. (2003). *Learning UML*. O'Reilly Media, Inc. https://www.oreilly.com/library/view/learning-uml/0596003447/
10. Hunt, V. D., Puglia, A., & Puglia, M. (2007). *RFID – A Guide to Radio Frequency Identification*. John Wiley & Sons, Inc. https://onlinelibrary.wiley.com/doi/book/10.1002/0470112255

Рецензент: д-р техн. наук, проф. В.В. Любченко, Національний університет «Одеська політехніка», Україна.

Автор: СОЛОВЕЙ Ілля Владиславович
здобувач вищої освіти факультету комп'ютерних наук
Харківський національний університет радіоелектроніки
E-mail – illia.solovei@nure.ua
ID ORCID: https://orcid.org/0009-0005-5715-2755

Автор: ВОРОЧЕК Ольга Григорівна
кандидат технічних наук, доцент кафедри програмної інженерії
Харківський національний університет радіоелектроніки
E-mail – olga.vorochek@nure.ua
ID ORCID: https://orcid.org/0000-0002-9054-9894

DEVELOPMENT OF A SOFTWARE SYSTEM FOR SEARCHING FOR LOST ITEMS

I. Solovei, O. Vorochek
Kharkiv National University of Radio Electronics, Ukraine

The article presents an approach to developing a software system to facilitate the efficient search for lost items. As part of the study, the authors designed an integrated system that includes server, web, and mobile versions, which provides broad opportunities for identifying and returning lost property.

In large urban agglomerations, the loss of personal items becomes a common problem. Existing solutions are often limited to preventive measures and do not provide sufficient support after the item's loss. The lack of an effective feedback and identification system complicates the lost property search process.

The developed system uses modern technologies such as Django for the backend, JavaScript for the web interface, and Flutter for creating cross-platform mobile applications. This approach makes it possible to expand the list of system users to the maximum extent. Django, a capable web framework, facilitates reliable data management and integration with various interfaces.

The study involved the development of a database to provide storage and processing of all the necessary information. Using REST API ensured efficient interaction between the system's server, web, and mobile parts. This approach allows different system components to exchange data in a unified format.

The system lets users register individuals and legal entities, which makes it possible to use it in a wide range of public spaces. It helps to effectively search for and return lost items on the street and in public places, including restaurants, theatres, and transport hubs where lost items often occur.

The software system created in this study addresses the critical problem of losing personal items. Widespread use of this system across various sectors could dramatically improve managing lost and found operations, turning a typically disorganised and ineffective process into an efficient, technology-driven one. The system's implementation has the potential to significantly reduce the time and resources spent on searching for lost property and increase the productivity of lost property offices.

Keywords: *software system, item search, location, lost property office.*