



Programowanie i bazy danych

„KitQfinder” – oprogramowanie dla schronisk dla zwierząt

Autor: Jacek Motyka

Gdańsk, 2020

Spis treści

1. Wstęp	3
2. Specyfikacja wymagań	6
3. Technologie i narzędzia.....	9
4. Prezentacja projektu	10
5. Weryfikacja rozwiązania	18
6. Podsumowanie i wnioski	19
7. Literatura	19

1. Wstęp

1.1. Rys historyczny

Problem bezdomności zwierząt w Polsce i na świecie nie jest nowy. Od wielu lat regulator, jak i podmioty prywatne, podejmują różne działania mające na celu ochronę życia i zdrowia zwierząt oraz zapewnienie im jak najlepszych warunków bytowych – zarówno w domach prywatnych jak i w instytucjach powołanych do ochrony zwierząt bezdomnych. Skutki tych działań bywają różne.

Ustawa z dnia 21 sierpnia 1997 r. o ochronie zwierząt nakłada na samorządy gminne szereg obowiązków związanych z zapewnieniem zwierzętom wolnobytujaćym opieki¹. Nie jest to zadanie łatwe, wiąże się też z szeregiem kosztów. W samym województwie Pomorskim koszt utrzymania schronisk w roku 2018 wyniósł 13.046.525 zł². Niestety, dobro zwierząt nie zawsze bywa priorytetem dla budżetu, jest też dla niego dużym obciążeniem. Średnia ilość bezdomnych psów jakimi zajęły się gminy (średnio na gminę) w 2018 r. to ok. 90.000. Średnia liczba psów wydanych z nadzorowanych schronisk (zwróconych właścicielom, adoptowanych lub sprzedanych) to natomiast 66.245³. Statystyka ta pokazuje, jak duża ilość zwierząt (statystyka uwzględnia również zwierzęta padłe) nie znajduje właściciela – często pomimo poszukiwań pupila przez jego właścicieli.

1.2. Cel projektu

Celem niniejszego projektu jest opracowanie oprogramowania umożliwiającego stworzenie ogólnopolskiej bazy danych zwierząt zagubionych lub znalezionych oraz umożliwienia łatwego i zautomatyzowanego przekazywania informacji o prawdopodobnym miejscu pobytu/właścicielu zagubionego zwierzęcia. Ma to na celu nie tylko pomóc zwierzętom wrócić do swoich domów, ale również odciążyć pracowników schronisk i wolontariuszy. Mniejsza ilość zwierząt w schroniskach spowodowana ich szybszym odbiorem przez opiekunów znacząco przełoży się też na koszty działalności schronisk. Pieniądze te będzie można zainwestować w poprawienie losu zwierzętom wciąż oczekującym na adopcję.

¹ Ustawa z dnia 21 sierpnia o ochronie zwierząt, Dz. U. 1997 Nr 111 poz. 724.

² <https://www.obrona-zwierzat.pl/aktualnosci/1715-bezdomnosc-zwierzat-w-polsce-w-2018-r.html>

³ <http://www.boz.org.pl/raport/2016/1.htm>

W perspektywie czasu możliwym jest poszerzenie dostępu do oprogramowania dla aktywistów prowadzących tymczasowe domy dla zwierząt oraz dodanie systemu do holistycznego zarządzania schroniskiem. Wielość funkcji w jednym miejscu, dostęp do aplikacji przez sieć web oraz łatwość obsługi może skutkować wzrostem świadomości społecznej problemu i realnym wpływem na dobrobyt zwierząt. Automatyczne zmiany w bazie danych schroniska w razie odebrania danego zwierzęcia usprawniłoby pracę opiekunów i wolontariuszy.

Na końcowym etapie możliwym byłoby umożliwienie dostępu do systemu każdemu chętnemu, zgłaszającemu obecność prawdopodobnie zagubionego zwierzęcia na wolności w danym obszarze. Stworzenie swoistej sieci społecznościowej łączącej ludzi wrażliwych na los zwierząt, zwracających uwagę na potencjalnie zagubione zwierzęta (np. wolnobytnące psy w obroży) umożliwiłoby zapobieganie trafiań zwierząt do schronisk, gdyż byłyby one znajdowane przez właścicieli nim znalazłyby je gminne służby. Dla wielu użytkowników informacja o podobnym zwierzęciu do zaginionego pupila widzianego w konkretnym miejscu i czasie może pomóc w samodzielnym odnalezieniu zagubionego zwierzęcia nim trafi do schroniska bądź stanie mu się krzywda.

1.3. Istniejące rozwiązania

Na rynku istnieją rozwiązania dostarczające oprogramowanie dla schronisk, jednak są one w większości komercyjne, nastawione na zysk oraz skupiają się na dostarczeniu rozwiązania do pojedynczego podmiotu. Z istniejących już rozwiązań na uwagę zasługują między innymi:

- System Informacji o Zwierzętach (PMCA Software) - oprogramowanie komputerowe dla schroniska składające się z modułów ewidencji zwierząt, zarządzania multimediami, leczenia zwierząt, bazy kontrahentów tj. osób oddających i pobierających zwierzęta ze schroniska, odpadów pochodzenia zwierzęcego, modułu zarządzania użytkownikami lokalnymi (przydzielanie praw dostępu do danych) oraz zawartością strony www⁴;
- Program Rejestracji (Geulincx) – darmowe oprogramowanie do ewidencji zwierząt wraz z dostępem do międzynarodowych baz danych⁵,
- Program Schronisko (Datum Software) – program kompleksowo wspomagający prowadzenie schronisk dla zwierząt, w tym między innymi umożliwiającą ewidencjonowanie zwierząt przyjętych i wydanych oraz przechowywanie aktualnego stanu schroniska, prowadzenia

⁴ <http://www.sioz.pl/AboutSIOZ>

⁵ <https://www.geulincx.pl/program-rejestracji>

rejestrów zgłoszeń zwierząt zaginionych i znalezionych z możliwością automatycznej analizy danych pod kątem zbieżności danych wprowadzanych w rejestrach zwierząt i zgłoszeń tworzenie różnego rodzaju zestawień dających⁶.

- Aplikacja mobilna „Na 4 łapy” (Volanto, Miasto Gdańsk) – jest to aplikacja mobilna dla Gdańskiego schroniska Promyk to rozwiązanie pozwalające na przeglądanie zdjęć i opisów zwierząt znajdujących się obecnie pod ich opieką. Użytkownik może w prosty sposób dodawać psy i koty do listy ulubionych oraz przekazywać datki na zwierzęta w formie elektronicznych mikropłatności⁷.
- Międzynarodowy program EUROPETNET oraz korzystająca z niego baza danych SAFE-ANIMAL – zgłaszanie zaginięcia oraz możliwość wyszukiwania znalezionej zwierzęcia poprzez nr wszczepionego chipu⁸. SAFE-ANIMAL jest międzynarodowym zbiorem danych zwierząt oraz ich właścicieli, bezpośrednio współpracuje z największymi polskimi miastami, w których realizowane są programy znakowania psów i kotów⁹.

To, co wyróżnia „KitQfinder” na tle konkurencji to przede wszystkim:

- umożliwienie szerokiego dostępu do bazy danych każdemu opiekunowi zatroskanemu o los swojego pupila darmowy dostęp dla każdego, *pro bono*;
- wyszukiwanie podobnych zwierząt bez względu na to, do którego schroniska trafiły. Brak ograniczenia do zwierząt „zachipowanych”, szacowanie podobieństwa w oparciu o dokładne dane wpisywane w formularz (w tym np. zaznaczanie miejsca o określonym kolorze na modelu zwierzęcia), zdjęcia i algorytmy uczenia maszynowego;
- budowanie sieci społecznościowej ludzi wrażliwych na los zwierząt poprzez brak ograniczenia zgłoszeń zwierząt znalezionych tylko do schronisk (domy tymczasowe, wolontariusze);
- umożliwienie samodzielnych poszukiwań zwierząt w oparciu o prawdopodobne miejsce pobytu zagubionego pupila udostępnione przez osobę, która widziała zwierzę, jednak nie udało się go złapać.

⁶ <http://www.datum.com.pl/schronisko.html>

⁷ <https://kodujdlapolski.pl/projects/aplikacja-dla-schroniska-promyk/>

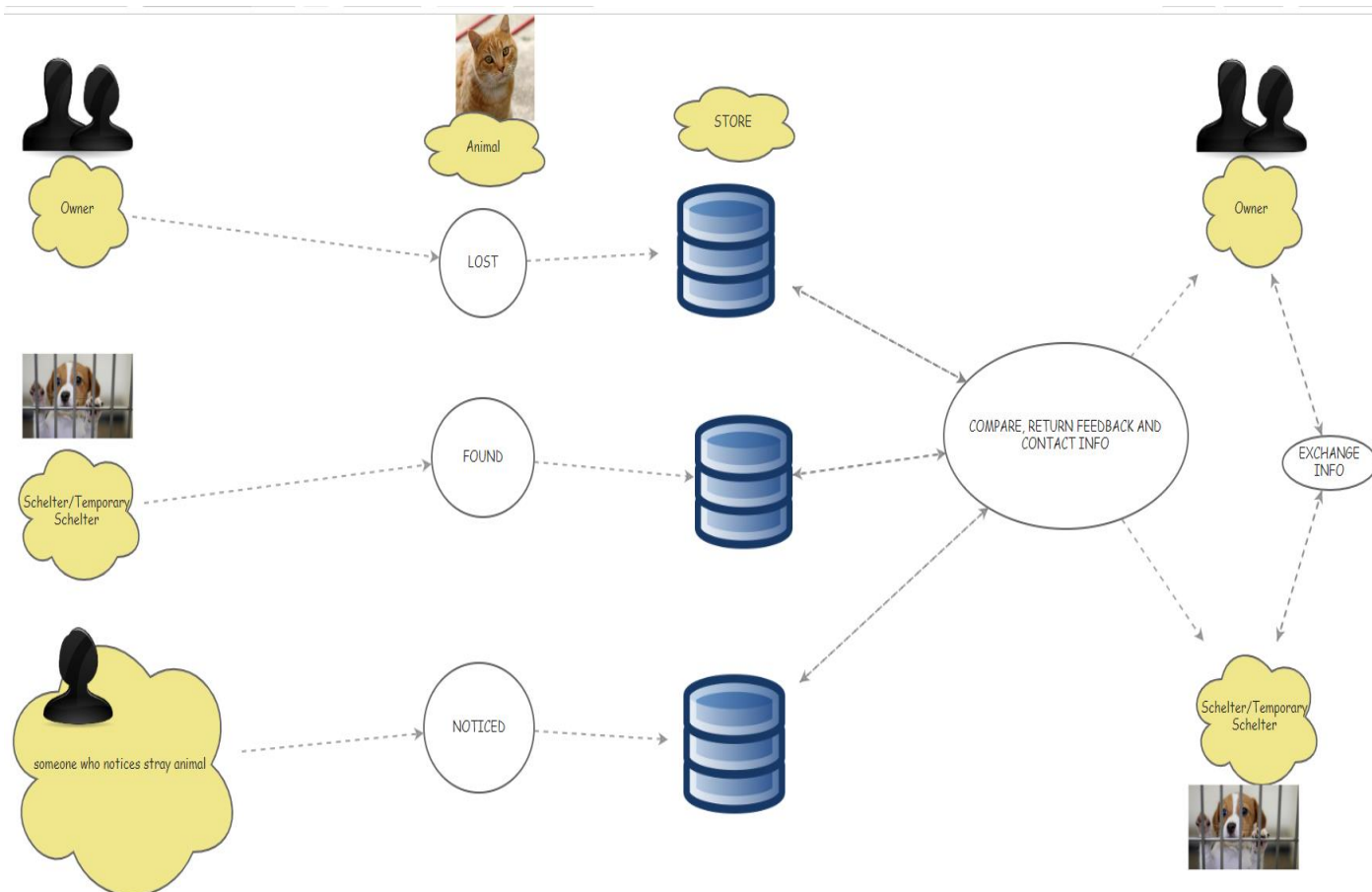
⁸ <https://www.europetnet.com/>

⁹ <https://www.safe-animal.eu>

2. Specyfikacja wymagań

2.1. Wymagania ogólne

- Dostarczenie ogólnopolskiego systemu zrzeszającego wszystkie schroniska i instytucje/osoby powiązane w jednej bazie danych
- Umożliwienie darmowego dostępu do oprogramowania każdej osobie zainteresowaniem odnalezieniem zagubionego zwierzęcia;
- Wyszukiwanie zwierząt odnalezionych i zagubionych przy dodawaniu zwierzęcia do bazy danych;



Rys. 1 – Rich Picture pokazujący schemat działania aplikacji

2.2. Wymagania funkcjonalne

2.2.1. Wymagania funkcjonalne związane z tworzeniem kont użytkowników

- Tworzenie konta nowego użytkownika danego typu:
 - Schronisko;
 - Dom tymczasowy;
 - Osoba która znalazła prawdopodobnie bezpieczne zwierzę, lecz nie udało się go schwytać (dalej: „Zainteresowany”);

- Właściciel zagubionego zwierzęcia;
- Logowanie na konta użytkownika danego typu;
- Weryfikacja duplikatów kont użytkowników każdego typu po adresie e-mail;

2.2.2. Dodawanie zwierząt do bazy danych

- Dodawanie przez użytkowników typu schronisko oraz dom tymczasowy zwierząt do bazy danych zwierząt zagubionych;
- Dodawanie przez użytkowników typu Zainteresowany zwierząt do bazy danych zwierząt potencjalnie zagubionych;
- Dodawanie przez użytkowników typu Właściciel zwierząt do bazy danych zwierząt potencjalnie zagubionych;
- Precyzyjnie określone opisy poszczególnych rodzajów zwierząt na podstawie formularzy i predefiniowanych terminów oraz:
 - Interaktywny piktogram zwierzęcia danego typu umożliwiający uproszczone oznaczenie umaszczenia zwierzęcia wraz ze wskazaniem kolorów;
 - Geolokalizacja miejsca zagubienia/potencjalnego bytowania (konto typu Zainteresowany);
 - Podanie numeru chip, jeśli dotyczy;
 - Dodanie zdjęcia ukazującego umaszczenie, kolor oraz cechy szczególne;
 - Opis tekstowy cech szczególnych.

2.2.3. System zarządzania schroniskiem

- Komplementarny system do zarządzania schroniskiem i jego zasobami, w tym ludzkimi;
- System zamówień leków i pożywienia;
- Kadry i płace;
- Ewidencja zwierząt na potrzeby wewnętrzne, w tym dokumentacja medyczna;
- E-recepty od zakontraktowanych lekarzy weterynarii;
- Czarna lista użytkowników;
- System umów adopcyjnych wraz z ich archiwizacją;
- Wydruki i statystyka;
- Dotacje oraz finansowanie ze zbiorów publicznych;
- Informacje o wolontariuszach.

2.2.4. Wyszukiwanie zwierząt podobnych

- Automatyczne przeszukiwanie bazy danych w momencie dodawania przez użytkownika nowego zwierzęcia;

- W przypadku gdy zwierzę dodaje użytkownik typu Schronisko, Dom tymczasowy lub Zainteresowany, przeszukiwana jest baza danych zwierząt zaginionych;
 - W przypadku gdy zwierzę dodaje użytkownik typu Właściciel, przeszukiwana jest baza danych zwierząt znalezionych oraz potencjalnie zaginionych;
- Wyszukiwanie oparte na zaawansowanych algorytmach wyszukiujących, stopniujących i szacujących podobieństwo odnalezienia zwierzęcia, również z użyciem uczenia maszynowego, w oparciu o dane z systemu – w tym analiza zdjęć, tekstu pisanego, formularzy, piktogramów;
- Prezentacja wyników wraz z danymi kontaktowymi – informacja o stopniu prawdopodobieństwa odnalezienia poszukiwanego zwierzęcia;
- Usuwanie z bazy danych zwierząt odnalezionych (nadzorowane przez Schroniska i Domy Tymczasowe).

3. Technologie i narzędzia

Do opracowania projektu użyte zostały następujące narzędzia i technologie:

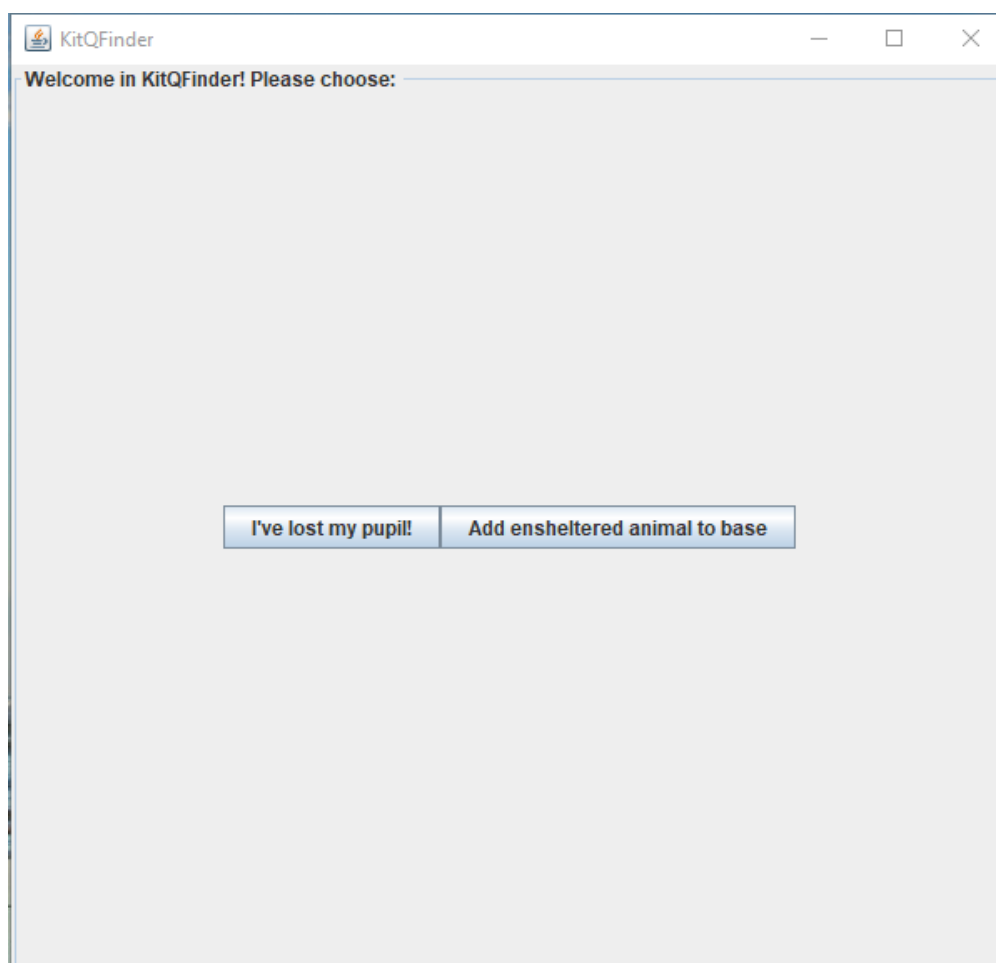
- a) Baza danych – MariaDB w wersji 10.4.11;
- b) Język zapytań - MySQL;
- c) Modelowanie bazy danych – MySQL Workbench;
- d) Połączenie z bazą danych – sterownik JDBC oraz zewnętrzna biblioteka `MySqlConnectionJava.jar`;
- e) Języki programowania – JAVA Standard Edition 13, HTML;
- f) Graficzny interfejs użytkownika – biblioteka JAVA Swing;
- g) Serwer lokalny – XAMPP;
- h) Środowisko programistyczne – IntelliJ Idea;
- i) Rich Picture - <https://insightmaker.com/>.

4. Prezentacja projektu

4.1. Panel wyboru rodzaju użytkownika

Wraz z rozwojem oprogramowania panel wybory użytkownika będzie ulegał coraz to dalszemu rozwojowi. Na końcowym etapie użytkownik będzie miał możliwość:

- a) utworzenia nowego konta użytkownika typu schronisko/osoba która znalazła bezpieczne zwierzę; albo
- b) utworzenia nowego konta użytkownika typu osoba, która zgubiła zwierzę; lub
- c) zalogowania się na już istniejące konto. W zależności od rodzaju konta oprogramowanie udostępniać będzie odmienne panele i funkcje.

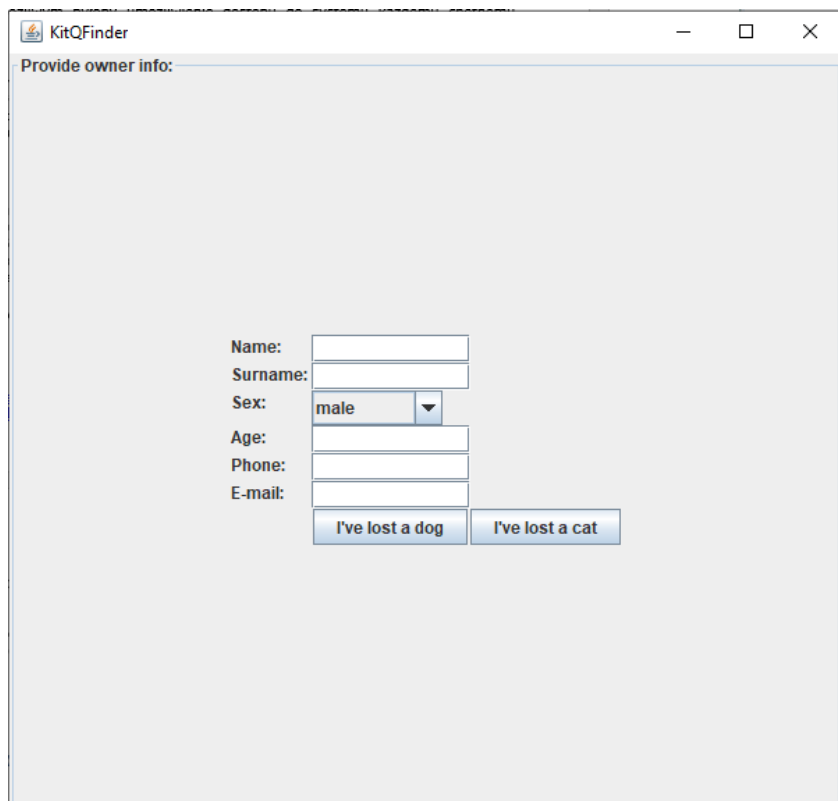


Rys. 2 - panel wyboru użytkownika

Na ten moment „KitQfinder” umożliwia dodanie zwierzęcia do bazy danych jako zwierzę zagubione, bądź znalezione.

4.2. Panel tworzenia użytkownika

Aby przypisać zwierzę do właściciela lub odpowiedniego schroniska, i w konsekwencji umożliwić znalezienie kontaktu do właściciela/schroniska, tworzony jest nowy użytkownik typu „Właściciel” bądź typu „Schronisko”.



The screenshot shows a window titled "KitQFinder" with standard Windows window controls (minimize, maximize, close). Below the title bar is a label "Provide owner info:". The form contains the following fields and controls:

- Name:
- Surname:
- Sex: (dropdown menu)
- Age:
- Phone:
- E-mail:

At the bottom of the form are two buttons: "I've lost a dog" and "I've lost a cat".

Rys. 3 - panel tworzenia nowego Właściciela

KitQFinder

Provide info about the shelter:

Name of the shelter:

Phone number:

E-mail address:

Street name:

Street number:

Flat no:

Post code:

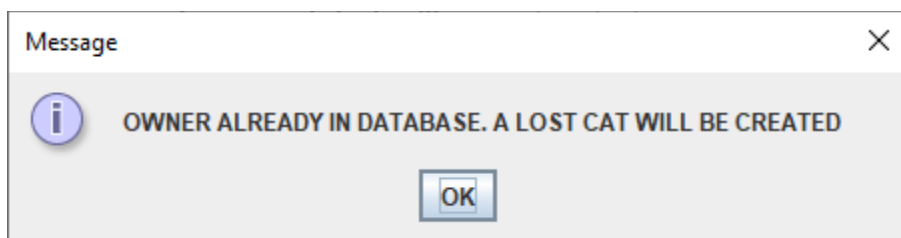
City:

Country:

Rys. 4 - panel tworzenia nowego Schroniska

Program poddaje wprowadzone dane weryfikacji. Jeśli Użytkownik (niezależnie od jego typu) nie znajduje się jeszcze w bazie danych, program stworzy nowego użytkownika, doda go do bazy danych oraz nada mu numer identyfikacyjny. Podczas tworzenia użytkownika wybierany jest także rodzaj tworzonego zwierzęcia, za pomocą przycisku, ze względu na fakt że na ten moment KitQfinder umożliwia dodawanie wyłącznie psów i kotów. W przyszłości gdy funkcjonalność zostanie rozszerzona metoda wskazywania zwierzęcia ulegnie zmianie.

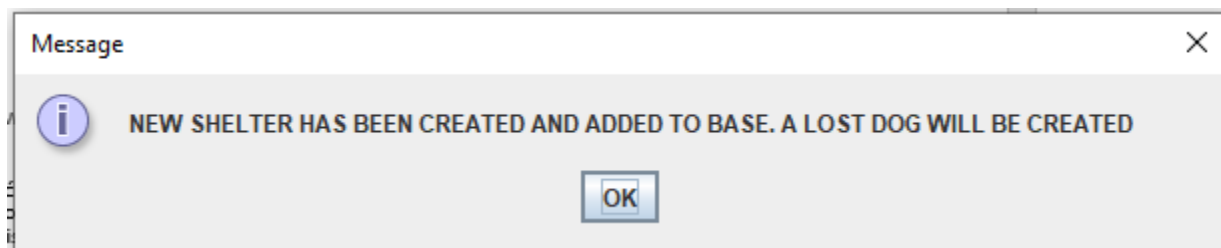
W przypadku gdy dany użytkownik znajduje się już w bazie danych program poinformuje użytkownika o tym fakcie, odzyska z bazy danych ID danego schroniska/właściciela oraz połączy go z tworzonym zwierzęciem.



Rys. 5 - komunikat o duplikacie Właściciela oraz utworzeniu nowego zwierzęcia typu Kot

idOwner	nameOwner	surnameOwner	sexOwner	ageOwner	phoneNoOwner	eMailOwner
107	Jacek	Motyka	male	29	606480056	jmotyka9@gmail.com

Rys. 6 - użytkownik w bazie danych. Nr ID to 107



Rys. 7 - komunikat o utworzeniu nowego Schroniska oraz utworzeniu nowego zwierzęcia typu Pies

idShelter	emailShelter	nameShelter	streetShelter	streetNoShelter	flatNoShelter	postCodeShelter	cityShelter	countryShelter	phoneNoShelter
39	nowe@schronisko.pl	Nowe Schronisko	Polna	23		81-243	Gdynia	Polska	545632342

Rys. 8 - nowe Schronisko w bazie danych

4.3. Panel tworzenia zwierzęcia

Po utworzeniu nowego użytkownika bądź pozyskaniu z bazy danych ID istniejącego użytkownika tworzone jest zwierzę. W zależności od wyboru dokonanego w panelu tworzenia użytkownika, otwierany jest kolejny panel pozwalający na utworzenie zwierzęcia danego rodzaju oraz dodanie go do bazy danych.

Poszczególne, predefiniowane cechy zwierząt są ustalane przez użytkownika. W zależności od rodzaju cechy określone są one:

- zmiennymi typu *Enum* określającymi kolor, płeć, stan zdrowia, budowę ciała, rasę oraz umaszczenie (tylko koty);
- zmiennymi typu *Boolean* określającymi fakt sterylizacji, posiadania obroży, plakietki z imieniem, ogona;
- zmiennymi typu *Long* określającymi przybliżone wymiary zwierzęcia;
- zmienna typu *String* określająca imię (bez znaczenia dla dalszego działania programu z zastrzeżeniem przypadków, gdy zmienna określająca fakt posiadania plakietki z imieniem ma wartość *TRUE*).

KitQFinder

Provide info about your cat:

Name:

Gender:

Health status:

Primary color:

Secondary color (optional):

Tertiary color (optional):

Bodytype:

Is your pupil sterilized?

Did your pupil have collar?

Did your pupil have a name tag?

Did your pupil have a tail?

Weight: 11

Height: 11

Length: 31

Breed:

Pattern:

Rys. 9 - panel tworzenia zwierzęcia typu Kot

KitQFinder

Provide info about your dog:

Name:

Gender:

Health status:

Primary color:

Secondary color (optional):

Tertiary color (optional):

Bodytype:

Is your pupil sterilized?

Did your pupil have collar?

Did your pupil have a name tag?

Did your pupil have a tail?

Weight: 9

Height: 11

Length: 31

Breed:

Rys. 10 - panel tworzenia zwierzęcia typu Pies

4.4. Wyszukiwanie zwierząt podobnych

Po naciśnięciu przycisku „Submit” program wykonuje następujące operacje:

- a) tworzony jest nowy obiekt typu Zwierzę (konkretnej klasy dziedziczącej po abstrakcyjnej klasie Zwierzę) - zmienna o nazwie „*animalSearchedFor*”;
- b) zwierzęciu przypisywany jest Użytkownik (typu Schronisko albo Właściciel);
- c) program zawęża poszukiwania podobnego zwierzęcia poprzez pobranie z bazy danych wyłącznie tych zwierząt, które mają szansę być zwierzęciem tym samym lub podobnym na podstawie cech niezmiennych i obiektywnych (np. płeć, fakt dokonania sterylizacji, rasa) i zwróceniu ich listy („*list*”);
- d) poszukiwane zwierzę jest zapisywane w bazie danych;
- e) program tworzy nowy obiekt klasy JPanel („*resultToBeDisplayed*”) pokazujący w obiektach typu JTextField wyniki, na bazie algorytmu oceniającego podobieństwo zwierząt z listy *list*, o której mowa w lit. c) powyżej do zwierzęcia *animalSearchedFor* dodanego do systemu w lit. a) powyżej. Generyczność oraz implementacja odpowiednich interfejsów zapewnia wyświetlanie odpowiedniego tekstu w zależności od tego, jakie typy danych (rodzaje zwierząt) otrzymuje funkcja *createResultPanel*.

```

public JPanel createResultPanel(ArrayList<T> list, Animal animalSearchedFor){

    JPanel resultToBeDisplayed = new JPanel();
    resultToBeDisplayed.setLayout(new BorderLayout(resultToBeDisplayed, BorderLayout.Y_AXIS));
    resultToBeDisplayed.setVisible(true);
    TitledBorder bf = BorderFactory.createTitledBorder("Thank you for using KitQFinder!");
    resultToBeDisplayed.setBorder(bf);
    ArrayList<T> mostProbableMatch = new ArrayList<T>();
    ArrayList<T> probableMatch = new ArrayList<T>();
    ///// determine what is to be displayed based on probability of a match
    Animal bestmatchSoFar = null;
    int matchProbability ;
    int highestMatchProbabilitySoFar = 0;
    for (T i:list) {
        matchProbability = i.compareAnimals(i, animalSearchedFor);

        if (matchProbability > highestMatchProbabilitySoFar) {
            highestMatchProbabilitySoFar = matchProbability;
            bestmatchSoFar = i;
        }
        else if (matchProbability >= 50) {
            mostProbableMatch.add(i);
        }
        else if (matchProbability >= 20){
            probableMatch.add(i);
        }
    }
    ///// first display most probable match together with contact button
    if (bestmatchSoFar != null) {
        String text = "<html> <h1> Hey, I think we may have found a match! </h1></html>";
        resultToBeDisplayed.add (new JLabel(text));
        resultToBeDisplayed.add(createTextField(bestmatchSoFar));
        resultToBeDisplayed.add(Box.createRigidArea(new Dimension( width: 0, height: 5)));
    }
    ///// secondly display other probable matches
    if (mostProbableMatch.size()>0) {
        String text = "<h2> Here are some other similar animals, maybe it's worth to check them out?</h2>";
        resultToBeDisplayed.add(new JLabel( text: "<html>" + text + "</html>"));
        for (T i:mostProbableMatch) {
            resultToBeDisplayed.add(createTextField(i));
            resultToBeDisplayed.add(Box.createRigidArea(new Dimension( width: 0, height: 5)));
        }
    }
    ///// then display a few less probable matches
    if (probableMatch.size()>0) {
        String text = "<html><h3> These animals are less similar, but maybe it's worth to try...</h3></html>";
        resultToBeDisplayed.add(new JLabel(text));
        for (T i:probableMatch) {
            resultToBeDisplayed.add(createTextField(i));
            resultToBeDisplayed.add(Box.createRigidArea(new Dimension( width: 0, height: 5)));
        }
    }
}

return resultToBeDisplayed;
}

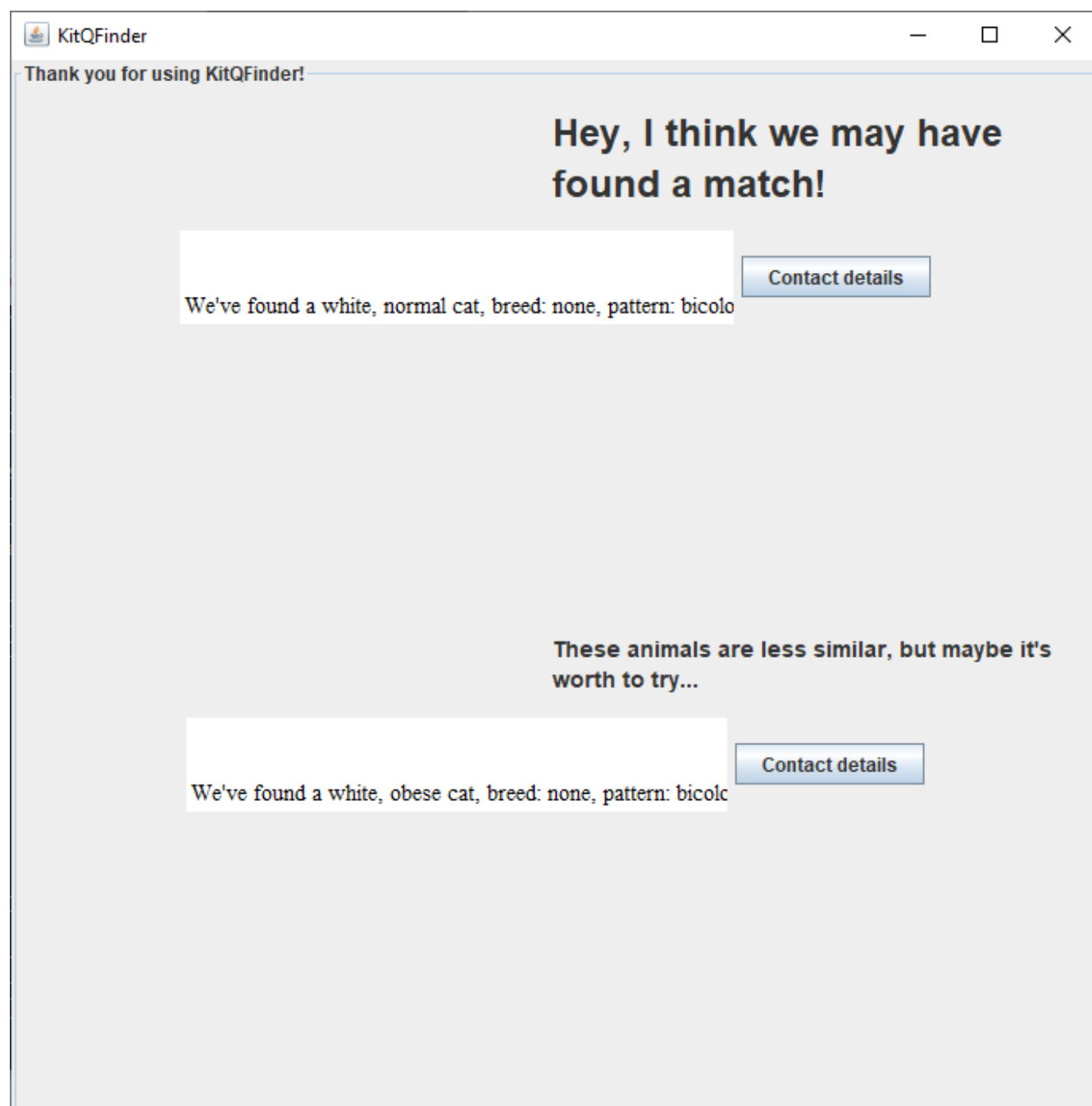
```

Rys. 11 - algorytm tworzący panel z wynikami końcowymi

4.5. Prezentacja wyników

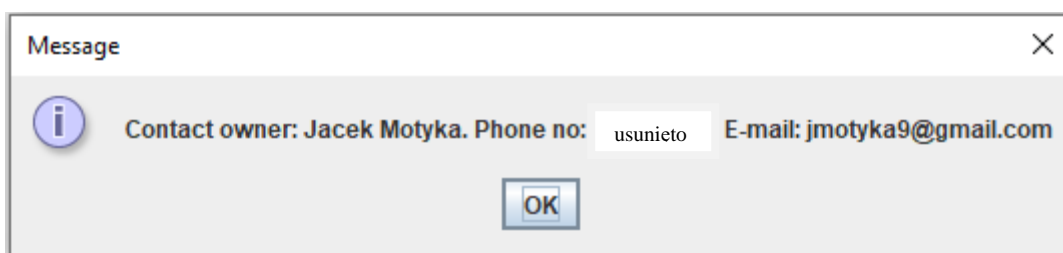
W rezultacie działania algorytmów opisanych w pkt. 4.4 tworzony jest panel z wynikami przedstawiającymi:

- a) najbardziej prawdopodobne dopasowanie wśród zwierząt z listy *list*, o której mowa w pkt. 4.4. lit. c);
- b) dopasowania mniej prawdopodobne, ale dające szansę na zbieżność. Porównanie bierze pod uwagę cechy, które po zagubieniu zwierzęcia mogły ulec zmianie bądź są ocenne i zależą od odbioru znalazcy (np. różnica w wymiarach zwierzęcia, budowie ciała, podobny kolor).



Rys. 12 - panel z wynikami przeszukiwania

W przypadku gdy użytkownik uzna, iż istnieje prawdopodobieństwo że znalezione zostało jego zwierzę, ma możliwość podglądu danych kontaktowych tego użytkownika, który zwierzę dodał do bazy danych. W zależności od tego, kto tworzy dane zwierzę, przeszukiwana jest inna tabela w bazie danych – w przypadku dodania znalezione zwierzęcia i umieszczenia go w schronisku przeszukiwana jest tabela ze zwierzętami zaginionymi i odwrotnie – w przypadku dodania zagubionego zwierzęcia przez właściciela, przeszukiwana jest tabela ze zwierzętami znalezionymi i przebywającymi w schroniskach.



Rys. 13 - dane kontaktowe na przykładzie Właściciela (na potrzeby prezentacji usunięto nr telefonu)

5. Weryfikacja rozwiązania. Napotkane problemy.

Na dzień powstania niniejszej pracy program spełnia oczekiwania autora, aczkolwiek widoczna jest potrzeba wielu poprawek oraz dalszego rozwijania rozwiązania, o których mowa w pozostałych rozdziałach niniejszej pracy. Najciekawsze problemy napotkane w trakcie tworzenia projektu to:

- a) Brak znajomości struktury programu (głównie podział na klasy) okienkowego.

Przykład: brak doświadczenia w programowaniu GUI, trudność z podziałem na klasy oraz wydajnym sposobem przekazywania obiektów pomiędzy poszczególnymi widokami

- b) Brak doświadczenia w przekazywaniu informacji kluczowych pomiędzy poszczególnymi obiektami składającymi się na interfejs graficzny.

Przykład: konieczność zaprogramowania przycisku, który otwiera nowy obiekt klasy JPanel, wymagający informacji o użytkowniku korzystającym z systemu. Konieczność znalezienia sposobu na przekazanie informacji o użytkowniku pomiędzy formularzem jego tworzenia, przyciskiem a nowym obiektem klasy JPanel.

- c) Problemy z połączeniem się z bazą danych. Brak znajomości klas języka Java ułatwiających pracę z bazą danych.

Przykład: problemy z dołączeniem sterownika JDBC do programu.

- d) Nieoptymalne nazewnictwo przyjęte w bazie danych.

Przykład: pomimo prawidłowej implementacji dziedziczenia i interfejsów obsługujących obiekty dziedziczące po abstrakcyjnej klasie „Animal” lub „Submitter” powstała konieczność powielania podobnych zapytań SQL ze względu na podobną strukturę tabel, ale inne nazwy dla kolumn (np. OwnerName/SchelterName zamiast „Name” w obu tabelach).

6. Podsumowanie i wnioski

Projekt, zdaniem autora, stanowi ciekawe rozwiązanie nie tylko na potrzeby rozwijania umiejętności programistycznych. Po rozbudowie systemu, implementacji wymagań opisanych w niniejszej pracy oraz poprawkach może stać się holistycznym narzędziem do prowadzenia ewidencji zwierząt w schroniskach. Przyjazny interfejs, komplementarność rozwiązania oraz daleko idące rozpowszechnienie dostępu do oprogramowania powinny stanowić czynniki prowadzące do coraz to dalej idącej popularyzacji rozwiązania zarówno wśród użytkowników nieinstytucjonalnych (zatroškani właściciele), jak i instytucjonalnych. Sukcesy w odnajdywaniu zwierząt (zależne od stopnia złożoności i poprawności zaimplementowanego algorytmu porównującego zwierzęta) przełożą się na dalszą popularyzację i ilość użytkowników, a tym samym – na bezpośrednią pomoc zwierzętom.

Rozbudowane funkcje dostarczane schroniskom mogłyby z sukcesem zastąpić oprogramowanie wykorzystywane do tej pory – często płatne. Jeden system dla zarządzania zamówieniami, ewidencją zwierząt, kontaktów z właścicielami oszczędziłby także czasu na naukę obsługi wielu systemów dla pracowników. Realna oszczędność czasu przełożyłaby się na jakość opieki nad samymi zwierzętami. Połączenie z ogólnodostępnym systemem ewidencji zwierząt poprzez CHIP mogłoby stanowić kolejny krok w rozwoju systemu.

Autor planuje dalszy rozwój projektu, głównie w celach edukacyjnych oraz prezentacji nabytych umiejętności zainteresowanym podmiotom. Wraz z rozwojem projektu nabyta wiedza, doświadczenie i świadomość istniejących i dostępnych rozwiązań ukazała konieczność wielu poprawek w istniejącym projekcie – np. tworzenie kont użytkowników, usprawnienie bazy danych i zapytań.

7. Literatura

7.1. Źródła internetowe

1. <http://www.boz.org.pl/raport/2016/1.htm>
2. <http://www.datum.com.pl/schronisko.html>
3. <https://www.europetnet.com/>
4. <https://www.geulincx.pl/program-rejestracji>
5. <https://kodujdlapolski.pl/projects/aplikacja-dla-schroniska-promyk/>
6. <https://www.obrona-zwierzat.pl/aktualnosci/1715-bezdomnosc-zwierzat-w-polsce-w-2018-r.html>
7. <https://www.safe-animal.eu>
8. <http://www.sioz.pl/AboutSIOZ>

7.2. Pozostałe źródła

1. Ustawa z dnia 21 sierpnia o ochronie zwierząt, Dz. U. 1997 Nr 111 poz. 724.
2. H. Schildt, „*Java. Kompendium programisty*”, Wydanie X, Helion SA 2019 r.