

POLITECHNIKA WROCŁAWSKA
WYDZIAŁ ELEKTRONIKI

KIERUNEK: INFORMATYKA (INF)

SPECJALNOŚĆ: INŻYNIERIA SYSTEMÓW INFORMATYCZNYCH (INS)

PRACA DYPLOMOWA
INŻYNIERSKA

Aplikacja webowa wspomagająca wybór rasy psa
Web application supporting choosing proper dog
breed

AUTOR:

Paweł Komorowski

PROWADZĄCY PRACĘ:

Dr inż. Tomasz Babczyński,
Katedra Informatyki Technicznej

OCENA PRACY:

WROCŁAW 2019

Spis treści

| | |
|---|----|
| Spis rysunków | 5 |
| Spis tabel | 6 |
| Spis listingów | 7 |
| Skróty | 8 |
| 1. Wstęp..... | 9 |
| 1.1. Wprowadzenie | 9 |
| 1.2. Cel i zakres pracy | 9 |
| 2. Przegląd istniejących rozwiązań | 11 |
| 2.1. Opis rozwiązań | 11 |
| 2.2. Istniejące rozwiązania..... | 11 |
| 2.2.1. IAMS | 11 |
| 2.2.2. Nestlé Purina Petcare | 12 |
| 2.2.3. Petbarn..... | 13 |
| 2.2.4. DogTime..... | 13 |
| 2.2.5. PodajŁape..... | 14 |
| 2.3. Podsumowanie..... | 14 |
| 3. Analiza wymagań..... | 15 |
| 3.1. Opis działania | 15 |
| 3.2. Wymagania funkcjonalne | 15 |
| 3.2.1. Wymagania funkcjonalne gościa | 15 |
| 3.2.2. Wymagania funkcjonalne administratora..... | 15 |
| 3.3. Wymagania нефункционалне | 15 |
| 4. Technologie i narzędzia | 16 |
| 4.1. Wybrane technologie | 16 |
| 4.1.1. Front-end | 16 |

| | | |
|--------|---|----|
| 4.1.2. | Back-end..... | 16 |
| 4.1.3. | System zarządzania bazą danych | 17 |
| 4.2. | Wybrane narzędzia | 17 |
| 4.2.1. | Środowisko programistyczne | 17 |
| 4.2.2. | Zarządzanie projektem | 17 |
| 4.2.3. | Zarządzanie bazą danych..... | 17 |
| 4.2.4. | Testowanie oprogramowania | 17 |
| 4.2.5. | Repozytoria kodu | 18 |
| 4.3. | Licencje | 18 |
| 5. | Analiza biznesowa..... | 19 |
| 5.1. | Analiza kosztów | 19 |
| 5.2. | Analiza strategii przedsięwzięcia | 19 |
| 5.2.1. | Analiza SWOT | 19 |
| 5.2.2. | Analizowane czynniki | 20 |
| 5.2.3. | Opis czynników | 20 |
| 5.2.4. | Wycena czynników | 21 |
| 5.2.5. | Wybór strategii | 22 |
| 5.2.6. | Podsumowanie | 23 |
| 6. | Projekt bazy danych | 25 |
| 6.1. | Analiza rzeczywistości | 25 |
| 6.2. | Model bazy danych..... | 26 |
| 6.2.1. | Diagram ERD | 26 |
| 6.2.2. | Opis tabel..... | 26 |
| 6.2.3. | Opis powiązań | 26 |
| 6.2.4. | Mechanizmy przetwarzania danych | 28 |
| 7. | Projekt aplikacji..... | 29 |
| 7.1. | Architektura aplikacji | 29 |

| | | |
|------|--|----|
| 7.2. | Interfejs graficzny | 29 |
| 7.3. | Specyfikacja wybranych funkcjonalności aplikacji | 31 |
| 7.4. | Połączenie z bazą danych | 32 |
| 7.5. | Zabezpieczenia na poziomie aplikacji | 32 |
| 8. | Implementacja i testowanie | 33 |
| 8.1. | Implementacja modelu i bazy danych | 33 |
| 8.2. | Implementacja kontrolerów | 34 |
| 8.3. | Implementacja widoków | 35 |
| 8.4. | Komunikacja z bazą danych | 36 |
| 8.5. | Logika aplikacji | 36 |
| 8.6. | Testy jednostkowe | 38 |
| 8.7. | Testy akceptacyjne | 39 |
| 9. | Podsumowanie | 41 |
| 9.1. | Efekt końcowy | 41 |
| 9.2. | Kierunki rozwoju | 42 |
| | Literatura | 43 |
| | Dodatek A | 45 |
| | Instrukcja obsługi użytkownika | 45 |
| | Spis funkcjonalności | 45 |
| | Funkcjonalności użytkownika | 45 |
| | Funkcjonalności administratora | 49 |
| | Dodatek B | 52 |
| | Opis dołączonej płyty CD | 52 |

Spis rysunków

| | |
|--|----|
| Rysunek 1. Zrzut ekranu z aplikacji IAMS | 12 |
| Rysunek 2. Zrzut ekranu z aplikacji Nestlé Purina Petcare | 12 |
| Rysunek 3. Zrzut ekranu z aplikacji Petbarn..... | 13 |
| Rysunek 4. Zrzut ekranu z aplikacji Dogime | 13 |
| Rysunek 5. Zrzut ekranu z aplikacji PodajŁape..... | 14 |
| Rysunek 6. Diagram ERD bazy danych..... | 27 |
| Rysunek 7. Diagram przypadków użycia..... | 30 |
| Rysunek 8. Diagram czynności logowania administratora | 31 |
| Rysunek 9. Diagram czynności dodawania rasy do bazy danych..... | 32 |
| Rysunek 10. Prezentacja funkcji L..... | 37 |
| Rysunek 11. Prezentacja funkcji gamma | 37 |
| Rysunek 12. Prezentacja funkcji lambda | 37 |
| Rysunek 13. Zrzut ekranu z narzędzia Selenium IDE prezentujący przykładowy test..... | 40 |
| Rysunek 14. Zrzut ekranu odnośników w nagłówku | 46 |
| Rysunek 15. Zrzut ekranu formularza wyszukiwania rasy | 46 |
| Rysunek 16. Zrzut ekranu wyniku wyszukiwania ras..... | 46 |
| Rysunek 17. Zrzut ekranu listy wszystkich ras | 47 |
| Rysunek 18. Zrzut ekranu szczegółów rasy | 47 |
| Rysunek 19. Zrzut ekranu galerii zdjęć rasy | 48 |
| Rysunek 20. Zrzut ekranu formularza logowania | 48 |
| Rysunek 21. Zrzut ekranu odnośników edycji oraz usuwania rasy | 49 |
| Rysunek 22. Zrzut ekranu odnośnika dodawania rasy..... | 49 |
| Rysunek 23. Zrzut ekranu formularza dodawania rasy | 50 |
| Rysunek 24. Zrzut ekranu galerii zdjęć z formularza edycji rasy | 51 |
| Rysunek 25. Zrzut ekran przycisku wylogowania | 51 |

Spis tabel

| | |
|--|----|
| Tabela 1. Podsumowanie kosztów związanych z wykonanie i utrzymanie aplikacji. | 19 |
| Tabela 2. Przedstawienie analizowanych cech..... | 20 |
| Tabela 3. Wycena wewnętrznych czynników analizy SWOT. | 22 |
| Tabela 4. Wycena zewnętrznych czynników analizy SWOT. | 22 |
| Tabela 5. Przebieg wybór strategii na podstawie uwzględnionych czynników. | 23 |
| Tabela 6. Wynik wyboru strategii. | 23 |
| Tabela 7. Zestawienie danych wchodzących w skład widoku Breed Info..... | 28 |

Spis listingów

| | |
|--|----|
| Listing 1. Fragment klasy modelu z adnotacjami ORM | 34 |
| Listing 2. Przykładowa klasa typu ENUM..... | 34 |
| Listing 3. Tworzenie widoku | 34 |
| Listing 4. Endpoint odpowiedzialny za obsługę żądania wyświetlenia informacji o rasie..... | 35 |
| Listing 5. Tworzenie repozytorium dla klasy BreedsInfo | 36 |
| Listing 6. Funkcja obliczająca stopień przynależności rozmiaru rasy | 38 |
| Listing 7. Przykładowy tekst jednostkowy sprawdzający poprawność logiki rozmytej..... | 39 |

Skróty

UML (*ang. Unified Modeling Language*)

ERD (*ang. Entity-Relationship Diagram*)

JPA (*ang. Java Persistence API*)

HTML (*ang. HyperText Markup Language*)

CSS (*ang. Cascading Style Sheets*)

SQL (*ang. Structured Query Language*)

SWOT (*ang. Strengths Weaknesses Opportunities Threats*)

JVM (*ang. Java Virtual Machine*)

MVC (*and. Model-View-Controller*)

ORM (*ang. Object-Relational Mapping*)

1. Wstęp

1.1. Wprowadzenie

Na świecie istnieje wiele ras psów i każda z nich charakteryzuje się własnym zestawem cech. Przez ich różnorodność lub przez nieświadomość odpowiedzialności idącej za takim wyborem ludzie nie przywiązują do tego należytej wagi. W konsekwencji wybór często pada na rasę nie pasującą do przestrzeni, którą dysponuje właściciel, albo do jego wyobrażenia na jej temat.

Kluczowym kryterium, które powinno się mieć na uwadze podczas wyboru psa, jest jego zdrowie oraz problemy z nim związane. Z powodu stałości swoich cech każda rasa ma zdefiniowaną podatność na choroby genetyczne. Dzięki temu możliwe jest uniknięcie wyboru choro-ego psa oraz znaczne ograniczenie wydatków związanych z leczeniem. Niestety, te ogólnodostępne informacje nie są zazwyczaj wykorzystywane. Czołowym przykładem jest kwestia buldogów francuskich – chociaż ich wybór jest odradzany przez weterynarzy, a hodowla w Holandii jest prawnie zakazana [1], nadal są one często kupowane.

1.2. Cel i zakres pracy

Celem pracy jest stworzenie aplikacji webowej wykorzystującej technologie bazodanowe. Tworzona aplikacja ma być prostą bazą wiedzy z możliwością wyszukiwania psa, która nie będzie odstraszać potencjalnych użytkowników natłokiem informacji, a będzie dawała te najbardziej istotne z podkreśleniem podatności na choroby każdej z ras.

W zakres pracy wchodzi przeprowadzenie czterech etapów: analiza wymagań, analiza biznesowa, projektowanie oraz implementacja.

1. Analiza wymagań – w skład analizy systemu wchodzi przedstawienie idei aplikacji wraz z już istniejącymi rozwiązaniami, sformułowanie wymagań funkcjonalnych i nie-funkcjonalnych oraz wybór technologii i narzędzi służących do jego realizacji.
2. Analiza biznesowa – przedstawi koszty wykonania projektu oraz analizę strategii przedsięwzięcia.
3. Projektowanie – proces projektowania będzie obejmował bazę danych, architekturę aplikacji, kwestie bezpieczeństwa oraz graficzny interfejs użytkownika. Zawarte w nim

będą opisy słowne projektowanych elementów oraz ilustrujące je diagramy UML [2] i ERD [3].

4. Implementacja – będzie obejmować implementacje bazy danych, logiki aplikacji oraz warstwy prezentacji, w których skład wejdą: opisy funkcjonalności, wykorzystanych wzorców projektowych, listingi kodu źródłowego oraz przeprowadzone testy elementów aplikacji.

Ostatnią częścią pracy inżynierskiej będzie instrukcja obsługi przeznaczona dla użytkownika systemu.

2. Przegląd istniejących rozwiązań

2.1. Opis rozwiązań

Na rynku istnieją różne aplikacje wspomagające wybór rasy psa. Są tworzone najczęściej przez portale zajmującą się tematyką dbania o swoje zwierzęta lub firmy produkujące produkty dla psów.

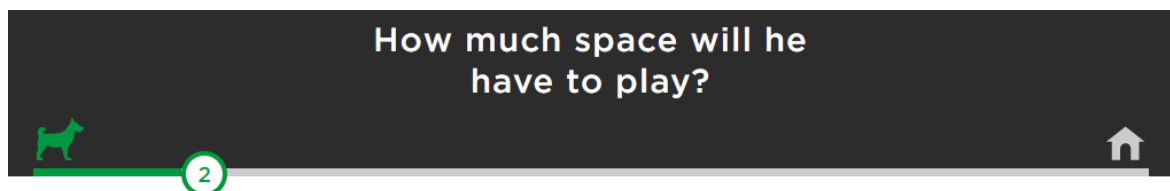
Konkurencyjne aplikacje charakteryzują się tym samym podejściem do procesu wybierania rasy psa. Dają potencjalnemu przyszłemu właścicielowi krótką, zawierającą około 15 pytań ankietę, która ma na celu zbadanie jego preferencji i stworzenie uproszczonego profilu psychologicznego. Poruszane są w nich banalne kwestie typu: „Czy lubisz biegać?” albo „Czy wolisz psa do głaskania czy psa obronnego?”. Takie podejście nie informuje właściciela, jaki jest naprawdę polecany pies oraz jakie wiążą się z nim wyzwania.

Omawiane aplikacje również mają formę aplikacji webowej, przy czym są zawsze fragmentem większego portalu. Po przetestowaniu każdej z wybranych pięciu można zauważyć, że trzy z nich są nieoptymalizowane. Przejścia między kolejnymi pytaniami wymagają długiego przeładowania strony, co jest uciążliwe dla użytkownika. Ważną kwestią w stosunku do polskiego rynku jest to, że w języku polskim działa tylko jedna aplikacja.

2.2. Istniejące rozwiązania

2.2.1. IAMS

IAMS to firma produkująca karmy dla psów i kotów w różnych grupach wiekowych. Szczyci się wysokiej jakości produktami projektowanymi przez zwierzęcych dietetyków, dzięki czemu mają dobrą renomę. Strona internetowa dostępna jest pod adresem zamieszczonym w literaturze [4].



Rysunek 1. Zrzut ekranu z aplikacji IAMS

2.2.2. Nestlé Purina Petcare

Jest to firma produkująca średniej jakości karmy i przysmaki dla zwierząt. Mimo konkurencji posiadającej lepsze, ale zarazem droższe karmy, jest popularnym wyborem wśród ludzi. Strona internetowa dostępna jest pod adresem zamieszczonym w literaturze [5].

What qualities would your ideal dog have?

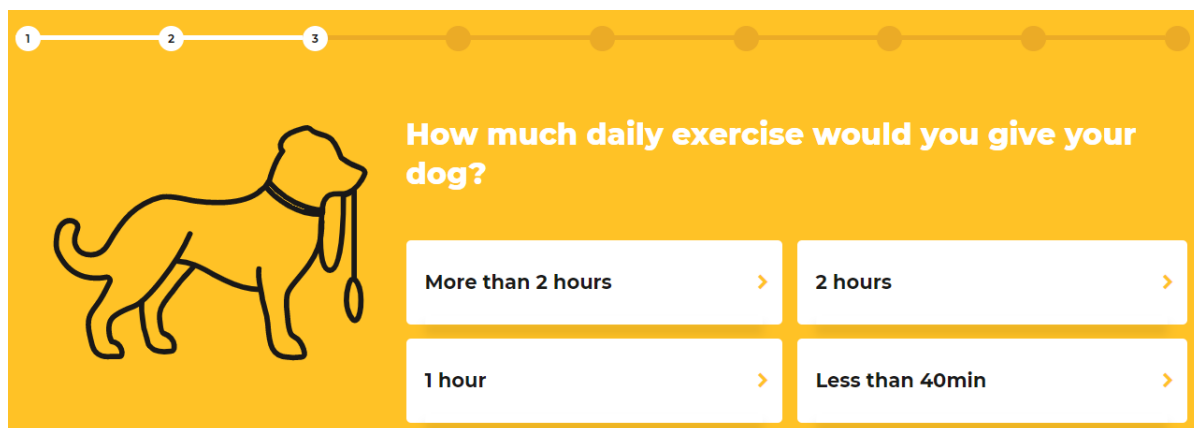
CHOOSE ONE OR MORE

- ☐ Protection
- ☒ Sporting or Hunting
- ☐ Training
- ☐ Being a Companion
- ☐ Hanging Out With Children
- ☐ Hanging Out With Our Pets
- ☐ Interacting With Strangers

Rysunek 2. Zrzut ekranu z aplikacji Nestlé Purina Petcare

2.2.3. Petbarn

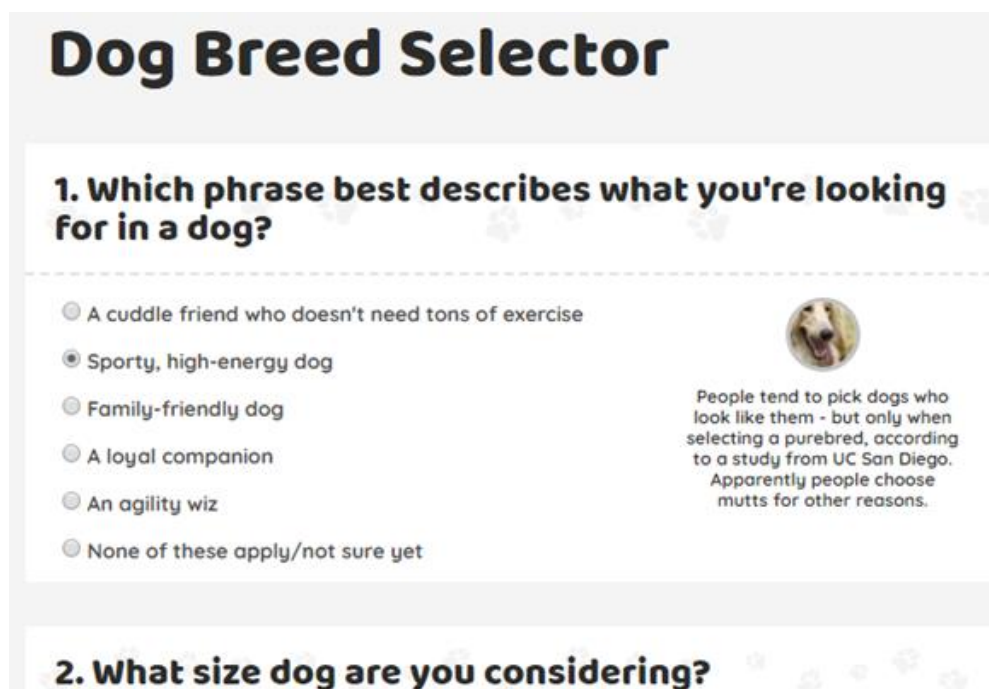
Jest to firma sprzedająca karmy dla zwierząt. Funkcjonuje głównie na rynku australijskim, a jej asortyment skierowany jest do szerokiego grona gatunków. Strona internetowa dostępna jest pod adresem zamieszczonym w literaturze [6].



Rysunek 3. Zrzut ekranu z aplikacji Petbarn

2.2.4. DogTime

Jest to portal internetowy posiadający obszerną bazę wiedzy, między innymi o zdrowiu, zachowaniach i sposobach wychowania psów. Wspiera również ideę adopcji zwierząt. Strona internetowa dostępna jest pod adresem zamieszczonym w literaturze [7].



Rysunek 4. Zrzut ekranu z aplikacji Dogime

2.2.5. PodajŁape

Jest to portal internetowy w języku polskim przeznaczony dla miłośników psów. Posiada informacje na temat istniejących w Polsce hodowli, sklepów zoologicznych, weterynarzy, gabinetów fryzjerskich, hoteli dla psów i wiele więcej. Strona internetowa dostępna jest pod adresem zamieszczonym w literaturze [8].

9. Charakter:

Ważne zwłaszcza dla przyszłych posiadaczy swojego pierwszego psa. Wybierając rasę, która lubi dominować musimy pamiętać, że nie jest ona polecana dla osób, które wcześniej nie miały psa.

- ☒ Nieistotne
- ☐ Uległy
- ☐ Pośredni
- ☐ Pies lubi dominować, jeżeli się mu tego nie zabroni

Rysunek 5. Zrzut ekranu z aplikacji PodajŁape

2.3. Podsumowanie

Spośród powyższych portali zajmującymi się psami najbliższej idei mojej aplikacji jest polskie rozwiązanie, ponieważ zadaje bardziej odpowiedzialne pytania – sugerują one, że posiadanie psa łączy się nie tylko z przyjemnościami.

3. Analiza wymagań

3.1. Opis działania

Skończony projekt ma mieć formę aplikacji webowej dostępnej z poziomu przeglądarki internetowej. Celem aplikacji jest ułatwienie osobom zainteresowanym nabyciem rasowego psa jego wyboru. Aplikacja prosi użytkownika o wybranie, na jakich cechach psa mu zależy, a następnie prezentuje optymalne wybory. Kluczową właściwością systemu ma być jasne informowanie nie tylko o zaletach danej rasy, ale także o wadach. Polecenia będą dokonywane nie tylko na podstawie oczekiwań użytkownika, ale także jego możliwości opieki nad zwierzęciem.

3.2. Wymagania funkcjonalne

3.2.1. Wymagania funkcjonalne gościa

- Strona główna umożliwiająca wyświetlanie poleceń na podstawie wybranych cech.
- Realizacja wyboru poleceń za pomocą logiki rozmytej.
- Podstrony zawierające pełną bazę wiedzy: opisy oraz zdjęcia dla każdej z wprowadzonych ras.
- Strona zawierająca pełną listę psów wraz z odnośnikami do podstron.
- Brak konieczności logowania do systemu dla gości.

3.2.2. Wymagania funkcjonalne administratora

- Panel administracyjny umożliwiający wprowadzanie, usuwanie i edycję ras oraz dodawanie i usuwanie zdjęć.

3.3. Wymagania niefunkcjonalne

- Prostota i szybkość użytkowania.
- Wyniki polecenia skupione na odpowiedzialności i bezpieczeństwie

4. Technologie i narzędzia

4.1. Wybrane technologie

4.1.1. Front-end

- HTML 5 [9] – jest najnowszą odsłoną standardu HTML, w rozwinięciu HyperText Markup Language – hipertekstowy język znaczników. Powstał jako rozszerzenie HTML 4 i XHTML o nowe znaczniki, zachowując równocześnie wsteczną kompatybilność. W odróżnieniu od poprzednich wersji HTML 5 miał na celu wprowadzenie interaktywności do statycznych stron internetowych.
- CSS [10] – Cascading Style Sheets, czyli kaskadowe arkusze stylów, służą do opisywania wyglądu stron WWW. Opisują, jak poszczególne elementy powinny być wyświetlane.
- FreeMarker [11] – jest biblioteką języka Java stworzoną przez firmę Apache. Umożliwia tworzenie szablonów – generuje tekst wyjściowy na podstawie szablonów napisanych w języku FreeMarker Template Language oraz przygotowanych przez program danych. W aplikacji zostanie użyty w celu generowania stron HTML.

4.1.2. Back-end

- Java 1.8 [12] – język programowania obiektowego ogólnego przeznaczenia. Jest szeroko stosowany w aplikacjach biznesowych. Jego kod jest kompilowany do kodu bajtowego, który następnie jest wykonywany przez wirtualną maszynę. Wykorzystywanie wirtualnej maszyny umożliwia bezproblemowe uruchomienie każdego programu napisanego w tym języku na dowolnym urządzeniu, które ją posiada. Został wybrany ze względu na posiadane przeze mnie doświadczenie programistyczne w tym języku.
- Spring Boot 2.2.0 [13] – jeden z elementów frameworka Spring, w skład którego wchodzi wiele projektów przeznaczonych do realizacji aplikacji w języku Java. Udostępnia funkcjonalności z innych Springowych projektów, jak na przykład JPA, oryginalnie dostępny w Spring Data. Został wybrany w celu nauki i poszerzania własnych kompetencji.

4.1.3. System zarządzania bazą danych

- MySQL [14] – darmowy, open-sourcowy system zarządzania relacyjnymi bazami danych firmy Oracle. Został wybrany, ponieważ pracowałem z nim już podczas studiów. Ponadto, jest wspierany przez większość serwerów, dzięki czemu nie będzie problemu z doбором takiego, na którym może zostać uruchomiona aplikacja.

4.2. Wybrane narzędzia

4.2.1. Środowisko programistyczne

- IntelliJ IDEA Community [15] w wersji 2019.2.4 – darmowe środowisko programistyczne przeznaczone dla języków wykorzystujących wirtualną maszynę Javy (JVM). Dzięki wielu dostępnym wtyczkom zostanie użyty również do pozostałych technologii użytych w back-endzie i front-endzie.

4.2.2. Zarządzanie projektem

- Apache Maven 3.6.2 [16] – jest narzędziem do zarządzania projektami w języku Java oraz automatyzującym proces ich budowania. Umożliwia również łatwe pozyskiwanie nowych bibliotek, które są udostępniane w Mavenowych repozytoriach [17].

4.2.3. Zarządzanie bazą danych

- MySQL Community Server [18] w wersji 8.0.18 – darmowy system zarządzania bazą danych przeznaczony dla MySQL firmy Oracle. Jest serwerem udostępniającym klientom bazę danych.
- phpMyAdmin [19] w wersji 4.9.1 – darmowe narzędzie napisane w języku PHP, umożliwiające administrowanie bazą danych MySQL przez Internet. Jego graficzny interfejs użytkownika wspiera wykonywanie najczęściej podejmowanych operacji, między innymi zarządzanie bazą danych, jej tabelami, relacjami czy indeksami. Umożliwia również wykonywanie poleceń w języku SQL.

4.2.4. Testowanie oprogramowania

- JUnit [20] – biblioteka służąca do pisania testów jednostkowych w języku Java. Testy jednostkowe sprawdzają poprawność działania najmniejszych elementów

systemu, umożliwiają sprawdzenie poprawności funkcjonalności bez konieczności uruchamiania całej aplikacji oraz pomagają w utrzymaniu czytelnej architektury aplikacji.

- Selenium IDE [21] – narzędzie do automatycznego przeprowadzania testów funkcjonalnych aplikacji webowych. Między innymi umożliwia nagranie testów, co pozwala na ich zrealizowanie bez umiejętności programowania.

4.2.5. Repozytoria kodu

- Github [22] – portal internetowy umożliwiający przechowywanie kodu źródłowego aplikacji oraz historię jego modyfikacji. Dzięki korzystaniu z tego narzędzia projekt jest zabezpieczony przez utratą kodu źródłowego spowodowaną awarią sprzętu, natomiast historia pomocna jest podczas śledzenia błędnie zmodyfikowanego kodu. Github wspiera również wspólną pracę nad projektami poprzez mechanizmy pracy nad zdalnym kodem.

4.3. Licencje

Wszystkie wymienione technologie i narzędzia są darmowe do wykorzystania komercyjnego.

5. Analiza biznesowa

5.1. Analiza kosztów

Analiza kosztów określa koszt wykonania omawianej w pracy aplikacji. W celu jego oszacowania zakładamy, że czas pracy programisty wyniesie 120 godzin, z czego 20 godzin zostanie przeznaczone na projektowanie systemu. Zakładamy również, że stawka godzinowa wyniesie 30zł.

Tabela 1. Podsumowanie kosztów związanych z wykonanie i utrzymanie aplikacji.

| Opis | Okres | Koszt [zł] |
|----------------------------|-------------|------------|
| Projekt i implementacja | Jednorazowo | 3600 |
| Hosting | Rok | 200 |
| Domena | Rok | 50 |
| Oprogramowanie i narzędzia | - | 0 |

Analiza kosztów pokazana w tabeli 1. pokazuje, że główną inwestycją związaną z realizacją projektu jest jej utworzenie. Koszty utrzymania gotowej aplikacji są niewielkie i wynoszą około 250zł rocznie.

5.2. Analiza strategii przedsięwzięcia

5.2.1. Analiza SWOT

Analiza SWOT [23] jest wiodącą metodą strategicznej analizy organizacji lub przedsięwzięcia. Łączy ona mocne oraz słabe strony przedsięwzięcia jako czynniki wewnętrzne z szansami oraz zagrożeniami jako czynniki zewnętrzne.

- Mocne strony określają umiejętności zespołu oraz posiadane zasoby, które mogą dać przewagę na rynku.
- Słabe strony mówią o czynnikach, które mają negatywny wpływ na realizację projektu i mogą wpłynąć na mocne strony.
- Szanse to sprzyjające projektowi okoliczności na rynku lub w społeczeństwie. Ich dobre wykorzystanie przynosi większe szanse na sukces przedsięwzięcia.

- Zagrożenia to utrudnienia zewnętrzne wpływające na projekt. Jeśli nie zostaną wyeliminowane będą miały na niego negatywny wpływ.

5.2.2. Analizowane czynniki

Po przeanalizowaniu sytuacji na rynku, trendów społecznych oraz posiadanych umiejętności, mające wpływ na projekt czynniki zostały zaprezentowane w tabeli 2.

Tabela 2. Przedstawienie analizowanych cech.

| Mocne strony | Słabe strony |
|--|---|
| <ol style="list-style-type: none"> 1. Wyszukiwanie lepsze niż u konkurencji. 2. Przyjazny, ilustrowany interfejs zachęci użytkowników. 3. Deweloper dobrze zna wymagane technologie. | <ol style="list-style-type: none"> 1. Mniejsza ilość informacji na temat psów może sugerować, że strona jest gorsza. 2. Informowanie o problemach może odstraszyć użytkowników. 3. Brak wiedzy związanej z marketingiem może utrudnić promowanie produktu. |
| Szanse | Zagrożenia |
| <ol style="list-style-type: none"> 1. Posiadanie zwierząt jest popularne. 2. Coraz głośniejsze są ruchy promujące świadome i odpowiedzialne posiadanie zwierząt. 3. Możliwość współpracy z firmami zoologicznymi. | <ol style="list-style-type: none"> 1. Konkurencyjne strony mają mocną, budowaną od dawna pozycję. 2. Strona może być narażona na ataki ze strony konkurencji. 3. Mało czasu na zrealizowanie projektu. |

5.2.3. Opis czynników

5.2.3.1. Mocne strony:

1. Wyszukiwanie lepsze niż u konkurencji - aplikacja umożliwi wyszukiwanie psów na podstawie ich cech. Taka funkcjonalność jest niedostępna u konkurencji.
2. Przyjazny ilustrowany interfejs zachęci użytkowników - prosty interfejs umożliwi sprawne korzystanie z aplikacji osobom nie mającym dużego kontaktu z komputerem.
3. Developer dobrze zna wymagane technologie - dzięki doświadczeniu dewelopera implementacji aplikacji przebiegnie sprawnie.

5.2.3.2. Słabe strony:

1. Mniejsza ilość informacji na temat psów może sugerować potencjalnym użytkownikom, że strona jest gorsza - strony konkurencji mają bardzo szczegółowe informacje

na temat ras psów, przez co mniejsza ich ilość może dawać mylne wrażenie o jakości informacji.

2. Informowanie o problemach może odstraszyć klientów - konkurencyjne strony informację o wadach ras umieszczają zwykle na końcu całego opisu, przez co ludzie mogą nie zwracać na nie uwagi. W tej aplikacji informacje o wadach będą podkreślane, w wyniku czego ludzie mogą się zniechęcić do używania jej.
3. Brak wiedzy związanej z marketingiem może utrudnić promowanie produktu.

5.2.3.3. Szanse:

1. Posiadanie zwierząt jest popularne - wśród młodych osób istnieje rosnąca tendencja do odkładania rodzicielstwa na rzecz posiadania zwierząt.
2. Coraz głośniejsze ruchy promujące świadome i odpowiedzialne posiadanie zwierząt - odpowiedzialne posiadanie zwierząt promują zarówno weterynarze, jak i organizacje prozwierzęce.
3. Możliwość współpracy z firmami zoologicznymi - firmy zoologiczne chętnie współpracują z przedsiębiorstwami promującymi posiadanie zwierząt.

5.2.3.4. Zagrożenia:

1. Konkurencyjne strony mają mocną, budowaną od dawna pozycję - wyszukiwarka internetowa na początku będzie pozycjonowała wyżej konkurencyjne strony.
2. Strona może być narażona na ataki ze strony konkurencji - jeśli konkurencja uzna aplikację za zagrożenie dla swojej pozycji, może podejmować kroki mające na celu osłabienie jej pozycji poprzez przeprowadzenie ataków hakerskich lub kłamliwe oceny.
3. Mało czasu na zrealizowanie projektu - deweloper musi pracować pod presją czasu.

5.2.4. Wycena czynników

Następnym krokiem analizy SWOT jest określenie jak silny wpływ dana cecha ma na projekt. Wyceny czynników znajdują się w tabeli 3. oraz tabeli 4.

Tabela 3. Wycena wewnętrznych czynników analizy SWOT.

| Lp. | Mocne lub słabe strony | Wycena czynnika | | | |
|-----|--|-----------------|----|---------|----|
| | | Słabiej | | Mocniej | |
| | | -2 | -1 | +1 | +2 |
| 1 | Wyszukiwanie lepsze niż u konkurencji. | | | | +2 |
| 2 | Przyjazny ilustrowany interfejs zachęci użytkowników. | | | +1 | |
| 3 | Deweloper dobrze zna wymagane technologie. | | | | +2 |
| 4 | Mniejsza ilość informacji na temat psów może sugerować, że strona jest gorsza. | -2 | | | |
| 5 | Informowanie o problemach może odstraszyć użytkowników. | | -1 | | |
| 6 | Brak wiedzy związanej z marketingiem może utrudnić promowanie produktu. | | -1 | | |

Tabela 4. Wycena zewnętrznych czynników analizy SWOT.

| Lp. | Szanse lub zagrożenia | Wycena czynnika | | | | |
|-----|--|-----------------|----|---|----|----|
| | | -2 | -1 | 0 | +1 | +2 |
| 1 | Posiadanie zwierząt jest popularne. | | | | | +2 |
| 2 | Coraz głośniejsze ruchy promujące świadome i odpowiedzialne posiadanie zwierząt. | | | | +1 | |
| 3 | Możliwość współpracy z firmami zoologicznymi. | | | | | +2 |
| 4 | Konkurencyjne strony mają mocną, budowaną od dawna pozycję | -2 | | | | |
| 5 | Strona może być narażona na ataki ze strony konkurencji | | -1 | | | |
| 6 | Mało czasu na zrealizowanie projektu | | -1 | | | |

5.2.5. Wybór strategii

W celu wybrania strategii należy odpowiedzieć na określone pytania dla każdej mocnej i słabej strony, a następnie twierdzącą odpowiedź zaznaczyć przy odpowiednich cechach. Te pytania to:

1. Czy zidentyfikowana mocna strona pozwoli wykorzystać nadarzającą się szansę?
2. Czy zidentyfikowana mocna strona pozwoli przezwyciężyć zagrożenie?

3. Czy zidentyfikowana słaba strona nie pozwoli na wykorzystanie się szansy?
4. Czy zidentyfikowana słaba strona wzmocni siłę oddziaływania zagrożenia?

Przebieg procesu doboru strategii został przedstawiony w tabeli 5. Zbiorczy wynik jest zawarty w tabeli 6.

Tabela 5. Przebieg wybór strategii na podstawie uwzględnionych czynników.

| Wybór strategii | | | Otoczenie | | | | | |
|-----------------|--------------|---|-----------|---|---|------------|---|---|
| | | | Szanse | | | Zagrożenia | | |
| | | | 1 | 2 | 3 | 1 | 2 | 3 |
| Przedsięwzięcie | Mocne strony | 1 | X | X | | X | | |
| | | 2 | X | X | | | | |
| | | 3 | | | | | | X |
| | Słabe strony | 1 | X | X | | X | | |
| | | 2 | X | X | | X | | |
| | | 3 | | | X | | | |

Tabela 6. Wynik wyboru strategii.

| | Szanse | Zagrożenia |
|--------------|-----------------------------|----------------------------|
| Mocne strony | Strategia agresywna - 4 | Strategia konserwatywna -2 |
| Słabe strony | Strategia konkurencyjna - 5 | Strategia defensywna - 2 |

5.2.6. Podsumowanie

Na podstawie danych z tabeli 3. można zauważyć, że głównym zagrożeniem projektu jest dobrze zbudowana pozycja konkurencji na rynku, a co za tym idzie – ich dobre pozycjonowanie w silnikach wyszukiwania. Rozwiązaniem tego problemu może być dodatkowa inwestycja w kampanię marketingową, która zapewni ruch na stronie do momentu wzmocnienia pozycji.

Na podstawie tabeli 6. można stwierdzić, że najlepszą strategią dla naszej aplikacji będzie strategia konkurencyjna. Polega ona na koncentrowaniu się na szansach przy równoczesnym minimalizowaniu zagrożeń.

6. Projekt bazy danych

6.1. Analiza rzeczywistości

Rasa psa posiada szereg kluczowych cech określających jego parametry:

- Nazwa - identyfikuje rasę psa.
- Grupa - grupa psów, do których przynależy dana rasa. Wszystkie psy w grupie charakteryzują się podobnymi cechami.
- Rozmiar - każda rasa jest przypisana do jednego ze zdefiniowanych rozmiarów: mały, średni, duży i olbrzymi.
- Waga - średnia waga rasy.
- Trudność w utrzymaniu czystości - zależy między innymi od rodzaju i długości sierści oraz budowy psa. Przykładowo Spaniele posiadające długie zwisające uszy mają skłonność do gromadzenia w nich brudu.
- Trudność w tresowaniu - podatność na tresurę. Inteligentniejsze psy łatwiej wyszkolić, ale wpływ na ten proces ma również charakter.
- Długość sierści - średnia długość sierści rasy określona jako krótka, średnia bądź długa.
- Typ sierści - rasa posiada jeden ze zdefiniowanych typów sierści: puchata, gładka i szorstka.
- Długość życia - średnia długość życia.
- Koszt kupna - średnia cena kupna psa danej rasy.
- Koszt miesięcznego utrzymania psa - w dużej mierze mówi o wydatkach na wyżywienie.
- Podatność na choroby - opisuje jakie są częste choroby genetyczne na jakie zapada dana rasa psa. Im rasa jest bardziej pierwotna tym ma ich mniej.

Oprócz wyżej wymienionych parametrów każda rasa będzie posiadała również pełen opis oraz galerię zdjęć.

Dla czterech z wymienionych cech: waga, długość życia, koszt kupna oraz koszt utrzymania, zostanie zastosowane wyszukiwanie wykorzystujące zbiory rozmyte. Z tego powodu konieczne jest przechowywanie parametrów ich funkcji przynależności, które będą przechowywane w bazie danych.

6.2. Model bazy danych

6.2.1. Diagram ERD

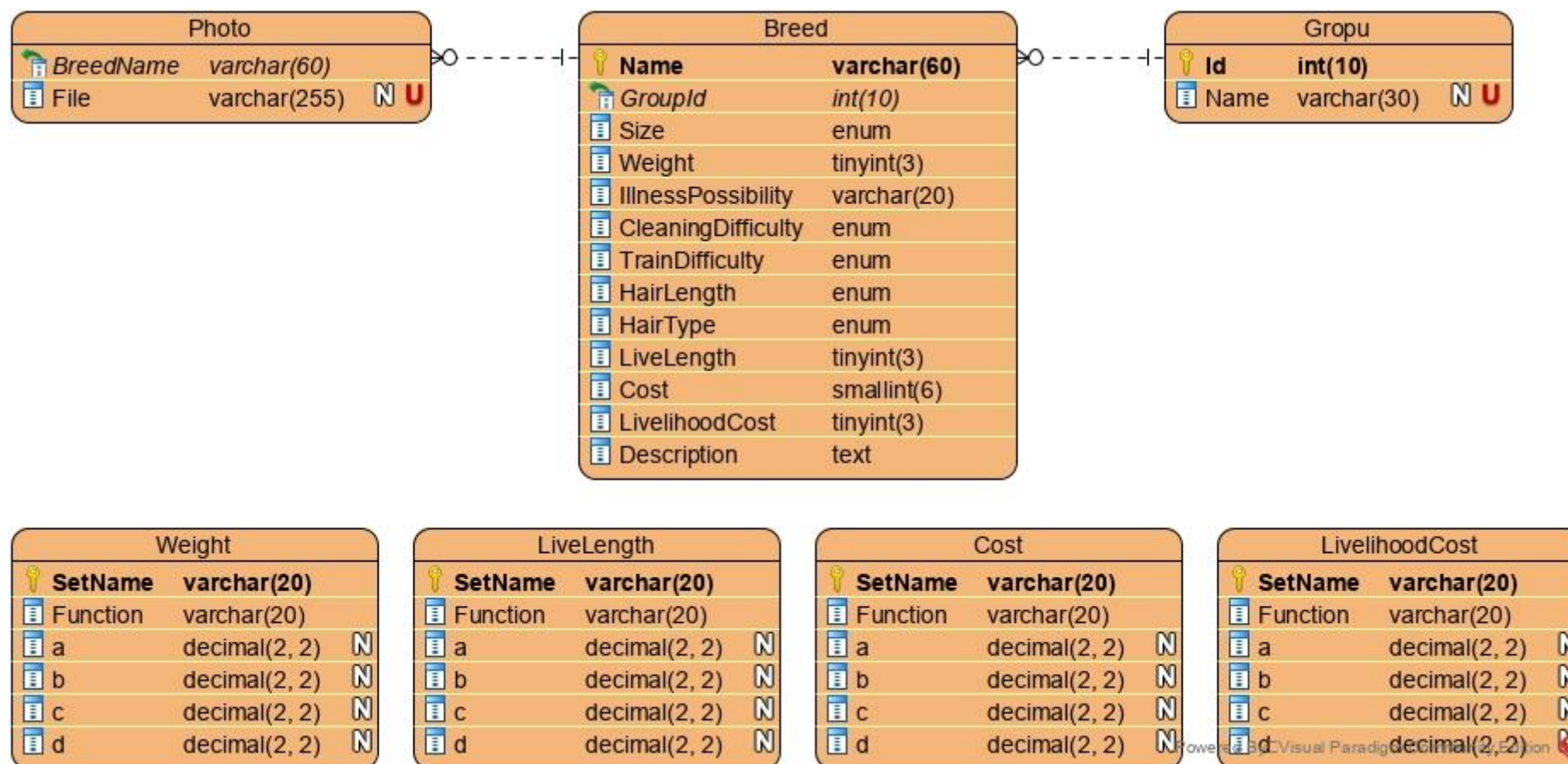
Diagram ERD bazy danych jest zaprezentowany na rysunku 6.

6.2.2. Opis tabel

- Breed - określa kluczowe cechy rasy psa opisane w sekcji 6.1 za wyjątkiem podatności na choroby. Zawiera również jej opis. Kluczem głównym tabeli jest nazwa rasy, która jednoznacznie ją identyfikuje.
- Photo - słaba encja posiadająca ścieżki do zdjęć rasy psa, do której się odnosi w kluczu obcym.
- Group - zawiera spis wszystkich zdefiniowanych przez związki kynologiczne grup psów.
- Weight, LiveLength, Cost, LivelihoodCost - posiadają parametry funkcji przynależności wraz z nazwą zbioru, którego dotyczą. W przypadku, gdy wykorzystywana funkcja przynależności nie wykorzystuje wszystkich dostępnych parametrów, te niewykorzystywane przyjmą wartość NULL. Każda z tabel dotyczy cechy wynikającej z jej nazwy i pokrywającej się z cechą ras z tabeli Breed.

6.2.3. Opis powiązań

- Tabele Breed i Photo - rasa psa może posiadać wiele zdjęć lub może nie mieć ich wcale. Zdjęcie musi dotyczyć jednej rasy.
- Tabele Breed i Group - rasa psa musi posiadać jedną grupę. Grupa psa może opisywać wiele ras, ale może też nie być przypisana do żadnej rasy.



Rysunek 6. Diagram ERD bazy danych

6.2.4. Mechanizmy przetwarzania danych

6.2.4.1. Widoki

Widok to zapytanie zapisane w bazie danych pozwalające pobierać rekordy ze zgromadzonych danych jak z tabeli. Dzięki temu mechanizmowi możliwe jest zdefiniowanie skomplikowanych zapytań w widoki, co upraszcza późniejszą pracę z bazą danych.

Dla przedstawionej bazy danych utworzony zostanie widok Breed_info prezentujący cechy psa z tabeli Breed ze sprecyzowaną grupą. Kolumny wchodzące w skład widoku zostały przedstawione w tabeli 7.

Tabela 7. Zestawienie danych wchodzących w skład widoku Breed Info

| Tabela | Breed | Group |
|---------|--|-------|
| Kolumny | Name, Size, Weight, CleaningDifficulty, TrainDifficulty, HairLength, HairType, LifeLength, Cost, LivelihoodCost, Description | Name |

6.2.4.2. Indeksy

W bazie danych MySQL zostanie użyty silnik InnoDB, który automatycznie tworzy indeksy na każdej kolumnie spełniającej podane kryteria:

- kolumna jest kluczem głównym,
- kolumna jest kluczem obcym,
- kolumna ma ograniczenie UNIQUE.

Dzięki tej własności wszystkie potrzebne indeksy zostaną utworzone automatycznie.

6.2.4.3. Kaskadowe usuwanie

Kaskadowe usuwanie występuje, gdy po usunięciu jednego rekordu trzeba usunąć inne z nim powiązane w celu utrzymania spójności bazy danych. W przypadku niniejszego projektu taka sytuacja występuje przy usuwaniu rekordów z tabeli Breed - usunięcie rekordu z tej tabeli pociąga za sobą usunięcie powiązanych z nim rekordów z tabeli Photo.

6.2.4.4. Mechanizmy bezpieczeństwa

Wykonywanie operacji na bazie danych wymaga konta użytkownika po jej stronie, do którego aplikacja będzie się logować podczas nawiązywania z nią połączenia. W celu zachowania środków ostrożności wspomniane konto powinno mieć tylko uprawnienia konieczne do zapewnienia poprawnego funkcjonowania aplikacji. W tym celu rozróżniamy dwa typy aktorów:

- Guest - ma możliwość pobierania informacji z bazy danych. Ponieważ aplikacja nie będzie posiadała wrażliwych informacji, nie ma restrykcji na dostęp do poszczególnych tabel i widoków.
- Admin - rozszerza uprawnienia aktora guest o możliwość dodawania, usuwania i aktualizowania informacji we wszystkich tabelach za wyjątkiem tabel Group, Weight, LifeLength, Cost, LivelihoodCost.

7. Projekt aplikacji

7.1. Architektura aplikacji

Do realizacji aplikacji została zastosowana architektura Model-Widok-Kontroler (MVC) [24]. Polega ona na oddzieleniu logiki aplikacji, warstwy prezentacji i komunikacji między nimi. Nazwa tego wzorca projektowego bezpośrednio odnosi się do wytyczonych zasad:

- Model - reprezentuje dane, na których pracuje aplikacja, komunikuje się z repozytoriami w celu ich pobierania oraz wysyłania.
- Widok - definiuje sposób prezentacji danych przetwarzanych przez aplikację w graficznym interfejsie użytkownika.
- Kontroler - reaguje na akcje użytkownika, przetwarza otrzymane dane, zarządza modelami i widokami.

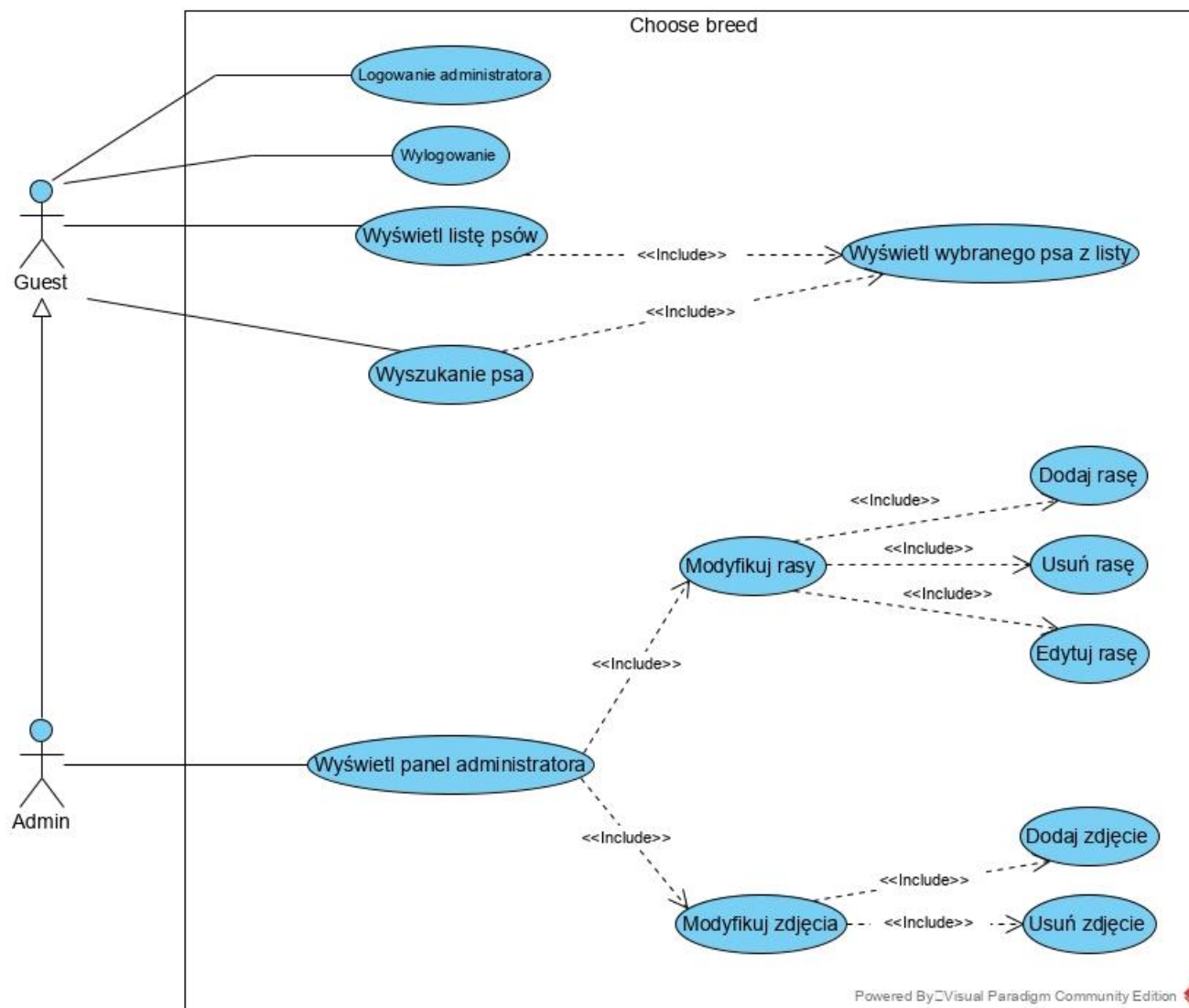
Sama logika zostanie napisana w sposób wielowarstwowy. Oznacza to, że każda z jej funkcjonalności, takich jak połączenie z bazą danych czy dokonywanie obliczeń, zostanie wyniesione do osobnych modułów, udostępniając jedynie interfejs do komunikacji między sobą.

7.2. Interfejs graficzny

Kluczowymi cechami interfejsu graficznego aplikacji będzie czytelność i prostota obsługi. Zostanie wykonany według wytycznych języka projektowania Material Design stworzonego przez firmę Google.

Z jego głównego menu będzie można przejść do panelu wyszukiwania ras, listy dostępnych ras oraz panelu logowania. W przypadku gdy użytkownik będzie zalogowany jako administrator, będzie miał również możliwość przejścia do panelu administratora.

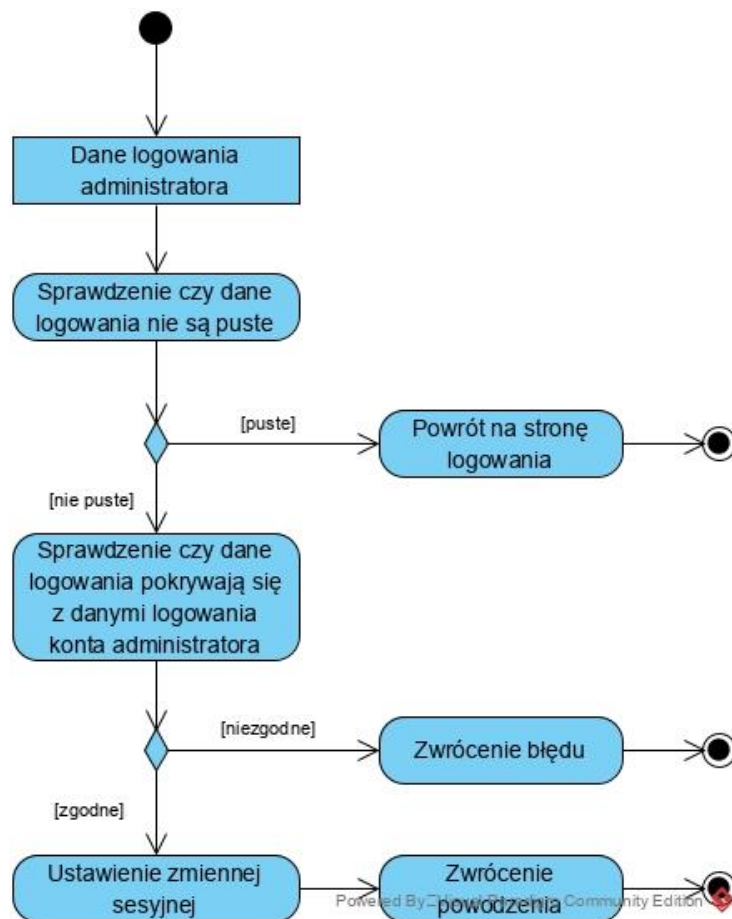
Możliwe działania, które mogą zostać podjęte przez aktorów aplikacji są przedstawione na diagramie przypadków użycia na rysunku 7.



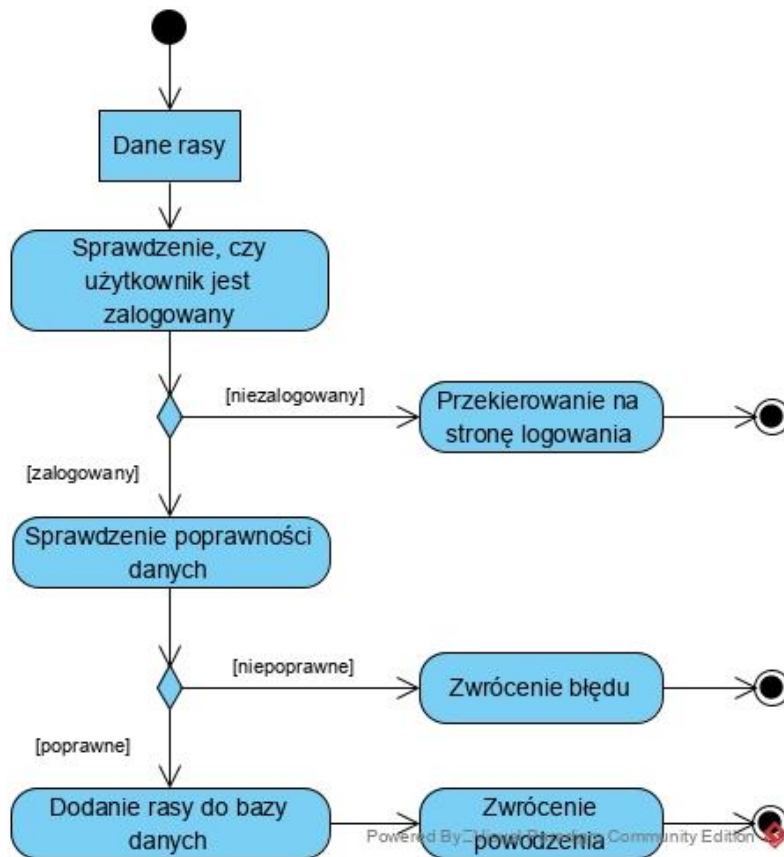
Rysunek 7. Diagram przypadków użycia

7.3. Specyfikacja wybranych funkcjonalności aplikacji

W celu przedstawienia specyfikacji wybranych funkcjonalności aplikacji zostaną użyte diagramy czynności. Rysunek 8. przedstawia proces logowania administratora. Rysunek 9. przedstawia proces dodawania rasy do bazy danych.



Rysunek 8. Diagram czynności logowania administratora



Rysunek 9. Diagram czynności dodawania rasy do bazy danych

7.4. Połączenie z bazą danych

Serwer bazy danych MySQL oraz serwer WWW obsługujący back-end aplikacji zostaną uruchomione na tej samej maszynie. Komunikacja między nimi odbędzie się z wykorzystaniem biblioteki Hibernate [25] dostępnej w frameworku Spring Boot. Wspiera on współpracę z wieloma systemami zarządzania bazą danych oraz umożliwia mapowanie obiektowo-relacyjne między utworzonymi obiektami w logice aplikacji a rekordami encji.

7.5. Zabezpieczenia na poziomie aplikacji

Dane logowania do konta administratora zostaną zapisane w pliku konfiguracyjnym logiki aplikacji. Hasło będzie zahashowane funkcją skrótu BCrypt [26], która została stworzona specjalnie z myślą o hashowaniu haseł statycznych

8. Implementacja i testowanie

8.1. Implementacja modelu i bazy danych

Model danych został w większości zaimplementowany wykorzystując podejście „code first”. Reprezentują go klasy Javy, dokładnie odwzorowujące encje przedstawione na diagramie ERD na rysunku 6. Ponadto, w celu zapewnienia poprawnych wartości dla kolumn typu ENUM zostały stworzone dodatkowe klasy typu wyliczeniowego, co zapobiegnie wprowadzeniu nieoczekiwanych wartości.

Stworzenie bazy danych sprowadziło się do wykorzystania techniki mapowania obiektowo-relacyjnego udostępnionego przez bibliotekę Hibernate. Aby ją zastosować, należało dodać do modelu stosowne adnotacje.

- `@Entity` – oznacza daną klasę jako encja. W jej argumencie można podać nazwę encji bazy danych, do której ma nastąpić mapowanie.
- `@Id` – oznacza dane pole klasy jako klucz główny encji. Hibernate wymaga, aby każda encja miała swój klucz główny.
- `@Column` – określa, że dane pole klasy jako kolumna encji. Dodatkowo można za jego pomocą ustawić różne opcje kolumny jak na przykład nazwę, możliwość przyjmowania wartości NULL, unikalność czy długość w przypadku tekstu.
- `@Enumerated` – umożliwia mapowanie klas typu ENUM.
- `@OneToMany/@ManyToOne/@ManyToMany` – określają połączenia między encjami. W jej argumencie można przekazać nazwę pola, które ma być odpowiedzialna za mapowanie. W przeciwnym wypadku Hibernate może zmapować relację dwukrotnie np.: tworząc dwie tabele łączące.
- `@OnDelete` – pozwala na określanie akcji przekazanej w argumencie, która zostanie podjęta podczas usuwania rekordu.

Problemem podczas mapowania okazało się stworzenie widoku, ponieważ Hibernate bezpośrednio nie wspiera takiej funkcjonalności. Ostatecznie rozwiązaniem zostało stworzenie widoku z poziomu bazy danych i mapowanie klasy reprezentującej widok na już istniejącą tabelę, co jest przykładem wykorzystania podejścia „database first”.

Listing 1. Fragment klasy modelu z adnotacjami ORM

```
@Entity
public class Breed {
    @Id
    @Column(name = "Name", length = 60)
    private String name;

    @Enumerated(EnumType.STRING)
    @Column(name = "Size", nullable = false, length = 20)
    private Size size;
    ...
    @OneToMany(mappedBy = "breed")
    @OnDelete(action = OnDeleteAction.CASCADE)
    private List<Photo> photos = new ArrayList<>();
}
```

Listing 2. Przykładowa klasa typu ENUM

```
public enum Size {
    MALY("Mały"),
    SREDNI("Średni"),
    DUZY("Duży"),
    OLBRZYM("Olbrzym");

    private String size;

    Size(String size) {
        this.size = size;
    }

    public String getSize() {
        return size;
    }
}
```

Listing 3. Tworzenie widoku

```
CREATE VIEW breeds_info AS
SELECT b.name AS name, g.name AS group_name, b.size AS size,
       b.weight AS weight, b.illness_possibility AS illness_possibility,
       b.cleaning_difficulty AS cleaning_difficulty, b.train_difficulty AS
       train_difficulty, b.hair_length AS hair_length, b.hair_type AS hair_type,
       b.live_length AS live_length, b.cost AS cost,
       b.livelihood_cost AS livelihood_cost, b.description AS description
FROM breed b INNER JOIN breed_group g ON b.breed_group_id = g.id
```

8.2. Implementacja kontrolerów

Aplikacja posiada dwa kontrolery: kontroler funkcjonalności użytkownika oraz kontroler funkcjonalności administratora.

Aby klasa Javy była postrzegana jako kontroler, wystarczy wykorzystać adnotację `@Controller` dostępną we frameworku Spring Boot. Dalej w kontrolerze, w celu określenia

endpointów obsługujących różne funkcjonalności, wykorzystuje się kolejne adnotacje, na przykład `@GetMapping` oraz `@PostMapping`, które w argumencie przyjmują ścieżki umożliwiające komunikację z nimi. Przekazywanie parametrów również odbywa za pomocą adnotacji:

- `@PathVariable` – określa parametr ukryty w ścieżce endpointu.
- `@RequestParam` – określa parametry przekazywane w adresie URL. W jej parametrze można określić, czy dany parametr jest wymagany. Posiada ona również funkcję konwertowania wielu wystąpień danego parametru w listę.
- `@ModelAttribute` – używane wraz z adnotacją `@PostMapping`. Przekształca ciało żądania HTTP [27] na obiekt Javy.

Jako parametr metody obsługującej endpoint można również podać klasę `Model` dostarczaną przez Springa. Umożliwia ona przekazanie danych do szablonów widoku.

Listing 4. Endpoint odpowiedzialny za obsługę żądania wyświetlenia informacji o rasie

```
@GetMapping("/breed/{name}")
public String showAllBreeds(@PathVariable String name, Model model){
    String nameDecoded;
    try {
        nameDecoded = URLDecoder.decode(name, "UTF-8");
    } catch (UnsupportedEncodingException e) {
        nameDecoded = name;
    }

    model.addAttribute("pageTitle", url);
    model.addAttribute("breed", breedInfoRepository.findByName(url).get(0));

    return "breed";
}
```

8.3. Implementacja widoków

Do implementacji widoków zostały wykorzystane szablony FreeMarkera. Posiada on zestaw instrukcji podobnych do znaczników HTML, które umożliwiają generowanie strony HTML na podstawie przekazanych danych. Odwoływanie się do danych odbywa się przez ich nazwy w konwencji `${nazwa_obiektu_z_danymi}`. Przy tworzeniu szablonów w przypadku tej aplikacji przydatna okazała się tylko instrukcja `<#list>`, która umożliwia iterację po elementach listy, oraz `<#include>`, która została wykorzystana do importowania nagłówka i stópki do głównych szablonów.

Oprócz szablonów generujących strony HTML zostały stworzone style CSS.

8.4. Komunikacja z bazą danych

Komunikacja z bazą danych odbywa się z użyciem repozytoriów dostępnych w JPA, która z kolei jest dostępna w frameworku Spring Boot.

Stworzenie repozytorium sprowadza się tylko to rozszerzenie własnego interfejsu przez interfejs `JpaRepository`, który jest interfejsem generycznym, przyjmującym zdefiniowaną w modelu encję oraz typ jej klucza głównego.

Interfejs repozytorium domyślnie dostarcza podstawowe funkcje umożliwiające komunikację z bazą danych, jak na przykład: `findAll`, `save` czy `deleteAll`. Inną wygodną funkcjonalnością repozytoriów jest automatyczna implementacja metod zadeklarowanych w odpowiedniej konwencji nazewnictwa. W przypadku, gdy zapytanie jest zbyt rozbudowane dla automatycznej implementacji, możliwe jest użycie adnotacji `@Query`, w której argumentem można podać własne zapytanie w języku SQL, a z kolei parametry zapytania podać w argumentach metody z adnotacją `@Param`.

Listing 5. Tworzenie repozytorium dla klasy `BreedsInfo`

```
public interface BreedInfoRepository extends JpaRepository<BreedsInfo, String>
{
    List<BreedsInfo> findByName(String name);
}
```

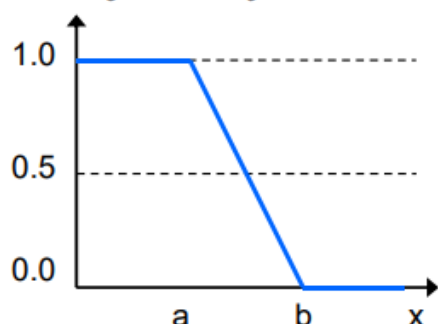
8.5. Logika aplikacji

Logika aplikacji ma za zadanie ocenić z użyciem logiki rozmytej stopień zgodności ras psów z oczekiwaniami wprowadzonymi przez użytkownika. W tym celu konieczne było wykorzystanie zbiorów rozmytych [28], które w formalny sposób określają pojęcia nieprecyzyjne. W naszym przypadku jest to określenie kosztu kupna, kosztu utrzymania i wieku rasy jako mały, średni lub duży oraz rozmiaru jako mały, średni, duży lub olbrzymi. Stopień, w którym dokładne parametry ras pokrywają się z tymi określeniami, jest definiowany przez funkcję przynależności.

Ocenianie ras również odbywa się na podstawie innych cech, które już nie są rozmyte, jak na przykład typ sierści. Ona, jak i wszystkie pozostałe mają z góry określone wartości, które mogą przyjąć. Z tego powodu są reprezentowane przez klasy typu `ENUM`. Ich ocena odbywa się w sposób zero-jedynkowy – albo rasa posiada daną wartość cechy albo nie.

Do realizacji logiki rozmytej w pracy zostały wybrane trzy podstawowe funkcje przynależności: dla określenia mały funkcja L, dla określenia duży lub ogromny w przypadku rozmiaru funkcja gamma, natomiast dla określeń pośrednich funkcja lambda.

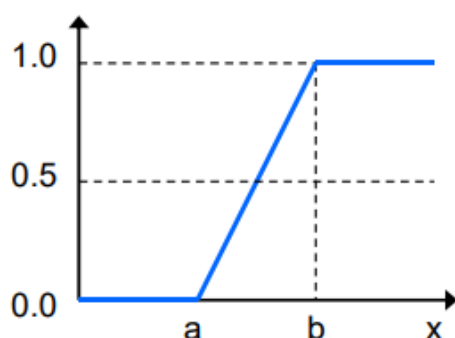
➤ funkcja klasy L



$$L_{a,b} = \begin{cases} 1 & \text{dla } x \leq a \\ (b-x)/(b-a) & \text{dla } a < x \leq b \\ 0 & \text{dla } x > b \end{cases}$$

Rysunek 10. Prezentacja funkcji L

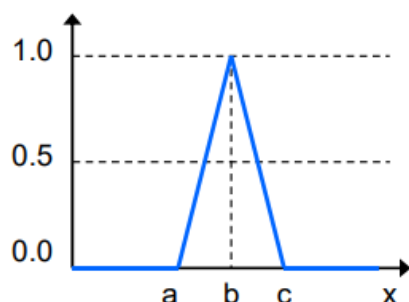
➤ funkcja klasy Γ (gamma)



$$\Gamma_{a,b} = \begin{cases} 0 & \text{dla } x \leq a \\ (x-a)/(b-a) & \text{dla } a < x \leq b \\ 1 & \text{dla } x > b \end{cases}$$

Rysunek 11. Prezentacja funkcji gamma

➤ funkcja klasy Λ (lambda)



$$\Lambda_{a,b,c} = \begin{cases} 0 & \text{dla } x \leq a \vee x \geq c \\ (x-a)/(b-a) & \text{dla } a < x \leq b \\ (c-x)/(c-b) & \text{dla } b < x < c \end{cases}$$

Rysunek 12. Prezentacja funkcji lambda

Logika aplikacji została napisana zgodnie z zasadami SOLID – rozszerzenie rozumowania rozmytego o nowe funkcje przynależności nie wymaga zmian w kodzie, a każda z klas wchodząca w skład logiki jest odpowiedzialna tylko za jej jedną funkcjonalność.

Samo obliczanie stopnia przynależności rasy do cechy sprowadza się do przedstawionych na rysunkach 10., 11. i 12. funkcji. W przypadku, gdy użytkownik zaznaczy więcej niż jedno określenie dla danej cechy, rasa zostanie oceniona na podstawie sumy zbiorów.

Listing 6. Funkcja obliczająca stopień przynależności rozmiaru rasy

```
public float checkWeight(BreedsInfo breedsInfo, List<String> weights){
    if(weights == null || weights.size() == 0)
        return 0;
    float maxScore = 0;
    float tempScore = 0;
    FuzzyParam chosenParam;
    for(String s : weights){
        chosenParam = getWeightParam(s);
        if(chosenParam != null){
            switch (chosenParam.getFunction()){
                case "L": tempScore = functionL(chosenParam.getA(),
                    chosenParam.getB(), breedsInfo.getWeight());
                    break;
                case "lambda": tempScore = functionLambda(chosenParam.getA(),
                    chosenParam.getB(), chosenParam.getC(),
                    breedsInfo.getWeight());
                    break;
                case "gamma": tempScore = functionGamma(chosenParam.getA(),
                    chosenParam.getB(), breedsInfo.getWeight());
                    break;
                default: tempScore = 0;
            }
        }
        if(tempScore > maxScore){
            maxScore = tempScore;
        }
    }
    return maxScore;
}
```

8.6. Testy jednostkowe

Testy jednostkowe objęły logikę aplikacji opisaną w poprzednim punkcie. Sprawdzają, czy dana rasa jest poprawnie oceniana na podstawie funkcji przynależności oraz wybranych przez użytkownika zbiorów oczekiwanych cech. Dzięki temu upewniliśmy się co do poprawności kluczowej funkcjonalności aplikacji, jaką jest poprawne ocenianie ras, a co za tym idzie – prezentowanie ich użytkownikowi w odpowiedniej kolejności.

Listing 7. Przykładowy tekst jednostkowy sprawdzający poprawność logiki rozmytej

```
@Test
void shouldReturnCorrectFunctionValue() {
    BreedsInfo breed = createBreed();

    List<FuzzyParam> costParams = new ArrayList<>();
    List<FuzzyParam> liveLengthParams = new ArrayList<>();
    List<FuzzyParam> livelihoodCostParams = new ArrayList<>();
    List<FuzzyParam> weightParams = new ArrayList<>();
    costParams.add(new FuzzyParam(createCostParam()));
    liveLengthParams.add(new FuzzyParam(createLiveLengthParam()));
    livelihoodCostParams.add(new FuzzyParam(createLivelihoodCostParam()));
    weightParams.add(new FuzzyParam(createWeightParam()));

    List<String> costs = new ArrayList<>();
    List<String> weights = new ArrayList<>();
    List<String> liveLengths = new ArrayList<>();
    List<String> livelihoodCosts = new ArrayList<>();
    costs.add("maly");
    weights.add("sredni");
    liveLengths.add("maly");
    livelihoodCosts.add("duzy");

    IndicatorFunctions indicatorFunctions = new IndicatorFunctions(costParams,
        liveLengthParams, livelihoodCostParams, weightParams);

    assertEquals(Math.round(indicatorFunctions.checkCost(breed, costs)*10), 5);
    assertEquals(Math.round(indicatorFunctions.checkWeight(breed,
        weights)*1000), 714);
    assertEquals(Math.round(indicatorFunctions.checkLivelihoodCost(breed,
        livelihoodCosts)*1000), 667);
    assertEquals(Math.round(indicatorFunctions.checkLiveLength(breed,
        liveLengths)), 0);
}
```

8.7. Testy akceptacyjne

Testy akceptacyjne zostały wykonane przy użyciu narzędzia Selenium IDE, dostępnego jako rozszerzenie przeglądarki internetowej. Polegały na nagraniu akcji użytkownika, które będą powtarzane przez narzędzie podczas uruchomienia testu. Ponadto, do każdego testu dodano dodatkową akcję sprawdzającą zgodność danych na stronie z oczekiwanymi wartościami, na przykład sprawdzenie, czy nazwa wybranej rasy jest zgodna z oczekiwaną.

| | | | |
|------------------------------------|-----------------------|---|-----------|
| Search tests... | http://localhost:8080 | | |
| ► Default Suite | | | |
| ▼ Search | | | |
| Search first small breed | | | |
| Search second big breed | | | |
| Search last breed with small c... | | | |
| ▼ Select | | | |
| Select third breed from all bre... | | | |
| | Command | Target | Value |
| 1 | open | / | |
| 2 | set window size | 968x1040 | |
| 3 | click | css=.grid__col--1-4:nth-child(2) > .checkbox__label | |
| 4 | click | css=.card__button | |
| 5 | click | css=.card:nth-child(2) .card__button | |
| 6 | click | css=.heading--2 | |
| 7 | assert text | css=.heading--2 | Chihuahua |

Rysunek 13. Zrzut ekranu z narzędzia Selenium IDE prezentujący przykładowy test

9. Podsumowanie

9.1. Efekt końcowy

Efektem pracy jest zaimplementowana aplikacja webowa wykorzystująca technologie bazodanowe i spełniająca wszystkie wymagania założone podczas procesu projektowania.

Jej główne funkcjonalności przeznaczone dla użytkownika to możliwość wyszukania ras względem wybranych cech oraz udostępnienie bazy wiedzy pozwalającej na zapoznanie się z informacjami na ich temat. Udało się również zaimplementować wymagane funkcje administratora, które są dostępne po zalogowaniu się do aplikacji. Umożliwiają one dodawanie, modyfikowanie i usuwanie ras oraz dodawanie i usuwanie ich zdjęć. Administrator nie posiada osobnego panelu do zarządzania rasami, a wszystkie podejmowane przez niego akcje są dostępne z widoku wszystkich ras. Zrealizowanie tych funkcjonalności pokryło wszystkie przypadki użycia określone na diagramie UML.

Zaprojektowana baza danych zaprezentowana na diagramie ERD zapewnia wszystkie dane potrzebne do poprawnego działania aplikacji. Została zaimplementowana przy pomocy mapowania obiektowo-relacyjnego dostępnego w bibliotece Hibernate. Mapowanie objęło zamianę klas Javy, reprezentujących model danych w zastosowanym wzorcu architektonicznym model-widok-kontroler, na encje w bazie danych. Jedynie widok w bazie danych zawierający wszystkie informacje na temat ras został stworzony poleceniem SQL w systemie zarządzania bazą danych, a utworzona tabela została połączona z klasą ją odzwierciedlającą również z wykorzystaniem ORM.

Funkcjonalność wyszukiwania psów wymaga oceny ras na podstawie wartości cech wybranych przez użytkownika. Część z nich, jak odporność na choroby czy trudność tresury, posiadają z góry narzucone możliwe wartości, dzięki czemu ocenianie psa na ich podstawie może następować w sposób zero-jedynkowy. Niektóre z cech jednak posiadają wartości liczbowe, które też należało przekształcić na ocenę. W tym celu zastosowano logikę rozmytą, która wartościom liczbowym przyporządkowuje określenia rozmyte. W ten sposób formularz wyszukiwania ras dodatkowo się uprościł, dając użytkownikowi możliwość wybrania wartości cech spośród określeń rozmytych zamiast wprowadzać przedział wartości liczbowych. Przykładowo cesze koszt zostały przyporządkowane trzy określenia rozmyte - mały, średni oraz duży, które dalej przy użyciu opisanych w pracy funkcji przynależności oceniają rasę. W przypadku, gdy

użytkownik wybrał wiele określeń rasa otrzymuje ocenę będącą maksymalną wartością spośród obliczonych funkcji.

Na podstawie ocen każda z ras otrzymuje procentową wartość, która określa, w jakim stopniu pasuje ona do wybranych przez użytkownika cech. Rasy na liście wyników są posortowane malejąco względem tej wartości.

Aplikacja zachowuje prostotę i czytelność oraz ogranicza prezentowane informacje o rasach do minimum, co było jednym z jej głównych założeń. Dzięki temu potencjalny użytkownik może szybko przyswoić poszukiwaną wiedzę, co odróżnia tę aplikację od konkurencyjnych portali, które dostarczają ją w dużych ilościach.

Prezentacja widoków aplikacji oraz sposobu działania znajduje się w dodatku A w formie instrukcji obsługi dla użytkownika.

9.2. Kierunki rozwoju

Aplikacja, mimo że spełnia wszystkie wymagania, może być dalej rozwijana. Istotnym elementem do rozszerzenia jest optymalny dobór funkcji przynależności oraz ich parametrów w celu ulepszenia systemu oceny psów. W tym celu należałoby przeprowadzić badania oraz skonsultować się z ekspertem posiadającym szeroka wiedzę na temat psów.

Aplikację można również rozszerzać o kolejne funkcjonalności, na przykład tworzenie i publikowanie artykułów, forum dyskusyjne o rasach, dodanie integracji z serwisem społecznościowym Facebook w celu logowania się za jego pomocą, dodawania komentarzy i publikowania na nim wybranej dla siebie rasy.

Literatura

- [1] Holandia oficjalnie zakazała hodowli buldogów francuskich i mopsów
<https://kochamyzwierzaki.pl/hodowla-mopsow-i-buldogow/> (01.12.2019)
- [2] Wprowadzenie, konsekwencje stosowania modelowania w projektach programistycznych (01.12.2019)
http://zofia.kruckiewicz.staff.iiar.pwr.wroc.pl/wyklady/IO_UML/Wyklad_1_INEK011.pdf
- [3] T. Connolly, C. Begg. Systemy baz danych Tom 1 Praktyczne metody projektowania implementacji i zarządzani, 2004
- [4] Strona główna portalu IASM
<https://www.iams.com> (01.12.2019)
- [5] Strona główna portalu Purina
<https://www.purina.com> (01.12.2019)
- [6] Strona główna portalu PETBARN
<https://www.petbarn.com.au> (01.12.2019)
- [7] Strona główna portalu dogtime
<https://dogtime.com> (01.12.2019)
- [8] Strona główna portalu PodajLape
<https://www.podajlape.pl> (01.12.2019)
- [9] What is HTML?
<https://www.yourhtmlsource.com/starthere/whatishtml.html> (01.12.2019)
- [10] TECH 101: THE ULTIMATE GUIDE TO CSS
<https://skillcrush.com/2012/04/03/css/> (01.12.2019)
- [11] What is Apache FreeMarker™?
<https://freemarker.apache.org/> (01.12.2019)
- [12] What is Java programming language?
<https://howtodoinjava.com/java/basics/what-is-java-programming-language/> (01.12.2019)
- [13] Spring Boot
<https://spring.io/projects/spring-boot> (01.12.2019)
- [14] MySQL
<https://searchoracle.techtarget.com/definition/MySQL> (01.12.2019)

- [15] IntelliJ IDEA
<https://www.jetbrains.com/idea/> (01.12.2019)
- [16] What is Maven?
<https://maven.apache.org/what-is-maven.html> (01.12.2019)
- [17] MVN Repository
<https://mvnrepository.com/> (01.12.2019)
- [18] MySQL Community Server
<https://dev.mysql.com/downloads/mysql/> (01.12.2019)
- [19] phpMyAdmin
<https://www.phpmyadmin.net/> (01.12.2019)
- [20] JUnit 5
<https://junit.org/junit5/> (01.12.2019)
- [21] Selenium
<https://selenium.dev/> (01.12.2019)
- [22] GitHub
<https://github.com/> (01.12.2019)
- [23] Analiza SWOT
<http://www.jaknapisac.com/analiza-swot/> (01.12.2019)
- [24] MVC Definition
<https://techterms.com/definition/mvc> (01.12.2019)
- [25] Hibernate
<https://hibernate.org/> (01.12.2019)
- [26] Kompendium bezpieczeństwa haseł – atak i obrona (część 1.)
<https://sekurak.pl/kompendium-bezpieczenstwa-hasel-atak-i-obrona/> (01.12.2019)
- [27] What is HTTP?
https://www.w3schools.com/whatis/whatis_http.asp (01.12.2019)
- [28] L.A. Zadeh. Fuzzy sets, Information and Control. 8, 3, 338–353, 1965
[https://doi.org/10.1016/S0019-9958\(65\)90241-X](https://doi.org/10.1016/S0019-9958(65)90241-X) (01.12.2019)

Dodatek A

Instrukcja obsługi użytkownika

Spis funkcjonalności

Aplikacja posiada funkcjonalności przeznaczone dla jej użytkowników oraz administratora.

1. Funkcjonalności użytkownika:
 - a. Wyszukiwanie ras
 - b. Dostęp do listy wszystkich ras
 - c. Dostęp do szczegółów rasy
 - d. Logowanie jako administrator
2. Funkcjonalności administratora
 - a. Dodawanie, edycja i usuwanie rasy
 - b. Dodawanie i usuwanie zdjęć
 - c. Wylogowanie

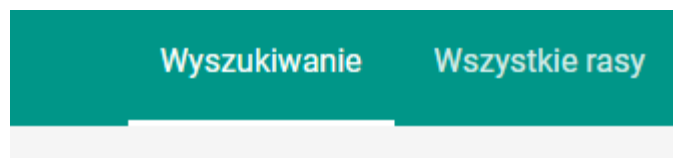
Funkcjonalności użytkownika

Wyszukiwanie ras

Formularz wyszukiwania ras jest dostępny na głównej stronie aplikacji pod adresem „/” oraz z poziomu każdej strony, ponieważ odnośnik do niej znajduje się w ich nagłówkach. Można na nim zaznaczyć wartości cech, które pies powinien posiadać, przy czym możliwe jest zaznaczenie wielu wartości oraz niezaznaczenie żadnej.

Po zaznaczeniu pól i naciśnięciu przycisku „Szukaj” zostaniemy przekierowani do listy wyników, gdzie każda pozycja zawiera nazwę rasy, procent zgodności jej cech z oczekiwanymi, oraz początek opisu. Po naciśnięciu na „Więcej” zostaniemy przekierowani do szczegółów rasy.

Rysunek 14. przedstawia odnośniki w nagłówku strony, 15. formularz wyszukiwania, a 16. wynik wyszukiwania.



Rysunek 14. Zrzut ekranu odnośników w nagłówku

Wyszukiwanie

Cechy

Rozmiar

☐ Mały

☐ Średni

☐ Duży

☐ Olbrzym

Skłonność do chorób

☐ Mała

☐ Średnia

☐ Duża

Długość życia

☐ Mała

☐ Średnia

☐ Duża

Koszt zakupu

☐ Mały

☐ Średni

☐ Duży

Koszt utrzymania

☐ Mały

☐ Średni

☐ Duży

Trudność czyszczenia

☐ Łatwo

☐ Średnio

☐ Trudno

Trudność tresury

☐ Łatwo

☐ Średnio

☐ Trudno

Długość sierści

☐ Krótka

☐ Średnia

☐ Długa

Typ sierści

☐ Gładka

☐ Puchata

☐ Szorstka

SZUKAJ

Rysunek 15. Zrzut ekranu formularza wyszukiwania rasy

Wyniki wyszukiwania

Labrador - 72%

W Wielkiej Brytanii i Stanach Zjednoczonych Labrador Retriever należy do najpopularniejszych ras psów. Także w wielu innych krajach europejskich przoduje w statystykach narodzin szczeniąt rasowych. Nic dziwnego, gdyż Labrador Retriever jak żadna inna rasa odznacza się wybitną lojalnością wobec człowieka oraz wrodzonym posłuszeństwem.

WIĘCEJ

Rysunek 16. Zrzut ekranu wyniku wyszukiwania ras

Dostęp do listy ras

Dostęp do listy ras jest możliwy pod adresem „/all” oraz tak jak formularz wyszukiwania poprzez odnośniki w nagłówku stron.

Każda pozycja listy zawiera nazwę psa, służącą również jako odnośnik do jego szczegółów, oraz wartości najistotniejszych cech.

Na rysunku 17. została przedstawiona lista wszystkich ras.

| Nazwa | Rozmiar | Skłonność do chorób | Długość życia | Cena [zł] |
|----------------------------------|---------|---------------------|---------------|-----------|
| Akita | Średni | Mała | 13 | 3 000 |
| Alaskan malamute | Duży | Średnia | 11 | 3 500 |
| Barbet | Średni | Średnia | 12 | 4 000 |
| Basset | Średni | Duża | 11 | 2 000 |
| Beagle | Średni | Średnia | 14 | 2 000 |
| Bernardyn | Olbrzym | Duża | 9 | 3 500 |

Rysunek 17. Zrzut ekranu listy wszystkich ras

Dostęp do szczegółów rasy

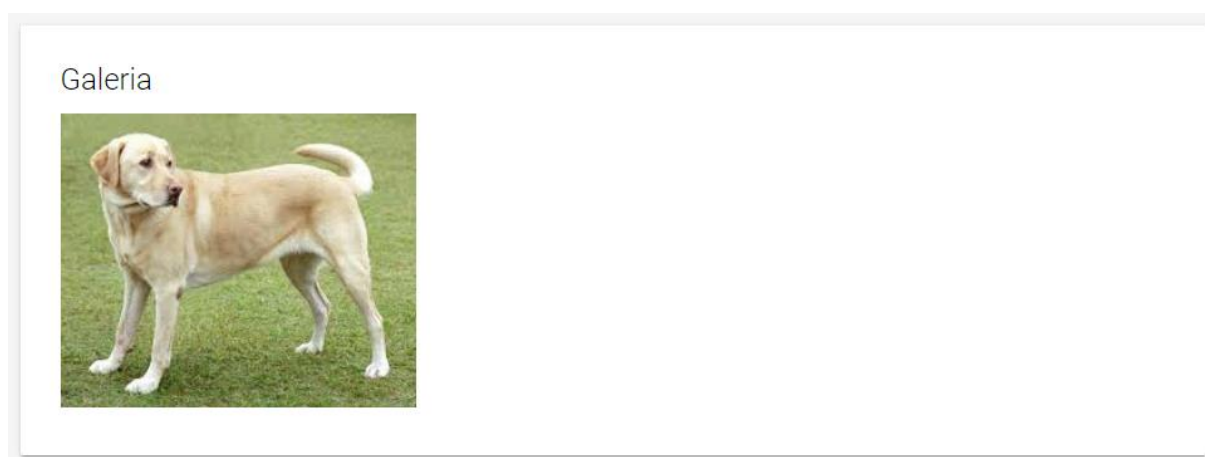
Dostęp do szczegółów danej rasy możliwy jest poprzez odnośniki dostępne w liście wyników wyszukiwania oraz w liście wszystkich ras.

Strona ze szczegółami zawiera nazwę rasy, jej pełny opis, spis wszystkich cech oraz galerię zdjęć.

Rysunek 18. przedstawia szczegóły danej rasy, a rysunek 19. jej galerię zdjęć.

| Opis | Cechy | | | | | | | | |
|---|---|-------|---------|-------|------------------------------|---------|------|------|----|
| <p>W Wielkiej Brytanii i Stanach Zjednoczonych Labrador Retriever należy do najpopularniejszych ras psów. Także w wielu innych krajach europejskich przoduje w statystykach narodzin szczeniąt rasowych. Nic dziwnego, gdyż Labrador Retriever jak żadna inna rasa odznacza się wybitną lojalnością wobec człowieka oraz wrodzonym posłuszeństwem. Jego naturalna potrzeba przypodobania się właścicielowi czyni z niego kompana niezwykle łatwego do ułożenia, wykazującego gotowość do współpracy i zdolność do przystosowania się. Labradora bez problemu można zabrać ze sobą wszędzie i wszędzie raczej będzie mile widzianym gościem. Obcych wita przyjaźnie, machając radośnie ogonem. To ciekawska, otwarta rasa, która najlepiej czuje się w towarzystwie.</p> | <table><thead><tr><th>Cecha</th><th>Wartość</th></tr></thead><tbody><tr><td>Grupa</td><td>Aportery, płochacze, dowodne</td></tr><tr><td>Rozmiar</td><td>Duży</td></tr><tr><td>Waga</td><td>35</td></tr></tbody></table> | Cecha | Wartość | Grupa | Aportery, płochacze, dowodne | Rozmiar | Duży | Waga | 35 |
| Cecha | Wartość | | | | | | | | |
| Grupa | Aportery, płochacze, dowodne | | | | | | | | |
| Rozmiar | Duży | | | | | | | | |
| Waga | 35 | | | | | | | | |

Rysunek 18. Zrzut ekranu szczegółów rasy



Rysunek 19. Zrzut ekranu galerii zdjęć rasy

Logowanie administratora

Formularz logowania dostępny jest pod adresem „/login” i nie prowadzi do niego żaden odnośnik ze strony.

W formularzu należy podać login oraz hasło administratora, po czym zatwierdzić je przyciskiem „Zaloguj”. Jeśli wpisane dane są poprawne zostaniemy przekierowani na stronę z listą wszystkich ras, w przeciwnym wypadku dostaniemy powiadomienie o błędnych danych.

Formularz logowania jest przedstawiony na rysunku 20.

The screenshot shows a login form titled "Logowanie" in a large, blue font. Below the title are two input fields. The first field is labeled "Użytkownik" in a small, gray font. The second field is labeled "Hasło" in a small, gray font. At the bottom of the form is a button labeled "ZALOGUJ" in a bold, teal font. The entire form is enclosed in a light gray border.

Rysunek 20. Zrzut ekranu formularza logowania

Funkcjonalności administratora

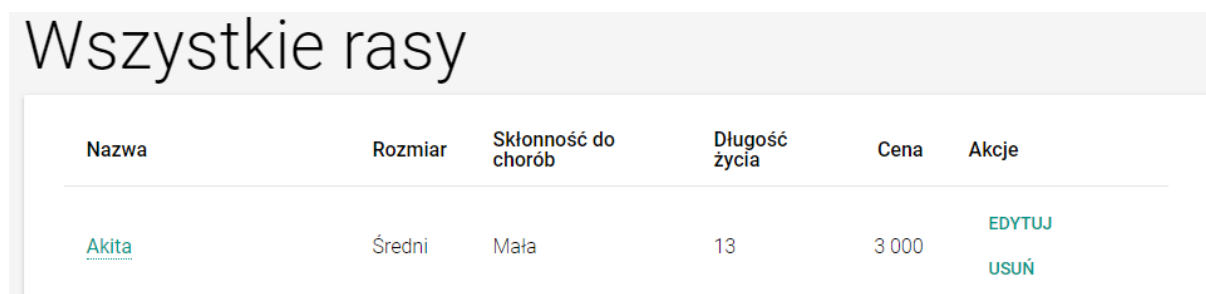
Dodawanie, edycja i usuwanie rasy

Dodawanie, edytowanie oraz usuwanie ras jest możliwe z poziomu listy wszystkich ras po zalogowaniu do aplikacji. Odnośniki do akcji edycji oraz usunięcia danej rasy jest dostępne są w dodatkowej kolumnie. Odnośnik do dawania rasy znajduje się na dole listy. Rysunek 21. przedstawia odnośniki edycji oraz usuwania rasy, a rysunek 22. odnośnik dodawania.

Formularz dodawani psa jest dostępny po naciśnięciu napisu „Dodaj”, który jest widoczny na rysunku 23. Umożliwia on wprowadzenie wszystkich danych rasy. Naciśnięcie przycisku „Zapisz” zapisuje rasę. Pola o ograniczonej puli wartości, jak na przykład rozmiar rasy, są wybieralne z listy, natomiast pozostałe trzeba wpisać ręcznie.

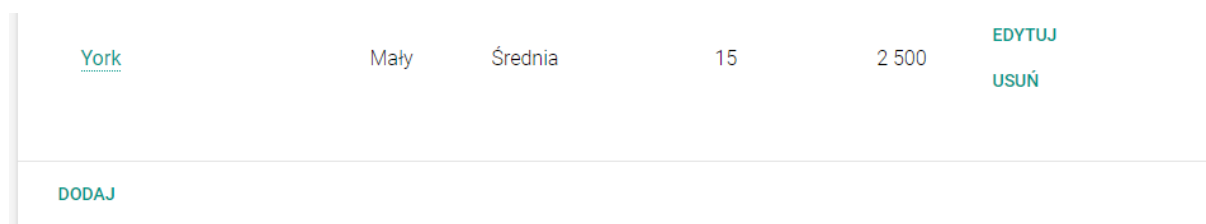
Formularz edycji jest analogiczny do formularza dodawania rasy, przy czym pole nazwy rasy jest nieedytowalne oraz możliwe jest z jego poziomu dodawanie oraz usuwanie zdjęć. Po wejściu w edycję wszystkie pola zostaną automatycznie uzupełnione o aktualne wartości.

Usunięcie rasy sprowadza się tylko na naciśnięcia przycisku „Usuń”.



| Nazwa | Rozmiar | Skłonność do chorób | Długość życia | Cena | Akcje |
|-----------------------|---------|---------------------|---------------|-------|--|
| Akita | Średni | Mała | 13 | 3 000 | EDYTUJ USUŃ |

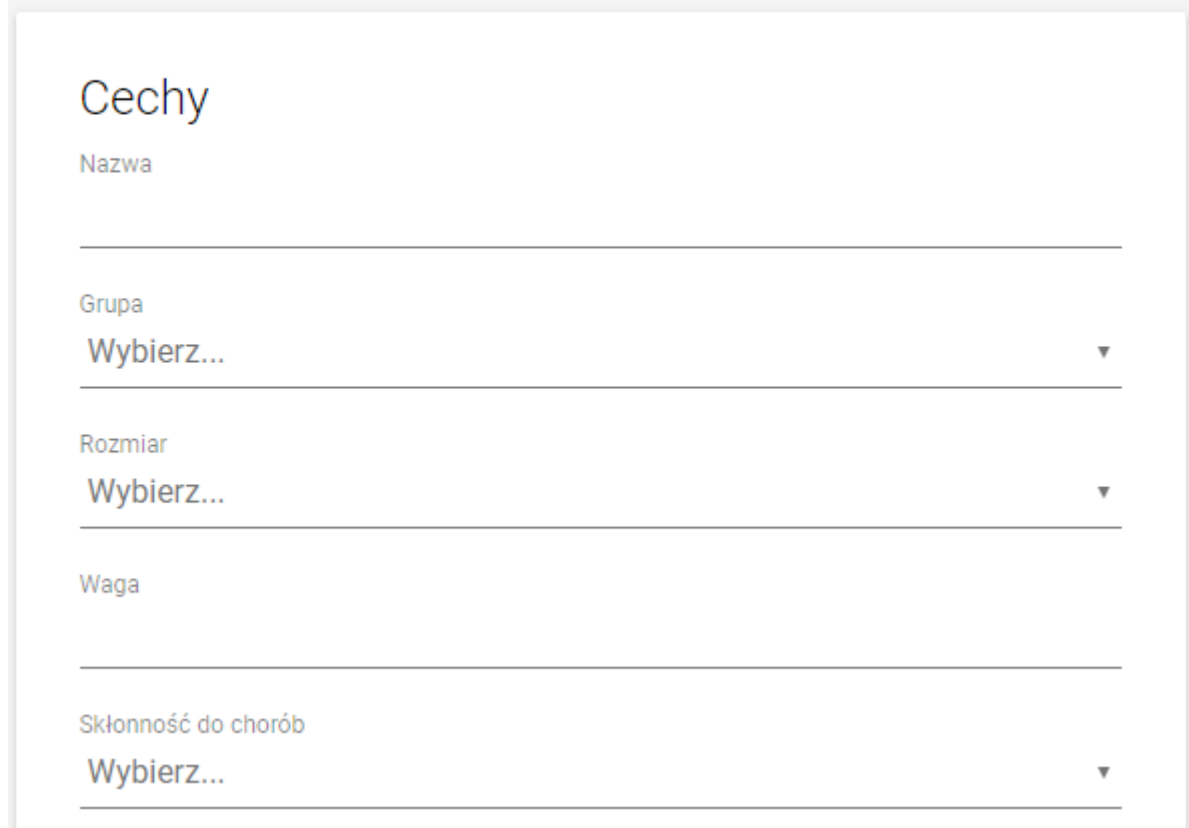
Rysunek 21. Zrzut ekranu odnośników edycji oraz usuwania rasy



| | | | | | |
|-----------------------|------|---------|----|-------|--|
| York | Mały | Średnia | 15 | 2 500 | EDYTUJ USUŃ |
| DODAJ | | | | | |

Rysunek 22. Zrzut ekranu odnośnika dodawani rasy

Dodawanie



Cechy

Nazwa

Grupa

Wybierz... ▼

Rozmiar

Wybierz... ▼

Waga

Sklonność do chorób

Wybierz... ▼

Rysunek 23. Zrzut ekranu formularza dodawania rasy

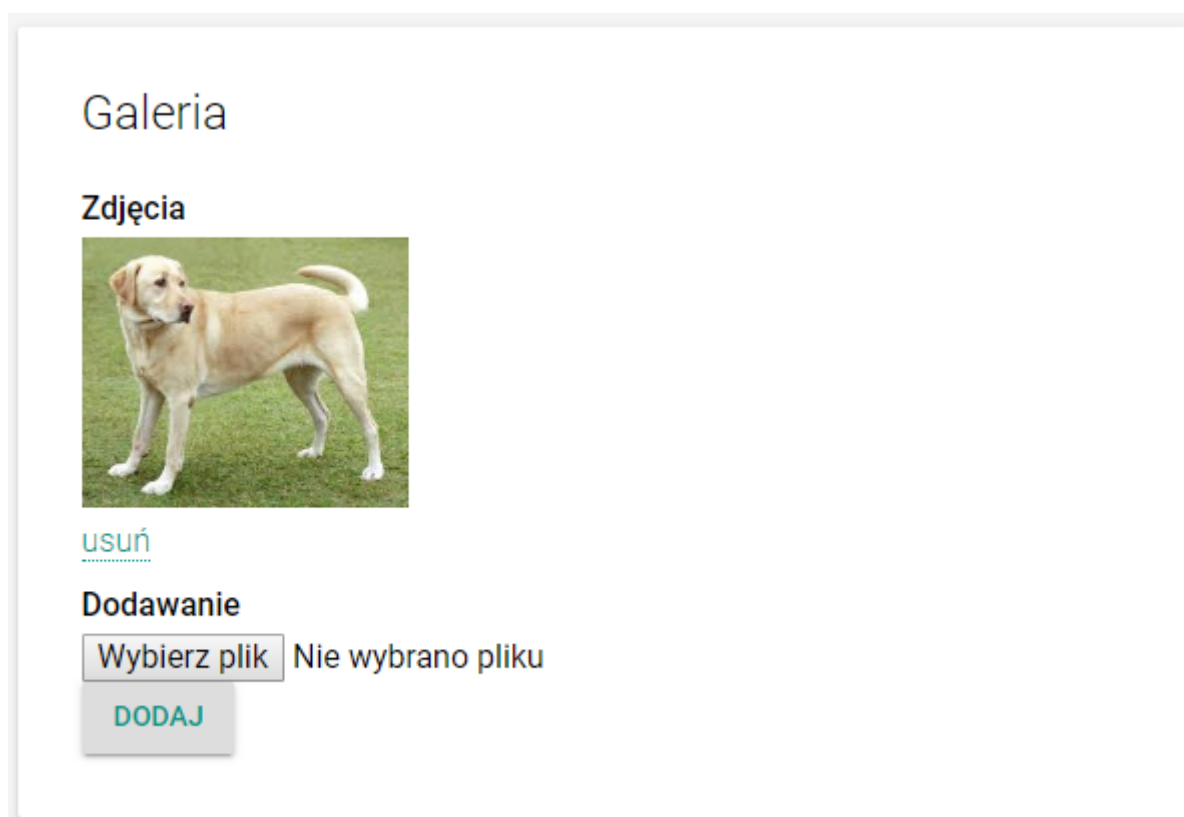
Dodawanie i usuwanie zdjęć

Dodawanie oraz usuwanie zdjęć możliwe jest z poziomu galerii zdjęć dostępnej na dole strony z formularzem edycji rasy.

Dodawanie zdjęcia polega na wybraniu pliku z formularza dostępnego po naciśnięciu przycisku „Wybierz plik”, a następnie zatwierdzeniu go przyciskiem „Dodaj”. Możliwe jest dodanie tylko jednego zdjęcia jednocześnie.

Usuwanie zdjęć, podobnie jak usuwanie ras, sprowadza się do naciśnięcia przycisku „Usuń” znajdującego się pod danym zdjęciem.

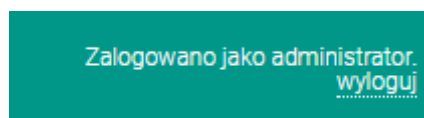
Galeria zdjęć z formularza edycji jest widoczna na rysunku 24.



Rysunek 24. Zrzut ekranu galerii zdjęć z formularza edycji rasy

Wylogowanie

Wylogowanie się z aplikacji spowoduje utratę możliwości podejmowania akcji przeznaczonych dla administratora. Przycisk służący do wylogowania pojawia się po zalogowaniu w stopce strony. Rysunek 25. przedstawia przycisk wylogowania.



Rysunek 25. Zrzut ekran przycisku wylogowania

Dodatek B

Opis dołączonej płyty CD

