

AI1	Dokumentacja projektu
Autor	Piotr Smoła, 125162
Kierunek, rok	Informatyka, II rok, st. stacjonarne (3,5-I)
Temat projektu	<i>Schronisko dla zwierząt</i>

Spis treści

1. Wstęp	3
Zaimplementowane funkcjonalności	3
Założenia Techniczne	4
Fragment rzeczywistości objęty działaniem aplikacji	4
2. Narzędzia i technologie	5
Framework Laravel	5
Baza danych	5
Konfiguracja połączenia z serwerem bazodanowym:	5
Frontend	6
Middleware	6
Kontrolery	7
Modele	7
Walidacja Danych	8
Seedowanie Danych	8
3. Baza danych	9
Diagram ERD	9
Opis rozwiązań zastosowanych w bazie	9
Migracje	9
Seedery	10
Opis Przykładowej Relacji w Modelach	11
4. GUI	12
Opis Interfejsu Użytkownika	12
Wybrane Widoki Aplikacji	12
5. Uruchomienie aplikacji	17
Wymagania	17
Kroki potrzebne do wykonania przy pierwszym uruchomieniu aplikacji	17
Kroki potrzebne do wykonania przy kolejnych uruchomieniach aplikacji	18
6. Funkcjonalności Aplikacji	19

7. Walidacja danych.....	34
Walidacja dla różnych typów żądań.....	34

1. Wstęp

Projekt dotyczy stworzenia aplikacji internetowej dla schroniska dla zwierząt przy użyciu frameworka Laravel. Aplikacja ma na celu usprawnienie zarządzania schroniskiem, umożliwiając pracownikom i administratorom efektywne zarządzanie zwierzętami, ich adopcjami oraz szczepieniami. Ponadto, aplikacja pozwala potencjalnym adopcyjnym rodzicom na przeglądanie dostępnych zwierząt, rejestrację oraz składanie wniosków o adopcję.

Zaimplementowane funkcjonalności

- I. **Rejestracja i logowanie użytkowników:**
 - Tylko zarejestrowani i zalogowani użytkownicy mogą korzystać z pełnych funkcji aplikacji, takich jak składanie wniosków o adopcję.
 - Nowi użytkownicy mogą samodzielnie się zarejestrować, a po rejestracji uzyskują dostęp do swojego konta.
- II. **Zarządzanie zwierzętami przez administratora:**
 - Administrator ma możliwość dodawania, edytowania oraz usuwania zwierząt w bazie danych.
 - Administrator zarządza również adopcjami oraz szczepieniami zwierząt, a także zmianą danych użytkowników czy ich usunięciem.
- III. **Automatyzacja procesów biznesowych:**
 - Aplikacja automatycznie oblicza status adopcji zwierząt oraz umożliwia śledzenie historii szczepień.
 - Obsługa zmiany dostępności zwierząt oraz inne operacje związane z zarządzaniem schroniskiem są wykonywane automatycznie.
- IV. **Funkcjonalności dla użytkowników:**
 - Użytkownicy mogą przeglądać dostępne zwierzęta, korzystając z różnych filtrów i sortowań.
 - Mogą składać wnioski o adopcję oraz zarządzać swoimi wnioskami.
 - Użytkownicy mają dostęp do swojego profilu, gdzie mogą edytować swoje dane oraz przeglądać historię adopcji.
- V. **Interfejs użytkownika:**
 - Aplikacja posiada responsywny interfejs użytkownika, zaprojektowany z użyciem silnika szablonów Blade.
 - Interfejs zawiera różnorodne elementy, takie jak formularze, przyciski, listy oraz dynamiczne elementy, które poprawiają doświadczenie użytkownika.
- VI. **Bezpieczeństwo:**
 - Aplikacja implementuje mechanizmy uwierzytelniania i autoryzacji, zapewniając bezpieczeństwo danych użytkowników.
 - Wszystkie wrażliwe operacje są zabezpieczone, a dostęp do funkcji administracyjnych jest ograniczony do uprawnionych użytkowników.

Założenia Techniczne

- **Framework:** Laravel 11.x
- **Baza danych:** MySQL
- **Frontend:** Szablony Blade, CSS, JavaScript
- **Backend:** PHP + Laravel
- **Inne technologie:** Bootstrap 5 do stylizacji elementów interfejsu

Aplikacja "Pet Shelter" jest zaprojektowana tak, aby zapewnić wygodę użytkowania zarówno dla klientów, jak i administratorów. Automatyzacja wielu procesów biznesowych znacznie upraszcza zarządzanie zwierzętami i procesami adopcyjnymi, co przekłada się na efektywność działania całego systemu.

Fragment rzeczywistości objęty działaniem aplikacji

Aplikacja obejmuje działalność schroniska dla zwierząt, które zajmuje się opieką nad bezdomnymi zwierzętami, szczepieniami oraz poszukiwaniem dla nich nowych domów. Fragment rzeczywistości, który jest objęty działaniem aplikacji, to cały proces związany z przyjmowaniem zwierząt do schroniska, ich rejestracją, zarządzaniem informacjami o nich, procesem adopcji, a także rejestrowaniem i zarządzaniem szczepieniami. Użytkownik aplikacji, będący pracownikiem schroniska, może dodawać nowe zwierzęta, aktualizować ich informacje, zarządzać procesem adopcji, a także rejestrować szczepienia. Zarejestrowany użytkownik (potencjalny adopcyjny właściciel) może przeglądać dostępne zwierzęta, składać wnioski o adopcję oraz zarządzać swoim profilem.

2. Narzędzia i technologie

Framework Laravel

Laravel to framework aplikacji webowych oparty na języku PHP. Został on stworzony przez amerykańskiego programistę - Taylora Otwell. Jest to zaawansowane narzędzie, które umożliwia efektywne tworzenie aplikacji webowych, wykorzystując przy tym nowoczesne techniki programowania oraz wzorce projektowe, co ułatwia tworzenie czytelnego i łatwego w utrzymaniu kodu.

W projekcie wykorzystujemy Laravel w wersji 11.x, który oferuje liczne funkcjonalności i usprawnienia, przede wszystkim wydajnościowe, w porównaniu do poprzednich wersji. Laravel jest udostępniany na podstawie licencji MIT, co oznacza, że możemy swobodnie korzystać, modyfikować i rozpowszechniać framework. Oficjalna dokumentacja Laravel jest dostępna na stronie Laravel Docs, gdzie można znaleźć szczegółowe informacje na temat funkcji, klas, interfejsów i sposobu użycia różnych komponentów frameworka.

Baza danych

Aplikacja wykorzystuje silnik bazodanowy MySQL. MySQL jest popularnym oprogramowaniem typu open source, dostępnym na licencji GNU General Public License. Jest to lekki oraz łatwy w użytkowaniu system bazodanowy zawierający liczne funkcjonalności. Każdy ma możliwość pobrania najnowszej wersji MySQL ze strony MySQL Downloads.

Konfiguracja połączenia z serwerem bazodanowym:

Aby skonfigurować połączenie z serwerem bazodanowym MySQL, należy edytować plik konfiguracyjny `.env` oraz `config/database.php`. W pliku `database.php` znajduje się sekcja `connections`, w której można zdefiniować ustawienia połączenia dla różnych środowisk (np. `development`, `production`). Jako przykładowa konfiguracja dostarczone są ustawienia umożliwiające połączenie się z każdą bazą danych wspieraną przez Laravel

Konfiguracja połączenia z bazą danych MySQL w pliku `database.php`:

```
'mysql' => [
    'driver' => 'mysql',
    'url' => env('DB_URL'),
    'host' => env('DB_HOST', '127.0.0.1'),
    'port' => env('DB_PORT', '3306'),
    'database' => env('DB_DATABASE', 'laravel'),
    'username' => env('DB_USERNAME', 'root'),
    'password' => env('DB_PASSWORD', ''),
    'unix_socket' => env('DB_SOCKET', ''),
    'charset' => env('DB_CHARSET', 'utf8mb4'),
    'collation' => env('DB_COLLATION', 'utf8mb4_unicode_ci'),
    'prefix' => '',
    'prefix_indexes' => true,
    'strict' => true,
    'engine' => null,
    'options' => extension_loaded('pdo_mysql') ? array_filter([
        PDO::MYSQL_ATTR_SSL_CA => env('MYSQL_ATTR_SSL_CA'),
    ]) : [],
],
```

W powyższym kodzie należy ustawić odpowiednie wartości dla parametrów host, port, database, username i password, zgodnie z konfiguracją Twojego serwera bazodanowego MySQL.

W przypadku pliku '.env' to służy on do przechowywania konfiguracyjnych zmiennych środowiskowych, które są specyficzne dla danego środowiska aplikacji, takich jak klucze API, ustawienia bazy danych oraz inne poufne informacje.

Konfiguracja połączenia z bazą danych MySQL w pliku .env:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=petsShelter
DB_USERNAME=root
DB_PASSWORD=
```

W powyższym kodzie należy ustawić odpowiednie wartości dla parametrów DB_CONNECTION, DB_HOST, DB_PORT, DB_DATABASE, DB_USERNAME i DB_PASSWORD, zgodnie z konfiguracją Twojego systemu bazodanowego MySQL. Ustawiając te zmienne w pliku .env, aplikacja Laravel będzie mogła poprawnie nawiązać połączenie z bazą danych MySQL, korzystając z tych ustawień.

Frontend

Aplikacja wykorzystuje Laravel Blade jako silnik szablonów do generowania widoków HTML. Blade jest stosunkowo prosty w użyciu, a jednocześnie pozwalając na wykorzystanie pełnych możliwości PHP bez komplikacji związanych z innymi silnikami szablonów. Szablony Blade w połączeniu z CSS i JavaScript, zapewnia dynamiczny, intuicyjny oraz responsywny interfejs użytkownika.

Middleware

Middleware w Laravelu służy do wykonywania różnych zadań pośrednich w trakcie przetwarzania żądań HTTP. W projekcie używamy middleware do sprawdzania uprawnień użytkowników, na przykład, czy użytkownik jest administratorem, czy ma odpowiednie role do wykonywania określonych działań.

```
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class IsAdmin
{
    public function handle(Request $request, Closure $next)
    {
        if (Auth::check() && Auth::user()->role == 'admin') {
            return $next($request);
        }
        abort(403, 'Unauthorized action.');
```

Kontrolery

Kontrolery w Laravelu organizują logikę aplikacji i odpowiadają za obsługę żądań HTTP. Przykładem może być kontroler 'AdoptionController', odpowiedzialny za realizację procesu adopcji i rezerwacji zwierząt.

```
class AdoptionController extends Controller
{
    public function adopt($id)
    {
        $pet = Pet::findOrFail($id);

        if (Auth::check()) {
            Adoption::create([
                'pet_id' => $pet->id,
                'customer_id' => Auth::id(),
                'adoption_date' => now(),
                'status' => 'completed'
            ]);

            return redirect()->route('user.profile')->with('success', 'You have adopted a pet');
        }

        return redirect()->route('login');
    }

    public function reserve($id)
    {
        $pet = Pet::findOrFail($id);

        if (Auth::check()) {
            Adoption::create([
                'pet_id' => $pet->id,
                'customer_id' => Auth::id(),
                'adoption_date' => now(),
                'status' => 'reserved'
            ]);

            return redirect()->route('user.profile')->with('success', 'You have reserved a pet');
        }

        return redirect()->route('login');
    }
}
```

Modele

Modele w Laravelu odpowiadają za interakcje z bazą danych. Każdy model jest odpowiedzialny za jedną tabelę w bazie danych. Przykładowo, model Pet odpowiada za operacje na tabeli pets, model User za operacje na tabeli users, a model Adoption za operacje na tabeli adoptions. Przykładowy model:

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Pet extends Model
{
    use HasFactory;

    protected $table = 'pets';

    protected $primaryKey = 'id';

    protected $fillable = [
        'name',
        'species',
        'breed',
        'age',
        'weight',
        'photo_path',
        'description',
    ];

    public $timestamps = false;

    public function adoptions()
    {
        return $this->hasMany(Adoption::class, 'pet_id');
    }

    public function vaccinations()
    {
        return $this->hasMany(Vaccination::class, 'pet_id');
    }
}
```

Walidacja Danych

Laravel oferuje prosty sposób walidacji danych za pomocą form requestów. Walidacja jest wykonywana zarówno po stronie serwera, jak i po stronie klienta, aby zapewnić, że dane wprowadzane przez użytkowników są poprawne i bezpieczne. Przykładowo, walidacja formularza rejestracji użytkownika sprawdza, czy adres email jest unikalny, czy hasło ma odpowiednią długość i czy pola nie są puste. Przykład walidacji dla dodawania nowego zwierzęcia do bazy:

```
public function storePet(Request $request)
{
    $request->validate([
        'name' => 'required|string|max:20',
        'species' => 'required|string|max:20',
        'breed' => 'required|string|max:50',
        'age' => 'required|integer',
        'weight' => 'required|numeric',
        'description' => 'required|string|max:250',
        'photo' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
    ]);
}
```

Seedowanie Danych

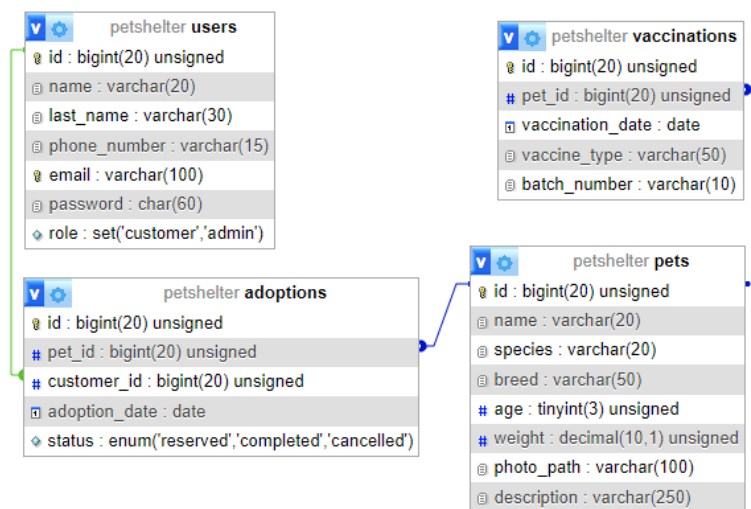
W celu załadowania przykładowych danych do bazy, używamy seederów. Seedery w Laravelu pozwalają na programowe wypełnienie bazy danych przykładowymi rekordami. Jest to szczególnie przydatne podczas testowania aplikacji i w fazie developmentu. Przykład seedera:

```
class PetsSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        DB::table('pets')->insert([
            [
                'name' => 'Max',
                'species' => 'Dog',
                'breed' => 'Labrador',
                'age' => 3,
                'weight' => 24.5,
                'photo_path' => 'photos/max.jpg',
                'description' => 'Friendly and energetic Labrador.',
            ],
            [
                'name' => 'Bella',
                'species' => 'Cat',
                'breed' => 'Persian',
                'age' => 2,
                'weight' => 4.3,
                'photo_path' => 'photos/bella.jpg',
                'description' => 'Calm and fluffy Persian cat.',
            ],
            [
                'name' => 'Charlie',
                'species' => 'Dog',
                'breed' => 'Beagle',
                'age' => 5,
                'weight' => 10.2,
                'photo_path' => 'photos/charlie.jpg',
                'description' => 'Loyal and curious Beagle.',
            ],
        ]);
    }
}
```

Wszystkie wymienione wyżej elementy technologiczne oraz narzędzia zostały starannie zintegrowane w aplikacji, co gwarantuje jej efektywne działanie, łatwość w utrzymaniu oraz potencjał do dalszego rozwoju.

3. Baza danych

Diagram ERD



Opis rozwiązań zastosowanych w bazie

Baza którą przygotowano dla projektu Systemu Zarządzania Schroniskiem dla Zwierząt nosi nazwę petShelter.

USERS: W tabeli users przechowujemy zarówno informacje o użytkownikach-klientach, którzy założyli konto w naszej aplikacji jak i również o administratorach. Tabela zawiera szczegółowe dane użytkowników, takie jak imię (name), nazwisko (last_name), email, numer telefonu (phone_number), hasło (password) oraz rolę, która to określa uprawnienia użytkownika (customer lub admin).

PETS: W tabeli pets przechowujemy informacje o zwierzętach znajdujących się w schronisku. Tabela zawiera dane takie jak imię zwierzęcia (name), gatunek (species), rasa (breed), wiek (age), waga (weight), ścieżkę do zdjęcia zwierzęcia (photo_path) oraz opis (description).

ADOPTIONS: W tabeli adoptions przechowujemy informacje o adopcjach zwierząt. Tabela zawiera pet_id i customer_id, które są kluczami obcymi do tabel pets i users, datę adopcji (adoption_date), oraz status adopcji (status), który może przyjmować wartości reserved, completed lub cancelled.

VACCINATIONS: W tabeli vaccinations przechowujemy informacje o szczepieniach zwierząt. Tabela zawiera pet_id, który jest kluczem obcym do tabeli pets, datę szczepienia (vaccination_date), rodzaj szczepionki (vaccine_type) oraz numer serii szczepionki (batch_number).

Migracje

Migracje w Laravelu służą do zarządzania strukturą bazy danych aplikacji. Migracje pozwalają na tworzenie, modyfikowanie i usuwanie tabel oraz wielu innych składowych bazy danych (np. klucze obce czy wyzwalacze) przy użyciu kodu PHP, co znacznie ułatwia zarządzanie bazą danych w trakcie rozwoju aplikacji. Każda migracja dziedziczy po klasie

Migration i zawiera dwie metody: up() i down(). Jako pierwsza wywoływana jest metoda up() od razu podczas uruchamiania migracji i służy ona do utworzenia struktury bazy danych. Metoda down() jest wywoływana podczas wycofywania migracji. Służy ona do usunięcia utworzonych wcześniej elementów bazy danych, jak na przykład tabele.

```
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('pets', function (Blueprint $table) {
            $table->id();
            $table->string('name', 20);
            $table->string('species', 20)->check('length(species) <= 20');
            $table->string('breed', 50)->check('length(breed) <= 50');
            $table->tinyInteger('age')->unsigned();
            $table->decimal('weight', 10, 1)->unsigned()->default(1.0)->check('weight > 0');
            $table->string('photo_path', 100);
            $table->string('description', 250);
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('pets');
    }
};
```

Seedery

Seedery w Laravelu służą do wypełniania tabel w bazie danych wcześniej przygotowanymi przykładowymi danymi. Jest to szczególnie przydatne podczas testowania i budowy aplikacji, pozwalając na szybkie załadowanie danych początkowych, testowych.

```
class UsersSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        DB::table('users')->insert([
            [
                'name' => 'John',
                'last_name' => 'Doe',
                'phone_number' => '123456789',
                'email' => 'john.doe@example.com',
                'password' => Hash::make('Haslo12345@'),
                'role' => 'admin',
            ],
            [
                'name' => 'Jane',
                'last_name' => 'Smith',
                'phone_number' => '987654321',
                'email' => 'jane.smith@example.com',
                'password' => Hash::make('Haslo12345@'),
                'role' => 'admin',
            ],
            [
                'name' => 'Alice',
                'last_name' => 'Johnson',
                'phone_number' => '456789123',
                'email' => 'alice.johnson@example.com',
                'password' => Hash::make('Haslo12345@'),
                'role' => 'customer',
            ],
            [
                'name' => 'Bob',
                'last_name' => 'Brown',
                'phone_number' => '789123456',
                'email' => 'bob.brown@example.com',
                'password' => Hash::make('Haslo12345@'),
                'role' => 'customer',
            ],
        ],
```

Opis Przykładowej Relacji w Modelach

Pomiędzy tabelami `pets` (zwierzęta) i `adoptions` (adopcje) w projekcie występuje relacja jeden do wielu (one-to-many). Taka relacja umożliwia, że jeden użytkownik może adoptować wiele zwierząt, zaś jedno zwierzę może mieć wiele rekordów w tabeli `adoptions`, np. jeden użytkownik tylko rezerwuje, zaś inny później adoptuje.

```
public $timestamps = false;

public function pet()
{
    return $this->belongsTo(Pet::class, 'pet_id');
}

public function user()
{
    return $this->belongsTo(User::class, 'customer_id');
}
```

W modelu `Pet` zdefiniowano metodę `adoptions()`, która ustanawia powiązanie jeden do wielu z modelem `Adoption`. Wykorzystuje ona metodę `hasMany()` i określa nazwę klucza obcego `pet_id`. W modelu `User` zdefiniowano metodę `adoptions()`, która ustanawia powiązanie jeden do wielu z modelem `Adoption` przy użyciu metody `hasMany()` i określa nazwę klucza obcego `customer_id`.

Dzięki zdefiniowaniu tych relacji, można łatwo uzyskać powiązane adopcje dla danego zwierzęcia poprzez dostęp do atrybutu `adoptions` na obiekcie `Pet`. Podobnie, można uzyskać adopcje powiązane z danym użytkownikiem poprzez dostęp do atrybutu `adoptions` na obiekcie `User`.

Relacja jeden do wielu umożliwia elastyczne odwzorowanie przypadków, w których jeden obiekt może być powiązany z wieloma innymi obiektami, co jest szczególnie przydatne w przypadku modelowania związku między zwierzętami a adopcjami, gdzie jedno zwierzę może mieć rekordy adopcji przez różnych użytkowników (na przykład uprzednia rezerwacja przez jednego z klientów, a późniejsza adopcja przez innego klienta), a jeden użytkownik może adoptować wiele zwierząt.

4. GUI

Opis Interfejsu Użytkownika

Graficzny Interfejs użytkownika (GUI) aplikacji Systemu Zarządzania Schroniskiem dla Zwierząt został zaprojektowany z myślą o prostocie i intuicyjnym użytkowaniu oraz estetyce. Aplikacja wykorzystuje Laravel Blade jako silnik szablonów, co umożliwia dynamiczne generowanie widoków HTML, przy jednoczesnej integracji z backendem. Całość interfejsu została wykonana przy użyciu nowoczesnych technologii frontendowych, takich jak Bootstrap, wykorzystany do stylizacji elementów interfejsu oraz JavaScript do obsługi dynamicznych elementów.

Wybrane Widoki Aplikacji

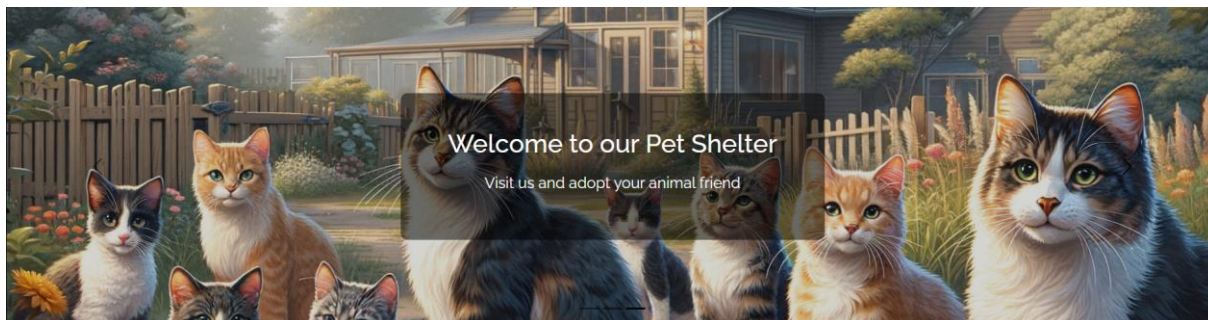
I. Strona główna aplikacji (index.blade.php)

Strona główna jest pierwszym widokiem, który użytkownik widzi po wejściu na stronę aplikacji. Składa się ona z:

- **Nagłówek:** Zawiera on nazwę aplikacji („Pet Shelter”) oraz pasek nawigacyjny umożliwiający przejście do pozostałych podstron aplikacji. Oprócz tego umieszczony został w nim przycisk odpowiedzialny za zmianę kolorystyki widoków aplikacji pomiędzy motywami ciemnym i jasnym oraz sekcję zawierającą odnośnik do logowania się, który w przypadku pomyślnego procesu logowania zmienia swoją zawartość oraz funkcjonalność



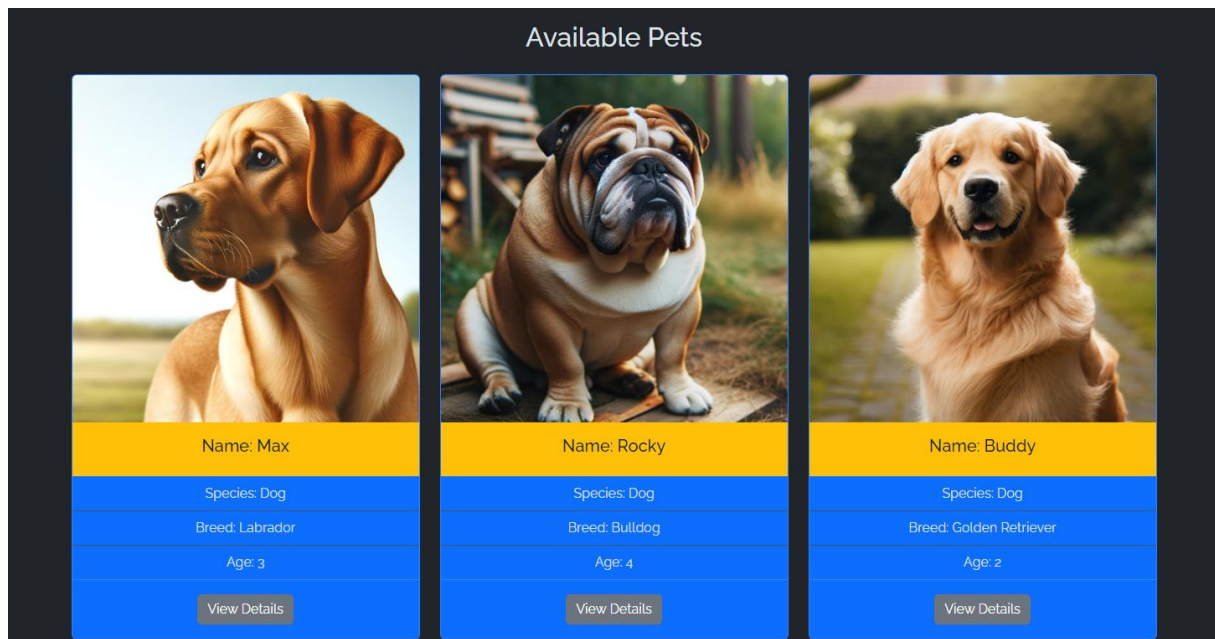
- **Karuzeli:** Jest to dynamiczny slider wyświetlający grafiki wzbogacające wygląd aplikacji i zachęcające odwiedzającego do aktywnego przeglądania jej zawartości.



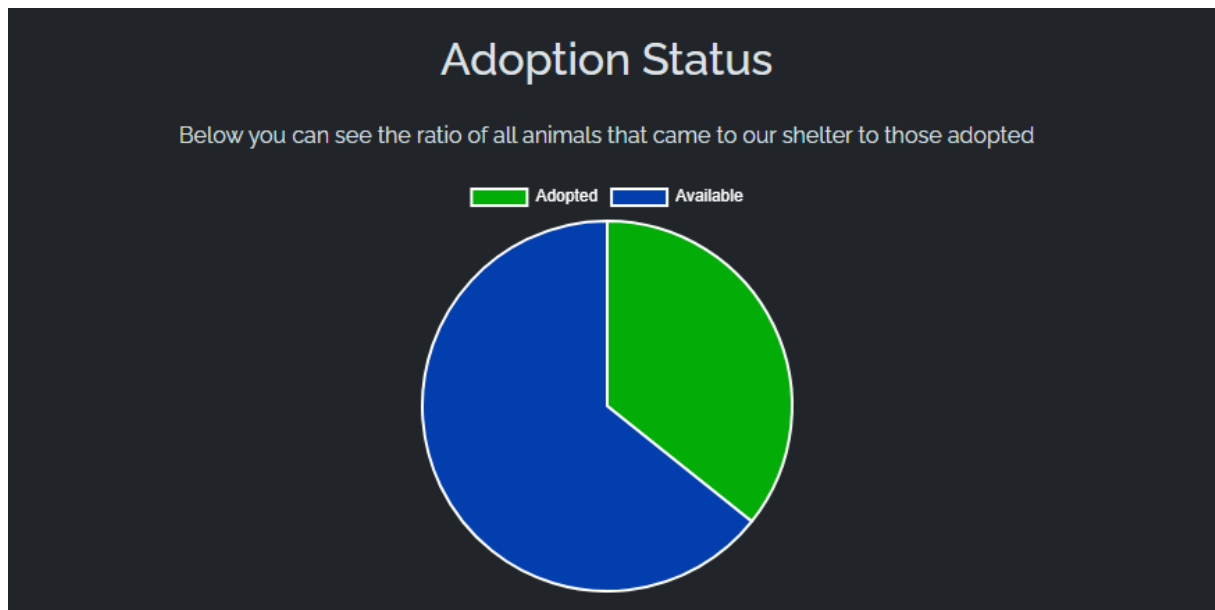
- **Sekcji „Animals That Have Found a Home”:** Jest to sekcja wyświetlająca sześć przykładowych zdjęć zwierząt które zostały adoptowane.



- **Kart dostępnych zwierząt:** Sekcja ta przedstawia sześć pierwszych zwierząt możliwych do adopcji. W każdej karcie zawarte są informacje o imieniu zwierzęcia, gatunku, rasie oraz wieku. Dostępny jest również przycisk przenoszący do widoku wszystkich informacji o danym zwierzęciu.



- **Sekcji ze statystykami:** Sekcja zawiera wykres kołowy prezentujący stosunek zwierząt adoptowanych do zwierząt dostępnych do adopcji.

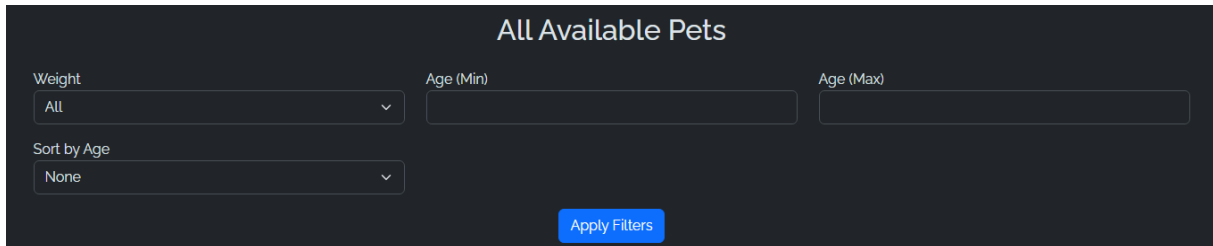


- **Stopki:** Zawiera ona informacje o autorze aplikacji, przykładowy adres email oraz adres fizyczny schroniska, cztery ikony będące odnośnikami do mediów społecznościowych oraz godziny otwarcia schroniska

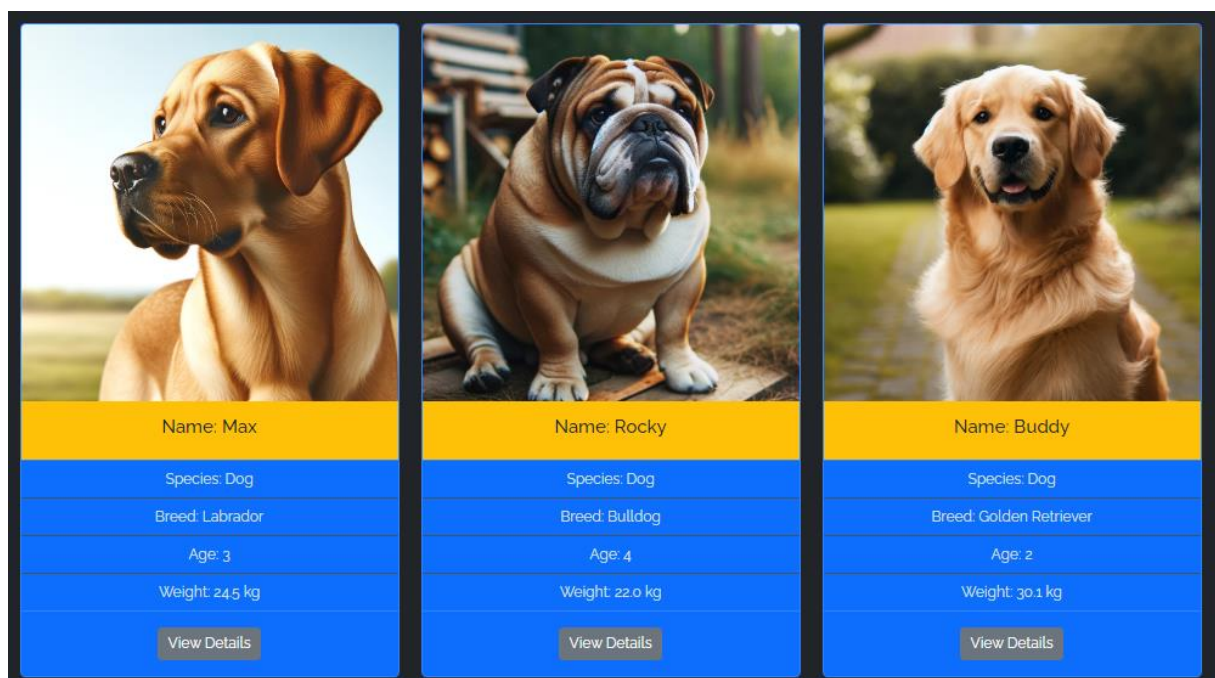
II. Widok wszystkich zwierząt (all.blade.php)

Jest to strona która zawiera karty wszystkich zwierząt z bazy możliwych do adopcji. Użytkownik ma możliwość filtrowania ich po wadze oraz wieku, a także sortowania od najmłodszych lub od najstarszych. Składa się ona z:

- **Sekcji filtrowania i sortowania:** Sekcja ta zawiera pola do filtrowania wszystkich dostępnych zwierząt według wieku i wagi oraz sortowania od najmłodszych lub od najstarszych. Każde z pól działa niezależnie od pozostałych, dzięki czemu jest możliwość zastosowania tylko jednego filtru jak również wszystkich na raz z dodatkowym sortowaniem








- **Kart dostępnych zwierząt:** Sekcja ta zawiera identyczne karty z informacjami o danym zwierzęciu, jak na stronie głównej. Domyślnie wyświetlane są wszystkie zwierzęta po dziewięć kart na stronę.



III. Widok wszystkich adopcji i rezerwacji (adoptions.blade.php)

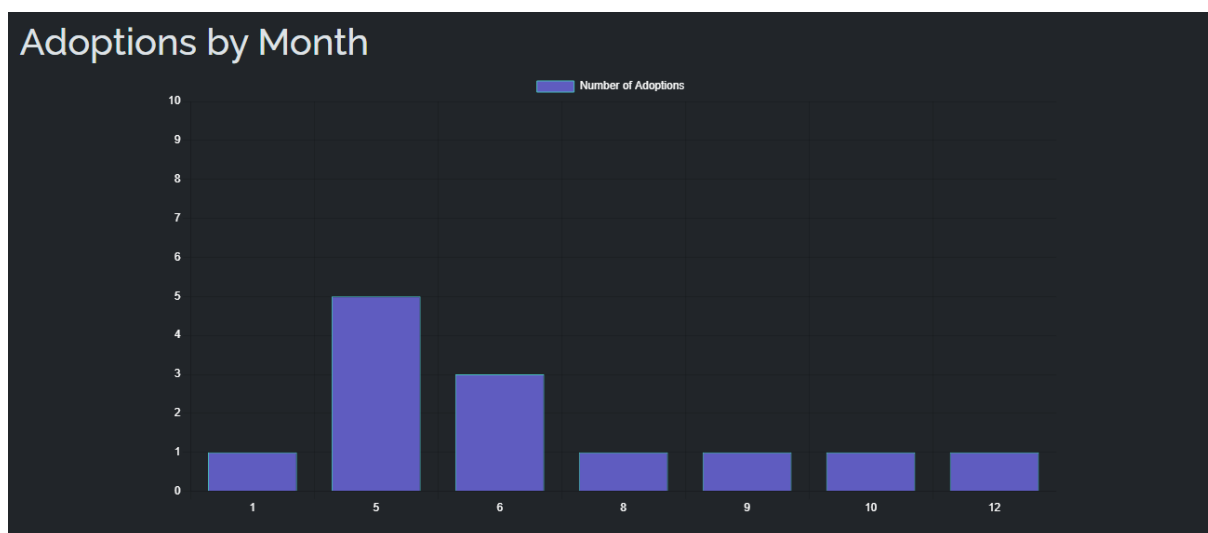
Jest to widok dostępny tylko dla administratorów aplikacji. Składa się z dwóch sekcji. Pierwsza z nich zawiera informacje o wszystkich rekordach adopcji, a także umożliwia zmianę statusu danego rekordu adopcji na jeden z trzech (completed, reserved, cancelled). Druga zawiera wykres ilustrujący ilość adopcji w danym miesiącu minionego roku.

- **Tabela z wszystkimi rekordami adopcji:** Wyświetla informacje o wszystkich rekordach adopcji w bazie danych oraz umożliwia zmianę statusu adopcji.

All Adoptions							
Photo	Pet ID	Pet Name	Species	Breed	Customer Email	Adoption Date	Status
	2	Bella	Cat	Persian	bob.brown@example.com	2023-08-15	completed
	3	Charlie	Dog	Beagle	charlie.davis@example.com	2023-09-10	reserved
	4	Lucy	Cat	Siamese	emily.wilson@example.com	2023-10-05	completed
	5	Rocky	Dog	Bulldog	david.miller@example.com	2023-12-01	cancelled
	6	Luna	Cat	Maine Coon	sophia.taylor@example.com	2024-01-20	cancelled

< 1 2 3 >

- **Wykres ilości adopcji:** Wykres słupkowy ilustruje ilość adopcji w roku minionym z podziałem na miesiące



Stylizacja i Responsywność

Wykonana aplikacja jest w pełni responsywna, co oznacza, że interfejs użytkownika dostosowuje się do różnych rozmiarów ekranów, od dużych ekranów komputerów stacjonarnych po małe ekrany urządzeń mobilnych. Stylizacja została wykonana z użyciem frameworku Bootstrap 5, co zapewnia spójny i estetyczny wygląd wszystkich elementów interfejsu, wraz z zachowaniem responsywności każdego z elementów interfejsu.

5. Uruchomienie aplikacji

Wymagania

1. **Pakiet XAMPP:** Pakiet XAMPP zawiera serwer Apache, bazę danych MySQL oraz interpreter PHP (należy zainstalować wersję zawierającą PHP w wersji 8.x), które są niezbędne do uruchomienia aplikacji Laravel. Pakiet XAMPP można za darmo pobrać ze strony [XAMPP Download](#).
2. **Composer:** Composer to narzędzie do zarządzania zależnościami w projekcie Laravel. Używane jest do instalacji wszystkich niezbędnych bibliotek i zależności frameworka. Composer można pobrać za darmo ze strony: [Composer Download](#).
3. **Visual Studio Code:** Visual Studio Code to popularne środowisko programistyczne (IDE) firmy microsoft, które umożliwia łatwe tworzenie oraz edytowanie wszelkiego rodzaju kodu. Visual Studio Code można pobrać za darmo ze strony: [VS Code Download](#).

Kroki potrzebne do wykonania przy pierwszym uruchomieniu aplikacji

1. Uruchomienie pakietu XAMPP:

- Po zainstalowaniu XAMPP, należy go uruchomić i włączyć wymagane do działania aplikacji Laravel moduły: Apache oraz MySQL.
- XAMPP zapewnia graficzny interfejs, który pozwala na łatwe zarządzanie usługami serwera.

2. Uruchomienie pliku start.bat:

- W folderze projektu znajduje się plik **start.bat**, który automatyzuje proces konfiguracji i uruchomienia aplikacji.
- Należy otworzyć folder projektu i uruchomić plik **start.bat** jako administrator.

```
%systemDrive%\xampp\mysql\bin\mysql -uroot -e "CREATE DATABASE IF NOT EXISTS petShelter;"

if %errorlevel% neq 0 msg %username% "Nie udało sie utworzyc bazy danych." && exit /b %errorlevel%

php -r "copy('.env.example', '.env');"

call composer install

call php artisan key:generate

call php artisan storage:link

call php artisan migrate:fresh --seed

call composer require --dev barryvdh/laravel-ide-helper

call php artisan ide-helper:generate

call php artisan serve

start http://127.0.0.1:8000

code .
```

- Skrypt start.bat wykonuje następujące operacje:
 - Jeśli nie utworzono wcześniej bazy petShelter, to tworzy ją.
 - Kopiuje plik konfiguracyjny **.env.example** do **.env**.
 - Instaluje wszystkie wymagane do działania aplikacji zależności za pomocą **composer install**.
 - Generuje klucz aplikacji za pomocą **php artisan key:generate**.
 - Tworzy 'symbolic link' dla przechowywanych w storage plików za pomocą **php artisan storage:link**.
 - Wykonuje migrację bazy danych oraz seeduje dane początkowe za pomocą **php artisan migrate:fresh --seed**.
 - Instaluje pakiet **barryvdh/laravel-ide-helper** dla ułatwienia programistycznego.
 - Uruchamia lokalny serwer deweloperski php za pomocą **php artisan serve**.
 - Otwiera w przeglądarce adres <http://127.0.0.1:8000>, który jest adresem uruchomionego serwera deweloperskiego
 - Otwiera projekt w Visual Studio Code.

3. Uruchomienie serwera deweloperskiego:

- Po wykonaniu się skryptu **start.bat**, aplikacja powinna być dostępna pod adresem <http://127.0.0.1:8000>.

Kroki potrzebne do wykonania przy kolejnych uruchomieniach aplikacji

1. Przejście do katalogu projektu:

- Należy uruchomić wiersz poleceń.
- Przejdź do katalogu, w którym znajduje się projekt aplikacji Laravel.

2. Uruchomienie serwera deweloperskiego:

- Uruchom serwer deweloperski za pomocą komendy: **php artisan serve**
- Aplikacja powinna być dostępna pod adresem <http://127.0.0.1:8000>.

6. Funkcjonalności Aplikacji

Wykonana aplikacja Systemu Zarządzania Schroniskiem dla Zwierząt oferuje szereg funkcji dostępnych dla usprawnienia pracy pracowników schroniska, a także dla klientów zainteresowanych ofertą schroniska. Poniżej przedstawione są wszystkie widoki wraz z zaimplementowanymi w nich funkcjami dostępnymi dla klientów i administratorów aplikacji.

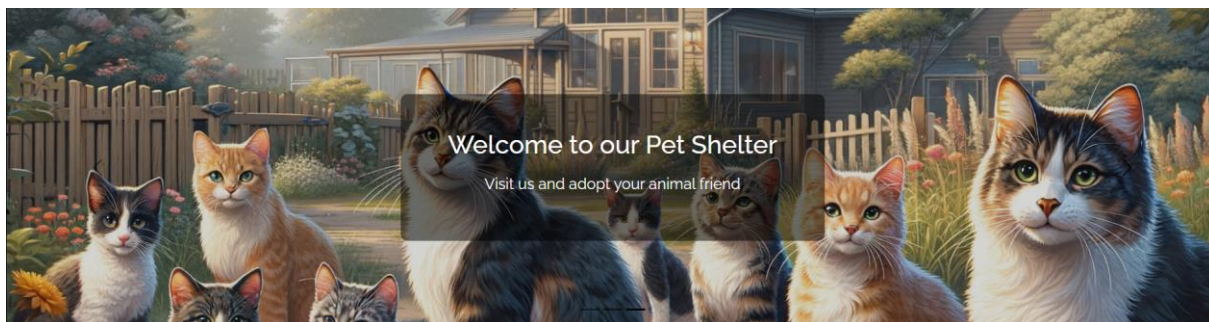
1. Strona główna aplikacji (index.blade.php)

Strona główna jest pierwszym widokiem, który użytkownik widzi po wejściu na stronę aplikacji. Składa się ona z:

- **Nagłówka:** Zawiera on nazwę aplikacji („Pet Shelter”) oraz pasek nawigacyjny umożliwiający przejście do pozostałych podstron aplikacji. Oprócz tego umieszczony został w nim przycisk odpowiedzialny za zmianę kolorystyki widoków aplikacji pomiędzy motywami ciemnym i jasnym oraz sekcję zawierającą odnośnik do logowania się, który w przypadku pomyślnego procesu logowania zmienia swoją zawartość oraz funkcjonalność



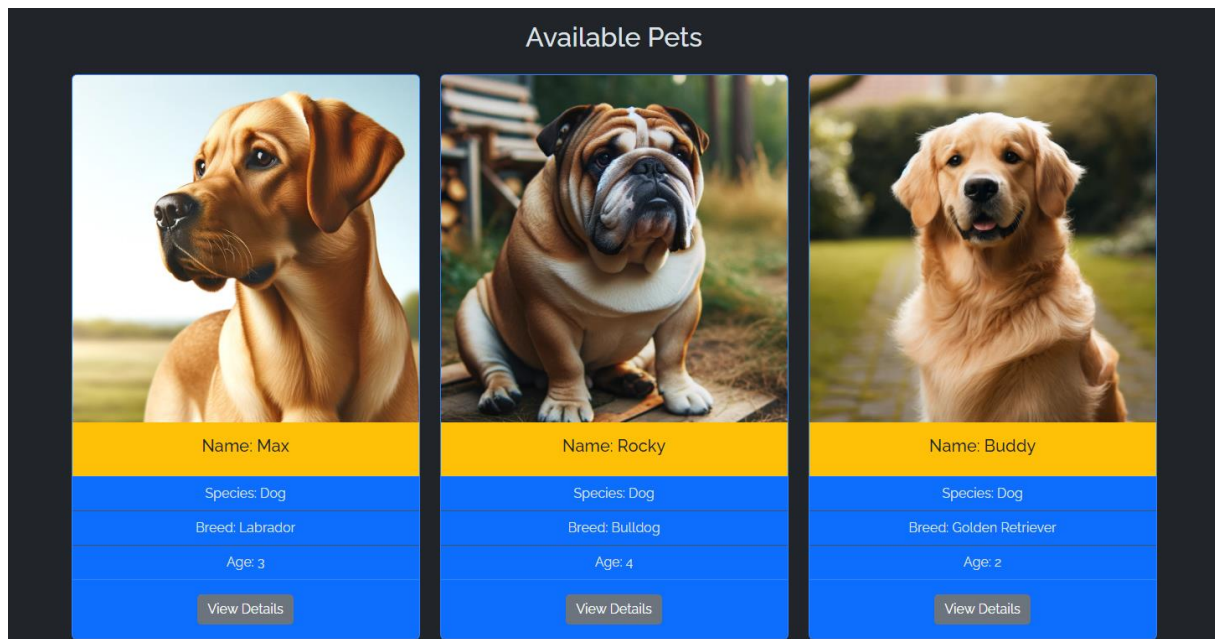
- **Karuzeli:** Jest to dynamiczny slider wyświetlający grafiki wzbogacające wygląd aplikacji i zachęcające odwiedzającego do aktywnego przeglądania jej zawartości.



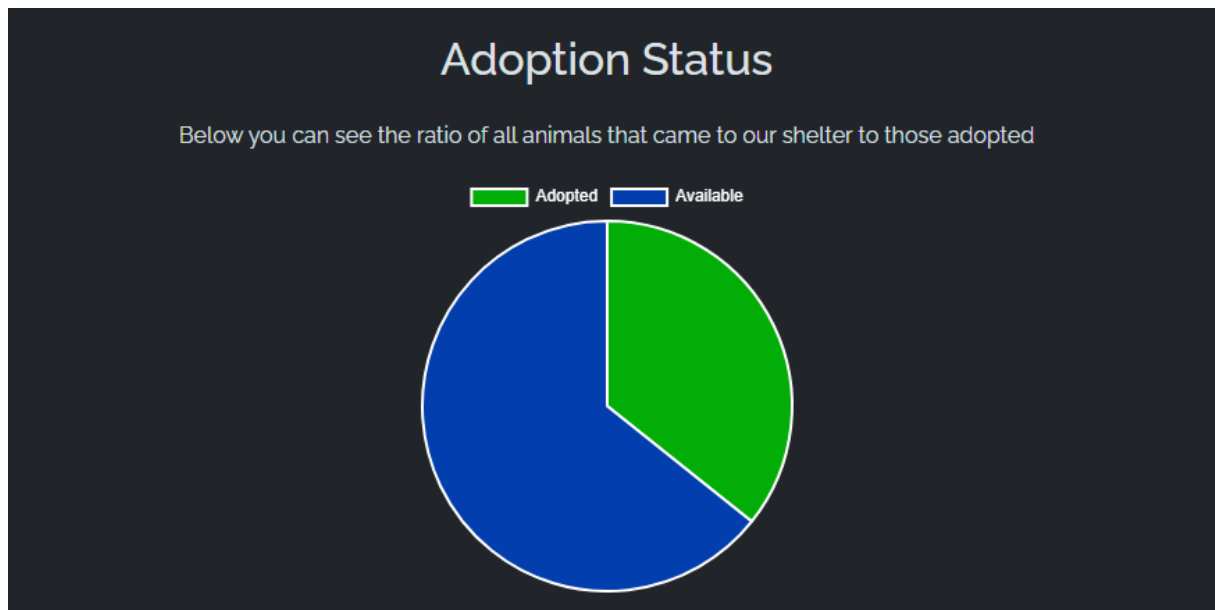
- **Sekcji „Animals That Have Found a Home”:** Jest to sekcja wyświetlająca sześć przykładowych zdjęć zwierząt które zostały adoptowane.



- **Kart dostępnych zwierząt:** Sekcja ta przedstawia sześć pierwszych zwierząt możliwych do adopcji. W każdej karcie zawarte są informacje o imieniu zwierzęcia, gatunku, rasie oraz wieku. Dostępny jest również przycisk przenoszący do widoku wszystkich informacji o danym zwierzęciu.



- **Sekcji ze statystykami:** Sekcja zawiera wykres kołowy prezentujący stosunek zwierząt adoptowanych do zwierząt dostępnych do adopcji.

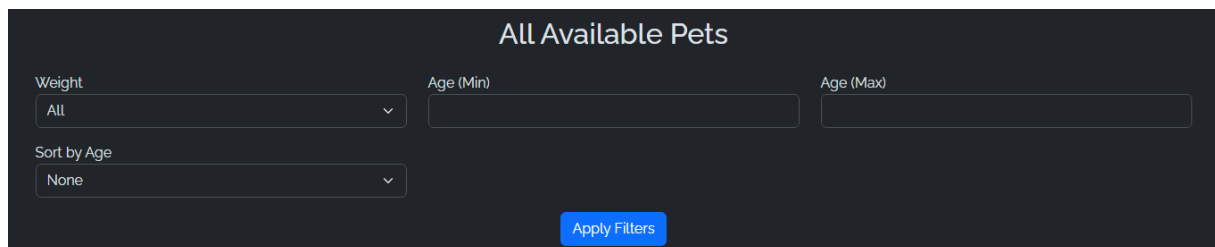


- **Stopki:** Zawiera ona informacje o autorze aplikacji, przykładowy adres email oraz adres fizyczny schroniska, cztery ikony będące odnośnikami do mediów społecznościowych oraz godziny otwarcia schroniska

2. Widok wszystkich zwierząt (all.blade.php)

Jest to strona która zawiera karty wszystkich zwierząt z bazy możliwych do adopcji. Użytkownik ma możliwość filtrowania ich po wadze oraz wieku, a także sortowania od najmłodszych lub od najstarszych. Składa się ona z:

- **Sekcji filtrowania i sortowania:** Sekcja ta zawiera pola do filtrowania wszystkich dostępnych zwierząt według wieku i wagi oraz sortowania od najmłodszych lub od najstarszych. Każde z pól działa niezależnie od pozostałych, dzięki czemu jest możliwość zastosowania tylko jednego filtru jak również wszystkich na raz z dodatkowym sortowaniem. Użytkownik ma możliwość w przypadku filtrowania po wadze jednego z dostępnych przedziałów: Small ($\leq 5\text{kg}$), Medium ($5.1\text{kg}-15\text{kg}$) oraz Large ($>15\text{kg}$). W polach przedziału wieku użytkownik może wprowadzić dowolną liczbę całkowitą w przedziale 0-30. W przypadku sortowania użytkownik ma do wyboru opcję sortowania od najstarszych lub od najmłodszych



All Available Pets

Weight: All

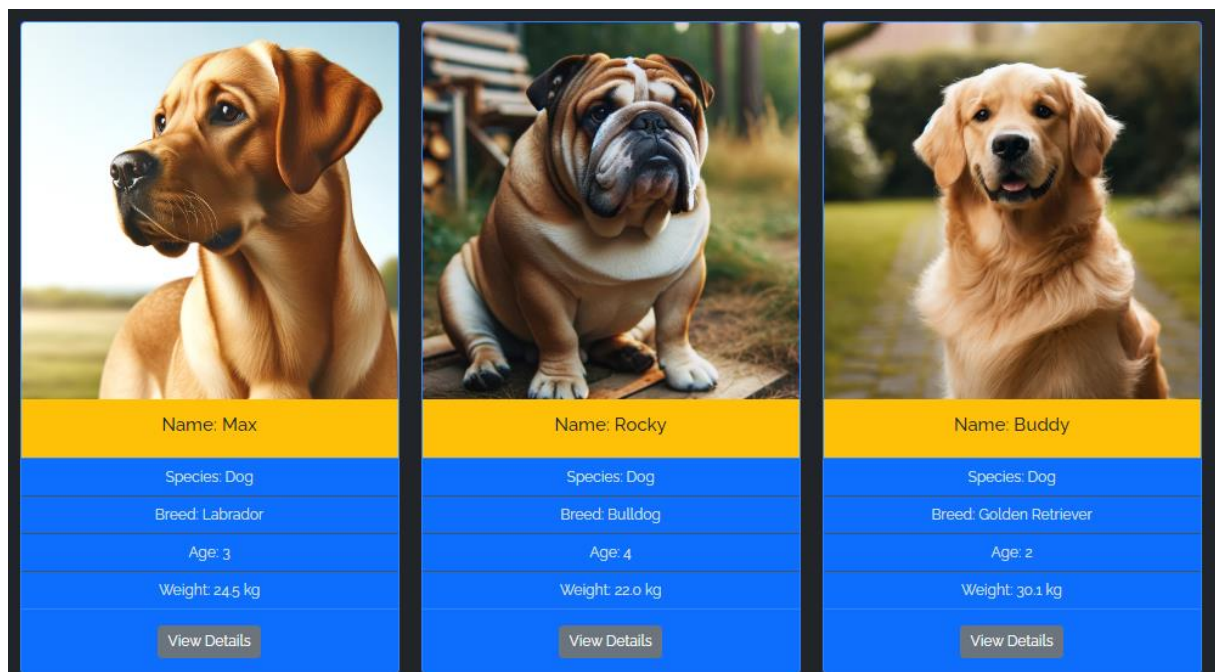
Age (Min):

Age (Max):

Sort by Age: None

Apply Filters

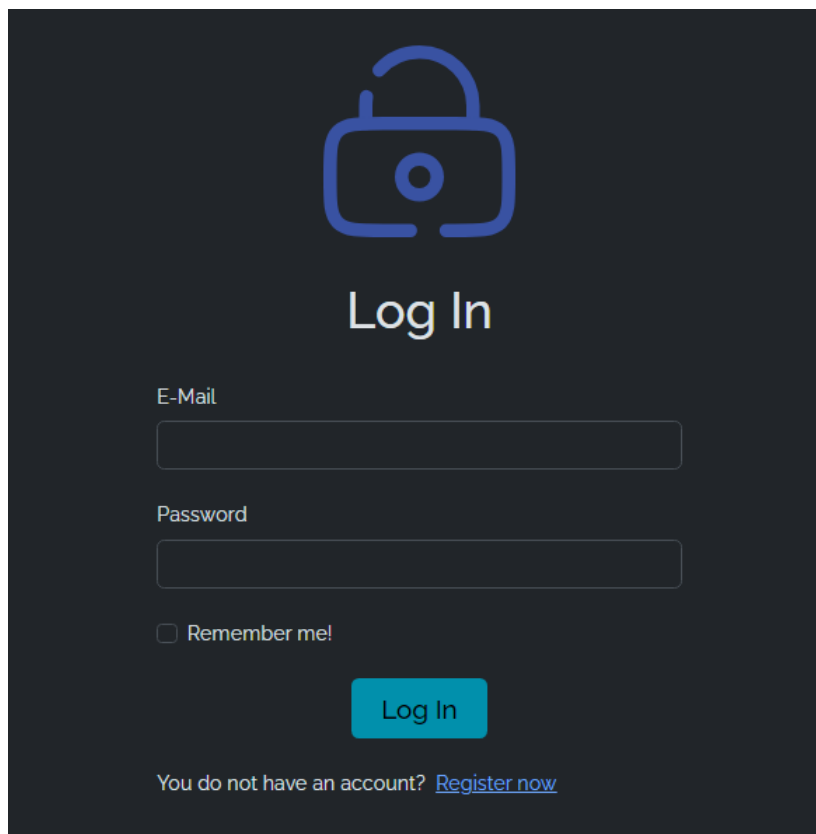
- **Kart dostępnych zwierząt:** Sekcja ta zawiera identyczne karty z informacjami o danym zwierzęciu, jak na stronie głównej. Domyślnie wyświetlane są wszystkie zwierzęta po dziewięć kart na stronę.



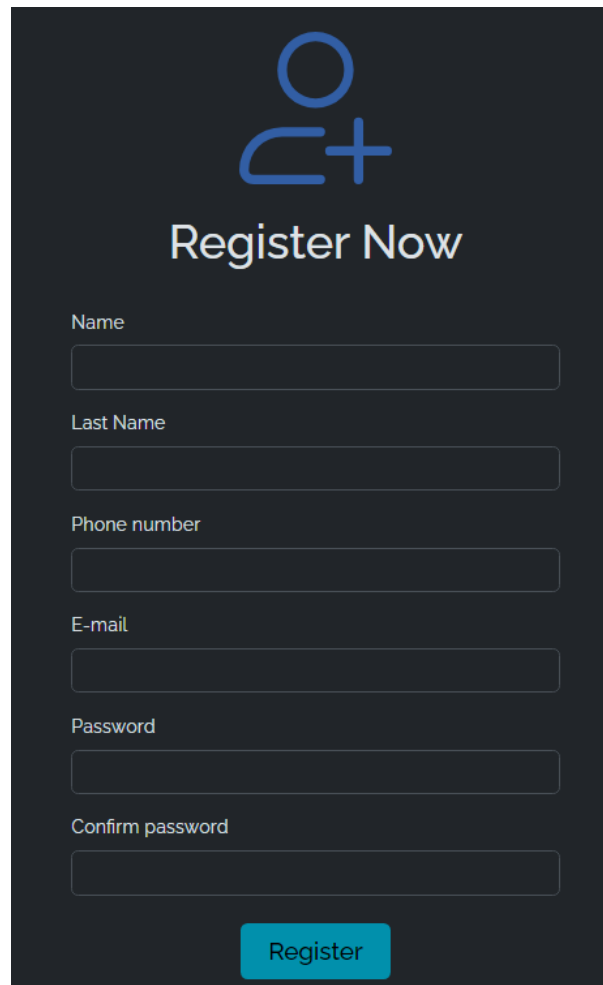
3. Panel Logowania oraz panel Rejestracji (login.blade.php i register.blade.php)

Panele logowania i rejestracji są dostępne tylko dla niezalogowanych użytkowników. Każdy z widoków składa się z odpowiednich pól wprowadzania danych. Dla panelu logowania jest to adres email (E-Mail) oraz hasło (Password), zaś dla panelu rejestracji są to: imię (Name), nazwisko (Last Name), numer telefonu (Phone number), adres email (E-mail), hasło (Password) oraz powtórzenie hasła (Confirm password). Do panelu logowania użytkownik może się dostać klikając strzałkę w prawej części navbaru, a następnie wybraniu opcji „Log In”, zaś do formularza rejestracji poprzez kliknięcie „Register now” w panelu logowania.

- **Panel logowania:** Widok umożliwiający zalogowanie się administratorom oraz klientom. Przykładowy email i hasło do zalogowania się jako administrator: john.doe@example.com, Hasło12345@. Do zalogowania się jako klient: alice.johnson@example.com, Hasło12345@. Po pomyślnym zalogowaniu się, użytkownik przenoszony jest do panelu użytkownika

The image shows a dark-themed login form. At the top center is a blue icon of a padlock with a keyhole. Below the icon, the text "Log In" is displayed in a large, white, sans-serif font. Underneath, there are two input fields: the first is labeled "E-Mail" and the second is labeled "Password". Both labels are in a small, light gray font. Below the password field is a checkbox with the text "Remember me!" next to it. At the bottom center of the form is a blue rectangular button with the text "Log In" in white. At the very bottom, there is a line of text: "You do not have an account? [Register now](#)", where the link is in blue and underlined.

- **Panel rejestracji:** Widok umożliwiający zarejestrowanie się nowych użytkowników, z domyślną rolą „customer”, z możliwością późniejszej zmiany typu konta przez jednego z administratorów.



The image shows a registration form on a dark background. At the top, there is a blue icon consisting of a circle above a stylized 'C' with a plus sign. Below the icon, the text 'Register Now' is displayed in a large, white, sans-serif font. The form contains six input fields, each with a label above it: 'Name', 'Last Name', 'Phone number', 'E-mail', 'Password', and 'Confirm password'. All labels and the text 'Register Now' are in white. The input fields are empty and have a light gray border. At the bottom of the form, there is a blue button with the word 'Register' in white text.

Register Now

Name

Last Name

Phone number

E-mail

Password

Confirm password

Register

IV. Panel użytkownika oraz widok zmiany danych i hasła (profile.blade.php oraz editdata.blade.php)

Strona panelu użytkownika pozwala użytkownikom na przeglądanie swoich danych oraz historii adopcji i rezerwacji. Umieszczony jest również przycisk odpowiedzialny za przeniesienie do widoku umożliwiającego edytowanie swoich danych oraz zmiany dotychczasowego hasła logowania.

- **Panel użytkownika:** Zbudowany jest z dwóch sekcji. Pierwsza wyświetla dane aktualnie zalogowanego użytkownika oraz zawiera dwa przyciski, z których pierwszy odpowiedzialny jest za przeniesienie do widoku zmiany danych i hasła, drugi zaś odpowiada za wylogowanie. Drugą sekcję stanowi tabela wyświetlająca informacje o adopcjach i rezerwacjach danego użytkownika. W przypadku, gdy zalogowany użytkownik jest administratorem, druga z sekcji jest niewidoczna, gdyż administrator nie może posiadać żadnych rekordów związanych z adopcjami.

The image shows a 'User Profile' interface. It has a dark theme. The top section is titled 'User Data' and contains the following information: Name: Olivia, Last Name: Thomas, Email: olivia.thomas@example.com, Phone Number: 741258963, and Customer ID: 10. Below this information are two buttons: 'Edit Password and Data' (grey) and 'Log Out' (red). The bottom section is titled 'Your Adoptions and Reservations' and contains a table with the following data:

Pet Name	Date of Adoption/Reservation	Species	Breed	Status
Rocky	2024-06-01	Dog	Bulldog	completed
Simba	2024-06-01	Cat	Bengal	reserved

- **Widok zmiany danych i hasła:** Składa się z dwóch sekcji. Pierwszą stanowi formularz zmiany danych użytkownika: imienia (Name), nazwiska (Last Name), numeru telefonu (Phone number), oraz adresu email (E-mail). Możliwa jest zmiana jednej lub wielu danych. Zmiany zatwierdzane są przyciskiem „Save Changes”. Drugą sekcję stanowi formularz zmiany hasła, w którym to należy podać obecne hasło oraz podać dwukrotnie nowe hasło. Zmianę hasła zatwierdzamy przyciskiem „Change Password”.



Account Settings

Name

Last Name

Phone Number

E-mail

Save Changes

Change Password

Current Password

New Password

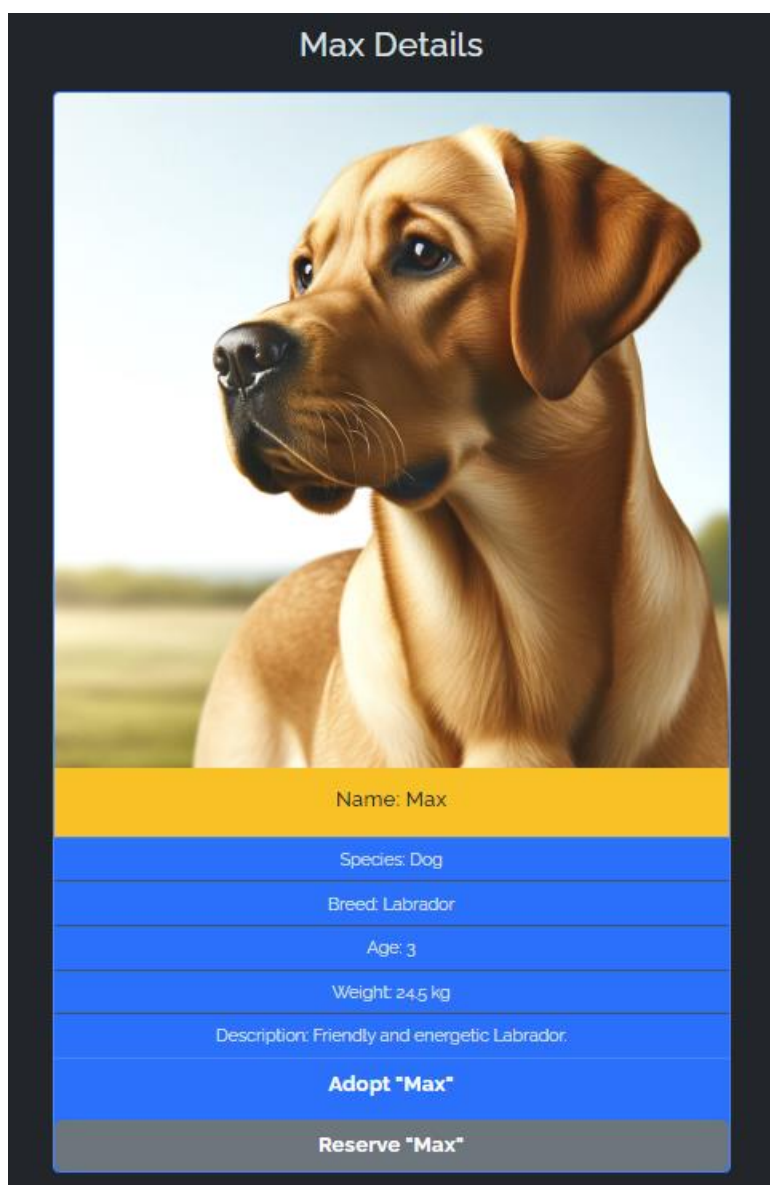
Confirm New Password

Change Password

V. Widok szczegółów zwierzęcia (show.blade.php)

Widok szczegółów dla danego zwierzęcia składa się z dwóch sekcji. Pierwsza zawiera wszystkie informacje z tabeli pets dla konkretnego zwierzęcia. Druga zaś wyświetla rekordy szczepień powiązane z danego zwierzaka. W przypadku wyboru przez zalogowanego klienta jednej z opcji: Adopt lub Reserve, wyświetlane jest odpowiednie okno modalne z informacjami powiązanymi z daną transakcją oraz przyciski do potwierdzenia lub anulowania operacji.

- **Sekcja informacji o zwierzęciu:** Składa się ona z nagłówka zawierającego imię zwierzęcia, zaś poniżej nagłówka znajduje się karta z informacjami o zwierzęciu, podobna do tych na stronie głównej oraz w widoku wszystkich zwierząt, jednakże ta zawiera wszystkie informacje o danym zwierzęciu, w tym opis. Poniżej danych zwierzęcia znajdują się dwa przyciski, które odpowiednio uruchamiają proces adopcji bądź rezerwacji zwierzęcia. W przypadku, gdy klient nie jest zalogowany tekst w przyciskach zmieniany jest na „Log in to adopt [imię zwierzęcia]”, „Log in to reserve [imię zwierzęcia]”. W przypadku gdy zalogowany jest administrator to działanie obu przycisków jest wyłączone



- **Sekcja z informacjami o szczepieniach:** Sekcja ta zawiera w formie tabeli informacje o wszystkich rekordach szczepień powiązanych z danym zwierzęciem. Wyświetlana jest data szczepienia, typ szczepionki oraz numer seryjny.

Vaccination History	
Date: 2023-01-15	
Vaccine Type: Rabies	
Batch Number: R12345	
Date: 2024-05-26	
Vaccine Type: Rabies	
Batch Number: AB1234	

- **Modal adopcji:** Jest to okno modalne wyświetlane w przypadku kliknięcia przycisku odpowiedzialnego za proces adopcji zwierzęcia, gdy klient jest zalogowany. Pełni ono rolę potwierdzenia operacji adopcji przez użytkownika umożliwiając jej zatwierdzenie bądź odrzucenie.

Adoption Confirmation

Are you sure you want to adopt "Max"? You will be responsible for this pet.

CancelConfirm

- **Modal rezerwacji:** Jest to okno modalne wyświetlane w przypadku kliknięcia przycisku odpowiedzialnego za proces rezerwacji zwierzęcia, gdy klient jest zalogowany. Pełni ono rolę potwierdzenia operacji rezerwacji przez użytkownika umożliwiając jej zatwierdzenie bądź odrzucenie. Wyświetlana jest informacja o czasie, przez jaki dane zwierzęcie będzie zarezerwowane (48 godzin), zaś po jego upływie administrator odpowiedzialny jest za anulowanie rezerwacji, gdy ta w ciągu 48 godzin nie zostanie zmieniona na adopcję

Reservation Confirmation






Are you sure you want to reserve "Max"? You will have 48 hours to adopt this pet.

CancelConfirm

VI. Widok wszystkich adopcji i rezerwacji (adoptions.blade.php)

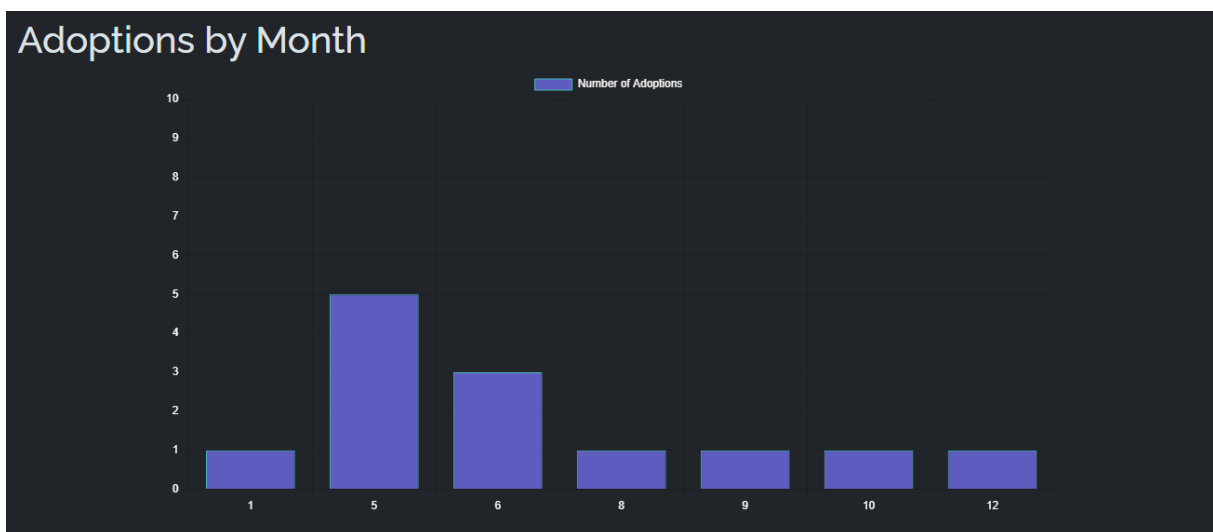
Jest to widok dostępny tylko dla administratorów aplikacji. Składa się z dwóch sekcji. Pierwsza z nich zawiera informacje o wszystkich rekordach adopcji, a także umożliwia zmianę statusu danego rekordu adopcji na jeden z trzech (completed, reserved, cancelled). Druga zawiera wykres ilustrujący ilość adopcji w danym miesiącu minionego roku.

- **Tabela z wszystkimi rekordami adopcji:** Wyświetla informacje o wszystkich rekordach adopcji w bazie danych oraz umożliwia zmianę statusu adopcji. Administrator ma do wyboru z listy rozwijanej jeden z trzech statusów: completed, reserved, cancelled. Każdy typ statusu ma przypisany odpowiedni kolor tła dla wyraźnego odznaczenia się każdego ze statusów

All Adoptions							
Photo	Pet ID	Pet Name	Species	Breed	Customer Email	Adoption Date	Status
	2	Bella	Cat	Persian	bob.brown@example.com	2023-08-15	completed
	3	Charlie	Dog	Beagle	charlie.davis@example.com	2023-09-10	reserved
	4	Lucy	Cat	Siamese	emily.wilson@example.com	2023-10-05	completed
	5	Rocky	Dog	Bulldog	david.miller@example.com	2023-12-01	cancelled
	6	Luna	Cat	Maine Coon	sophia.taylor@example.com	2024-01-20	cancelled

< 1 2 3 >

- **Wykres ilości adopcji:** Wykres słupkowy ilustruje ilość adopcji w roku minionym z podziałem na miesiące



VII. Widok wszystkich użytkowników (users.blade.php)

Widok dostępny tylko dla zalogowanego administratora, wyświetlający informacje o wszystkich kontach użytkowników aplikacji. Umożliwia on zmianę danych osobowych, adresu email oraz numeru telefonu, a także usuwania danego konta użytkownika. Dodatkowo jest dostępna opcja zmiany roli dla danego konta z „customer” na „admin”, w przypadku chęci dodania nowego administratora aplikacji, który uprzednio zarejestrował się poprzez formularz rejestracyjny. Administratorem może zostać tylko użytkownik który nie posiadał wcześniej żadnych rekordów adopcji.

ID	Name	Last Name	Email	Phone Number	Actions
1	John	Doe	john.doe@example.com	123456789	<button>Edit</button> <button>Delete</button>
2	Jane	Smith	jane.smith@example.com	987654321	<button>Edit</button> <button>Delete</button>
4	Bob	Brown	bob.brown@example.com	789123456	<button>Edit</button> <button>Delete</button>
5	Charlie	Davis	charlie.davis@example.com	321654987	<button>Edit</button> <button>Delete</button>
6	Emily	Wilson	emily.wilson@example.com	654321789	<button>Edit</button> <button>Delete</button>

Name

Last Name

Email

Phone Number

☒ Administrator

Save











VIII. Widoki wszystkich zwierząt, edycji ich danych oraz dodawania nowego zwierzęcia (pets.blade.php, editPet.blade.php oraz addPet.blade.php)

Widok wszystkich zwierząt, edycji ich danych oraz dodawania nowego zwierzęcia są opcjami dostępnymi tylko dla zalogowanego administratora.

- **Widok wszystkich zwierząt:** Wyświetla wszystkie informacje o każdym ze zwierząt jakie istnieją w bazie danych. Zawarte są informacje takie jak imię zwierzęcia (name), jego unikalny identyfikator (ID), gatunek (species), rasa (breed), waga (weight), wiek (age), opis zwierzęcia (description), a także miniaturowe zdjęcie (Photo). Poza informacjami o zwierzętach w każdym wierszu znajdują się przyciski odpowiedzialne kolejno za przenoszenie do widoku edycji danych zwierzęcia, usuwanie zwierzęcia oraz przenoszenie do widoku szczepienia dla danego zwierzęcia. Istnieje również opcja dodawania nowych zwierząt do bazy danych dostępna po kliknięciu przycisku „Add Pet”

All Pets

Add Pet

ID	Name	Species	Breed	Weight	Age	Description	Photo	Actions
1	Max	Dog	Labrador	24.5	3	Friendly and energetic Labrador.		Edit Delete Vaccinate
2	Bella	Cat	Persian	4.3	2	Calm and fluffy Persian cat.		Edit Delete Vaccinate
3	Charlie	Dog	Beagle	10.2	5	Loyal and curious Beagle.		Edit Delete Vaccinate
4	Lucy	Cat	Siamese	3.8	1	Playful and talkative Siamese cat.		Edit Delete Vaccinate
5	Rocky	Dog	Bulldog	22.0	4	Strong and gentle Bulldog.		Edit Delete Vaccinate
6	Luna	Cat	Maine Coon	6.5	3	Large and friendly Maine Coon.		Edit Delete Vaccinate
7	Buddy	Dog	Golden Retriever	30.1	2	Loving and obedient Golden Retriever.		Edit Delete Vaccinate
9	Bailey	Dog	Poodle	7.3	1	Intelligent and active Poodle.		Edit Delete Vaccinate
10	Nala	Cat	Ragdoll	4.8	2	Affectionate and calm Ragdoll.		Edit Delete Vaccinate
11	Daisy	Dog	Dachshund	6.1	3	Curious and lively Dachshund.		Edit Delete Vaccinate

< 1 2 >

- Widok edycji danych zwierzęcia:** Zawiera formularz odpowiedzialny za zmianę danych konkretnego zwierzęcia w bazie danych. Możliwe do zmiany są dane takie jak: imię (name), gatunek (species), rasa (breed), waga (weight), wiek (age), opis zwierzęcia (description), a także możliwość przesłania nowego zdjęcia dla danego zwierzęcia, przy czym obecne dane są automatycznie wprowadzone w odpowiednie pola, aby użytkownik wiedział jak wyglądają obecne dane i wiedział jakich zmian chce dokonać. Zmiany zatwierdzane są kliknięciem przycisku „Save” bądź odrzucane przyciskiem „Cancel”

Edit Pet

Name

Max

Species

Dog

Breed

Labrador

Age

3

Weight

24.5

Description

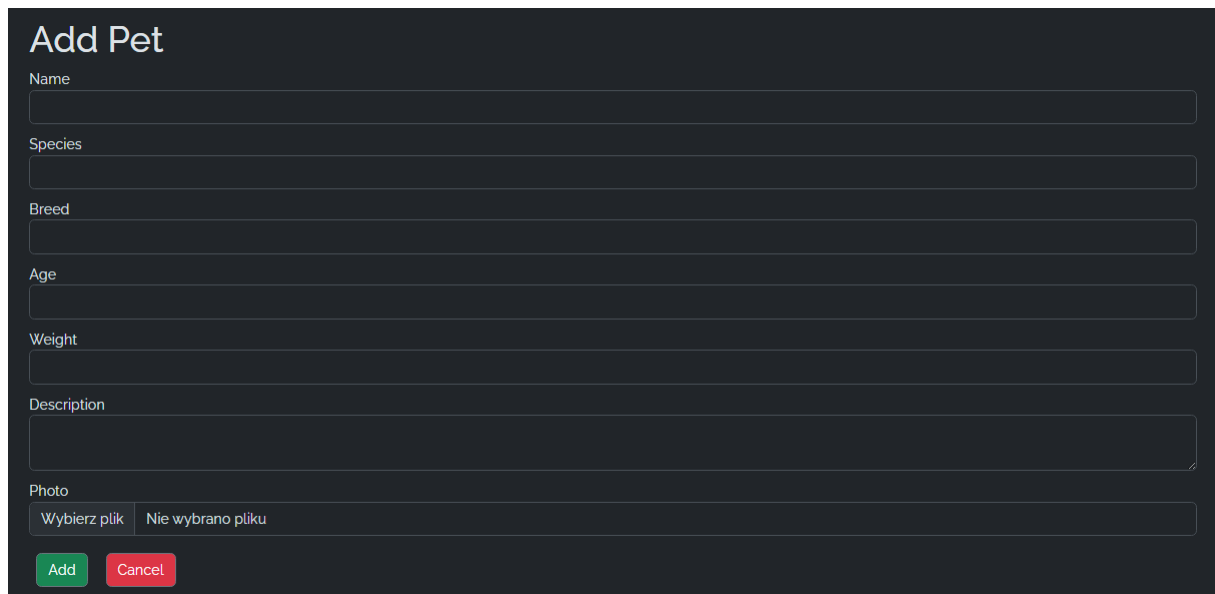
Friendly and energetic Labrador.

Photo

Wybierz plik Nie wybrano pliku

Save Cancel

- **Widok dodawania zwierzęcia:** Zawiera formularz identyczny jak w przypadku edycji danych zwierzęcia. Wprowadzone dane zatwierdzone są przyciskiem „Add” po czym następuje zapisanie danych do bazy, lub też jest możliwość anulowania dodawania nowego zwierzęcia poprzez kliknięcie przycisku „Cancel”



Add Pet

Name

Species

Breed

Age

Weight

Description

Photo

Wybierz plik Nie wybrano pliku

Add Cancel

IX. Widok szczepień (vaccinations.blade.php)

Widok ten składa się z dwóch sekcji. Pierwsza z nich zawiera formularz dodawania nowego szczepienia, druga zaś zawiera tabelę wyświetlającą wszystkie rekordy szczepień z bazy danych.

- **Sekcja dodawania nowego szczepienia:** Zawiera formularz, który poprzez podanie odpowiedniego ID zwierzęcia (Pet ID), daty szczepienia (Vaccination Date), typu szczepionki (Vaccine Type), serii szczepionki (Batch Number) umożliwia dodanie nowego rekordu szczepienia dla danego zwierzęcia. Dodawanie rekordu jest zatwierdzone przyciskiem „Add Vaccination”



Vaccinations Management

Pet ID

Vaccination Date

Vaccine Type

Batch Number

Add Vaccination

- **Sekcja wszystkich szczepień:** Sekcja ta zawiera tabelę wyświetlającą wszystkie szczepienia z bazy danych. Oprócz tego każdy wiersz zawiera przycisk do usuwania danego szczepienia.

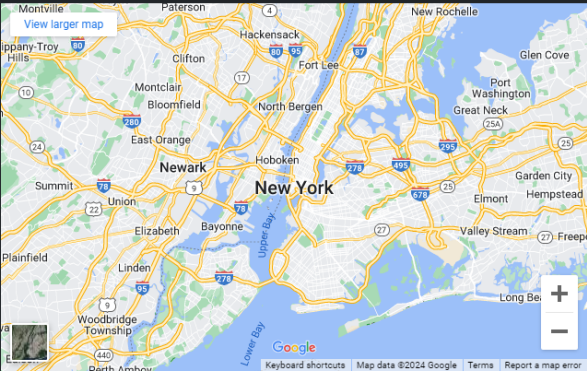
ID	Pet ID	Vaccination Date	Vaccine Type	Batch Number	Actions
1	1	2023-01-15	Rabies	R12345	Delete
2	2	2023-02-20	Feline Distemper	F67890	Delete
3	3	2023-03-25	Parvovirus	P11223	Delete
4	4	2023-04-30	Calicivirus	C44556	Delete
5	5	2023-05-15	Bordetella	B77889	Delete
6	6	2023-06-20	Leptospirosis	L99001	Delete
7	7	2023-07-25	Lyme Disease	L22334	Delete
9	9	2023-09-15	Feline Leukemia	F88900	Delete
10	10	2023-10-20	Canine Hepatitis	C12345	Delete
11	11	2023-11-25	Panleukopenia	P67890	Delete

X. Strona About Us (about.blade.php)

Strona ta składa się z dwóch sekcji. Pierwsza z nich zawiera krótki opis schroniska, druga zaś zawiera dane kontaktowe i adresowe wraz z mapą google ustawioną na konkretny adres.

About Us

Welcome to Pet Shelter, where we are dedicated to providing loving homes for animals in need. Our mission is to rescue, rehabilitate, and rehome pets that have been abandoned or neglected. We believe every animal deserves a chance to live a happy and healthy life. At Pet Shelter, we work tirelessly to ensure that every pet receives the care and attention they deserve. Our team of volunteers and professionals are committed to creating a safe and nurturing environment for all our animals, preparing them for their forever homes.



Contact Us

Address: 1234 Shelter St. Animal City, AC 56789

Email: petshelter@example.com

Phone 1: 123-456-7890

Phone 2: 987-654-3210

XI. Strona regulaminu schroniska (regulamin.blade.php)

Strona ta zawiera regulamin korzystania z aplikacji oraz usług schroniska

Terms and Conditions

1. Registration

- Any user wishing to utilize the services of Pet Shelter must register by providing their actual personal details, including their full name, address, and email address.
- Users are required to update their personal information on the service in the event of any changes.

2. Animal Selection

- Users have access to a wide selection of animals available for adoption or reservation at Pet Shelter.
- The status and availability of each animal are clearly marked.

3. Adoption and Reservation

- Adopting or reserving an animal is possible by completing the necessary forms and agreeing to the terms and conditions.
- After submitting the adoption or reservation request, the user receives confirmation along with details regarding the next steps and required documentation.
- The maximum period for a reservation is 2 days, after which the reservation will expire if not finalized.

4. Responsibility for Animal Welfare

- Customers are responsible for the well-being and proper care of the animal from the moment of adoption or during the reservation period.
- Any harm or neglect towards the animal during this period may result in legal action and additional charges.

5. Termination of Agreement

- Pet Shelter reserves the right to terminate the agreement in cases of breach of terms, including failure to provide adequate care for the animal.

7. Walidacja danych

W aplikacji "Pet Shelter" walidacja danych jest kluczowym elementem zapewniającym spójność bazy oraz poprawność i bezpieczeństwo operacji przeprowadzanych przez użytkowników. Laravel oferuje mechanizm walidacji, który pozwala na łatwe definiowanie reguł walidacji dla różnych żądań.

Walidacja dla różnych typów żądań

1. Dodawanie Zwierzęcia (storePet)

- **Opis:** Żądanie dodania nowego zwierzęcia do bazy danych.
- **Reguły walidacji:**
 - name: wymagane, tekst, maksymalnie 20 znaków.
 - species: wymagane, tekst, maksymalnie 20 znaków.
 - breed: wymagane, tekst, maksymalnie 50 znaków.
 - age: wymagane, liczba całkowita.
 - weight: wymagane, liczba.
 - description: wymagane, tekst, maksymalnie 250 znaków.
 - photo: wymagane, obrazek, formaty: jpeg, png, jpg, gif, svg, maksymalnie 2048 kB.

Kod:

```
public function storePet(Request $request)
{
    $request->validate([
        'name' => 'required|string|max:20',
        'species' => 'required|string|max:20',
        'breed' => 'required|string|max:50',
        'age' => 'required|integer',
        'weight' => 'required|numeric',
        'description' => 'required|string|max:250',
        'photo' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
    ]);
}
```

2. Aktualizacja Zwierzęcia (updatePet)

- **Opis:** Żądanie aktualizacji istniejącego zwierzęcia.
- **Reguły walidacji:**
 - name: wymagane, tekst, maksymalnie 20 znaków.
 - species: wymagane, tekst, maksymalnie 20 znaków.
 - breed: wymagane, tekst, maksymalnie 50 znaków.
 - age: wymagane, liczba całkowita.
 - weight: wymagane, liczba.

- description: wymagane, tekst, maksymalnie 250 znaków.
- photo: opcjonalne, obrazek, formaty: jpeg, png, jpg, gif, svg, maksymalnie 2048 kB.

Kod:

```
public function updatePet(Request $request, $id)
{
    $pet = Pet::findOrFail($id);

    $request->validate([
        'name' => 'required|string|max:20',
        'species' => 'required|string|max:20',
        'breed' => 'required|string|max:50',
        'age' => 'required|integer',
        'weight' => 'required|numeric',
        'description' => 'required|string|max:250',
        'photo' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
    ]);
}
```

3. Aktualizacja Użytkownika przez administratora (updateUser)

- **Opis:** Żądanie aktualizacji danych użytkownika.
- **Reguły walidacji:**
 - name: wymagane, tekst, maksymalnie 20 znaków, tylko litery.
 - last_name: wymagane, tekst, maksymalnie 30 znaków, tylko litery.
 - email: wymagane, format email, maksymalnie 100 znaków, unikalne w tabeli users z wyłączeniem bieżącego użytkownika.
 - phone_number: wymagane, tekst, maksymalnie 15 znaków, tylko cyfry.
- **Dodatkowa logika:**
 - Sprawdzenie, czy użytkownik ma jakiekolwiek adopcje. Jeśli tak, nie można przypisać mu roli admin.

Kod:

```
public function updateUser(Request $request, $id)
{
    $user = User::findOrFail($id);

    $request->validate([
        'name' => 'required|string|max:20|alpha',
        'last_name' => 'required|string|max:30|alpha',
        'email' => 'required|email|max:100|unique:users,email,' . $id,
        'phone_number' => 'required|string|max:15|regex:/^[0-9]+$/',
    ]);

    $hasAdoptions = Adoption::where('customer_id', $user->id)->exists();

    if ($request->has('admin') && $hasAdoptions) {
        return redirect()->route('admin.users')->withErrors(['admin' => 'Cannot assign admin role to a user with existing adoptions.']);
    }
}
```

4. Zmiana Hasła (changePassword)

- **Opis:** Żądanie zmiany hasła użytkownika.
- **Reguły walidacji:**
 - current_password: wymagane.
 - new_password: wymagane, tekst, minimalnie 8 znaków, zawiera małe i duże litery, cyfry oraz znak specjalny, musi być potwierdzone.

Kod:

```
public function changePassword(Request $request)
{
    $request->validate([
        'current_password' => 'required',
        'new_password' => 'required|confirmed|min:8|regex:/[a-z]/|regex:/[A-Z]/|regex:/[0-9]/|regex:/[!@#$%&?&#&]/',
    ], [
        'current_password.required' => 'Current password is required.',
        'new_password.required' => 'New password is required.',
        'new_password.confirmed' => 'Passwords do not match.',
        'new_password.min' => 'Password must be at least 8 characters.',
        'new_password.regex' => 'Password must contain at least one uppercase letter, one lowercase letter, one digit, and one special character.',
    ]);

    $user = Auth::user();

    if (!Hash::check($request->current_password, $user->password)) {
        return back()->withErrors(['current_password' => 'Current password does not match.']);
    }
}
```

5. Logowanie (authenticate)

- **Opis:** Żądanie logowania użytkownika.
- **Reguły walidacji:**
 - email: wymagane, format email, maksymalnie 100 znaków.
 - password: wymagane, tekst, minimalnie 8 znaków, zawiera małe i duże litery, cyfry oraz znak specjalny.

Kod:

```
public function authenticate(Request $request)
{
    $credentials = $request->validate([
        'email' => ['required', 'email', 'max:100'],
        'password' => ['required', 'string', 'min:8', 'regex:/[a-z]/', 'regex:/[A-Z]/', 'regex:/[0-9]/', 'regex:/[!@#$%&?&#&]/'],
    ]);
}
```

6. Rejestracja (register)

- **Opis:** Żądanie rejestracji nowego użytkownika.
- **Reguły walidacji:**
 - name: wymagane, tekst, maksymalnie 20 znaków, tylko litery.
 - last_name: wymagane, tekst, maksymalnie 30 znaków, tylko litery.
 - email: wymagane, format email, maksymalnie 100 znaków, unikalne w tabeli users.
 - password: wymagane, tekst, minimalnie 8 znaków, zawiera małe i duże litery, cyfry oraz znak specjalny, musi być potwierdzone.

- phone_number: wymagane, tekst, maksymalnie 15 znaków, tylko cyfry.

Kod:

```
public function register(Request $request)
{
    $validatedData = $request->validate([
        'name' => 'required|string|max:20|alpha',
        'last_name' => 'required|string|max:30|alpha',
        'email' => 'required|email|max:100|unique:users,email',
        'password' => [
            'required',
            'confirmed',
            'min:8',
            'regex:/[a-z]/',
            'regex:/[A-Z]/',
            'regex:/[0-9]/',
            'regex:/[@$!%*#?&]/'
        ],
        'phone_number' => 'required|string|max:15|regex:/^[0-9]+$/ ',
    ], [
```

7. Dodawanie Szczepienia (storeVaccination)

- **Opis:** Żądanie dodania nowego szczepienia do bazy danych.
- **Reguły walidacji:**
 - pet_id: wymagane, musi istnieć w tabeli pets.
 - vaccination_date: wymagane, data, nie wcześniejsza niż 1 stycznia 2020, nie późniejsza niż dzisiaj.
 - vaccine_type: wymagane, tekst, maksymalnie 50 znaków.
 - batch_number: wymagane, tekst, maksymalnie 10 znaków.

Kod:

```
public function storeVaccination(Request $request)
{
    $request->validate([
        'pet_id' => 'required|exists:pets,id',
        'vaccination_date' => 'required|date|after_or_equal:2020-01-01|before_or_equal:today',
        'vaccine_type' => 'required|string|max:50',
        'batch_number' => 'required|string|max:10',
    ]);
}
```

Podsumowanie

Mechanizmy walidacji zastosowane w aplikacji "Pet Shelter" gwarantują poprawność danych wprowadzanych przez klientów oraz administratorów na każdym etapie korzystania z aplikacji. Dzięki precyzyjnie zdefiniowanym regułom walidacji, zarówno po stronie klienta, jak i serwera, aplikacja działa sprawnie i bezpiecznie. Każdy proces, od rejestracji i logowania, przez dodawanie i zarządzanie

zwierzętami, po zmiany w profilach użytkowników oraz zarządzanie adopcjami i szczepieniami, jest odpowiednio chroniony przed błędami danych. Dzięki temu aplikacja minimalizuje ryzyko błędów, zapewnia integralność danych oraz oferuje pozytywne doświadczenie użytkownikom.