



**Politechnika
Śląska**

PROJEKT INŻYNIERSKI

Aplikacja wspomagająca bayesowską filtrację obrazów cyfrowych

Radosław BRYŚ

Nr albumu: 262833

Kierunek: Informatyka

Specjalność: Grafika Komputerowa i Oprogramowanie

PROWADZĄCY PRACĘ

dr inż. Alina Momot

KATEDRA Informatyki Stosowanej

Wydział Automatyki, Elektroniki i Informatyki

Gliwice 2024

Tytuł pracy

Aplikacja wspomagająca bayesowską filtrację obrazów cyfrowych

Streszczenie

Celem prac było wykonanie aplikacji implementującej filtr bayesowski. Projekt składa się z trzech współpracujących ze sobą aplikacji i bazy danych MySQL. Większość projektu została wykonana w języku JavaScript. Jedynie implementacje filtrów bayesowskiego i medianowego zostały wykonane w języku C. W wyniku prac uzyskano działającą aplikację internetową, przy użyciu której można filtrować obrazy.

Słowa kluczowe

przetwarzanie obrazów, filtr bayesowski, JavaScript, aplikacja internetowa

Thesis title

Application Supporting bayesian Digital Image Filtering

Abstract

The aim of the project was to create an application implementing a Bayesian filter. The project consists of three cooperating applications and a MySQL database. The majority of the project was developed in JavaScript. Only the implementations of the Bayesian and median filters were done in C. As a result of the work, a functional web application was obtained, which allows for image filtering.

Key words

image processing, bayesian filter, JavaScript, web application

Spis treści

1	Wstęp	1
1.1	Wprowadzenie	1
1.2	Cel i zakres pracy	1
1.3	Charakterystyka rozdziałów	2
2	Analiza tematu	3
2.1	Podstawy	3
2.1.1	Czym jest obraz	3
2.1.2	Czym jest bitmapa	3
2.1.3	Cyfrowe przetwarzanie obrazów	4
2.2	O filtrowaniu obrazów	4
2.2.1	Filtracja liniowa	5
2.2.2	Filtry dolnoprzepustowe	5
2.2.3	Filtry górnoprzepustowe	6
2.2.4	Filtr medianowy	6
2.2.5	Filtr bayesowski	6
2.3	Istniejące rozwiązania	8
2.3.1	Photofilters	8
2.3.2	PhotoMania	9
2.3.3	Snapchat	10
3	Wymagania i narzędzia	11
3.1	Wymagania funkcjonalne	11
3.2	Wymagania нефункционалне	12
3.3	Przypadki użycia	12
3.4	Wykorzystane narzędzia	13
3.4.1	JavaScript	13
3.4.2	Node.js	14
3.4.3	npm	15
3.4.4	Express.js	15
3.4.5	jimp	15

3.4.6	React.js	15
3.4.7	Vite	16
3.4.8	Git	16
3.4.9	ESLint	16
3.4.10	C	17
3.4.11	MySQL	17
3.4.12	Postman	17
4	Specyfikacja zewnętrzna	19
4.1	Wymagania sprzętowe i systemowe	19
4.2	Instalacja	19
4.2.1	Pobranie repozytorium	19
4.2.2	Przygotowanie bazy danych	20
4.2.3	Instalacja aplikacji serwerowej	21
4.2.4	Instalacja aplikacji klienckiej	21
4.3	Sposób aktywacji	21
4.4	Kategorie użytkowników	22
4.5	Sposób obsługi	22
4.5.1	Zwykły użytkownik	22
4.5.2	Administrator	25
4.6	Zabezpieczenia	26
4.7	Działanie filtra bayesowskiego	27
4.8	Redukcja szumów	28
5	Specyfikacja wewnętrzna	31
5.1	Architektura aplikacji	31
5.2	Struktura danych	31
5.3	Przepływ danych	32
5.4	Ważniejsze algorytmy	34
5.5	Zabezpieczenia	37
5.6	Wykorzystane moduły i biblioteki	37
5.6.1	pthread.h	37
5.6.2	body-parser	37
5.6.3	cors	37
5.6.4	dotenv	37
5.6.5	js-sha256	38
5.6.6	mysql2	38
5.6.7	sanitize-filename	38
5.6.8	uuid	38

6	Weryfikacja i walidacja	39
6.1	Środowisko testowe	39
6.2	Testowanie w trakcie tworzenia	40
6.2.1	Aplikacja filtrująca	40
6.2.2	Aplikacja backendowa	40
6.2.3	Aplikacja frontendowa	40
6.3	Wykryte błędy	40
6.4	Testy bezpieczeństwa	41
7	Podsumowanie i wnioski	43
7.1	Weryfikacja założeń	43
7.2	Potencjalne kierunki rozwoju aplikacji	43
7.3	Wady projektu	44
	Bibliografia	46
	Spis skrótów i symboli	49
	Lista dodatkowych plików, uzupełniających tekst pracy	51
	Spis rysunków	53

Rozdział 1

Wstęp

1.1 Wprowadzenie

Wzrok jest jednym z pięciu podstawowych zmysłów człowieka. Umożliwia on swobodne poruszanie się w fizycznym otoczeniu. Funkcjonowanie bez niego jest znacznie utrudnione. Posiadanie wzroku umożliwia również cieszenie się z krajobrazów, dokonywanie ekspresji artystycznej wizualnej i generalne korzystanie z obrazów.

Projekt ten dotyczy właśnie strefy wizualnej. W tej pracy autor zaprezentuje prostą aplikację, która umożliwi wyostrażanie, wygładzanie, uwypuklanie obrazów. Umożliwi ona również usuwanie szumów i wykrywanie krawędzi w obrazie. Dokona on tego dzięki zastosowaniu służących do tego filtrów.

Nie będzie to aplikacja szczególnie zaawansowana. Ciężko będzie ją porównać z istniejącymi w internecie rozwiązaniami takimi jak strona photofilters.com. Nie będzie ona też w takim stopniu rozbudowana jak popularny, służący rozrywce Snapchat.

Będzie to natomiast prezentacja umiejętności i talentu autora oraz przykład praktycznego zastosowania użytych w tym projekcie technologii.

1.2 Cel i zakres pracy

Celem tej pracy jest wytworzenie aplikacji komputerowej służącej do filtrowania obrazów cyfrowych. Ma to być implementacja szeregu prostych filtrów oraz filtru bayesowskiego, który zostanie opisany w dalszych rozdziałach. Będzie to aplikacja webowa składająca się z czterech części.

Aplikacji przeglądarkowej, w której będzie można wybrać znajdujący się na dysku użytkownika plik oraz wybrać filtr, który zostanie wykorzystany. Następnie wybrana grafika będzie przesyłana w żądaniu do serwera, który wykona filtrację i odeśle nowy obraz. Aplikacja przeglądarkowa będzie również umożliwiała logowanie się zarejestrowanych użytkowników. Umożliwi ona również administratorom dodawanie i usuwanie użytkowni-

ków, i przeglądanie ich listy.

Aplikacja serwerowa będzie odpowiadała za nasłuchiwanie żądań ze strony aplikacji webowej, wysyłanie żądań do bazy danych i wywoływanie aplikacji filtrującej, gdy ze strony przeglądarki internetowej przysłany zostanie plik i prośba o filtrację.

Baza danych będzie zawierać tablice użytkowników i administratorów oraz hasze haseł do logowania tychże użytkowników.

Aplikacja filtrująca będzie swego rodzaju bazą filtrów i algorytmów. Będzie ona mogła być samodzielnie wywoływana z linii komend. Będzie przyjmować jako parametry nazwę oznaczającą dany filtr, ścieżkę do pliku graficznego, który ma być przetworzony oraz ścieżkę z nazwą pliku wyjściowego.

Jest to praca jednego autora, który projekt i dokumentację wykona samodzielnie. Wykorzystał w tym celu takie technologie jak: język programowania JavaScript, język programowania C, środowisko uruchomieniowe Node, biblioteka React i platformy Express. React jest biblioteką języka JavaScript, która została wykorzystana do napisania strony internetowej będącej aplikacją kliencką w tym projekcie. Platforma Express została natomiast wykorzystana do napisania aplikacji serwerowej, również w języku JavaScript. Szczegółowy opis tych i innych technologii użytych w tym projekcie zostanie zamieszczony w dalszych rozdziałach.

1.3 Charakterystyka rozdziałów

Pierwszy rozdział stanowi wstęp do pracy. Przedstawione zostały w nim ogólne założenia projektu. W drugi rozdział stanowi część teoretyczną dokumentu. Zawarto w nim wprowadzenie w dziedzinę, implementowane algorytmy wraz z opisem oraz przegląd istniejących rozwiązań. Trzeci rozdział skupia się na wymaganiach projektowych i użytych narzędziach. Czwarty rozdział opisuje wymagania sprzętowe aplikacji, instrukcję instalacji oraz przykłady użycia aplikacji. Piąty rozdział skupia się na strukturze wewnętrznej aplikacji, jej działaniu i użytych bibliotekach. Szósty rozdział opisuje sposób wykonania i testowania aplikacji. Siódmy stanowi zakończenie pracy.

Rozdział 2

Analiza tematu

2.1 Podstawy

2.1.1 Czym jest obraz

Jednymi z najstarszych dobrze znanych obrazów są malowidła naskalne z Lascaux. Przedstawiają one bogatą prehistoryczną faunę i sceny polowania. Są one zatem pewnym zapisem doświadczeń i postrzegania świata w umysłach praludzi, nawet jeżeli nie są zbyt obiektywnym i dokładnym przedstawieniem materialnej rzeczywistości. Zwierzęta są koślawe a łowcy przypominają patyczaki, ale to nie jest problemem, gdyż obraz jest pewnego rodzaju abstrakcją - abstrakcją rzeczywistości, która jest dobra tak długo, jak jest zrozumiała dla odbiorcy.

Obrazy mogą również przedstawiać idee (np. \$ oznacza walutę dolara), dźwięki (litery alfabetu), symbolizować organizacje (np. godło państwa) lub znane marki. Generalnie są one podstawową, drugą po mowie, formą komunikacji współczesnego człowieka.

2.1.2 Czym jest bitmapa

Obrazy mogą przyjmować różne postacie. Mogą to być rzeźby, płaskorzeźby, obrazy olejne na płótnie, litery kute w skale, nuty na pięciolinii, ale mogą to być też obrazy cyfrowe wyświetlane przez tysiące pikseli na ekranach komputerów, telefonów, telewizorów. Podstawową formą przechowywania tych obrazów są bitmapy zapisane w pamięci urządzeń cyfrowych. Istnieją również grafiki zapisane w postaci wektorowej, ale ta praca ich nie dotyczy.

Bitmapa, zwana także piksel-mapą, jest to tablica bitów, w której kolejne ciągi bitów kodują kolejne piksele obrazu. Piksel jest to najmniejszy element rastra obrazu do którego można się odwołać. Do przedstawiania obrazu stosuje się przeważnie siatki kwadratowe.

Przykładowym formatem pliku wykorzystującego bitmapę jest format BMP [8]. Plik tegoż formatu składa się między innymi z trzech głównych części. Nagłówek zawierającego

ogólne i szczegółowe informacje o bitmapie, palety kolorów definiującej zakres obsługiwanych barw oraz tablicy pikseli zawierającej samą bitmapę. Program graficzny mając dostęp do tych informacji jest w stanie wygenerować na wyświetlaczu zdefiniowany w pliku obraz.

Zależnie od zakresu wyświetlanych barw jeden piksel w formacie .bmp może być reprezentowany przez szeroki zakres bitów. Przez jeden bit w przypadku obrazów czarno-białych składających się jedynie z pikseli czarnych i pikseli białych. Przez osiem bitów w przypadku obrazu czarno-białego zawierającego 256 odcieni, od czerni przy wartości 0, do bieli przy wartości 255. Generalnie osiem bitów to jeden bajt i on reprezentuje wartości binarne w zakresie od 0 do 255. 24-bitowy format koduje obraz o przestrzeni kolorów sRGB (ang. standarised Red, Green, Blue, najpowszechniej wykorzystywana przestrzeń barw składająca się z ponad 16 mln odcieni). Każdy z trzech bajtów koduje wartość jednej z barw składowych. Mianowicie kolory czerwony, zielony i niebieski. 32-bitowy obraz natomiast definiuje również wartość alfa danego piksela, czyli jego przeźroczystość.

2.1.3 Cyfrowe przetwarzanie obrazów

Cyfrowe przetwarzanie obrazów to ogólnie mówiąc wykorzystywanie komputerów cyfrowych do przetwarzania obrazów cyfrowych przy wykorzystaniu danych algorytmów. Operacje przetwarzania obrazów można podzielić na [15]: przekształcenia punktowe, przekształcenia geometryczne, przekształcenia kontekstowe, przekształcenia widmowe i przekształcenia morfologiczne.

W przekształceniach punktowych piksele zmieniane są niezależnie od elementów sąsiednich. Są to np. operacje przyciemniania lub rozjaśniania obrazu. Przesunięcia obrazu lub odbicia względem danej osi są przekształceniami geometrycznymi.

Przekształcenia poprzez użycie filtrów są przekształceniami kontekstowymi. Polegają one na wyznaczeniu nowej wartości elementu zależnie od poprzedniej wartości elementu i jego otoczenia. To właśnie filtracji obrazów będzie dotyczyć ta praca.

Przekształcenia widmowe wykorzystują transformację Fouriera do wytworzenia widma obrazu. Następnie na tym widmie wykonywane są dane operacje (np. usunięcie elementów o niskiej częstotliwości), by następnie poprzez odwrotną transformację Fouriera zrekonstruować obraz.

W przekształceniach morfologicznych element obrazu jest modyfikowany tylko wtedy, gdy spełniony zostanie zadany warunek logiczny. Są one zwykle wykonywane iteracyjnie, aż do spełnienia zadanego warunku logicznego [15].

2.2 O filtrowaniu obrazów

Filtrowanie jest jedną z podstawowych metod przetwarzania obrazów. Są to operacje kontekstowe. Dla uzyskania wartości punktu trzeba wykonać obliczenia na wielu elemen-

tach sąsiadujących. Najczęściej otoczenie piksela definiowane jest jako kwadratowa maska składająca się z tegoż piksela i wszystkich otaczających go pikseli. Te maski mogą mieć wymiar 3 na 3 piksele, 5 na 5, 7 na 7 itd.

Z powodu kontekstowości filtracji obrazu, operacja ta nie może być wykonywana na elementach leżących na brzegach bitmapy. Jest kilka możliwych sposobów poradzenia sobie z tym problemem. Można pominąć filtrację dla obrzeżnych pikseli i skopiować je do obrazu wynikowego niezmiennie. Można te piksele usunąć i pomniejszyć obraz wynikowy. Można też dołożyć do obrazu filtrowanego ramkę z pikseli o określonej wartości (np. czarnego koloru) i wtedy dokonać filtracji. W tym projekcie autor postanowił pominąć filtrację dla elementów granicznych i przekopiować je takie jakie są.

Proces filtracji przeprowadza się na wszystkich składowych obrazu osobno, tzn. Jeżeli każdy piksel składa się z czterech bajtów oznaczających składowe R, G, B i alfa, to algorytm wykonuje się na każdej z tych składowych.

Cały proces filtracji jest szczegółowo opisany w „Komputerowa analiza i przetwarzanie obrazów” [15].

2.2.1 Filtracja liniowa

Filtracja liniowa jest najprostszym rodzajem filtracji. Dla każdego punktu obrazu wylicza się średnią ważoną z jego otoczenia. Wagi są zapisywane w postaci macierzy. Poniżej przedstawiona jest filtracja na podstawie filtra o masce 3 na 3 (wzory pochodzą z „Filtrowanie obrazów - Algorytmy i Struktury Danych” [11]) :

$$\begin{bmatrix} w_{-1,-1} & w_{0,-1} & w_{1,-1} \\ w_{-1,0} & w_{0,0} & w_{1,0} \\ w_{-1,1} & w_{0,1} & w_{1,1} \end{bmatrix} \quad (2.1)$$

gdzie wartości $w_{-1,-1}, w_{0,-1}, \dots, w_{0,1}, w_{1,1}$ stanowią wagi dla piksela $a_{i,j}$ i jego sąsiadów. Natomiast wartość piksela $a_{i,j}$ po dokonaniu filtracji opisana jest wzorem:

$$a'_{i,j} = \frac{s}{\sum_{n=-1}^1 \sum_{m=-1}^1 w_{n,m}} \quad (2.2)$$

gdzie:

$$s = \sum_{n=-1}^1 \sum_{m=-1}^1 w_{n,m} a_{i+n,j+m} \quad (2.3)$$

2.2.2 Filtry dolnoprzepustowe

Działanie filtrów dolnoprzepustowych charakteryzuje się przepuszczaniem elementów o małej częstotliwości. Służą one do eliminacji zakłóceń obrazu. Filtry tego typu powodują

również efekt rozmycia obrazu co jest niepożądanym skutkiem ubocznym. Rozmycie to jest najsilniejsze w przypadku prostego filtra uśredniającego. By temu zapobiec stosuje się filtry ważone ze zwiększoną wartością wagi dla centralnych pikseli.

Filtr uśredniający jest najprostszym filtrem dolnoprzepustowym. Wszystkie wagi jego maski mają wartość $w_{nm} = 1$.

2.2.3 Filtry górnoprzepustowe

Służą one do wykrywania elementów powodujących szybkie zmiany wartości. Wykrywają krawędzie, kontury i drobne elementy. Powodują wizualne wyostrenie obrazu kosztem wzmocnienia zakłóceń i szumów, co jest odwrotnym działaniem w stosunku do filtrów dolnoprzepustowych. Przykładowa maska filtra górnoprzepustowego została przedstawiona poniżej:

$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix} \quad (2.4)$$

2.2.4 Filtr medianowy

Filtr medianowy jest filtrem nieliniowym i nie ma wyznaczonej maski. Działa on poprzez analizowanie danego obszaru obrazu wejściowego, w którego centrum znajduje się filtrowany aktualnie piksel. Algorytm ten wybiera piksel, którego wartość jest medianą, spośród analizowanego zbioru i wstawia go do obrazu wyjściowego. Służy on do tłumienia szumów. Jeśli dany jest dyskretny zestaw danych x , to filtr medianowy można zdefiniować jako:

$$y_n = \text{median}(x_{n-k}, \dots, x_n, \dots, x_{n+k}) \quad (2.5)$$

gdzie $N = 2k + 1$ to długość okna filtra. Filtr medianowy zawsze wybiera już istniejący punkt z danych, a nie tworzy nowy [3].

2.2.5 Filtr bayesowski

Sekcja ta dotyczy nowego algorytmu zaproponowanego w czasopiśmie „Studia Informatica” w artykule „Filtracja obrazów cyfrowych z wykorzystaniem bayesowskiego ważonego uśredniania” [12] przez panią dr inż. Alinę Momot. Jest to algorytm służący do wyznaczania wartości maski filtra konwolucyjnego. Filtr ten ma służyć do tłumienia szumów podobnie jak filtr uśredniający, czy filtr medianowy.

Filtr uśredniający cechuje się niszczeniem drobnych szczegółów i krawędzi przetwarzanych obrazów. Filtr ten wprowadza też szумы i zakłócenia poprzez ich „rozmywanie” na większym obszarze.

Filtr medianowy jest lepszy od filtra uśredniającego pod tym względem, że nie wprowadza zakłóceń do obrazu wyjściowego, ani nie niszczy drobnych szczegółów tak bardzo jak ten drugi. Niestety ma on także większą złożoność obliczeniową.

Filtr bayesowski to „adaptacyjny rozmyty filtr ważonego uśredniania, który rozpatruje piksele w „oknie” filtru jako zbiór rozmyty i każdy piksel w tym „oknie” jest charakteryzowany funkcją przynależności stanowiącą właściwą wagę tego piksela” [12]. Został on zaproponowany jako alternatywa dla filtru uśredniającego i filtru medianowego. Opracowano go na podstawie algorytmu EBWA (ang. empirical bayesian weighted averaging), który profesjonalnie wykorzystywany jest by tłumić zakłócenia w sygnale elektrokardiograficznym. Zakładając, że:

$$t = [f(x - R, y - R), \dots, f(x + R, y + R)] \quad (2.6)$$

gdzie t to wektor zawierający wszystkie piksele maski, R to promień maski, $g(x, y)$ (które obliczamy) to wartość piksela wyjściowego, algorytm można opisać następującymi krokami:

1. Oblicz $g(x, y)^{(0)}$ jako średnią arytmetyczną wektora t . Oblicz wariancję dla średniej arytmetycznej i jeśli jest ona większa od zera, to ustawić $k = 1$ i rozpocznij iterację.
2. Oblicz parametr $\beta^{(k)}$ i parametry $\alpha_i^{(k)}$ dla $i = 1, 2, \dots, D$. D to liczba elementów w wektorze t .

$$\beta^{(k)} = (g(x, y)^{(k-1)})^{-2} \quad (2.7)$$

$$\alpha_i^{(k)} = (t_i - g(x, y)^{(k-1)})^{-2} \quad (2.8)$$

3. Oblicz uśrednioną wartość k -tej iteracji $g(x, y)^{(k)}$

$$g(x, y)^{(k)} = \frac{\sum_{i=1}^D \alpha_i^{(k)} t_i}{\beta^{(k)} + \sum_{i=1}^D \alpha_i^{(k)}} \quad (2.9)$$

4. Sprawdź czy $(g(x, y)^{(k)} - g(x, y)^{(k-1)})^2 > \epsilon$, jeśli tak to ustaw $k = k + 1$ i kontynuuj od punktu 2. ϵ jest to parametr ustawiony przez użytkownika.

Powyższy algorytm jest wykonywany osobno na wszystkich trzech składowych piksela. Wartości składowych piksela mieszczą się w zakresie od 0 do 255. Parametr $\beta^{(k)}$ jest zawsze dodatni. Natomiast parametr $\alpha_i^{(k)}$ może być nieokreślony. W przypadku gdy $t_i -$

$g(x, y)^{(k-1)} = 0$ parametr $\alpha_i^{(k)}$ należy ustawić na wartość wyższą niż zwykle (np. 255). Składowa piksela pod indeksem i ma wtedy wartość średniej $g(x, y)^{(k)}$ w tej iteracji.

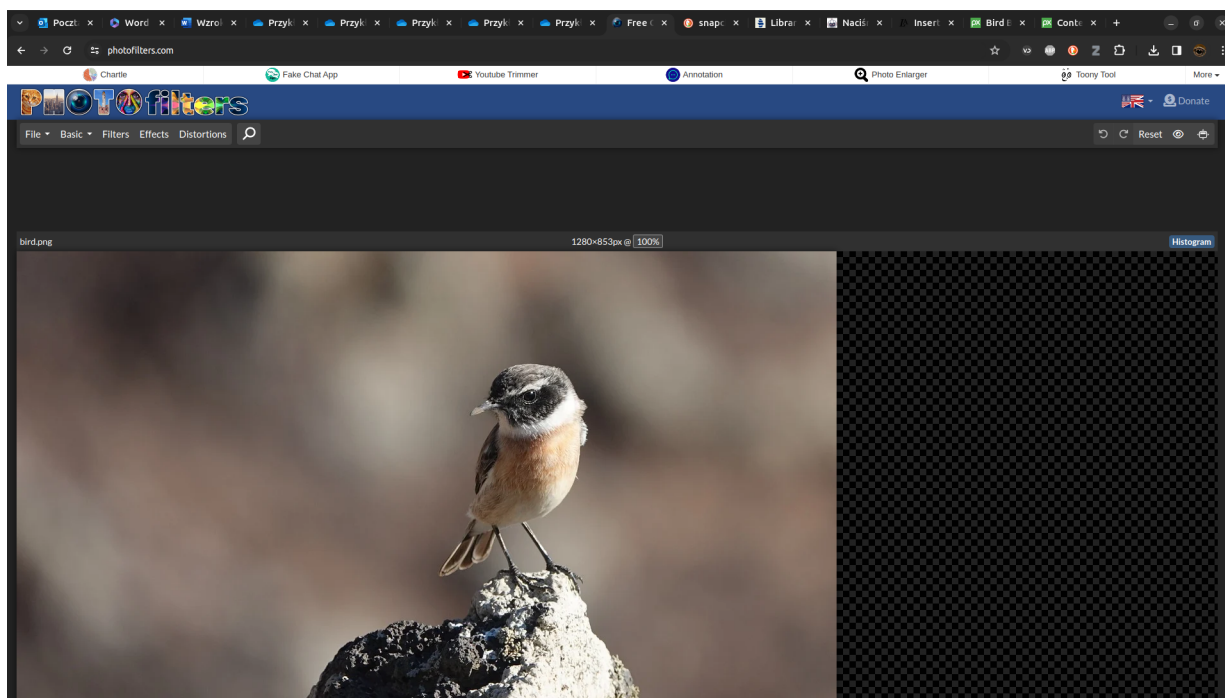
2.3 Istniejące rozwiązania

W internecie dostępne są liczne aplikacje umożliwiające filtrację obrazów, lecz implementują przeważnie filtry artystyczne. Są to bardziej zaawansowane filtry dające szeroki zakres efektów. Umożliwiają modyfikację palety barw, wykrywanie krawędzi, tworzenie efektu obrazu w podczerwieni i wiele innych.

Aplikacja którą opisuje ten dokument zawiera natomiast implementację prostych filtrów liniowych i filtru bayesowskiego. Implementacji filtru bayesowskiego nie znajdziemy natomiast w internecie.

2.3.1 Photofilters

Photofilters (Rysunek 2.1) to strona internetowa bardzo podobna do aplikacji tworzonej przez autora pracy. Znaleźć ją można pod adresem: <https://www.photofilters.com/>. Można w niej wybrać obraz z dysku, po czym wyświetli się on w podglądzie poniżej paska narzędzi. W zakładce „Basic” można wybrać m.in. takie podstawowe operacje jak przycięcie obrazu do oczekiwanego rozmiaru, rotacja obrazu lub zmiana temperatury barw.



Rysunek 2.1: Zrzut ekranu przedstawiający stronę photofilters.com.

W zakładce „Filters” znajduje się możliwość wyboru któregoś, z wielu rozmaitych za-

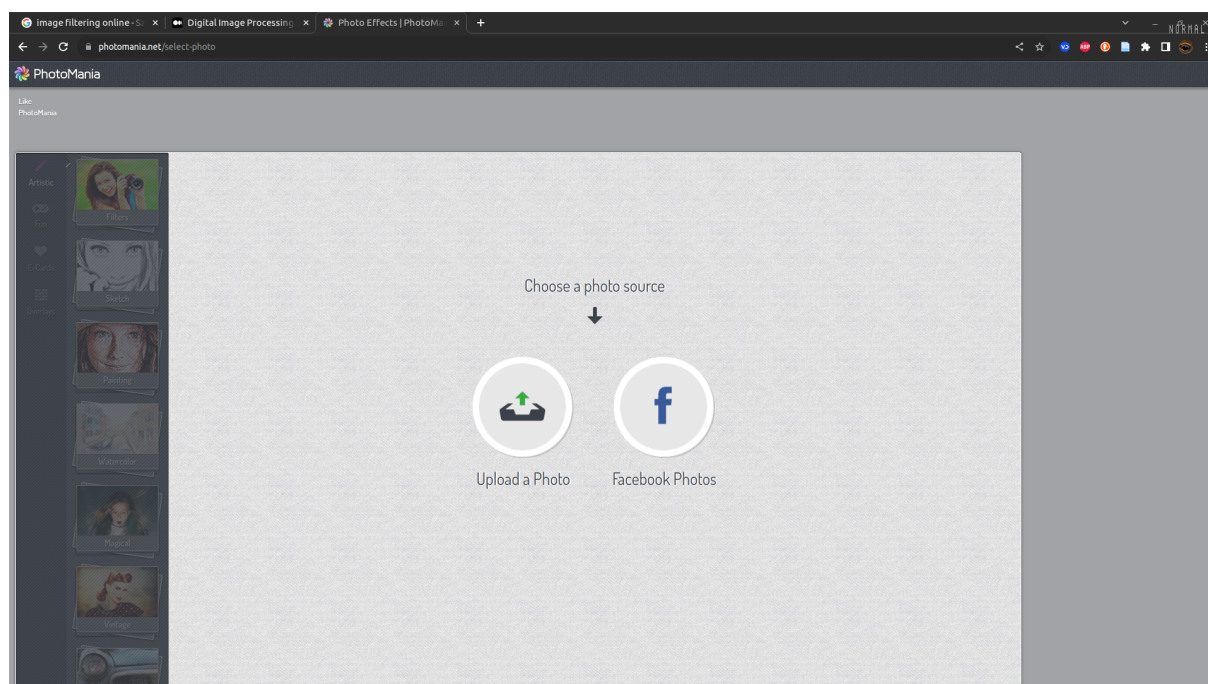
awansowanych filtrów. Można tu znaleźć filtr wykrywający krawędzie we wszystkich kierunkach i z dużą dokładnością. Daje to dużo lepszy efekt od filtrów zaimplementowanych przez autora pracy. W zakładce „Effects” natomiast można wybrać różne zniekształcenia, które sprawiają wrażenie odbicia w „krzywym zwierciadle”.

Aplikacja Photofilters nie posiada natomiast możliwości rejestrowania czy logowania użytkowników. Odróżnia ją to od projektu autora pracy.

2.3.2 PhotoMania

Photomania (Rysunek 2.2) to aplikacja webowa bardzo podobna do photofilters.com. Znaleźć ją można pod adresem: <https://photomania.net/select-photo>. Służy ona do nakładania na zdjęcia różnych filtrów i efektów o charakterze artystycznym. Zdjęcia można pobrać z dysku. Aplikacja ta posiada również możliwość połączenia się z facebookiem i pobrana znajdujących się tam zdjęć. Niestety w czasie pisania tej pracy funkcjonalność ta była wyłączona.

Część efektów, z których można skorzystać wymaga instalacji specjalnego rozszerzenia do przeglądarki Google Chrome. Nie jest to rozwiązanie korzystne dla użytkownika, gdyż „zaśmieca” ono jego przeglądarkę. W rozszerzeniach mogą się również znajdować luki zabezpieczeń, które mogą być wykorzystane do kradzieży danych i haseł użytkownika.



Rysunek 2.2: Zrzut ekranu przedstawiający aplikację PhotoMania.

2.3.3 Snapchat

Jest to aplikacja i usługa multimedialna służąca do przesyłania zdjęć i wiadomości między użytkownikami. Cechą charakterystyczną tej aplikacji jest to, że zdjęcia i wiadomości przesłane są dostępne przez krótki czas [16]. Jest to przede wszystkim aplikacja na smartfony tworzona w filozofii „mobile first”.

Aplikacja ta posiada funkcjonalność „snapchat stories”. Polega ona na tym, że materiały opublikowane jako „story” są publicznie dostępne przez 24 godziny. W tym czasie użytkownicy mogą oglądać tę treść dowolną ilość razy [13].

Zdjęcia w snapchacie mogą być personalizowane przy pomocy różnych efektów wizualnych i naklejek. Snapchat w swoich filtrach, znanych jako „soczewki” (ang. lenses), wykorzystuje technologię wykrywania twarzy, by w czasie rzeczywistym, w podglądzie, renderować np. psi nos czy kocie uszy. Część filtrów i naklejek dostępna jest tylko w określonych miejscach. Znane są one jako geofiltry.

Snapchat posiada możliwość rejestracji i logowania użytkowników, którzy mogą w aplikacji, w bazach danych snapchata, przechowywać swoje zdjęcia i materiały wideo. Materiały te są szyfrowane.

Aplikacja ta posiada również szereg pomniejszych funkcjonalności. Szczegółowe informacje o nich i o historii tej aplikacji można znaleźć na poświęconej snapchatowi stronie wikipedii [16].

Rozdział 3

Wymagania i narzędzia

3.1 Wymagania funkcjonalne

Wymagania funkcjonalne to wymagania, które opisują jakie funkcje spełniać ma aplikacja, nie określają one tego, w jaki sposób ma ona te funkcje wykonywać. Aplikacja powinna spełniać poniższe wymagania funkcjonalne:

- Użytkownicy mają możliwość zalogowania się do aplikacji.
- Użytkownicy mogą posiadać uprawnienia administratorskie.
- Aplikacja ma weryfikować logujących się użytkowników i administratorów.
- Aplikacja ma umożliwiać przysyłanie plików graficznych, które mają zostać przefiltrowane.
- Wyświetlony ma być podgląd przesłanego pliku graficznego.
- Wyświetlona ma być przefiltrowana grafika.
- Aplikacja ma mieć zaimplementowany szereg przykładowych filtrów liniowych, filtr medianowy i filtr bayesowski.
- Administrator ma mieć dostęp do specjalnego panelu administratora.
- Administrator ma mieć możliwość dodawania nowych użytkowników.
- Administrator ma mieć dostęp do listy istniejących użytkowników.
- Administrator ma mieć możliwość usuwania istniejących użytkowników.
- Administrator ma mieć możliwość zmiany uprawnień użytkowników.
- Wszystkie operacje wykonywane przez użytkowników i administratorów mają być autoryzowane.

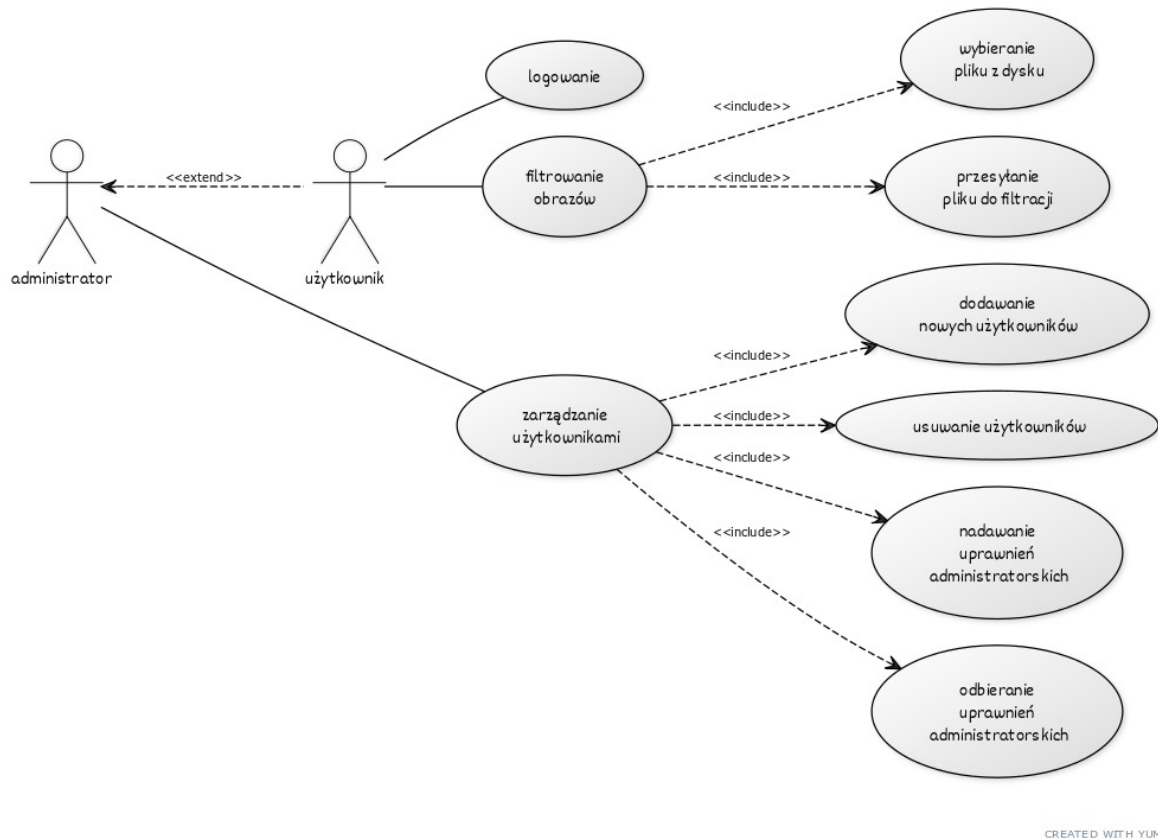
3.2 Wymagania niefunkcjonalne

Wymagania niefunkcjonalne określają sposób działania danego systemu, czyli w jaki sposób spełnia on swoje funkcje. Aplikacja powinna spełniać następujące wymagania niefunkcjonalne:

- Aplikacja ma być zrealizowana w modelu client-server.
- Aplikacja ma być zrealizowana zgodnie z wzorcem projektowym MVC (ang. model view controller).
- Widokiem ma być strona internetowa zrealizowana z użyciem biblioteki React.
- Kontrolerem ma być aplikacja serwerowa zrealizowana przy użyciu platformy programistycznej Express.
- Modelem ma być tablica użytkowników w relacyjnej bazie danych MySQL.
- Po stronie serwera ma być zrealizowany osobny program służący do filtrowania obrazów. Będzie on wywoływany przez kontroler.
- Algorytmy medianowy i bayesowski, ze względu na swoją złożoność, mają być zrealizowane wielowątkowo w języku C.
- Elementy napisane w języku Java Script mają być zgodne ze standardem ES2021.
- Komunikacja pomiędzy klientem a serwerem odbywa się zgodnie ze stylem REST.

3.3 Przypadki użycia

Na podstawie wymagań funkcjonalnych został sporządzony diagram przypadków użycia (rysunek 3.1).



Rysunek 3.1: Diagram przypadków użycia

3.4 Wykorzystane narzędzia

W tym podrozdziale opisane zostaną narzędzia, które zostały wykorzystane do utworzenia prezentowanej w tym dokumencie aplikacji. Większość z nich to moduły i aplikacje związane z językiem Java Script.

3.4.1 JavaScript

JavaScript znany jest także jako ECMAScript. Jest to wieloparadygmatowy język wysokiego poziomu. Wykorzystuje on elementy zarówno programowania funkcyjnego, jak i obiektowego. W 2023 roku 98,7% stron internetowych wykorzystywało Java Script [9].

Język ten powstał w 1995 roku. Pierwotnie jako język skryptowy do przeglądarki Netscape Navigator. Szybko jednak został zaimplementowany przez inne przeglądarki internetowe po wprowadzeniu standardu ECMAScript w 1997 roku.

Był to pierwotnie język interpretowany, wykonywany linia po linii przez przeglądarkę. Dzisiaj jednak silniki takie jak V8, kompilują Java Script do silnie zoptymalizowanego kodu maszynowego. Jest to język dynamicznie typowany, co oznacza, że typy zmiennych

są określane podczas wykonywania programu. Jest on również słabo typowany, co znaczy, że typy zmiennych mogą się zmieniać w trakcie wykonywania programu.

Dziedziczenie jest oparte na prototypach. Każdy obiekt ma swój prototyp. Pola i metody obiektów znajdują się w tak zwanym łańcuchu prototypów. Możliwe jest dynamiczne i elastyczne tworzenie obiektów. Często bez potrzeby pisania klas.

Jest to główny język wykorzystany w tym projekcie, zarówno po stronie serwera jak i przeglądarki.

3.4.2 Node.js

Node jest to środowisko uruchomieniowe języka Java Script. Jest to oprogramowanie otwarte i wieloplatformowe. Node jest oparty o silnik V8 [7]. Jest to wysokowydajny silnik implementujący standard ECMAScript. V8 jest również wykorzystywany w przeglądarce Google Chrome.

Aplikacja napisana pod Node.js jest wykonywana na jednym wątku i nie tworzy osobnego procesu dla każdego żądania klienta. Node wykorzystuje pętlę zdarzeń. Programy pisane przy pomocy biblioteki standardowej Node charakteryzuje to, że są one nieblokujące i asynchroniczne.

Pętla zdarzeń (ang. event loop) polega na tym, że program stale nasłuchuje nowych zdarzeń (np. nowe zapytanie do serwera), przetwarza je tak by nie nastąpiły blokady i wywołuje związane z nimi funkcje (tak-zwane callbacki).

Asynchroniczność polega na tym, że program nie oczekuje na zakończenie asynchronicznej operacji (takiej jak odczyt z bazy danych), tylko rejestruje związany z nią callback. Po zakończeniu asynchronicznej operacji callback jest wstawiany do kolejki i wykonywany w pętli zdarzeń.

Callback to funkcja, która jest przekazywana jako argument do funkcji asynchronicznej. Służą one do tego by określić, jakie operacje powinny być wykonywane po zakończeniu asynchronicznego zadania. Umożliwiają one ciągłe wykonywanie asynchronicznego programu. W programach napisanych w standardzie ECMAScript2017 lub nowszym, często wykorzystuje się zamiast callbacków konstrukcję `async/await`.

W tym projekcie Node jest głównym środowiskiem uruchomieniowym wykorzystywanym przez wszystkie jego części. Aplikacja backendowa jest w pełni wykonywana przez Node. Aplikacja filtrująca jest wykonywana w Node, a elementy napisane w C są wywoływane jako podproces (ang. child process), przy użyciu funkcji z biblioteki standardowej Node. Aplikacja frontendowa mimo, że jest wykonywana w przeglądarce, to jest ona wpierw budowana i serwowana przez narzędzie Vite, które to działa w oparciu o Node.

3.4.3 npm

Npm jest to menedżer pakietów wykorzystywany przez Node. Jest to także największy na świecie rejestr oprogramowania [2]. Składa się on z trzech części: strony internetowej, narzędzia CLI (ang. Command Line Interface) oraz rejestru. Służy on do pobierania oraz instalowania modułów nie będących częścią biblioteki standardowej Node. Służy on również do wykonywania skryptów związanych z aplikacjami napisanymi pod Node (np. skrypty uruchomieniowe czy testowe).

W tym projekcie jest on wykorzystywany do instalacji potrzebnych modułów oraz do uruchamiania aplikacji backendowej w fazie produkcji.

3.4.4 Express.js

Express.js jest to popularna platforma programowania sieciowego oparta o Node.js. Służy ona do pisania aplikacji serwerowych obsługujących zapytania HTTP (ang. Hyper Text Transfer Protocol to główny protokół według, którego odbywa się komunikacja w internecie). Jest to platforma bardzo minimalistyczna, lecz napisano pod nią wiele bibliotek do pracy z ciasteczkami, sesjami, logowaniem użytkowników, parametrami URL (ang. Uniform Resource Locator to unikatowy ciąg znaków służący do odnajdowania obiektów w internecie), danymi z zapytań POST i wiele innych [5].

W niniejszym projekcie express.js został wykorzystany do napisania aplikacji backendowej, która odpowiada za komunikację z bazą danych, za wywoływanie aplikacji filtrującej obrazy, za weryfikację i autoryzację użytkowników oraz za odpowiadanie na żądania aplikacji frontendowej.

3.4.5 jimp

Jimp jest to napisana w języku Java Script biblioteka służąca do przetwarzania obrazów [10]. Działa w środowisku Node. Ma ona licencję MIT, czyli otwartoźródłową licencję umożliwiającą swobodne kopiowanie, modyfikowanie i rozpowszechnianie oprogramowania. Zmodyfikowane oprogramowanie można rozpowszechniać pod jakąkolwiek inną licencją.

W tym programie została wykorzystana do odczytu pliku bitmapy. Zwraca ona bufor danych zawierający zapis obrazu. Ten bufor zostaje przetworzony do tablicy, na której wykonywana jest filtracja. Przetworzony obraz jest zapisywany z powrotem do bufora, na podstawie którego jimp tworzy plik wyjściowy.

3.4.6 React.js

React jest to otwarta (ang. Open Source) biblioteka służąca do tworzenia interfejsów użytkownika [6]. Główną cechą odróżniającą React od czystego Java Scriptu, jest to, że

jest on deklaratywny. Programista opisuje co chce osiągnąć, a nie sekwencję kroków, która ma być wykonana.

Program napisany w React składa się z komponentów - samodzielnych odseparowanych fragmentów kodu, które są wielokrotnego użytku. Do tworzenia komponentów wykorzystuje się przeważnie rozszerzenie języka Java Script o nazwie JSX (ang. JavaScript Syntax Extension). Przypomina on swoją strukturą HTML (ang. Hyper Text Markup Language to język, którym opisuje się strukturę stron internetowych).

Do komunikacji między komponentami React wykorzystuje props, czyli parametry, które są przekazywane od komponentu rodzica, do komponentu dziecka. Komunikacja przebiega tylko w dół drzewa komponentów.

W tym projekcie React został wykorzystany do napisania aplikacji frontendowej.

3.4.7 Vite

Vite to proste, szybkie narzędzie deweloperskie służące do tworzenia projektów w językach Java Script i Type Script. Obsługuje on takie platformy jak Vue i React. Składa się z dwóch części: serwera deweloperskiego i komendy służącej do budowania programu (ang. build command) [17]. Komenda ta generuje zoptymalizowany plik wyjściowy.

Podczas pisania tego projektu Vite zostało wykorzystane do testowego uruchamiania aplikacji frontendowej, a także do kompilacji jego kodu do ostatecznego pliku.

3.4.8 Git

Git jest to rozproszony system kontroli wersji. Charakteryzuje się on tym, że projekt jest przechowywany na wielu różnych gałęziach [1]. Osobni programiści mogą pracować na osobnych gałęziach. Gałęzie można dzielić na nowe (ang. branch out) lub scalać ze sobą (ang. merge).

Podczas pisania tego projektu, ze względu na ograniczony rozmiar projektu, wykorzystywana była jedna gałąź. Była ona przechowywana na serwerze GitHub i komputerze lokalnym. Powstałe zmiany były sukcesywnie wprowadzane na serwer, a w przypadku uszkodzenia projektu (np. przypadkowe skasowanie któregoś pliku), poprzednia wersja była pobierana z serwera.

3.4.9 ESLint

ESLint jest to linter języka Java Script. Linting to rodzaj statycznej analizy kodu, która służy do wymuszania jednolitego stylu pisanego kodu. Linting umożliwia również wykrywanie błędów w semantycznie poprawnym kodzie przed jego wykonaniem.

W tym projekcie został on wykorzystany do formatowania części napisanych w języku Java Script.

3.4.10 C

C to język programowania szerokiego zastosowania [4]. Został zaprojektowany w 1970 roku przez Dennisa Ritchie. Jest to język wymagający ręcznego operowania pamięcią. Programy napisane w C charakteryzują się stosunkowo krótkim czasem wykonania. Przy zastosowaniu odpowiednich bibliotek język ten umożliwia programowanie wielowątkowe.

Jest to język proceduralny. Program napisany w C składa się z funkcji. C jest statycznie typowany, czyli typ zmiennej nie może ulec zmianie podczas wykonywania programu. C stosuje się w szerokim zakresie aplikacji, od oprogramowania superkomputerów do oprogramowania mikrokontrolerów.

W tym projekcie język C został wybrany do implementacji algorytmów filtracji medianowej i bayesowskiej. Ze względu na złożoność obliczeniową tych algorytmów zdecydowano się na implementację wielowątkową. Java Script nie umożliwia programowania wielowątkowego. W tym projekcie części napisane w C były kompilowane przy pomocy programu GCC.

3.4.11 MySQL

MySQL to otwartoźródłowy system zarządzania bazą danych. Jest on rozwijany, rozprawdany i wspierany przez Oracle Corporation [14]. System ten zarządza relacyjną bazą danych. System ten składa się z wielowątkowego serwera SQL wspierającego różne narzędzia do zarządzania bazą danych, a także wiele API (ang. Application Programming Interface) służących do łączenia bazy danych z programem.

W tym projekcie MySQL wykorzystano do utworzenia jednotablicowej bazy użytkowników. Do zdefiniowania bazy danych, tablicy i użytkownika, którego dane wykorzystuje backend aplikacji, użyto graficznego programu dbeaver community edition.

3.4.12 Postman

Postman to narzędzie służące do tworzenia i używania interfejsów programowania API (ang. Application Programming Interface). Oprogramowanie to upraszcza kolaborację w tworzeniu API. Stanowi ono także repozytorium, w którym można gromadzić zapytania do API, przypadki testowe, wyniki testów, dane pomiarowe i wszystko inne związane z API. Postman stanowi kompletny zestaw narzędzi służących do projektowania, testowania i dokumentacji interfejsów. W tym projekcie został wykorzystany w procesie pisania aplikacji backendowej, gdyż umożliwiał wysyłanie do niej zapytań HTTP oraz weryfikację poprawności odpowiedzi [18].

Rozdział 4

Specyfikacja zewnętrzna

4.1 Wymagania sprzętowe i systemowe

Do uruchomienia aplikacji potrzebny jest komputer z systemem operacyjnym linux. Program był testowany z użyciem dystrybucji Ubuntu w wersji 22.04. Uruchomienie na innym systemie operacyjnym niż linux się nie powiedzie, gdyż część projektu została skompilowana pod ten system. Uruchomienie na systemie Windows wymagałoby rekompilacji tych części oraz zmiany fragmentów kodu, które te części wywołują.

By uruchomić program potrzebny jest również Node.js w wersji 18.18.0 bądź nowszej. Potrzebny jest również menedżer pakietów npm w wersji 9.8.1 bądź nowszej. Konieczna jest również konfiguracja bazy danych MySQL.

Do uruchomienia aplikacji frontendowej potrzebna jest przeglądarka Google Chrome w najnowszej wersji. Przeglądarki Mozilla Firefox oraz Microsoft Edge również powinny działać, choć aplikacja nie była pod nie testowana.

4.2 Instalacja

Poniższa instrukcja instalacji przedstawia instalację programu na komputerze z systemem Ubuntu 22.04. Instalacja na innych dystrybucjach linuxa może przebiegać nieco inaczej.

4.2.1 Pobranie repozytorium

Proces instalacji należy rozpocząć od pobrania repozytorium z projektem. Repozytorium znajduje się pod adresem <https://github.com/radekbys/imageFilteringApp>. Repozytorium należy pobrać przy pomocy narzędzia Git lub ręcznie jako folder skompresowany ZIP.

4.2.2 Przygotowanie bazy danych

Kolejnym krokiem powinna być instalacja aplikacji MySQL, lecz przed tym trzeba zaktualizować listę pakietów programu apt. Wykonuje się to komendą: `sudo apt update`. Następnie należy zainstalować aplikację przy pomocy komendy:

```
sudo apt install mysql-server
```

By upewnić się, że MySQL jest uruchomiony poprawnie należy wykonać:

```
sudo systemctl start mysql.service
```

W następnej kolejności należy nadać hasło użytkownikowi root. By to zrobić należy wejść w terminal mysql przy pomocy komendy: `sudo mysql`, a następnie wpisać komendę:

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY  
'password';
```

(w miejsce `password` należy podać wybrane hasło). Następnie należy opuścić terminal mysql przy pomocy komendy: `exit`.

Po nadaniu hasła należy uruchomić komendę:

```
sudo mysql_secure_installation
```

Po uruchomieniu tej komendy program zapyta użytkownika czy chce zainstalować komponent `VALIDATE PASSWORD`. Po zainstalowaniu lub niezainstalowaniu tego komponentu program zada szereg pytań, na które należy odpowiedzieć.

W kolejnym kroku należy utworzyć użytkownika, który będzie wykorzystywany przy komunikacji z bazą danych. By to zrobić trzeba wejść w terminal mysql przy pomocy komendy: `mysql -u root -p`, by następnie wywołać komendę:

```
CREATE USER 'username'@'localhost' IDENTIFIED BY 'password';
```

W miejsca `username` i `password` należy podać nazwę użytkownika i przypisane mu hasło. Następnie należy opuścić mysql komendą: `exit`.

W pobranym repozytorium z projektem znajduje się plik `database.sql`, który to jest zapisem przykładowej bazy danych. By załadować go do nowej bazy danych, należy w terminalu wywołać komendy:

```
mysql -u root -p -e "create database database_name";
```

(w miejsce `database_name` wstawić nazwę nowej bazy) oraz komendę:

```
mysql -u root -p database_name < ścieżka/database.sql
```

(w miejsce `ścieżka` wstawić ścieżkę do pliku).

Następnie należy nadać uprawnienia utworzonemu użytkownikowi. By to zrobić trzeba wejść w terminal mysql komendą: `mysql -u root -p`, by z tego terminalu wywołać komendę:

```
GRANT INSERT, UPDATE, DELETE, SELECT on exampleDatabase.Users TO 'eUsr'@'localhost'  
WITH GRANT OPTION;
```

(w miejsca `exampleDatabase` i `eUsr` wstawić nazwę bazy danych i nazwę użytkownika). Jeżeli wszystkie komendy udało się poprawnie wywołać to baza danych jest już gotowa.

4.2.3 Instalacja aplikacji serwerowej

W pierwszej kolejności należy otworzyć s terminal wewnątrz folderu `filterApp`. Folder ten znajduje się w repozytorium. Należy w tym folderze wywołać komendę: `npm i`. Komenda ta zainstaluje potrzebne moduły.

W drugim kroku należy nawigować do folderu `backend` i ponownie wywołać: `npm i`.

W trzecim kroku należy zdefiniować zmienne środowiskowe. Wewnątrz folderu `backend` należy zlokalizować plik `.env.example`. Następnie należy utworzyć plik o nazwie `.env` i skopiować do niego zawartość pliku `.env.example`. Następnie należy podmienić dane w pliku `.env`.

Plik `.env` zawiera adres portu, na którym aplikacja serwerowa będzie nasłuchiwać, ciąg znaków wykorzystywany do szyfrowania tokenów uwierzytelniających, nazwę bazy danych, adres serwera MySQL, nazwę wykorzystywanego użytkownika MySQL oraz hasło tego użytkownika.

4.2.4 Instalacja aplikacji klienckiej

Instalację należy rozpocząć od otwarcia terminalu w folderze `frontend`. Następnie należy wywołać komendę `npm i`, po czym trzeba wywołać komendę: `npm run build`. Komenda ta skompiluje pliki projektu do jednego pliku wyjściowego.

W kolejnym kroku należy utworzyć plik `.env` i skopiować do niego zawartość pliku `.env.example`. Jeśli serwer został skonfigurowany na maszynie lokalnej, na porcie 3000, to nic już nie trzeba zmieniać. W przeciwnym wypadku należy podmienić adres w pliku `.env`.

4.3 Sposób aktywacji

Aby móc korzystać z aplikacji należy uruchomić zarówno aplikację serwerową jak i kliencką. Aby uruchomić serwer należy zlokalizować plik `index.js` znajdujący się w folderze `backend`. Plik ten należy uruchomić w terminalu przy pomocy komendy: `node index.js`. W terminalu powinna się pojawić informacja o nasłuchiwanu.

Aby uruchomić aplikację kliencką należy w folderze `frontend` otworzyć terminal. Następnie trzeba wywołać komendę: `npm run preview`. Program można również uruchomić komendą: `npm run dev`. Uruchomi to program w trybie deweloperskim, tak by można go było debugować. Niezależnie od tego, w którym trybie zostanie uruchomiony program, w konsoli wyświetli się adres, który należy wpisać w przeglądarce by wyświetlić stronę logowania. By korzystać z programu obie aplikacje muszą być uruchomione jednocześnie.

4.4 Kategorie użytkowników

Użytkownicy dzielą się na zwykłych użytkowników i administratorów. Zwykli użytkownicy mają dostęp do panelu filtracji. Mogą tam wybrać zdjęcie z dysku, wybrać jeden filtr z listy i wykonać filtrację. Przefiltrowany obraz wyświetli się w przeglądarce. Użytkownik może ten obraz pobrać. By to zrobić musi wybrać ten obraz prawym przyciskiem myszy i z menu kontekstowego wybrać opcję `save image as...`.

Administratorzy mogą robić wszystko to co użytkownicy. Mają oni także dostęp do panelu administracyjnego, z którego mogą dodawać, usuwać, zmieniać uprawnienia użytkowników. Robią to poprzez wypełnianie znajdujących się w tym panelu formularzy.

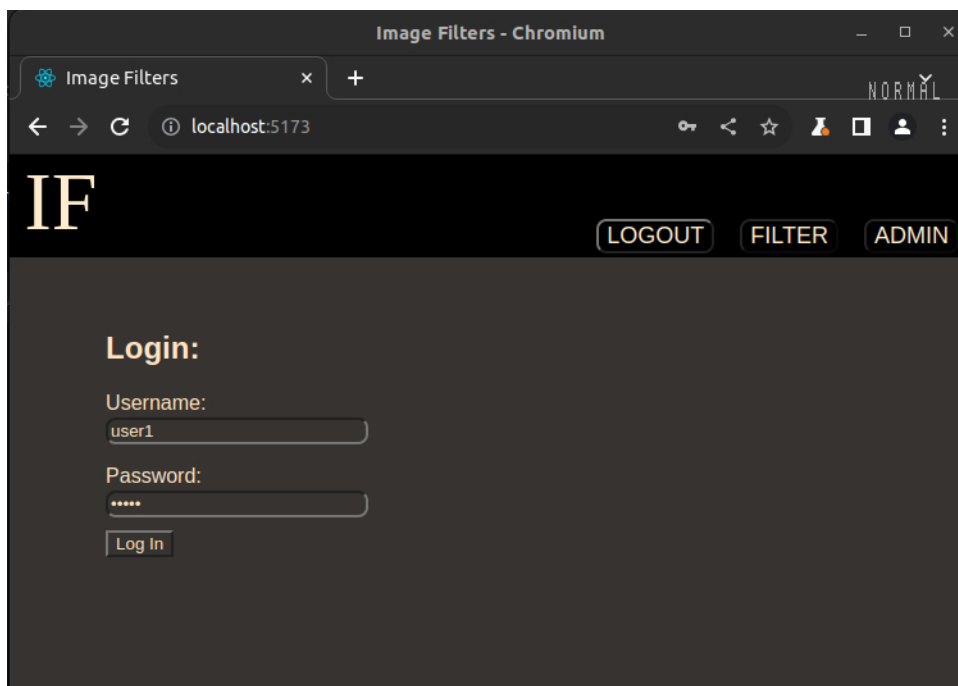
4.5 Sposób obsługi

W tym podrozdziale przedstawione zostaną dwa scenariusze korzystania z systemu, jeden dla zwykłego użytkownika i jeden dla administratora. Użyte w poniższym przykładzie zdjęcie psa pochodzi z portalu pixabay. Jest ono objęte jest wolną licencją umożliwiającą rozpowszechnianie, modyfikowanie zdjęć bez potrzeby podawania oryginalnego autora. Zdjęcie można znaleźć pod adresem: <https://pixabay.com/photos/dog-puppy-golden-retriever-pet-8272860/>.

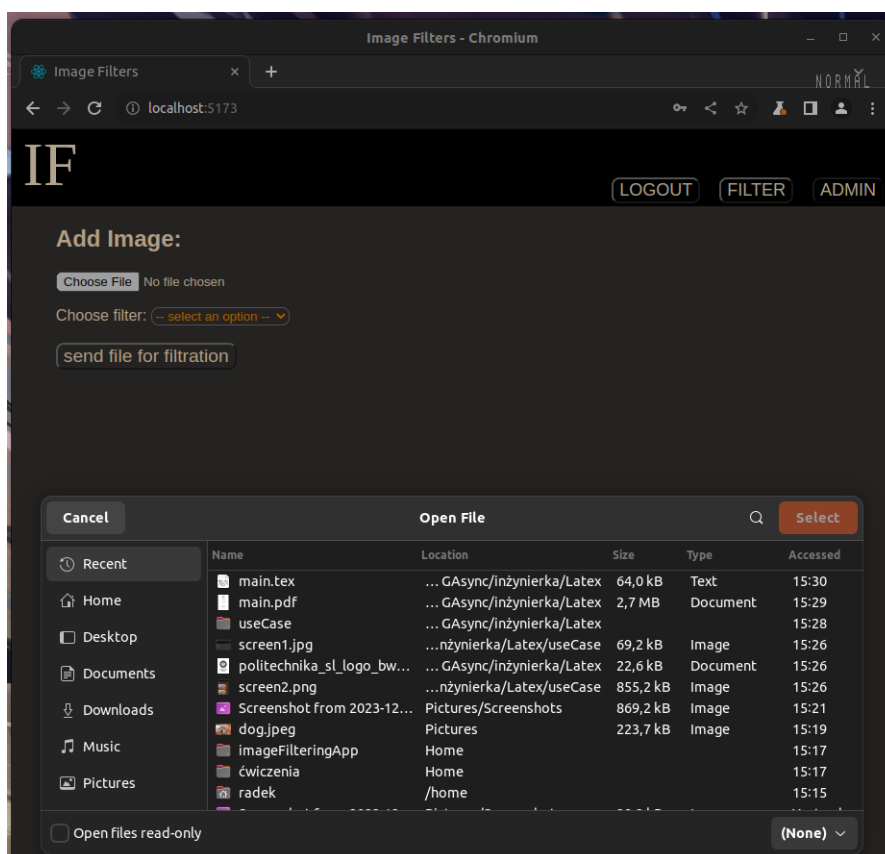
4.5.1 Zwykły użytkownik

W przykładowej bazie danych (której zrzut jest w repozytorium), znajduje się użytkownik o nazwie `user1`, a jego hasło to również `user1`. Loguje się on do systemu wpisując login i hasło. Następnie lewym przyciskiem myszy wybiera przycisk `Log In`, a następnie wybiera przycisk `FILTER` znajdujący się w prawym, górnym rogu ekranu. Logowanie zobrazowano na rysunku 4.1.

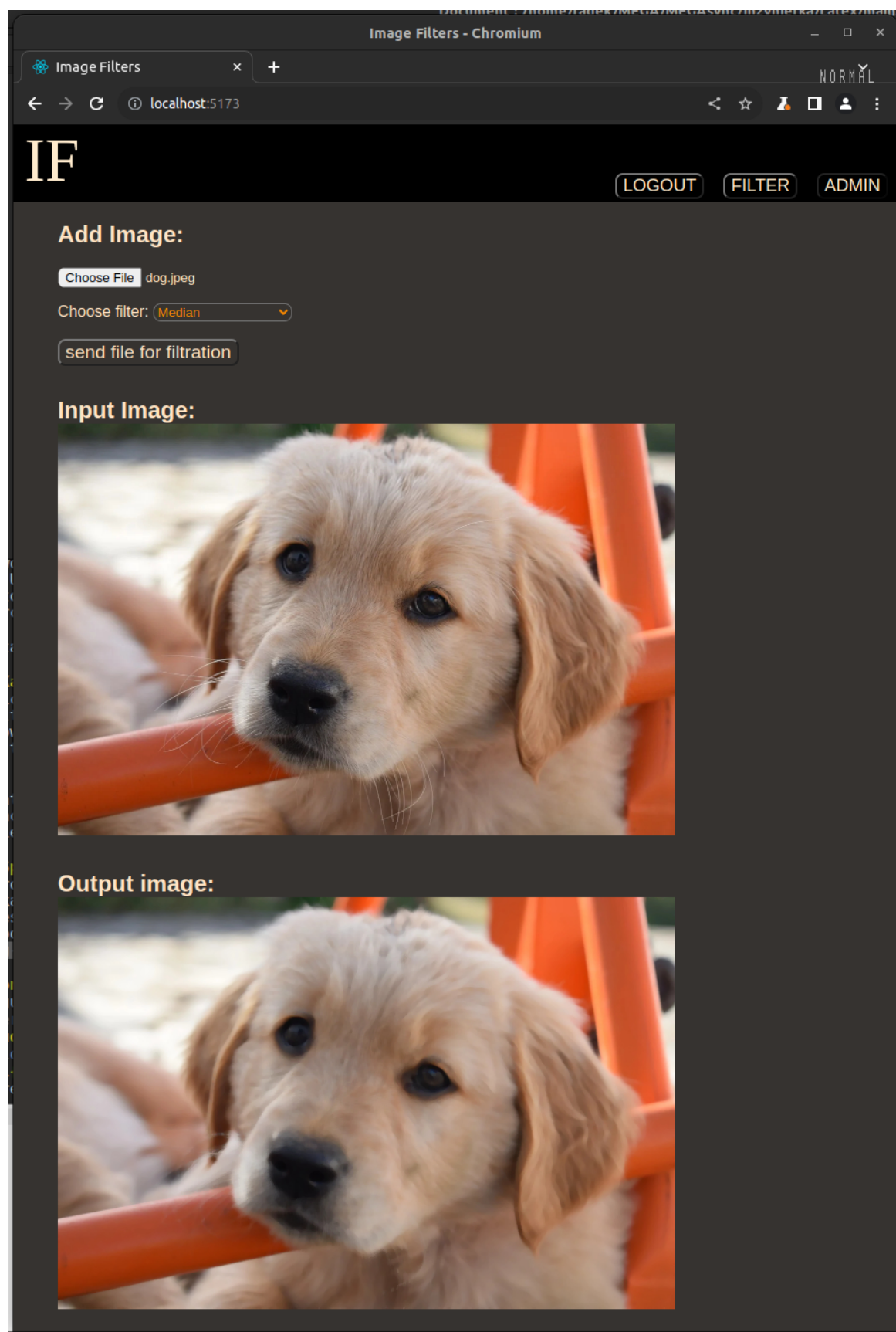
Po przejściu do panelu filtracji użytkownik używa przycisku `Choose File`, po czym wyświetlone zostaje mu okienko wyboru pliku przedstawione na rysunku 4.2. W następnym kroku wybiera on filtr medianowy z listy rozwijanej, po czym używa przycisku `send file for filtration`. Efekt działania przedstawiono na rysunku 4.3.



Rysunek 4.1: Zrzut ekranu przedstawiający panel logowania.



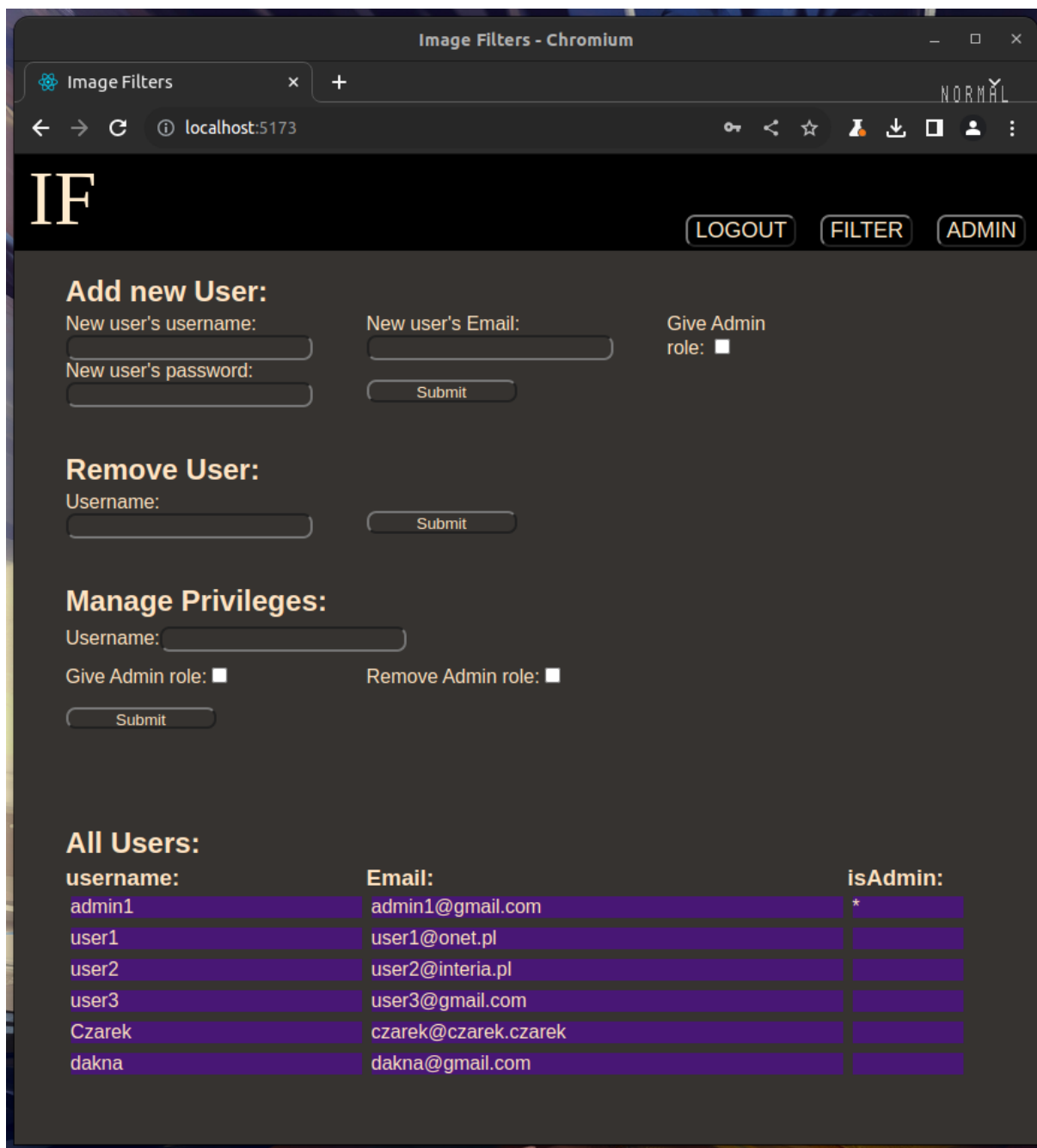
Rysunek 4.2: Zrzut ekranu przedstawiający okienko wyboru pliku.



Rysunek 4.3: Zrzut ekranu przedstawiający wynik działania filtru medianowego.

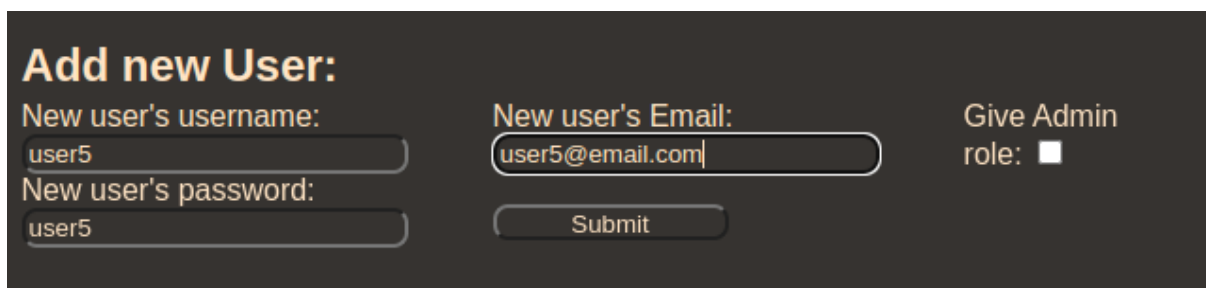
4.5.2 Administrator

W bazie danych znajduje się administrator o nazwie admin1 i hasle admin1. Ma on za zadanie dodanie do systemu nowego użytkownika o nazwie user5, hasle user5, emailu user5@email.com. Użytkownik ten nie ma mieć uprawnień administratora. W celu wykonania zadania admin1 loguje się do systemu podobnie jak zwykły użytkownik. Po zalogowaniu odblokowuje się przycisk ADMIN. Po wybraniu tego przycisku zostaje wyświetlony panel administracyjny. Panel ten został przedstawiony na rysunku 4.4.



Rysunek 4.4: Zrzut ekranu przedstawiający panel administracyjny.

W kolejnym kroku administrator wypełnia formularz służący do dodawania nowego użytkownika. Wypełniony formularz przedstawiony jest na rysunku 4.5. Po wybraniu przycisku **Submit** nowy użytkownik wyświetla się z dołu listy użytkowników (rysunek 4.6). Po wykonaniu tych kroków zadanie zostało spełnione i administrator wylogowuje się przy pomocy przycisku **LOGOUT**.



Add new User:

New user's username:

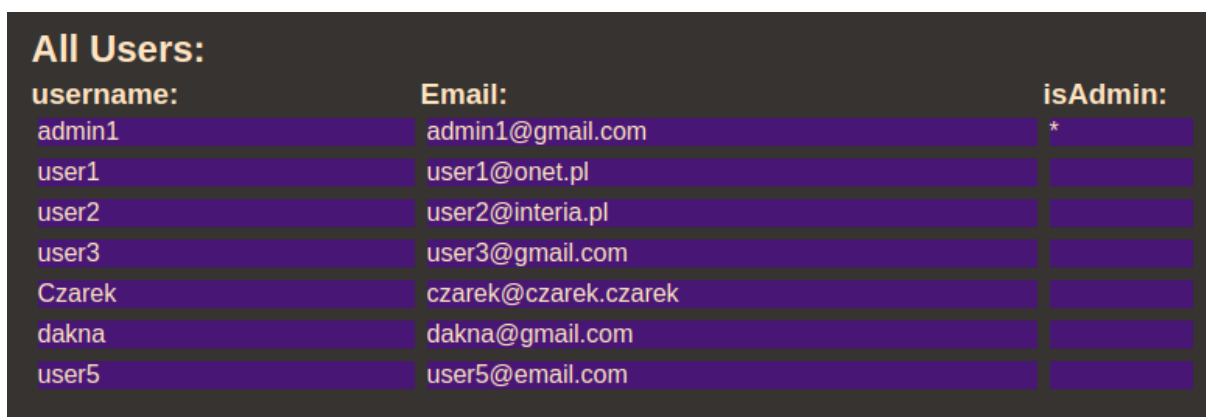
New user's Email:

New user's password:

Submit

Give Admin role: ☐

Rysunek 4.5: Zrzut ekranu przedstawiający wypełniony formularz dodawania nowego użytkownika.



username:	Email:	isAdmin:
admin1	admin1@gmail.com	*
user1	user1@onet.pl	
user2	user2@interia.pl	
user3	user3@gmail.com	
Czarek	czarek@czarek.czarek	
dakna	dakna@gmail.com	
user5	user5@email.com	

Rysunek 4.6: Zrzut ekranu przedstawiający listę użytkowników.

4.6 Zabezpieczenia

System logowania do aplikacji oparty jest o technologię JSON Web Token. Baza danych nie przechowuje haseł użytkowników, lecz hashe utworzone przy pomocy algorytmu sha256. Przy próbie logowania aplikacja porównuje hash hasła logującego się z tym przechowywanym w bazie danych. Jeżeli nazwa użytkownika i hasło się zgadza, to generowany jest token uwierzytelniający, który to jest odsyłany w odpowiedzi. Token ten jest ważny przez 30 minut. Przy kolejnych zapytaniach klienta token zostaje umieszczony w nagłówku http. Jeżeli z zapytaniem został przesłany niepoprawny token, to serwer zwróci błąd.

Elementy przesłane z klienta, będące częścią używanych przez serwer nazw plików są oczyszczane ze złośliwych znaków przy pomocy modułu `sanitize-filename`, który to zostanie opisany wraz z ważniejszymi bibliotekami w następnym rozdziale. Elementy przeka-

zywane do zapytań SQL są przekazywane jako zmienne. Chroni to przed wstrzyknięciem złośliwego kodu do bazy danych (przykład takiego zapytania: `await this.pool.query('DELETE FROM Users WHERE username = ?;', [username]);`)

4.7 Działanie filtru bayesowskiego

Działanie filtru bayesowskiego zostało udokumentowane. Rysunek 4.7 przedstawia zdjęcie przed filtracją, natomiast rysunek 4.8 przedstawia zdjęcie przefiltrowane.



Rysunek 4.7: Zdjęcie przedstawiające psa.



Rysunek 4.8: Zdjęcie przedstawiające psa, które zostało przetworzone filtrem bayesowskim.

4.8 Redukcja szumów

Niektóre z filtrów mogą posłużyć do redukcji szumów w obrazie. Przykładami takich filtrów są filtry medianowy, bayesowski i uśredniający. Zaszumiony obraz został przedstawiony na rysunku 4.9. Na rysunku 4.10 przedstawione jest zdjęcie, którego szum został zredukowany filtrem uśredniającym, na rysunku 4.11 filtrem medianowym, a na rysunku 4.12 filtrem bayesowskim.



Rysunek 4.9: Zdjęcie przedstawiające psa, które zostało zaszumione.



Rysunek 4.10: Zdjęcie przedstawiające psa, które zostało zaszumione i przetworzone filtrem uśredniającym.



Rysunek 4.11: Zdjęcie przedstawiające psa, które zostało zaszumione i przetworzone filtrem medianowym.



Rysunek 4.12: Zdjęcie przedstawiające psa, które zostało zaszumione i przetworzone filtrem bayesowskim.

Rozdział 5

Specyfikacja wewnętrzna

5.1 Architektura aplikacji

Projekt został wykonany zgodnie z wzorcem projektowym MVC (ang. Model View Controller). Widok projektu stanowi aplikacja frontendowa zawarta w folderze „frontend”. Jest to aplikacja przeglądarkowa stanowiąca graficzny interfejs użytkownika. Komunikuje się ona z aplikacją serwerową przy pomocy protokołu HTTP, zgodnie z zasadami REST. REST (ang. Representational State Transfer) to styl pisania oprogramowani, charakteryzujący się bezstanowością komunikacji, jednorodnością interfejsów i skalowalnością interakcji między komponentami.

Kontroler projektu stanowią dwie aplikacje. Aplikacja backendowa znajdująca się w folderze „backend” oraz aplikacja filtrująca obrazy zawarta w folderze „filterApp”. Aplikacja backendowa odpowiada za: uwierzytelnianie i autoryzację użytkowników, komunikację z bazą danych, odpowiadanie na zapytania frontendu, wywoływanie aplikacji filtrującej i przekazywanie jej tymczasowych plików z obrazami do filtracji. Aplikacja filtrująca natomiast odpowiada za odczyt obrazów, przetwarzanie ich i zwracanie obrazu wynikowego.

Model aplikacji stanowią baza danych i pliki tymczasowe z bitmapami. Baza danych składa się z jednej tablicy zawierającej dane użytkowników i ich uprawnienia (informację czy dany użytkownik jest administratorem, czy też nie). Pliki tymczasowe to obrazy przesłane przez użytkownika i ich przetworzone wersje. Są one kasowane po odesłaniu wyniku filtracji.

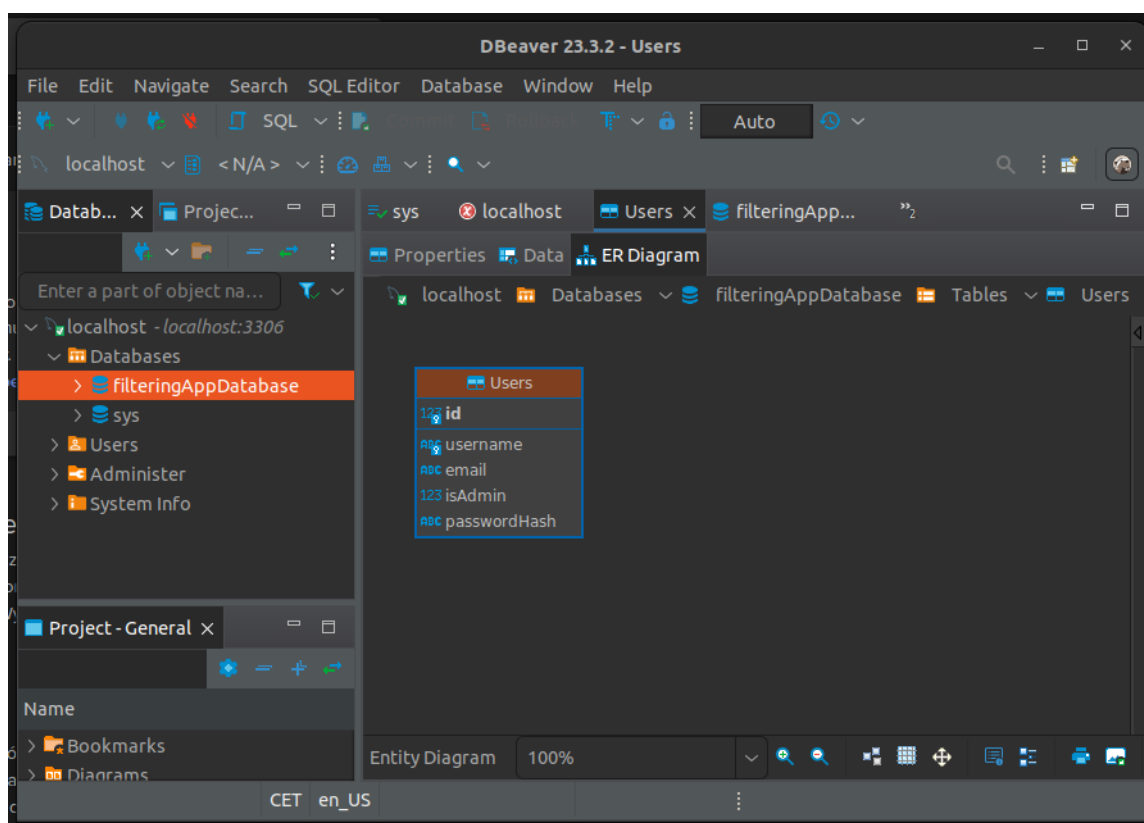
5.2 Struktura danych

W skład projektu wchodzi baza danych o nazwie `filteringAppDatabase`, która składa się z jednej tablicy o nazwie `Users`. Schemat bazy danych przedstawiony jest na rysunku 5.1. Tablica ta posiada następujące pola:

- `id` - klucz główny, liczba całkowita, jednoznacznie identyfikuje encję w bazie danych.

Podczas dodawania nowej encji do bazy id jest automatycznie inkrementowane i przypisywane do nowego wpisu.

- username - unikatowy ciąg znaków jednoznacznie identyfikujący użytkownika aplikacji.
- email - ciąg znaków, umożliwia skontaktowanie się z użytkownikiem.
- isAdmin - liczba całkowita mogąca przyjąć wartość 0 lub 1. Mówi ona czy dany użytkownik ma uprawnienia administratorskie.
- passwordHash - ciąg znaków, zawiera on skrót hasła o stałej długości. Został on utworzony przy pomocy algorytmu sha256 i potrzebny jest on do zalogowania się użytkownika.



Rysunek 5.1: Zrzut ekranu przedstawiający schemat bazy danych.

5.3 Przepływ danych

Aplikacja frontendowa i serwerowa komunikują się ze sobą przy pomocy zapytań HTTP, natomiast komunikacja pomiędzy serwerem a aplikacją filtrującą odbywa się poprzez zapis i odczyt plików na dysku. Aplikacja serwerowa przekazuje aplikacji filtrującej

ścieżki do pliku wejściowego i wyjściowego. Wywołanie aplikacji filtrującej przedstawione jest na rysunku 5.2.

Komunikacja między frontendem a serwerem odbywa się następująco: strona internetowa korzysta z funkcji `fetch()`, która jest częścią standardu ECMAScript i służy do komunikacji z zewnętrznym api. Przykładowe wywołanie tej funkcji przedstawione jest na rysunku 5.3. Aplikacja serwerowa natomiast do odpowiadania wykorzystuje metody routingowe pochodzące od metod HTTP (rysunek 5.4).

```
// save and filter the file
const path = resolve('../filterApp/index.js');
const username = sanitize(req.body.user);
const command = `node ${path} ${filterName}
  ${`${username}-input${filenameExtension}`}
  ${`${username}-output${filenameExtension}`}
  ${Number(req.body.epsilon)}`;
await writeFile(
  `../images/${username}-input${filenameExtension}`,
  req.body['file-base64'],
  { encoding: 'base64' },
);
await promisifiedExec(command);
```

Rysunek 5.2: Zrzut ekranu przedstawiający zapis pliku z obrazem i wywołanie podprogramu filtrującego.

```
useEffect(() => {
  const sequence = async () => {
    const res = await fetch(
      `${import.meta.env.VITE_REACT_APP_SERVER_URL}/admin/user`, {
        method: 'GET',
        headers: {
          'Json-Web-Token': String(localStorage.getItem('token')),
        },
      },
    );

    if (res.status !== 200) {
      // console.log(await res.json());
      window.alert(`Error ${await res.json().error}`);
      return;
    }

    setAllUsers(await res.json());
  };
  sequence();
}, [getUsersTrigger]);
```

Rysunek 5.3: Zrzut ekranu przedstawiający wywołanie funkcji `fetch`.

```

filterRouter
  .get('/user', authorization, async (req, res) => {
    try {
      const response = (await dbHandler.getUsers()).map((user) => ({
        username: user.username,
        email: user.email,
        isAdmin: Boolean(user.isAdmin),
      }));
      res.json(response);
    } catch (error) {
      res.status(error.status || 500);
      res.send({ error: error.message });
    }
  })
}

```

Rysunek 5.4: Przykład użycia funkcji routingowej `get`, która zwraca listę wszystkich użytkowników.

5.4 Ważniejsze algorytmy

W niniejszej pracy zaimplementowano trzy ważne algorytmy: algorytm filtracji liniowej, algorytm filtracji medianowej i algorytm filtracji bayesowskiej. Algorytmy te zostały opisane w rozdziale „Analiza tematu”. Implementacja algorytmu filtracji liniowej została przedstawiona na rysunku 5.6, filtracji medianowej na rysunku 5.5 a filtracji bayesowskiej na rysunku 5.7.

```

for(int i = first; i < last; i++){
  //get all the neighboring pixels into array
  for(int j = - MASK_RADIUS; j<= MASK_RADIUS; j++){
    for(int k = - MASK_RADIUS; k <= MASK_RADIUS; k++){
      index = ((*argument).in_row * j)+(k*BYTES_PER_PIXEL) + i;
      array[((j+2)*5)+(k+2)] =(*argument).image[index];
    }
  }
  //sort the array
  qsort(array, 25,sizeof(unsigned int), cmpfunc);
  //put the result into copy image
  (*argument).image_copy[i] = array[12];
}

```

Rysunek 5.5: Implementacja algorytmu filtracji medianowej.

```

// function that filters the image using simple filters
// buffer is the image data in buffer format, width and height are the image dimensions in pixels
// filter is an array of arrays containing the filters weights
static simpleFilter(buffer, width, height, filter) {
  // transfers the image data into two separate arrays
  const data = [...buffer];
  const copyData = [...buffer];

  // if filter lenght = 3 then pixelOffset = 1, if length =5 then pixelOffset = 3, etc.
  const pixelOffset = (filter.length - 1) / 2;

  // first and last few rows mus be skipped
  for (let i = pixelOffset; i < height - pixelOffset; i += 1) {
    // last and first columns also need to be skipped
    for (let j = pixelOffset * 4; j < (width * 4) - (pixelOffset * 4); j += 1) {
      // one pixel takes 4 bytes of data
      // every 4th byte codes no color, needs to be skipped
      if ((j + 1) % 4 !== 0) {
        // sum of all neighboring pixels multiplied by the filter weights
        let sum = 0;
        // divisor is the sum of filter weights
        let divisor = 0;

        // iterates over the whole filter matrix
        for (let k = 0; k < filter.length; k += 1) {
          for (let l = 0; l < filter[0].length; l += 1) {
            // finds the corresponding pixel in the image
            const componentAkl = data[
              (i + k - pixelOffset) * width * 4 + j + (l - pixelOffset) * 4
            ];
            if (componentAkl) {
              // adds the filter weight
              divisor += filter[k][l];
              // multiplies the pixel by weight and adds it to the sum
              sum += filter[k][l] * componentAkl;
            }
          }
        }
        // makes sure the result is an integer in the value range of a byte
        let result = parseInt(sum / divisor, 10);
        if (result < 0) result = 0;
        if (result > 255) result = 255;

        copyData[(i * width * 4) + j] = result;
      }
    }
  }
  // returns the filtered image in buffer format
  return Buffer.from(copyData);
}

```

Rysunek 5.6: Implementacja algorytmu filtracji liniowej.

```

for(int i = first; i < last; i++){
    //get all the neighboring pixels into array
    for(int j = - MASK_RADIUS; j<= MASK_RADIUS; j++){
        for(int k = - MASK_RADIUS; k <= MASK_RADIUS; k++){
            index = ((*argument).in_row * j)+(k*BYTES_PER_PIXEL) + i;
            t[((j+2)*5)+(k+2)] =(*argument).image[index];
        }
    }
    // 1. Initialize g(x, y)(0) as the arithmetic average of t. If the sample variance of t is
    // greater than zero set the iteration index k = 1 else stop.

    g_xy0=0;
    for(int j = 0; j<MASK_SIZE; j++) g_xy0+=t[j];
    g_xy0 = g_xy0/25.0; //arithmetic average

    t_variance = 0;
    for(int j = 0; j<MASK_SIZE; j++) t_variance+=(t[j] - g_xy0)*(t[j] - g_xy0);
    t_variance = t_variance/25.0; //variance of the vector t

    if(t_variance <= 0){
        (*argument).image_copy[i] = g_xy0;
        continue;
    }

    g_xyk = g_xy0;

    for(int k = 1; k++){ //set the iterator k=1
        // 2. Calculate the hyperparameter  $\alpha_i$  for  $i = 1, 2, \dots, D$  and  $\beta(k)$ 
        B_k = 1.0 / (g_xyk*g_xyk);
        for(int j =0; j<MASK_SIZE; j++) a_ik[j]=1.0/((t[j]-g_xyk)*(t[j]-g_xyk));

        // 3. Update the average g(x, y)(k) for kth iteration.
        sum_aik_ti = 0;
        sum_aik = 0;
        for(int j =0; j<MASK_SIZE; j++){
            sum_aik_ti += a_ik[j]*t[j];
            sum_aik += a_ik[j];
        }
        g_xyk_new = sum_aik_ti/(B_k + sum_aik);

        // 4. If  $g(x, y)(k) - g(x, y)(k-1) > \epsilon$  then  $k \leftarrow k + 1$  and go to 2, else stop.
        g_k_minus_prev_gk_square = (g_xyk_new-g_xyk)*(g_xyk_new-g_xyk);
        if(g_k_minus_prev_gk_square <= (*argument).epsilon){
            (*argument).image_copy[i] = g_xyk_new;
            break;
        }
        if(g_xyk_new!=g_xyk_new){
            (*argument).image_copy[i] = g_xyk;
            break;
        }
        g_xyk = g_xyk_new;
    }
}

```

Rysunek 5.7: Implementacja algorytmu filtracji bayesowskiej.

5.5 Zabezpieczenia

Dostęp do funkcjonalności aplikacji został zabezpieczony przy pomocy technologii JSON Web Token. Podczas logowania serwer sprawdza podane login i hasło. Jeśli są poprawne to tworzy token uwierzytelniający i przekazuje go aplikacji klienckiej, która go zapisuje w local storage. Proces ten nazywa się uwierzytelnianiem.

Aplikacja kliencka przy wysyłaniu kolejnych zapytań do serwera umieszcza token w nagłówku zapytania. Jeśli użytkownik spełnia określone wymagania, czyli jest poprawnym użytkownikiem w przypadku prośby o filtrację, lub ma uprawnienie administratora w przypadku prośby o modyfikację użytkowników, to żądanie zostaje spełnione i użytkownik uzyskuje poprawną odpowiedź. W przeciwnym wypadku aplikacja serwerowa zwróci błąd. Proces ten nazywa się autoryzacją. Token uwierzytelniający wygasa po 30 minutach.

5.6 Wykorzystane moduły i biblioteki

5.6.1 pthread.h

Plik nagłówkowy języka C zawierający deklaracje i mapowania interfejsów służących do programowania wielowątkowego. Zawiera ona także szereg stałych wykorzystywanych przez te funkcje. Została wykorzystana do wielowątkowej implementacji algorytmów bayesowskiego i medianowego.

5.6.2 body-parser

Moduł do node.js służący do przetwarzania ciał przychodzących żądań HTTP. Ma licencję MIT. Został wykorzystany w aplikacji serwerowej.

5.6.3 cors

Moduł do node.js umożliwiający korzystanie z mechanizmu CORS (ang. Cross Origin Resource Sharing), który to umożliwia dzielenie zasobów między serwerami znajdującymi się w różnych domenach. Wymagana była do komunikacji między aplikacją serwerową (napisaną w express.js) a aplikacją kliencką (napisaną w react.js). Moduł ten objęty jest licencją MIT.

5.6.4 dotenv

Moduł do node.js umożliwiający korzystanie ze zmiennych środowiskowych. Pliki te są przechowywane w pliku o końcówce .env znajdującym się w folderze aplikacji serwerowej. Pliki te wczytywane są raz podczas uruchamiania programu. Zmiana w pliku .env nie jest

odzwierciedlona w stale działającym programie. Moduł ten ma licencję BSD-2-Clause, która podobnie jak ISC i MIT jest licencją wolnego oprogramowania.

5.6.5 js-sha256

Moduł do node.js zawierający prostą funkcję do tworzenia skrótów przy użyciu algorytmu sha256. Obsługuje format utf-8. Wykorzystany w aplikacji serwerowej do tworzenia skrótów haseł użytkowników. Moduł ten ma licencję MIT.

5.6.6 mysql2

Moduł do node.js umożliwiający łączenie się z bazą danych MySQL oraz wykonywanie zapytań SQL z poziomu programu napisanego w JavaScriptcie. Został wykorzystany w aplikacji serwerowej. Oparty jest na licencji MIT.

5.6.7 sanitize-filename

Moduł do node.js zawierający funkcję do usuwania niebezpiecznych znaków z nazw plików. Bez tego modułu nieuprawniony użytkownik aplikacji mógłby wprowadzić zmiany w systemie, na którym działa aplikacja serwerowa. Funkcjonalność ta została wykorzystana w aplikacji serwerowej. Objęta jest licencją ISC.

5.6.8 uuid

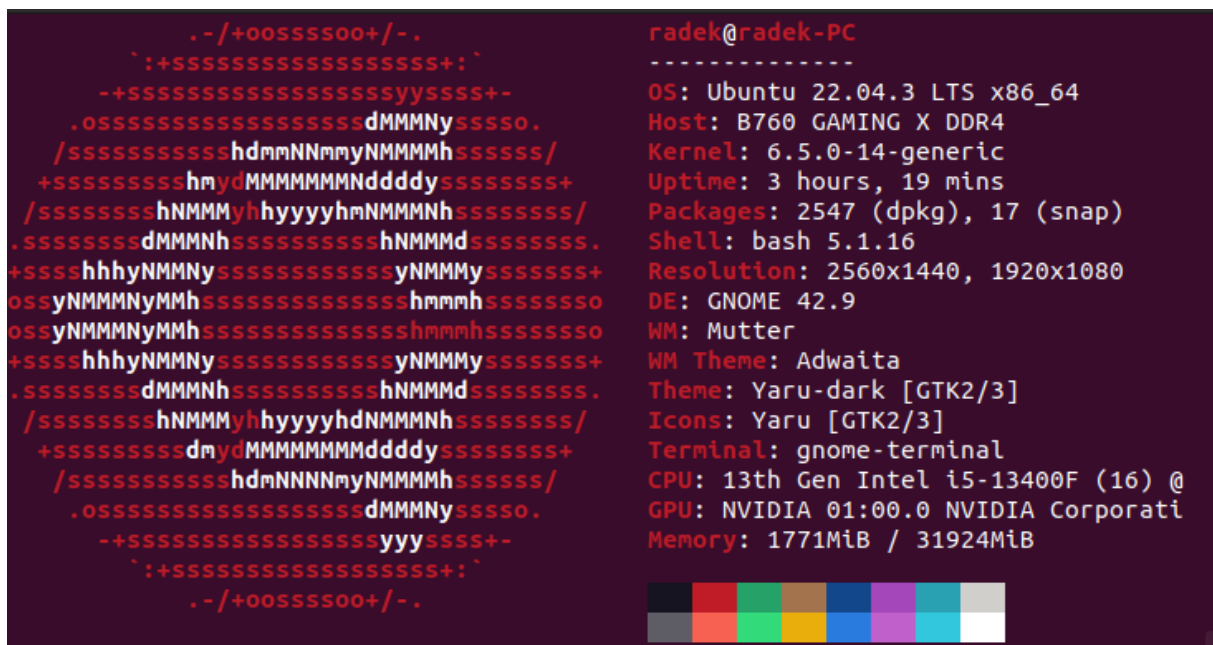
Moduł do node.js służący do tworzenia unikatowych identyfikatorów. Są one zazwyczaj wykorzystywane do identyfikowania encji w bazach danych. Ma to tę zaletę, że nie można odgadnąć identyfikatorów kolejnych wpisów w tablicy. W tym projekcie wykorzystano te identyfikatory do nazywania plików tymczasowych tworzonych przez aplikację serwerową i aplikację filtrującą. Moduł ten ma licencję MIT.

Rozdział 6

Weryfikacja i walidacja

6.1 Środowisko testowe

Projekt był testowany na komputerze osobistym z systemem Ubuntu 22.04. Program był uruchamiany z poziomu programu Visual Studio Code. Aplikacja kliencka była testowana w przeglądarce chromium. Dodatkowe dane dotyczące systemu przedstawione są na rysunku 6.1.



```
.-/+00ssss00+/- .
':+ssssssssssssssss+:`
-+ssssssssssssssssyyssss+-
.ossssssssssssssssdMMMnyssssso.
/ssssssssssshdmmNNmyNMMMMhssssss/
+ssssssssshmydMMMMMMNdddyssssssss+
/ssssssssshNMMMyhhyyyhNMMMMhssssssss/
.ssssssssdMMMhssssssssshNMMMdssssssss.
+ssssshhhyNMMNysssssssssssyNMMMyssssss+
ossyNMMNyMMhssssssssssshmmhssssssso
ossyNMMNyMMhssssssssssshmmhssssssso
+ssssshhhyNMMNysssssssssssyNMMMyssssss+
.ssssssssdMMMhssssssssshNMMMdssssssss.
/ssssssssshNMMMyhhyyyhNMMMMhssssssss/
+sssssssssdmydMMMMMMNdddyssssssss+
/ssssssssssshdmmNNmyNMMMMhssssss/
.ossssssssssssssssdMMMnyssssso.
-+ssssssssssssssssyyssss+-
':+ssssssssssssssss+:`
.-/+00ssss00+/- .

radek@radek-PC
-----
OS: Ubuntu 22.04.3 LTS x86_64
Host: B760 GAMING X DDR4
Kernel: 6.5.0-14-generic
Uptime: 3 hours, 19 mins
Packages: 2547 (dpkg), 17 (snap)
Shell: bash 5.1.16
Resolution: 2560x1440, 1920x1080
DE: GNOME 42.9
WM: Mutter
WM Theme: Adwaita
Theme: Yaru-dark [GTK2/3]
Icons: Yaru [GTK2/3]
Terminal: gnome-terminal
CPU: 13th Gen Intel i5-13400F (16) @
GPU: NVIDIA 01:00.0 NVIDIA Corporati
Memory: 1771MiB / 31924MiB
```

Rysunek 6.1: Informacje systemowe uzyskane komendą neofetch.

6.2 Testowanie w trakcie tworzenia

6.2.1 Aplikacja filtrująca

Wszystkie trzy aplikacje tworzące niniejszy projekt były implementowane w sposób przyrostowy. Kolejne funkcjonalności były testowane zaraz po ich implementacji. Aplikacja filtrująca jest programem konsolowym. Była ona na bieżąco testowana przy pomocy debuggera i terminala, w którym wyświetlane były wyniki działania kolejnych funkcjonalności aplikacji. Wywołania funkcji `console.log()` zostały usunięte z ostatecznej wersji aplikacji. Do testowania użyte zostały przykładowe zdjęcia, na których obserwowane były postępy w implementacji algorytmów. Aplikacja filtrująca została zaimplementowana jako pierwsza.

6.2.2 Aplikacja backendowa

Aplikacja backendowa była testowana przy pomocy debuggera, konsoli, a także przy pomocy programu postman. W pierwszej kolejności zostały zaimplementowane endpointy aplikacji, które były testowane przy pomocy Postmana i mockupu bazy danych. Mockup stanowiła prosta tablica obiektów. Testowe zapytania HTTP zostały zaprojektowane w postmanie, a następnie były przesyłane do aplikacji backendowej. Odpowiedzi serwera były weryfikowane w postmanie. Gdy wszystkie endpointy zostały sprawdzone, aplikacja serwerowa została połączona z bazą danych. Po usunięciu mockupu API serwera było testowane przy pomocy postmana.

6.2.3 Aplikacja frontendowa

Aplikacja frontendowa była testowana przy pomocy przeglądarki chromium i zaimplementowanych w niej narzędzi deweloperskich. Aplikacja ta powstała w trzech fazach. W pierwszej fazie zaimplementowana została cała struktura aplikacji, a także wewnętrzna logika aplikacji. W drugiej fazie formularze tej aplikacji zostały połączone z serwerem przy pomocy funkcji `fetch()`. Następnie aplikacja była ręcznie testowana w przeglądarce. W czasie ręcznego testowania implementowane były ostateczne poprawki. Wyniki działania aplikacji można obejrzeć w rozdziale „Specyfikacja zewnętrzna”.

6.3 Wykryte błędy

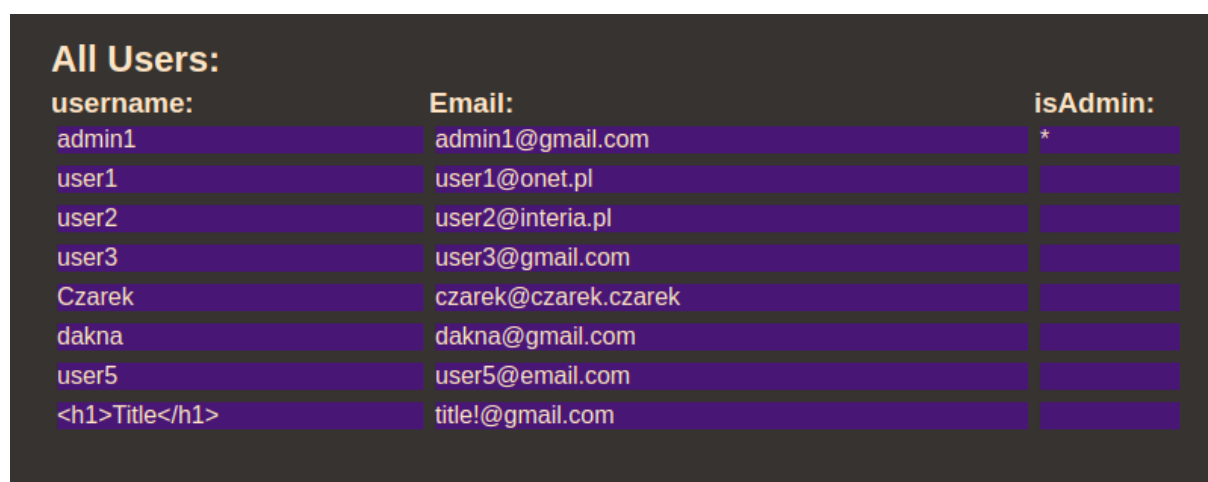
Częstym błędem przy implementacji algorytmów filtracji było przetworzenie jedynie jednej czwartej obrazu. Spowodowane było to tym, że piksele wewnątrz programu były kodowane na czterech bajtach informacji, zatem filtracja musiała być wykonana cztery razy więcej niż jest pikseli w obrazie.

Innym błędem, którego wykrycie zajęło wiele czasu była literówka w nazwie bazy danych. Uniemożliwiła ona połączenie się serwera z bazą danych. Błąd został wykryty dopiero, gdy kod programu i informacje o bazie danych zostały przekazane chatbotowi ChatGPT.

Program był testowany dla następujących formatów plików: JPG, PNG, GIF, TIFF, BMP. Dla formatów JPG, PNG i GIF program działa prawidłowo. Nie działa on dla formatu TIFF, gdyż te pliki nie wyświetlają się poprawnie w przeglądarce. Natomiast dla formatu BMP czasami pojawiał się błąd, w którym obraz wynikowy zmieniał kolor na czerwony. Metodą prób i błędów ustalono, że spowodowane jest to opcjonalnym nagłówkiem o paletce barw, który może zawierać plik BMP. Program działa poprawnie dla 24-bitowego obrazu BPM bez informacji o paletce barw w nagłówku.

6.4 Testy bezpieczeństwa

Podczas testów sprawdzono działanie prostego ataku hakerskiego XSS (ang. cross site scripting), podczas którego spróbowano utworzyć użytkownika o nazwie: „<h1>Title</h1>”. Użytkownika tego udało się utworzyć i można się z tego konta zalogować, ale na liście użytkowników nazwa ta wyświetla się jako zwykły tekst a nie jako nagłówek (rysunek 6.2). Świadczy to o tym, że panel administracyjny jest niepodatny na tego typu ataki.



All Users:		
username:	Email:	isAdmin:
admin1	admin1@gmail.com	*
user1	user1@onet.pl	
user2	user2@interia.pl	
user3	user3@gmail.com	
Czarek	czarek@czarek.czarek	
dakna	dakna@gmail.com	
user5	user5@email.com	
<h1>Title</h1>	title!@gmail.com	

Rysunek 6.2: Przedstawienie nieudanej próby ataku XSS.

Rozdział 7

Podsumowanie i wnioski

7.1 Weryfikacja założeń

W tym projekcie udało się zrealizować wszystkie założenia projektowanej aplikacji. Powstał zestaw programów będący kompletną i funkcjonalną aplikacją webową. Projekt ten stanowi rzetelne odzwierciedlenie umiejętności autora, który samodzielnie zaimplementował wszystkie części projektu: bazę danych, algorytmy filtrów, aplikację serwerową zgodną z REST oraz stronę internetową. Projekt ten był dla autora okazją do zapoznania się z nową technologią, mianowicie biblioteką React, a także możliwością zweryfikowania i utrwalenia jego wiedzy oraz umiejętności programistycznych.

W ukończonej aplikacji można tworzyć i usuwać użytkowników, co umożliwia dawanie i odbieranie do niej dostępu konkretnym osobom. Osoby, które mają dostęp do tej aplikacji mogą wybrać konkretny plik z obrazem, dokonać filtracji jednym z zaimplementowanych filtrów, mogą pobrać obraz wynikowy, jednocześnie mają podgląd obrazu przed i po filtracji.

7.2 Potencjalne kierunki rozwoju aplikacji

Przed dalszą rozbudową projekt ten należy wzbogacić dodatkowo o zestaw testów automatycznych zrealizowanych przy pomocy odpowiednich narzędzi. Dalsza rozbudowa jest jak najbardziej możliwa. dodanie kolejnych filtrów wymaga modyfikacji aplikacji filtrującej i aplikacji frontendowej. Nie wymaga natomiast zmian w aplikacji backendowej i bazie danych. Dodanie funkcjonalności zapisu obrazów użytkownika w bazie danych wymaga modyfikacji bazy danych, aplikacji frontendowej oraz serwera.

Jednym z możliwych kierunków rozwoju aplikacji jest implementacja algorytmów filtracji dla karty graficznej. Przykładową technologią, którą można w tym celu wykorzystać są rdzenie CUDA (ang. Compute Unified Device Architecture) kart graficznych firmy Nvidia. CUDA to platforma programowania równoległego i API programistyczne służące do

pisania i wykonywania kodu szerokiego zastosowania na kartach graficznych. Taka implementacja umożliwiłaby jeszcze szybsze i sprawniejsze dokonywanie filtracji.

7.3 Wady projektu

Główną wadą projektu jest to, że polega ona na wywoływaniu podprocesów, a informacje są przekazywane do nich poprzez zapis i odczyt plików z dysku. W przypadku filtracji dużych plików graficznych prowadzi to do poważnego spowolnienia aplikacji przez wielokrotne operacje odczytu i zapisu na dysku. Aby temu zapobiec aplikacja filtrująca powinna być zintegrowana z aplikacją backendową w jeden program. Natomiast części napisane w języku C powinny być wywoływane z kodu programu jako dodatki (ang. *addon*) zgodne ze środowiskiem Node, a nie jako osobne podprogramy, podprocesy jak to ma miejsce teraz.

Kolejną częścią wymagającą poprawek jest aplikacja kliencka. Liście użytkowników w panelu administracyjnym brakuje stronicowania, wyświetlani są wszyscy użytkownicy, a nie 10 czy 20 rekordów. Gdyby liczba użytkowników aplikacji zwiększyła się do kilkuset mogłoby to stanowić pewien problem. Dodatkowo kod tej aplikacji jest zorganizowany w dużych plikach, zatem część logiki należałoby wydzielić z komponentów (ang. *custom hooks*).

Bibliografia

- [1] *About - Git*. 2023. URL: <https://git-scm.com/about> (term. wiz. 13.12.2023).
- [2] *About npm / npm Docs*. 2023. URL: <https://docs.npmjs.com/about-npm> (term. wiz. 12.12.2023).
- [3] Wesley Bylsma. *Algorithm Alley*. URL: <http://www.drdobbs.com/parallel/algorithm-alley/184411079> (term. wiz. 11.11.2023).
- [4] *C (programming language)*. 2023. URL: [https://en.wikipedia.org/w/index.php?title=C_\(programming_language\)&oldid=1189830224](https://en.wikipedia.org/w/index.php?title=C_(programming_language)&oldid=1189830224) (term. wiz. 14.12.2023).
- [5] *Express/Node introduction - Learn web development / MDN*. 2023. URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction (term. wiz. 10.12.2023).
- [6] *Getting started with React - Learn web development / MDN*. 2023. URL: https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_getting_started (term. wiz. 12.12.2023).
- [7] *Introduction to Node.js / Node.js*. 2023. URL: <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs> (term. wiz. 10.12.2023).
- [8] Kashif Iqbal. *BMP - Image File Format*. 2023. URL: <https://docs.fileformat.com/image/bmp/> (term. wiz. 13.10.2023).
- [9] *JavaScript*. 2023. URL: <https://en.wikipedia.org/w/index.php?title=JavaScript&oldid=1188939975> (term. wiz. 12.12.2023).
- [10] *jimp*. 2023. URL: <https://www.npmjs.com/package/jimp> (term. wiz. 12.12.2023).
- [11] Tomasz Lubiński. *Filtrowanie obrazów - Algorytmy i Struktury Danych*. 1999. URL: <http://www.algorytm.org/przetwarzanie-obrazow/filtrowanie-obrazow.html> (term. wiz. 14.10.2023).
- [12] Alina Momot. „filtracja obrazów cyfrowych z wykorzystaniem Bayesowskiego ważonego uśredniania”. W: *Studia Informatica* 31.2A (89) (2010), s. 347–362.
- [13] Elise Moreau. *Here's What You Need to Know About Posting Stories on Snapchat*. 2021. URL: <https://www.lifewire.com/what-is-a-snapchat-story-3486000> (term. wiz. 14.11.2023).

- [14] *MySQL :: MySQL 8.0 Reference Manual :: 1.2.1 What is MySQL?* 2023. URL: <https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html> (term. wiz. 14.12.2023).
- [15] Przemysław Korohoda Ryszard Tadeusiewicz. *Komputerowa analiza i przetwarzanie obrazów*. Kraków: Fundacja Postępu Telekomunikacji, 1997. ISBN: 83-86476-15-X.
- [16] *Snapchat*. 2023. URL: <https://en.wikipedia.org/w/index.php?title=Snapchat&oldid=1177600994> (term. wiz. 14.11.2023).
- [17] *Vite*. 2023. URL: <https://vitejs.dev> (term. wiz. 14.12.2023).
- [18] *What is Postman? Postman API Platform*. 2024. URL: <https://www.postman.com/product/what-is-postman/> (term. wiz. 16.01.2024).

Dodatki

Spis skrótów i symboli

API ang. Application Programming Interface

BMP ang. bitmap, popularny format plików graficznych

BSD-2-Clause wolna licencja oprogramowania, ang. Berkley Software Distribution 2-Clause license

CLI ang. Command Line Interface

ECMA ang. European Association for Standardizing Information and Communication Systems

ES2021 ECMAScript 2021, najnowszy w czasie pisania tej pracy standard Java Script

GIF format pliku graficznego

HTTP ang. Hypertext Transfer Protocol

ISC otwarta licencja oprogramowania

JPG format pliku graficznego

JSX ang. JavaScript Syntax Extension

MIT otwarta licencja oprogramowania

MVC wzorzec projektowy ang. model view controller

PNG format pliku graficznego

REST ang. representational state transfer

SQL ang. Structured Query Language

sRGB ang. standarised Red, Green, Blue, najpowszechniej wykorzystywana przestrzeń barw składająca się z ponad 16 mln odcieni

TIFF format pliku graficznego

XSS ang. cross site scripting, rodzaj ataku hakerskiego

Lista dodatkowych plików, uzupełniających tekst pracy

W systemie do pracy dołączono dodatkowe pliki zawierające:

- źródła programu,
- film pokazujący działanie opracowanego oprogramowania,

Spis rysunków

2.1	Zrzut ekranu przedstawiający stronę photofilters.com.	8
2.2	Zrzut ekranu przedstawiający aplikację PhotoMania.	9
3.1	Diagram przypadków użycia	13
4.1	Zrzut ekranu przedstawiający panel logowania.	23
4.2	Zrzut ekranu przedstawiający okienko wyboru pliku.	23
4.3	Zrzut ekranu przedstawiający wynik działania filtra medianowego.	24
4.4	Zrzut ekranu przedstawiający panel administracyjny.	25
4.5	Zrzut ekranu przedstawiający wypełniony formularz dodawania nowego użytkownika.	26
4.6	Zrzut ekranu przedstawiający listę użytkowników.	26
4.7	Zdjęcie przedstawiające psa.	27
4.8	Zdjęcie przedstawiające psa, które zostało przetworzone filtrem bayesowskim.	27
4.9	Zdjęcie przedstawiające psa, które zostało zaszumione.	28
4.10	Zdjęcie przedstawiające psa, które zostało zaszumione i przetworzone filtrem uśredniającym.	28
4.11	Zdjęcie przedstawiające psa, które zostało zaszumione i przetworzone filtrem medianowym.	29
4.12	Zdjęcie przedstawiające psa, które zostało zaszumione i przetworzone filtrem bayesowskim.	29
5.1	Zrzut ekranu przedstawiający schemat bazy danych.	32
5.2	Zrzut ekranu przedstawiający zapis pliku z obrazem i wywołanie podpro- gramu filtrującego.	33
5.3	Zrzut ekranu przedstawiający wywołanie funkcji fetch.	33
5.4	Przykład użycia funkcji routingowej get, która zwraca listę wszystkich użyt- kowników.	34
5.5	Implementacja algorytmu filtracji medianowej.	34
5.6	Implementacja algorytmu filtracji liniowej.	35
5.7	Implementacja algorytmu filtracji bayesowskiej.	36

6.1	Informacje systemowe uzyskane komendą neofetch.	39
6.2	Przedstawienie nieudanej próby ataku XSS.	41