Streetview

Dokumentacja projektowa PZSP2

Wersja 2.0

24.04.23

Semestr 23L

Zespół nr 8 w składzie:

Franek Kozioł

Andrii Demydenko

Minh Nguyen Con

Mentor zespołu: Krystian Radlak

Spis treści

[1 Wprowadzenie 2](#_Toc136614666)

[1.1 Cel projektu 2](#_Toc136614667)

[1.2 Wstępna wizja projektu 2](#_Toc136614668)

[1.3 Problematyka problemu 2](#_Toc136614669)

[2 Analiza wymagań 3](#_Toc136614670)

[2.1 Wymagania użytkownika i biznesowe 3](#_Toc136614671)

[2.2 Wymagania funkcjonalne i niefunkcjonalne 3](#_Toc136614672)

[2.3 Przypadki użycia 3](#_Toc136614673)

[3 Definicja architektury 4](#_Toc136614674)

[4 Dane trwałe 4](#_Toc136614675)

[5 Specyfikacja analityczna i projektowa 5](#_Toc136614676)

[6 Projekt standardu interfejsu użytkownika 5](#_Toc136614677)

[7 Specyfikacja testów 6](#_Toc136614678)

[7.1 Testowanie ręczne 6](#_Toc136614679)

[7.1.1 Testy zgodności 6](#_Toc136614680)

[7.1.2 Renderowanie obrazu sferycznego 6](#_Toc136614681)

[7.1.3 Interakcja użytkownika 6](#_Toc136614682)

[7.2 Testy jednostkowe 6](#_Toc136614683)

[*8* *Wirtualizacja/konteneryzacja* 7](#_Toc136614684)

[9 Bezpieczeństwo 7](#_Toc136614685)

[10 Podręcznik użytkownika 7](#_Toc136614686)

[11 Podręcznik administratora 7](#_Toc136614687)

[12 Podsumowanie 8](#_Toc136614688)

[13 Bibliografia 9](#_Toc136614689)

# Wprowadzenie

## Cel projektu

Celem projektu jest wytworzenie wolnego i otwartego oprogramowania umożliwiającego przeglądanie zbioru zdjęć sferycznych w przeglądarce internetowej z uwzględnieniem urządzeń o niskiej mocy obliczeniowej, urządzeń mobilnych, urządzeń z ograniczonym dostępem do internetu i przeglądarek bez wsparcia dla JavaScript.

## Wstępna wizja projektu

Jako, że projekt ma nie korzystać z javascript, do „obracania się”, będą wykorzystywane przyciski: góra, dół, prawo, lewo. Kliknięcie przycisku spowoduje wyrenderowanie obrazka lekko obróconego w pożądaną stronę.

## Problematyka problemu

Problem tego projektu polega na przekształceniu zdjęcia wejściowego (odwzorcowania równoodległościowego) na projekcję prostoliniową (Oba zdjęcia są w formacie jpg).

A picture containing art, building, light

Description automatically generated

Możemy rzucić promień ze środka kuli przez powierzchnię kuli w punkcie P1, który styka się z płaszczyzną w punkcie P, gdzie płaszczyzna jest styczna do kuli w punkcie S, jak pokazano na rysunku. To odwzorowanie punktu P1 na P na powierzchni jest tym, co chcemy uzyskać

A picture containing diagram, sketch, line, text

Description automatically generated

Punkt S ma szerokość i długość geograficzną (λ, ϕ) = (0,0) i dlatego leży na równiku. Powyżej są równania przekształcenia stycznej do płaszczyzny w punkcie S o szerokości geograficznej ϕ i długości geograficznej λ, dla rzutu o długości środkowej λ0 i szerokości geograficznej środkowej ϕ1.

# Analiza wymagań

## Wymagania użytkownika i biznesowe

Wymagania biznesowe:

* Aplikacja ma być open source

Wymagania użytkownika:

* Aplikacja ma być szybka
* Aplikacja ma być niezawodna

Wymagania systemowe:

* Aplikacja ma działać na urządzeniach nie wspierających javascript
* Aplikacja ma działać na urządzeniach o niskiej mocy obliczeniowej

## Wymagania funkcjonalne i niefunkcjonalne

Wymagania funkcjonalne:

* Aplikacja umożliwia oglądanie zdjęć sferycznych
* Aplikacja umożliwia zmianę perspektywy poprzez używanie przycisków: góra, dół, lewo, prawo.
* Przyciski zmieniają kont patrzenia o 10 stopni w odpowiednią stronę.
* Nie ma limitu obracania, tj. można obracać się w nieskończoność horyzontalnie i wertykalnie.
* Można zmienić przybliżenie obrazka używając przycisków + i -.
* Aplikacja umożliwia poruszanie się po mapie stworzonej z wielu zdjęć, poprzez klikanie w kropki odpowiadające miejscom zdjęć w danej przestrzeni.
* Kierunek patrzenia i przybliżenie są zachowane po zmianie miejsca na mapie.

Wymagania niefunkcjonalne:

* Aplikacja funkcjonuje na urządzeniach nie wspierających javasript.
* Aplikacja funkcjonuje na urządzeniach o niskiej mocy obliczeniowej, takich jak telefony czy tablety.

## Przypadki użycia

A picture containing text, screenshot, diagram, line

Description automatically generated

# Definicja architektury

Diagram architektury:

A diagram of a server

Description automatically generated with medium confidence

# Dane trwałe

Obrazy będą przechowywane na komputerze host, a podczas wdrażania kontenera dockera, katalog ten jest montowany w kontenerze, wraz z plikiem mapy w formacie json.

# Specyfikacja analityczna i projektowa

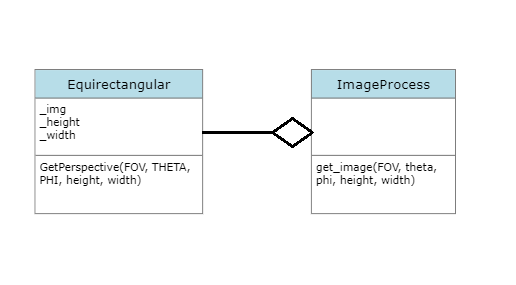
Odnośnik do repozytorium: https://github.com/congminh1254/360-webview-nojs

Aplikacja zbudowana przy użyciu Pythona i Flask. Biblioteka OpenCV używana do renderowania zdjęć.

Wykorzystana została architektura mikroserwisów z mikroserwisem Flask odpowiedzialnym za renderowanie zdjęć i zwracanie ich do klienta

Zdjęcia przechowywane jako pliki w systemie plików. Wygenerowana mapa zapisana w formacie JSON. Aplikacja uzyskuje dostęp do plików i map w celu renderowania zdjęć po stronie serwera i zwracania ich klientowi.

Diagram klas:

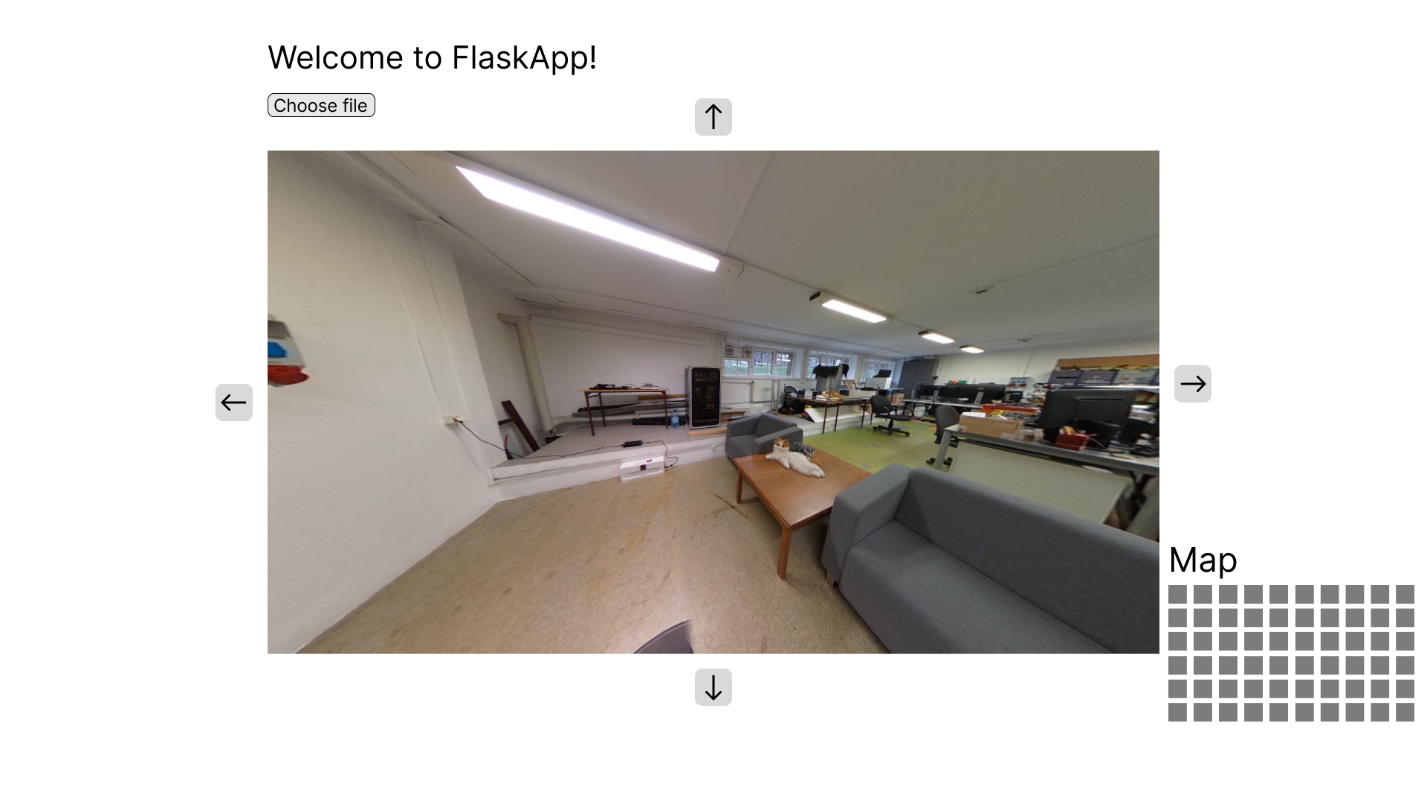


Statystyki:

* Liczba plików: 18
* Liczba linii kodu: 427
* Liczba testów jednostkowych: 2

# Projekt standardu interfejsu użytkownika

Przy użyciu programu Figma, stworzyliśmy mockup ekranu aplikacji:



Interfejs jest na tyle prosty, że nie ma potrzeby użycia wireframes, wireflows itd.

# Specyfikacja testów

Podejście testowe skupiło się na testowaniu manualnym ze względu na charakter projektu i brak zależności JavaScript. Zastosowano następujące metody badawcze:

## Testowanie ręczne

Testy ręczne odegrały kluczową rolę w weryfikacji poprawności renderowania obrazów sferycznych w różnych przeglądarkach, w tym w przeglądarkach nie wspierających JavaScript. Tester ręcznie wszedł w interakcję z aplikacją przeglądarkową, wykonując następujące czynności:

### Testy zgodności

Przeglądarka została przetestowana w różnych systemach operacyjnych i przeglądarkach, aby zapewnić bezproblemową kompatybilność na wielu platformach.

### Renderowanie obrazu sferycznego

Przetestowano różne obrazy sferyczne o różnych rozdzielczościach, współczynnikach proporcji i formatach, aby zapewnić dokładne renderowanie.

### Interakcja użytkownika

Interakcja z interfejsem przeglądarki, wykonywanie typowych czynności, takich jak lewo, prawo, góra, dół i powiększanie sferycznych obrazów.

## Testy jednostkowe

Zaimplementowano również zestaw testów jednostkowych w celu zweryfikowania krytycznych funkcji matematycznych. Te testy jednostkowe koncentrowały się przede wszystkim na funkcjach odpowiedzialnych za przekształcenia współrzędnych i obliczenia matematyczne.

Należy zauważyć, że ze względu na charakter projektu nieobsługujący języka JavaScript oraz brak złożonych interakcji lub zewnętrznych zależności testy jednostkowe były ograniczone. Większość wysiłków związanych z testowaniem była skierowana na testowanie ręczne, aby zapewnić zadowalające wrażenia użytkownika.

# *Wirtualizacja/konteneryzacja*

Docker i kontenery będą używane w tym projekcie do łatwego wdrażania i zarządzania aplikacją w różnych środowiskach. Aplikacja zostanie umieszczona w kontenerze i spakowana w obraz Dockera, który można łatwo wdrożyć na dowolnym komputerze hosta z zainstalowanym Dockerem. Kontener będzie miał preinstalowane wszystkie niezbędne zależności i konfiguracje, co ułatwi uruchamianie aplikacji bez obaw o problemy ze zgodnością. Takie podejście zapewni również izolację i bezpieczeństwo, ponieważ kontener działa niezależnie od komputera hosta i innych kontenerów.

# Bezpieczeństwo

Cała komunikacja między klientem a korzysta z protokołu HTTPS, który szyfruje przesyłane dane. Zapobiega to podsłuchiwaniu i manipulowaniu danymi.

# Podręcznik użytkownika

Użytkownik może obracać widok używając przycisków ze strzałkami. Aby zmienić zdjęcie/lokacje w której się znajduje, należy kliknąć na jeden z przycisków reprezentujących mapę.

# Podręcznik administratora

Instrukcja dla administratora, jak zainstalować projekt:

1. Sklonuj repozytorium z https://github.com/congminh1254/360-webview-nojs na lokalną maszynę.

2. Pobierz i rozpakuj zestaw danych zdjęć 360 stopni na swoim komputerze głównym.

3. Otwórz terminal i przejdź do katalogu głównego sklonowanego repozytorium.

4. Aby wyczyścić poprzedni kontener Docker, uruchom następujące polecenie:

```

docker stop 360-webview && docker rm 360-webview

```

5. Aby skompilować bieżącą wersję projektu do obrazu Dockera, uruchom następującą komendę:

```

docker build . -t 360-webview

```

6. Aby wdrożyć kontener Docker z zamontowanym katalogiem zestawu danych w /python-docker/data, uruchom następującą komendę:

```

docker run -d -it --name 360-webview --restart always --mount type=bind,source=$HOME/data\_folder,target=/python-docker/data -p 9001:9001 360-webview

```

To polecenie uruchomi kontener Docker o nazwie `360-webview`, który automatycznie uruchomi się ponownie po ponownym uruchomieniu systemu (`--restart always`). Zamontuje również katalog zestawu danych znajdujący się w `$HOME/data\_folder` w katalogu `/python-docker/data` kontenera, w którym aplikacja będzie szukać zdjęć 360 stopni. Na koniec zmapuje port 9001 kontenera na port 9001 hosta (`-p 9001:9001`), w którym aplikacja będzie dostępna z przeglądarki internetowej.

7. Opcjonalnie: aby skonfigurować Nginx do ruchu proxy z Internetu do portu 9001, wykonaj następujące kroki:

- Zainstaluj Nginx, jeśli nie jest jeszcze zainstalowany w twoim systemie.

- Otwórz plik konfiguracyjny Nginx, zwykle znajdujący się w `/etc/nginx/nginx.conf`, za pomocą edytora tekstu.

- Dodaj następujący blok kodu w bloku `http`:

```

server {

listen 80;

server\_name example.com;

location / {

proxy\_pass http://localhost:9001;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

}

}

```

- Zastąp `example.com` nazwą swojej domeny lub adresem IP.

- Zapisz i zamknij plik, a następnie uruchom ponownie Nginx za pomocą następującego polecenia:

```

sudo systemctl restart nginx

```

- Teraz powinieneś mieć dostęp do aplikacji z przeglądarki internetowej, przechodząc do „http://example.com” lub swojego adresu IP.

# Podsumowanie

Mocne strony:

* Open source
* Małe wymagania sprzętowe
* Nie potrzebuje javascript
* Proste w obsłudze

Słabe strony:

* Proste, nieimponujące UI
* Brak możliwości „przeciągania” myszką, aby się obracać
* Brak możliwości „chodzenia” (tylko klikanie na mapę)

# Bibliografia

OpenCV: https://docs.opencv.org/

Flask: https://flask.palletsprojects.com/en/2.1.x/

Docker: https://docs.docker.com/