

**AGH**

**AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE**  
WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ

## Podstawy grafiki komputerowej Dokumentacja projektu

Aleksander Siekliński

Karol Kozłowski

Michał Opióła

## Spis treści

<b>1</b>	<b>Tytuł projektu i jego autorzy</b>	<b>3</b>
<b>2</b>	<b>Opis projektu</b>	<b>3</b>
<b>3</b>	<b>Założenia wstępne przyjęte w realizacji projektu</b>	<b>3</b>
3.1	Wymagania podstawowe projektu . . . . .	3
3.2	Wymagania dodatkowe projektu . . . . .	3
<b>4</b>	<b>Analiza projektu</b>	<b>4</b>
4.1	Dane wejściowe . . . . .	4
4.2	Dane wyjściowe . . . . .	4
4.3	Struktury danych . . . . .	4
4.4	Interfejs Użytkownika . . . . .	4
4.4.1	Okno aplikacji . . . . .	4
4.4.2	Lewa i środkowa część interfejsu użytkownika . . . . .	5
4.4.3	Prawa część interfejsu użytkownika . . . . .	6
4.5	Zadania . . . . .	7
4.6	Narzędzia programistyczne . . . . .	7
<b>5</b>	<b>Podział pracy i analiza czasowa</b>	<b>7</b>
<b>6</b>	<b>Opracowanie i opis niezbędnych algorytmów</b>	<b>8</b>
<b>7</b>	<b>Kodowanie</b>	<b>8</b>
<b>8</b>	<b>Testowanie</b>	<b>9</b>
<b>9</b>	<b>Wdrożenie, raport i wnioski</b>	<b>9</b>

## 1 Tytuł projektu i jego autorzy

W ramach przedmiotu Podstawy Grafiki Komputerowej mieliśmy zrealizować projekt o tytule "NAPISY NA ZDJĘCIACH". Autorami są Aleksander Siekliński, Karol Kozłowski i Michał Opiola - studenci Informatyki Stosowanej na Wydziale Fizyki i Informatyki Stosowanej Akademii Górniczo-Hutniczej im. Stanisława Staszica w Krakowie.

### Aleksander Siekliński

Głównym zadaniem było przygotowanie środowiska, jak i oprawy graficznej do projektu oraz praca nad algorytmami potrzebnymi do poprawnego działania programu.

### Karol Kozłowski

Głównym zadaniem było znalezienie i zaimportowanie bibliotek potrzebnych do programu oraz wdrażanie i analizowanie nowych algorytmów.

### Michał Opiola

Głównym zadaniem było przygotowanie i aktualizowanie niniejszego sprawozdania, testowanie kolejnych wersji programu oraz analiza ewentualnych błędów.

## 2 Opis projektu

Celem projektu jest stworzenie wyspecjalizowanego programu do obsługi danych EXIF i IPTC. Są to dwa najpopularniejsze standardy. Informacje te nie są bezpośrednio dostępne do odczytu, jak i zapisu. Potrzebny do tego jest program, który pobierze dane zdjęcia z komputera użytkownika oraz wyświetli na ekranie ukryte dane.

## 3 Założenia wstępne przyjęte w realizacji projektu

### 3.1 Wymagania podstawowe projektu

Program powinien pobierać katalog z którego zdjęcia będą przeglądane. Po lewej powinny znajdować się miniatury zdjęć z katalogu, a po prawej stronie informacje EXIF i IPTC. Przy dwukrotnym kliknięciu zdjęcie powinno się powiększyć do rozmiaru okna. Przy dwukrotnym kliknięciu program powinien powrócić do wyświetlania miniatur.

### 3.2 Wymagania dodatkowe projektu

Program powinien generować plik tekstowy, w którym umieszczone są nazwy zdjęć oraz odpowiadające im dane, dodatkowo powinien wypisywać dane na zdjęcia i zapisywać je pod inną nazwą.

## 4 Analiza projektu

### 4.1 Dane wejściowe

Program wczytuje dane z plików w formacie JPG File i JPEG File. Większość zdjęć ma dane EXIF, ale już mało które mają dane IPTC.

### 4.2 Dane wyjściowe

Program zapisuje dane w odpowiadających im pierwotnie rozszerzeniach. Edytowany plik JPG zostanie zapisany z rozszerzeniem .jpg, plik JPEG zostanie zapisany z rozszerzeniem .jpeg.

### 4.3 Struktury danych

W projekcie korzystamy z kontenera `std::vector` z biblioteki C++ Standard Template Library (STL).

Kontenera `std::vector` używamy do:

- trzymania danych o ścieżce prowadzącej do zdjęcia
- przechowywania danych EXIF i IPTC

Stworzyliśmy własną klasę o nazwie `GUIMyFrame1`.

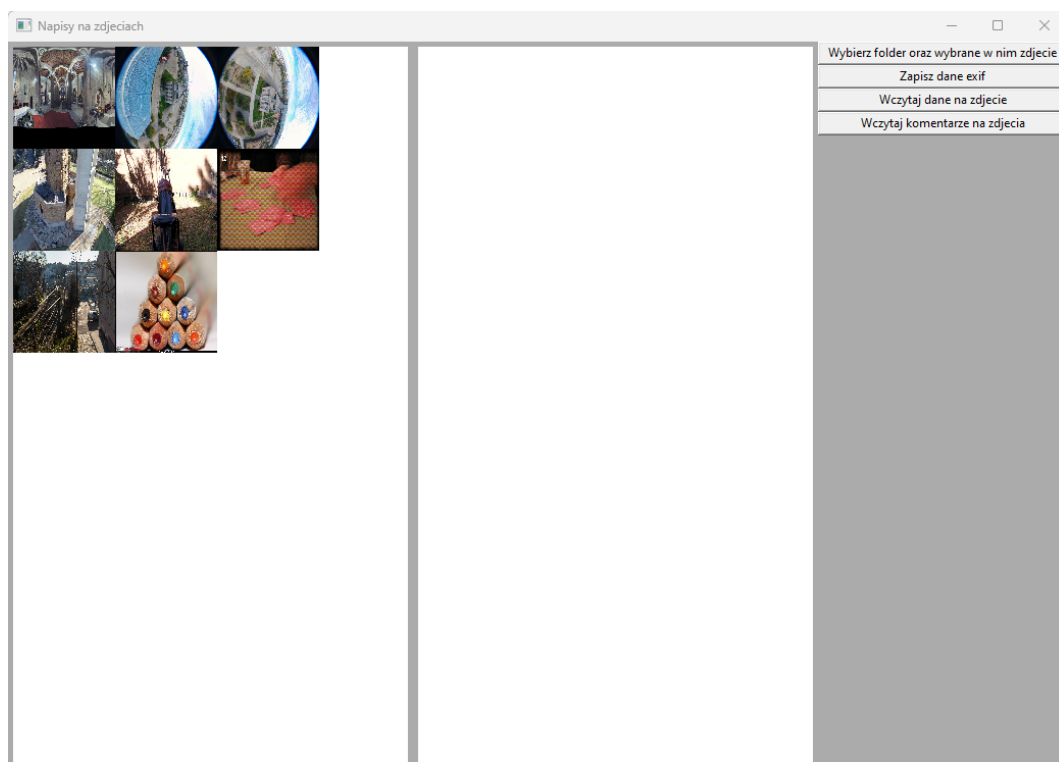
Korzystamy także z klas pochodzących z bibliotek takich jak:

- `WxWidgets`
- `TinyEXIF.h`

### 4.4 Interfejs Użytkownika

#### 4.4.1 Okno aplikacji

Jak można zauważyć na poniższym zdjęciu główny panel aplikacji został podzielony na trzy części. Po prawej stronie znajduje się menu zarządzania. W środkowej części zostają wyświetlane informacje EXIF i IPTC wybranego obrazka. Natomiast po lewej stronie znajdują się miniatury zdjęć pobrane z wczytanego folderu.




Rysunek 1: Wygląd interfejsu użytkownika po wczytaniu folderu

#### 4.4.2 Lewa i środkowa część interfejsu użytkownika

Po wczytaniu folderu ze zdjęciami po lewej stronie pojawi nam się ich zmniejszona graficzna reprezentacja. Możemy wybrać dane zdjęcie klikając na nie dwa razy, wtedy powiększy się ono do rozmiaru ramki. Możemy wyjść z powiększenia ponownie klikając myszką na zdjęcie.

Gdy zdjęcie zostanie wybrane w środkowej części pojawią się nam dane EXIF oraz IPTC. W przypadku ich braku użytkownik zostanie poinformowany, że takie dane nie zostały zapisane na danym zdjęciu.



interfejs - powiększenie zdjęcia.png

Rysunek 2: Caption

#### 4.4.3 Prawa część interfejsu użytkownika

Prawa część interfejsu odpowiada za wczytywanie i zapisywanie zdjęć. Dostępne są 4 opcje:

- Wybierz folder oraz wybrane w nim zdjęcie - wybranie tej opcji przekieruje użytkownika do przeglądarki plików. Tam użytkownik może wybrać folder ze zdjęciami, który chce wczytać do programu.
- Zapisz dane exif - Zapisuje dane ze zdjęć do pliku tekstowego.
- Wczytaj dane na zdjęcie - zapisuje zdjęcia z wybranego katalogu z ujawnionymi danymi EXIF i IPTC.
- Wczytaj komentarze na zdjęcia - zapisuje zdjęcia z komentarzem wczytanym z pliku "loader.txt".

Wybierz folder oraz wybrane w nim zdjęcie
Zapisz dane exif
Wczytaj dane na zdjęcie
Wczytaj komentarze na zdjęcia

Rysunek 3: Caption

## 4.5 Zadania

Projekt można podzielić na mniejsze części:

1. Analiza projektu, jego wymagań, używanego środowiska oraz bibliotek.
2. Stworzenie graficznego interfejsu.
3. Opracowanie i implementacja funkcji obsługujących wczytanie obrazów oraz ich wyświetlanie.
4. Opracowanie i implementacja funkcji wczytujących i wyświetlających dane EXIF i IPTC.
5. Opracowanie i implementacji funkcji edytujących zdjęcia i zapisujących je jako nowe pliki ze zmienioną nazwą.
6. Testowanie, analizowanie i poprawianie błędów.
7. Testy globalne.
8. Sporządzenie dokumentacji

## 4.6 Narzędzia programistyczne

Do naszego projektu używaliśmy Github, aby sprawnie i wygodnie aktualizować postępy projektu.

Kod pisaliśmy w programie Visual Studio Code.

Oprawa graficzna programu została zrobiona przy użyciu biblioteki wxWidgets. Dostęp do danych EXIF i IPTC umożliwiły nam biblioteki TinyEXIF oraz EXIF2.

## 5 Podział pracy i analiza czasowa

1. Wstępne omówienie projektu - 6h
  - Omówienie projektu oraz podstawowych i dodatkowych założeń.
  - Dostosowanie środowiska programistycznego na każdym z komputerów.
2. Stworzenie interfejsu graficznego - 4h
3. Grupowe omawianie projektu oraz ewentualnych zmian - 7 h
4. Implementacja algorytmów - 34h

- Funkcje obsługi zdjęć
  - Funkcje obsługi danych EXIF i IPTC.
  - Funkcje obsługi zapisywania i edycji zdjęć
  - Repozytorium na Github
5. Testowanie i edycja błędów - 17 h
- Testowanie poszczególnych funkcji
  - Poprawianie błędów i optymalizacja programu
  - Testowanie całego programu na różnych komputerach.
6. Dokumentacja - 18h
- Sporządzenie dokumentacji
  - Edycja błędów oraz aktualizacja dokumentacji podczas projektu

## 6 Opracowanie i opis niezbędnych algorytmów

W naszej aplikacji używamy klas i funkcji z dwóch bibliotek:

1. Biblioteki "wxWidgets" odpowiedzialnej za graficzny interfejs oraz przetwarzanie informacji o zdjęciach, jak i o działaniach użytkownika.
2. Biblioteki "TinyEXIF" odpowiedzialnej za pobieranie danych EXIF ze zdjęć i przekonwertowanie ich na przyjazny dla programisty format.

Po konsultacji postanowiliśmy użyć pliku iptcprint.exe, który jest efektem wcześniejszego skompilowania biblioteki przez autora zamiast używać całej biblioteki. Plik ten pozwala nam pobrać dane IPTC ze zdjęcia do pliku txt, z którego następnie są one wyświetlane w aplikacji

## 7 Kodowanie

Do poprawnego działania programu napisaliśmy własną klasę "GUIMyFrame1" która posiada struktury:

- bitmapexif, bitmapphoto - zawiera tymczasowe informacje exif oraz zdjęcie
- images - kontener trzymający informacje exif wszystkich zdjęć wczytanych z folderu
- s - przechowuje informacje o ścieżce do zdjęcia
- dire - przechowuje informacje o ścieżce do folderu, który przechowuje zdjęcia
- path - kontener trzymający informacje o ścieżce do każdego zdjęcia

oraz funkcje takie jak:

- MainFormClose - funkcja czyszcząca dane



- `WxPanel_Repaint`, `Repaint` - aktualizuje wygląd zdjęć w aplikacji
- `m.button1.click` - odpowiada za wczytanie zdjęć z folderu
- `m.button2.click` - zapisuje dane ze zdjęć do pliku tekstowego
- `m.button3.click` - umieszcza na obrazku dane w postaci białego napisu
- `m.button4.click` - umieszcza komentarz z pliku `loader.txt` na zdjęciu
- `m_panel_2lclick`, `m_panel_1lclick` - funkcje odpowiedzialne za powiększanie i pomniejszanie zdjęć
- `AddExifAndIptc` - funkcja wyświetlająca dane EXIF i IPTC po powiększeniu obrazka

## 8 Testowanie

Początkowo testy skupiały się na kompilowaniu się pojedynczych funkcji, następnie uruchamialiśmy cały program po zrobieniu każdego bloku wyznaczonego w punkcie 4.5. Testowaliśmy na wielu różnych danych wejściowych pobieranych z Internetu.

Na końcu były testy globalne polegające na testowaniu programu przez wielu użytkowników.

## 9 Wdrożenie, raport i wnioski

Cały projekt przebiegał sprawnie. Kluczowym elementem było znalezienie i zaimplementowanie odpowiednich bibliotek. Gotowe funkcje w bibliotekach `TinyEXIF`, jak i `EXIF2` w bardzo przystępnym formacie pobierały dane ze zdjęć, przez co projekt sprowadzał się do pobrania danych z plików tekstowych i zapisania ich w formie jakiej zażyczy sobie użytkownik.

Jedyny problem sprawiło zaimportowanie biblioteki `EXIF2`, gdyż nie mogliśmy jej skompilować na Windows. Po wielokrotnych porażkach postanowiliśmy skonsultować się z prowadzącym, który poradził nam pobranie skompilowanej już przez autora funkcji, która akurat jest nam potrzebna do projektu zamiast importować całą bibliotekę. Tak też zrobiliśmy i zadziałało.

Projekt nauczył nas wielu rzeczy jak np. optymalizowania czasu, pracy w zespole, zyskaliśmy doświadczenie w operowaniu Githubem, pull'owaniu i merge'owaniu requestów.