RAPORT KOŃCOWY - POS

Artur Radko 188805

Marek Omilian 188759

Jan Reza 188866

Łukasz Saldat 188714

1.

Opracowanie wymagań, modele UML zidentyfikować klienta, użytkowników końcowych i inne zainteresowane strony temat (stakeholders); zgromadzić i przeanalizować szczegółowe wymagania funkcjonalne i pozafunkcjonalne zadań; określić scenariusze użycia i stworzyć nadającą się do czytania specyfikację funkcjonalną; opcjonalnie zawrzeć w niej diagramy UML scenariuszy użycia; stworzyć model UML kluczowych elementów projektowanego rozwiązania; przeanalizować możliwość użycia dodatkowych bibliotek i narzędzi do realizacji zadania; opracować krótką specyfikację techniczną podsumowującą te informacje wraz z harmonogramen pracy; przedstawić specyfikację do zaliczenia; przygotować harmonogram pracy oraz kamienie milowe.	Microsoft Visio	10
---	-----------------	----

Opracowanie wymagań

2. Napisanie oprogramowania

Oprogramowanie realizującego przetwarzanie obrazów (po uzgodnieniu z prowadzącym, można realizować inny, własny projekt)

Założenia:

- · program uruchamiany z linii poleceń;
- program z pliku *.ini powinien odczytać ścieżki źródłowe i docelowe (do analizy pliku ini można wykorzystać dowolną darmową bibliotekę, np. iniParser);
- program odczytuje wszystkie pliki z obrazami ze ścieżki źródłowej;
- program wykonuje przetworzenie odczytanych obrazów (wielowątkowo) detekcja konturów;
- przetworzone obrazy zapisujemy w folderze docelowym;
- · w folderze docelowym zapisujemy matrycę miniatur plików źródłowych oraz matrycę miniatur obrazów docelowych (jeden plik zawierać powinien sklejone ze sobą zmniejszone obrazki źródłowe, drugi plik - sklejone obrazki po przetworzeniu, aby można było szybko zobaczyć efekty działania programu).

Visual Studio 2013 i C++ wraz z biblioteką OpenCV

20*

Proponuję najpierw zapoznać się z poszczególnymi elementami projektu:

- · analiza pliku INI;
- · odczytywanie, zapisywanie i wyświetlanie obrazów w OpenCV;
- · zasady operacji na macierzach/obrazach w OpenCV;
- · wybranie i sprawdzenie metody przetwarzania obrazów.

Uwagi dodatkowe:

- w pliku INI podajemy ścieżkę do folderu z plikami wejściowymi progam powinien przeczytać wszystkie pliki z tego folderu;
- program powienien umożliwiać pracę z dużą liczbą plików >500;
- · obrazki miniaturki powinny mieć zachowane proporcje wysokość/szerokość;

Zdjęcia testowe do testowania oprogramowania (ZIP)

3. Wykorzystanie narzędzia do pracy grupowej			l
 uzyskać dostęp do repozytorium systemu kontroli wersji (zewnętrzny hosting GitHub, Bitbucket.); wyznaczyć osobę odpowiedzialną za utrzymanie porządku w repozytorium (maintainer); zaplanować strukturę repozytorium mając na uwadze używane środowisko IDE, język 	GIT + GitHub	10	
programowania, dodatkowe artefakty (grafiki UI, specyfikacje itp.).			

- <u>GitHub</u>

Rezoot final version 1.0		12e1b6b · 1 hour ago ① 27 Commits
build	Revert "coworker test"	5 days ago
dokumentacja	documentation update	2 hours ago
outlmages	documentation update	2 hours ago
testImages	documentation update	2 hours ago
CMakeLists.txt	clean	2 days ago
☐ Datas.cpp	final version 1.0	1 hour ago
DatasImages.h	final version 1.0	1 hour ago
☐ Images.cpp	final version 1.0	1 hour ago
Config.ini	final version 1.0	1 hour ago
main.cpp	final version 1.0	1 hour ago

4.

4. Wykorzystanie systemu śledzenia błędów • uzyskać dostęp do repozytorium systemu śledzenia błędów (zewnętrzny hosting GitHub, Bitbucket.); • wyznaczyć osobę odpowiedzialną za utrzymanie porządku w repozytorium (maintainer). 10 • wykryte podczas cykli testowania błędy raportować do systemu śledzenia błędów; Bugzilla • opcjonalnie zastosować metodę test-driven development do zapewnienia jakości; • zademonstrować użycie testów jednostkowych do walidacji kluczowych elementów oprogramowania. rozwiązany 2025-06-11 00002 Dodanie komentarzy nowy 2025-06-09

@Lukasz

- Bugzilla

5.

5. Wykorzystanie wątków Windows ¹⁰

```
// Tworzenie instancji obiektu Images i listy watkow
Images images;
std::vector<std::thread> threads;
int count = 0;

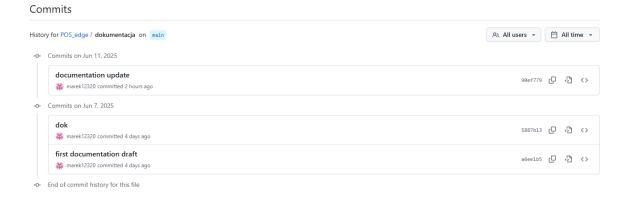
// Przetwarzanie plikow z rozszerzeniem .jpg i .png
processFilesWithExtension("jpg", images, threads, count);
processFilesWithExtension("png", images, threads, count);

// Oczekiwanie na zakonczenie wszystkich watkow
for (auto& t : threads) {
    if (t.joinable()) {
        t.join();
        count++;
    }
}
```

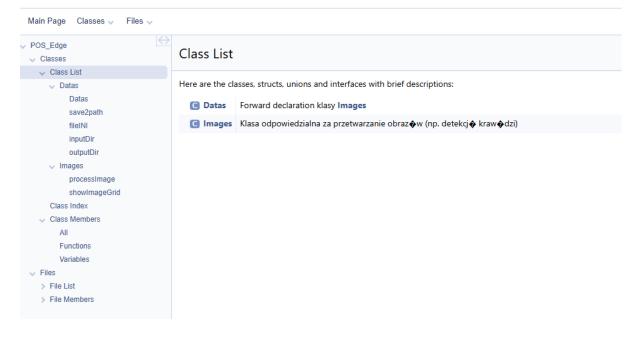
6.

	6. Wykorzystanie IPC	Windows	10	
- 1	1			ı

7. Wykonanie dokumentacji projektu • podczas implementacji zaangażować odpowiedni nakład pracy na dokumentowanie API (komentarze zgodne z Doxygen) oraz inne komentarze w kodzie, wyjaśniające szczególnie zagmatwane algorytmy, mechanizmy, zależności; • generować dokumentację HTML z komentarzy Doxygen na bieżąco, a nie po zakończeniu projektu; • przedstawić dokumentację do zaliczenia.



POS_Edge 1.0



8.

B. Wykonanie profilowania projektu (hotspot'y) użyć profiler do wykrycia hot spotów w opracowanym programie i przeprowadzić próbę ich optymalizacji.	Visual Studio Profiler	10

9.

9. Wykonanie wykrywania błędów				
Sprawdzić integralność danych w programie, np.:				
wykryć "memory leak"	Visual	Studio	+	10
 wykryć wykraczanie wskaźników (np. korzystając z makra _ASSERTE(_crtcheckMemory()); w trybie DEBUG) 	CrtCheckN	Memory		