1. Tytuł projektu i autorzy projektu

Współtwórca projektu	Przydzielone zadania
Antoni Cepak	Implementacja algorytmu odpowiedzialnego za przygotowanie wynikowego kształtu obrazu indeksu. Sporządzenie dokumentacji projektu.
Maciej Kubicki	Opracowanie głównego okna aplikacji oraz implementacja algorytmu wczytującego bitmapy do zaindeksowania ze wskazanego katalogu. Sporządzenie dokumentacji projektu.
Jakub Bolechała	Opracowanie algorytmu dopasowującego nazwy plików do dostępnej przestrzeni indeksu. Udział w tworzeniu dokumentacji.

2. Opis projektu

Projekt Fotoindeks zakładał utworzenie prostego programu pozwalającego na tworzenie indeksów zdjęć. Plik wynikowy powstały w procesie indeksowania składać się ma z 20 miniaturek. Wielkość wynikowej bitmapy ma wynieść 800x600. Miniaturki mają być rozmieszczone w 4 rzędach po 5. Każdy obrazek ma zostać podpisany nazwą pliku z którego pochodzi, w przypadku, kiedy jego nazwa przekroczy dostępną na podpis przestrzeń jego nazwa ma zostać skrócona. Jeśli nazwa zostanie skrócona to na jej początku pojawi się znak tyldy(~). Można pokusić się o stworzenie prostego graficznego interfejsu użytkownika.

3. Założenia wstępne przyjęte w realizacji projektu

Tworzenie indeksów będzie procesem składającym się z dwóch etapów. Pierwszy z nich – lekki – nie obejmuje swoim zakresem właściwego algorytmu indeksacji. Umożliwia użytkownikowi swobodną konfigurację właściwego procesu indeksacji. Użytkownik może wskazać folder, który zostanie przeszukany pod kątem obrazów z rozszerzeniami .bmp. Po dokonaniu wyboru może on upewnić się, że dokonał właściwego wyboru kontrolując aktualnie wybraną ścieżkę wyświetlaną w górnej części panelu. Dodatkowo dostarczona ilość elementów, które zostaną objęte procesem indeksacji. Dzięki temu użytkownik może zorientować się jak kosztowny będzie to proces. Drugi etap, czyli proces właściwej indeksacji nie powiedzie się w przypadku, gdy wskazana ścieżka to katalogu z plikami będzie błędna bądź w podanym przez użytkownika katalogu nie znajdą się żadne pliki w rozszerzeniem .bmp. Użytkownik zostanie poinformowany o przyczynie błędu.

4. Analiza projektu

- **specyfikacja danych wejściowych**
  Dane bezpośrednimi danymi wejściowymi specyfikowanymi przez użytkownika jest ścieżka do katalogu, który zostanie przeszukany pod kątem obecności obrazów do zindeksowania oraz tytuły wynikowych bitmap umieszczane w lewym dolnym rogu bitmapy. Pośrednimi danymi wejściowymi (wynikającymi z wyspecyfikowanych przez użytkownika parametrów) są same bitmapy oraz ich nazwy.
- **opis oczekiwanych danych wyjściowych**
  Dane wyjściowe to bitmapa 800x600 zawierająca wyskalowane bitmapy znalezione we wskazanym katalogu. Bitmapa ta zwierać ma 20 miniaturek z podpisami będącymi nazwą pełną nazwą pliku, bądź jej skróconą wersją. Na bitmapie znajdować się ma komentarz

użytkownika charakterystyczny dla konkretnego pliku indeksu. Nazwa plików wynikowych to „bitmapan.bmp”, gdzie n to numer porządkowy bitmapy nadany w kolejności tworzenia.

- **decyzja o wyborze narzędzi programistycznych**
  Do realizacji projektu wybrano środowisko programistyczne DEV C++ w połączeniu z biblioteką graficzną wxWidgets. Projekt był tworzony na platformie Windows. Dokonano takiego wyboru ze względu na wstępne zaznajomienie się ze wspomnianym środowiskiem oraz biblioteką w ramach zajęć ponadto zestaw ten okazał się być w pełni wystarczający do wykonania projektu. Projekt działa na pewno w wersjach wxDEV 7.3.1.3 oraz 7.4.2 (w tych wersjach kompiluje się bezproblemowo, także powinien działać też w innych wersjach).

5. Podział pracy i analiza czasowa

Projekt ze względu na swoją prostotę nie wymaga dużych nakładów pracy. Całkowity czas pracy nad projektem i dokumentacją szacowany jest na mniej niż 15h. Przewiduje się, że największe obciążenie spadnie na Macieja Kubickiego w przypadku zaistnienia takiej sytuacji prace nad dokumentacją projektu zostaną odpowiednio przeniesione na innych członków zespołu.

Harmonogram prac.

W fazie analizy projektu zdecydowano, ze w pierwszej kolejności powinno zostać utworzone główne okno programu oraz mechanizm wczytujące ścieżki do plików do zaindeksowania. Wprawdzie zadanie to nie blokuje zadania związanego z przygotowaniem właściwych plików z indeksami, jednak dzięki temu unikniemy problemów z niekompatybilnością dwóch części programów, nie będzie więc konieczne poświęcenie czasu na omawianie szczegółów związanych z ostateczną postacią poszczególnych funkcji. Osoba wykonująca drugie zadanie dostosuje się do zaproponowanego przez pierwszą interfejsu. Trzecie zadanie związane ze skracaniem nazw jest najbardziej niezależnym więc może zostać wykonane na dowolnym etapie wykonania, ustalono jedynie jej ostateczną postać.

6. Opracowanie i opis niezbędnych algorytmów

Program składa się jedynie z kilku prostych metod, których napisanie nie wymagało dokonania długiego procesu analizy a opis zaimplementowanych funkcji znajduje się w następnym punkcie.

7. Kodowanie

Opis funkcji

- **std::vector <std::string> whatIsIn(const char* str)**
  funkcja zwraca wektor stringów z nazwami plików o rozszerzeniu .bmp z katalogu wskazanego jako parametr funkcji.
- **void Projekt1Frm::WxButton1Click2(wxCommandEvent& event)**
  Funkcja ta obsługuje zdarzenie wciśnięcia przycisku „Wybierz folder”. Wyświetlane jest okno dialogowe w którym można wybrać katalog z plikami do indeksacji. Następnie wołana jest funkcja **whatIsIn** następuje zaktualizowanie wartości plików które zostaną zaindeksowane po wybraniu przycisku in „Zaindeksuj”.

- **wxString cutString(wxString name)**
  Funkcja ta skraca nazwę pliku w taki sposób aby nie przekroczyć dostępnego na podpis miejsca, w przypadku obcięcia nazwy, nazwa jest skracana i poprzedzona tyldą (~).
- **void Projekt1Frm::WxButton2Click(wxCommandEvent& event)**
  Funkcja ta obsługuje zdarzenie kliknięcia przycisku „Zindeksuj". Sprawdzana jest poprawność wskazanej ścieżki w przypadku niepoprawnej ścieżki użytkownik jest informowany o przez okno dialogowe. Następnie analizowane jest ilość plików do zaindeksowania. Brak plików do zaindeksowanie również skutkuje wyświetleniem komunikatu.
  Właściwy algorytm indeksacji zbudowany jest z 2 pętli. Zewnętrzna porusza się po indeksie będącym liczną naturalną i z zakresu $[0, ceil(liczba\_bitmap\_do\_zaindeksowania\ /\ 20)]$, po każdym obiegu tej pętli tworzona jest nowa bitmapa zawierająca miniaturki indeksowanych plików. Wewnętrzna pętla o indeksie $j \in [0, 19]$, $j \in \mathrm{N}$ rysuje przeskalowaną bitmapę na pozycji wyliczonej na podstawie indeksu j. Nazwa pliku do załadowania i wczytania pobierana jest z wcześniej przygotowanego wektora z posortowanymi nazwami plików. Plik, który zostanie wczytany wyznaczany jest na podstawie indeksów pętli $i * 20 + j$. Obrazek jest pozycjonowany przy pomocy tablicy points specyfikującej żądane położenia miniaturek. Użytkownik jest proszony o wyspecyfikowanie komentarza który zostanie umieszczony na wynikowej bitmapie.

8. Testowanie

W projekcie nie użyto narzędzia wykonującego testy automatyczne. Dużo bardziej efektywne wydały się być testy akceptacyjne. Utworzono ponad 20 plików z bitmapami o różnej wielkości 10x10 do 1000x1000, z różnej długości nazwami od nazwy 1 znakowej do 100 znakowej. Umieszczono je w katalogu Images. Następnie wykonano indeksację bitmap znajdujących się we wskazanym katalogu. Następnie oceniono bitmapy będące wynikiem indeksacji. Spełniały one wymienione w projekcie wymagania.

9. Wdrożenie, raport i wnioski

W czasie testów wykryto drobne usterki polegające na nie do końca dopracowanym interfejsie użytkownika. Po wybraniu katalogu z plikami do zaindeksowania i wybraniu opcji indeksacji użytkownik po pomyślnym procesie indeksacji nie otrzymuje komunikatu zwrotnego informującego o pomyślnym przebiegu procesu, mimo iż operacja przebiegła pomyślnie. W ostatecznej wersji umieszczony zostanie dodatkowy komunikat. Dodatkowym zaobserwowanym problemem może okazać się też konieczność wprowadzania podpisów dla każdego pliku indeksu. Może to być uciążliwe w przypadku indeksowania dużej ilości plików. W następnej wersji zostanie wprowadzony dodatkowy checkbox umożliwiający jednorazowe wpisanie komentarze, który zostanie umieszczony na wszystkich bitmapach wynikowych.