BAZA DANYCH PRACOWNIKÓW FRIMY Z INTERFEJSEM GRAFICZNYM

Autor: Bartosz Kawa

Akademia Górniczo-Hutnicza

Spis treści

Tabela Oznaczeń

OOP	Object-Oriented Programming
GUI	Graphical User Interface

1.Wstęp

Aplikacja Bazy Danych Pracowników posiadająca GUI. Celem aplikacji jest możliwość, odczytywania z pliku i zapisywania informacji o pracownikach danej firmy oraz praca na odczytanej bazie danych. Aplikacja powinna być tworzona z wykorzystaniem OOP.

Dostepne operacje na bazie danych to wyświetlanie pełnej bazy danych w miejscu do tego przeznaczonym, wyszukiwanie osób po numerze ID lub imieniu i nazwisku, dodawanie oraz usuwanie danych z bazy , tworzenie nowych baz danych oraz zarządzanie hasłem dla konkretnej bazy danych.

Każda baza danych edytowana przy pomocy aplikacji posiada swoje hasło które może być zmieniane w aplikacji, jeśli baza nie posiada hasła jest możliwość ustawienia hasła. Nowe hasło zostanie ustawione do momentu ponownej zmiany więc po zapisaniu bazy danych nowe hasło również zostanie zapisane.

2.Wymagania systemowe (*requirements*)

1. Zarys

Aplikacja przewiduje korzystanie z baz danych zapisanych na komputerze w pliku txt lub json, przy czym preferowane jest używanie baz danych w plikach json. Aplikacja powinna działać przede wszystkim na systemie linux (Linux Mint lub Ubuntu).

2. Opis

Aplikacja opiera się na oknie głównym z którego możliwe będzie zarządanie bazą danych oraz wykonywanie kolejnych operacji. Każda baza danych powinna posiadać własne hasło które jest znane dla pracowników zajmujących się zarządzaniem bazą danych i jest ono wczytywane wraz z otwieraniem bazy danych. Aby praca przy pomocy aplikacji była bardziej przejrzysta ważniejsze funkcje posiadają osobne okna. Aplikacja będzie funkcjonować na komputerach osobistych z systemem linux. Aplikacji ma zapewniać możliwość otwierania, tworzenia oraz zapisywania bazy danych. Ponadto powinna mieć możliwość wyświetlania całej bazy oraz przeszukiwania jej w celu znaleźenia konkretnego pracownika w bazie. Dla administratorów możliwe będzie usuwanie oraz dodawanie osób do bazy danych. Administrator będzie pracował w trybie ROOT który jest dostepny po podaniu hasła dla konkretnej bazy danych , hasło można zmienić znając aktualne hasło dla tej konkretnej bazy. Nowoutworzona baza danych nie posiada hasła więc możliwe jest swobodne przechodzenie między trybami zostawiając puste pole w miejscu gdzie należy wpisać hasło. W bazie danych bez hasła można będzie również w każdej chwili ustawić dowolne hasło.

Diagram przedstawiający docelowy wygląd bazy danych w aplikacji

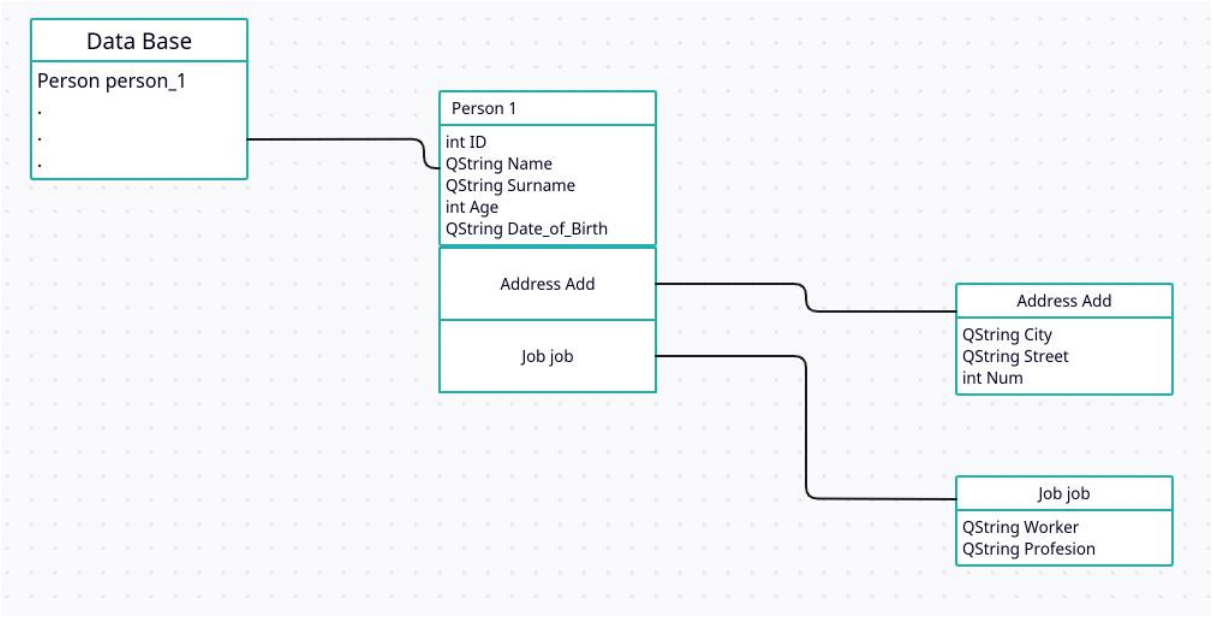


Diagram 2-1. *Przykładowy wygląd bazy danych*

Diagram przedstawia w jaki sposób będzie zorganizowana baza danych w aplikacji. Docelowo będzie to wektor obiektów klasy Person składających się z elementów przedstawionych na diagramie.

Klasy oraz zależności między nimi zostały dokladnie przedstawione w punkcie 4.Projekt techniczny.

3. Wymagania

Projekt zakłada wykorzystanie programu Qt do tworzenia aplikacji okienkowych aby stworzyć GUI programu do obsługi bazy danych. Do realizacji zostaną wykorzystane wymagane podstawowe biblioteki języka C++ oraz te dostepne przy zainstalowanym programie Qt.

Dodatkowo do operacji na plikach json wymagane jest posiadanie biblioteki powszechnie dostępnej biblioteki json.hpp, która powinna zostać umieszczona w katalogu z zasobami aplikacji.

Obsługa plików json realizowana przy użyciu kodu z pliku json.hpp ogólnie dostępnego na https://github.com/kmprograms/CppBasics/tree/master/07_Json

3.Funkcjonalność (*functionality*)

Diagram use case

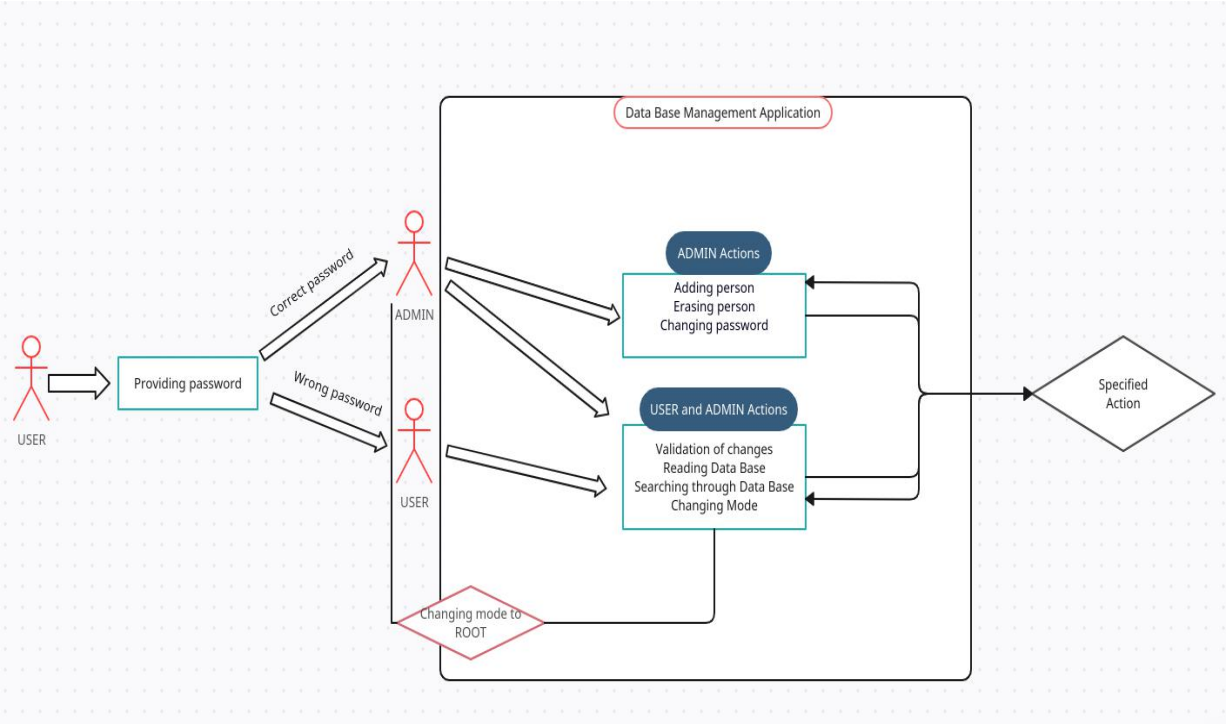


Diagram 3-1. *Use case*

Powyższy diagram przedstawia schemat użycia aplikacji po utworzeniu lub otworzeniu pliku bazy danych.

Podstawą funkcjonalności aplikacji jest możliwość tworzenia, zarządzania oraz edytowania bazy danych w formacie txt lub json. Dane w bazie muszą być zapisane w odpowiedni sposób aby możliwe było ich odpowiednie odczytanie, jednak po odczytaniu aplikacja zapisuje bazę zawsze w odpowiedni sposób z zachowaniem odpowiedniego szablonu danych w pliku bazy danych.

Pełna funkcjonalność aplikacji jest dostepna w trybie ROOT (po zmianie trybu przez podanie aktualnego hasła dla danej bazy danych , w oknie pokazanym w Podreczniku użytkownika Rysunek 5-3) , zwykły użytkownik nie ma możliwości dokonywania zmian w bazie danych (dodawania i usuwania informacji oraz oczywiście zmiany hasła). Okno dodawania osób do bazy jest przedstawione Podręczniku użytkownika Rysunek 5-4 , natomiast Okno usuwania osób z bazy pokazane jest na Rysunek 5-5.

W trybie ROOT możliwe jest dodawanie oraz usuwanie danych o kulku osobach jednocześnie lub pojedynczo, funkcjonalność ta ma na celu ułatwienie zarządania danymi w aplikacji.

Istotną funkcjonalnością jest przeszukiwanie bazy (widoczne w Onkie głównym w Podręczniku uzytkownika Rysunek 5-1) w celu znalezienia informacji o konkretnej osobie i wyświtlenia wych informacji. Wyszukiwać można po imieniu i nazwisku lub po numerze ID który jest przydzielany przy dodawaniu osony do bazy danych, każda osoba ma indywidualny numer który nie zostanie przydzielony do innej osoby nawet jeśli osoba z tym numerem nie będzie już w bazie danych.

Dodatkowo aplikacja posiada funkcję służącą do odświerzania i zatwierdzania zmian, dzieki tej funkcji uzytkownik ma pewność że wprowadzone zmiany w bazie danych zostały wprowadzone, jest to również opcja widoczna w Onkie głównym przedstawionym na Rysunku 5-1 .

4.Projekt techniczny (*technical design*)

Diagram klas

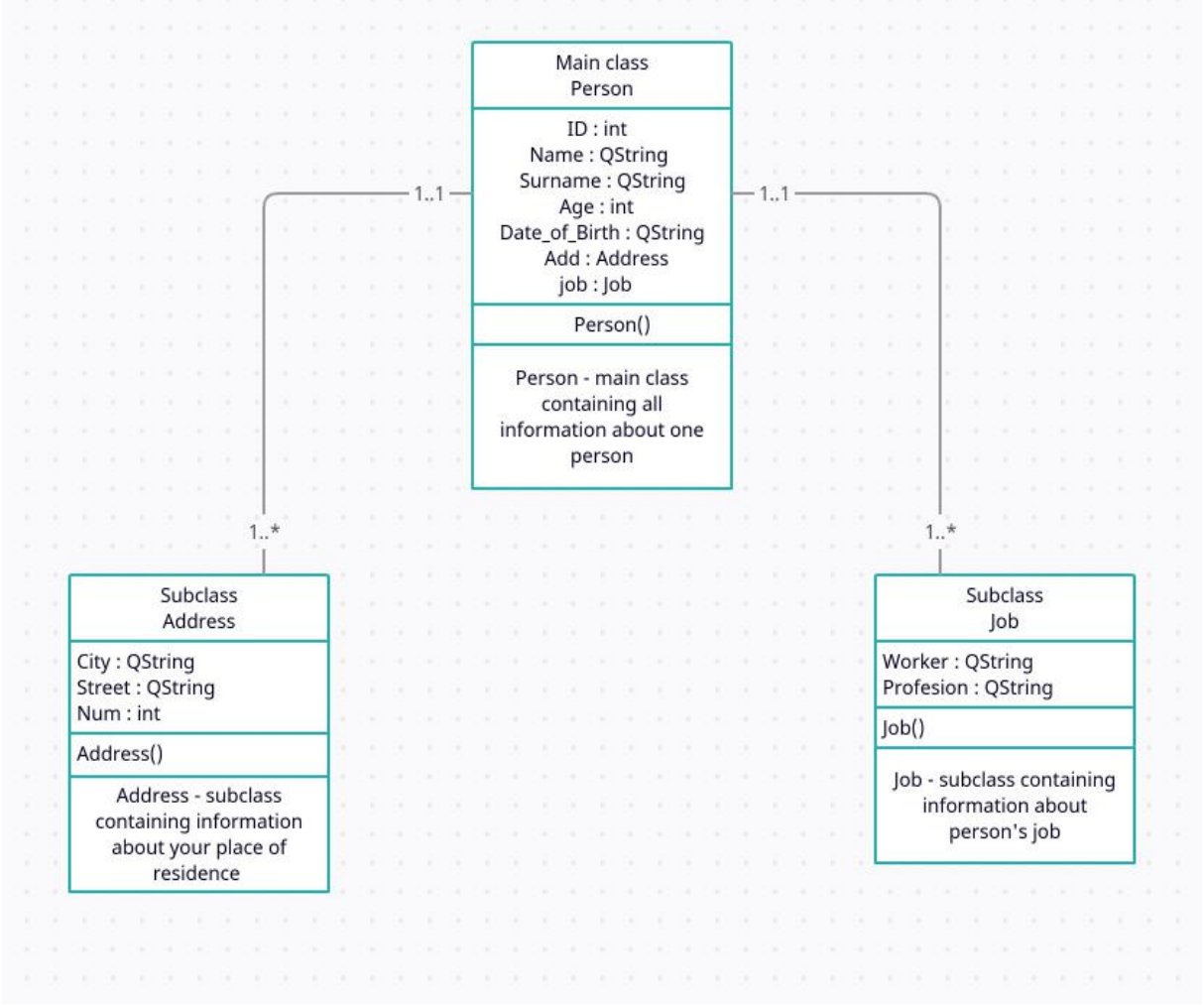


Diagram 4-1. *Diagram klas*

Na diagramie Diagram 4-1 został przedstawiony układ klas użytych w aplikacji wraz z krótkimi opisami co dana klasa przechowuje.

QString - jest to typ zmiennej dostepny w Qt podobny do zwykłego typu string ale pozwalający na więcej operacji oraz ułatwijący pracę z ciągami znaków

Job - klasa przechowuje informację o tym czym zajmuje się dany pracownik oraz jaki jest jego status. Klasa ta Job posiada swój konstruktor oraz dwa atrybuty. Wszystkie atrybuty są jako public.

Pierwszym atrybutem klasy jest zmianna typu **QString** o nazwie **Worker** , zmienna ta określa czy dana osoba jest dalej pracownikiem firmy czy może przestała już pracować w firmie ale jednak firma chce wciąż przechowywać dane , atrybut jest także przewidziany jako zmienna do

przechowywania statusu pracownika , czyli czy aktywnie pracuje badź jest na urlopie i tym podobne. Drugi artrybut klasy jest zmienną typu **QString** o nazwie **Profesion** który przechowyje dane o zawodzie bądź funkcji wykonywanej w firmie przez daną osobę.

Address - klasa przechowuje dane o miejscu zamieszkania pracownika. Klasa poza konstruktorem posiada trzy atrybuty. Wszystkie atrybuty są jako public.

Pierwszy atrybut opisuje miasto zamieszkania pracownika i jest to zmienna typu **QString** o nazwie **City**. Drugim atrybutem jest podobnie zmienna typy **QString** o nazwie **Street** która zawiera infomację o ulicy na której mieszka dany pracownik natomiast trzecim atrybutem jest zmienna typu **int** o nazwie **Num** której wartość określa numer mieszkania.

Person - jest to główna klasa wykorzystywana do przechowywania informacji o danum pracowniku i zbiera wszytkie dane wykorzystując do tego poprzednio omówione klasy. Klasa posiada siedem atrybutów oraz konstruktor. Wszystkie atrybuty są jako public.

Pierwszym atrybutem klasy jest numer **ID** który zostaje przydzielony osobie przydodawaniu jej do bazy danych , numer ten jest unikalny dla każdego pracownika i jeden numer nie może zostać przydzielony ponownie nawet jeśli pracownika który miał taki numer ID nie ma już w bazie danych , jest to zabieg mający na celu zapobieganie wszelikm pomyłkom i zapewniający możliwość identyfikacji kożdej osoby która miała styczność z firmą. Drugi oraz trzeci atrybut przechowują kolejno imię i nazwisko pracownika, są to zmienne typu **QString** o nazwach **Name** i **Surname**. Kolejnym atrybutem jest zmienna typu **int** o nazwie **Age** przechowująca wiek pracownika. Piąty atrybut to zmianna typu QString przechowująca informację o dacie urodzenia pracownika, nazwa zmiennej to Date_of_Birth.

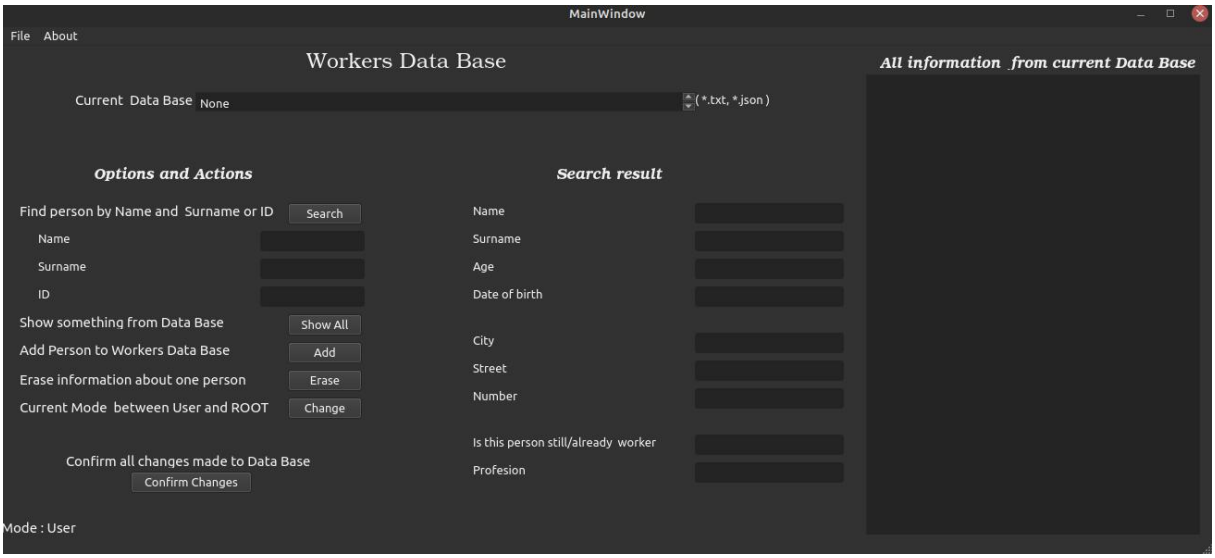
Pozostałe dwa atrybuty to obiekty wcześniej wymienionych klas **Address** i **Job** które dopełniają pełen zestaw informacji o konkretnym pracowniku.

Poza wymienionymi oraz opisanymi klasami aplikacja używa również wbudowanych klas Qt dostępnych w Qt Creator który to został użyty do wykonania aplikacji. Klasy Qt zostały wykorzystane do tworzenia oraz obsługi okien aplikacji.

Działanie całej aplikacji opiera się na połączonej pracy wszystkich wyżej wymienionych klas użytych w projektowaniu aplikacji za pomocą programu Qt Creator.

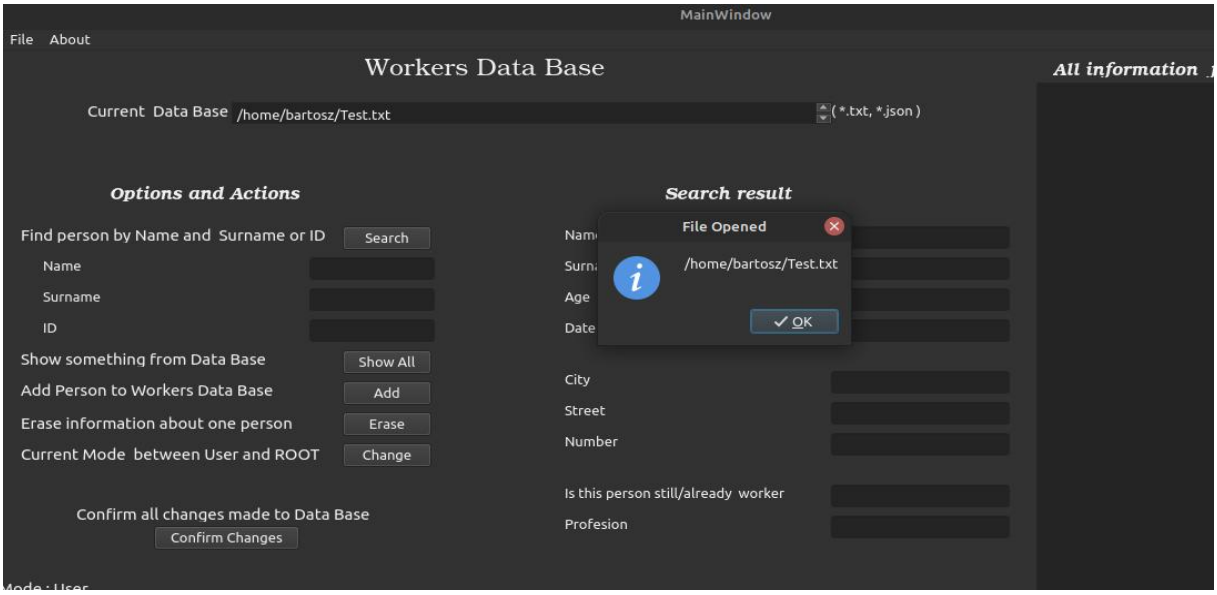
5.Opis wykonanych testów (*testing report*)

W celu testowania poprawnego działania aplikacji zostały przeprowadzaone testy manualne na dwóch przykładowych bazach, jedna w formacie txt oraz druga w formacie json. Sprawdzone zostały podstawowe funkcjonalności pod kątem poprawności działania oraz zachowania aplikacji.

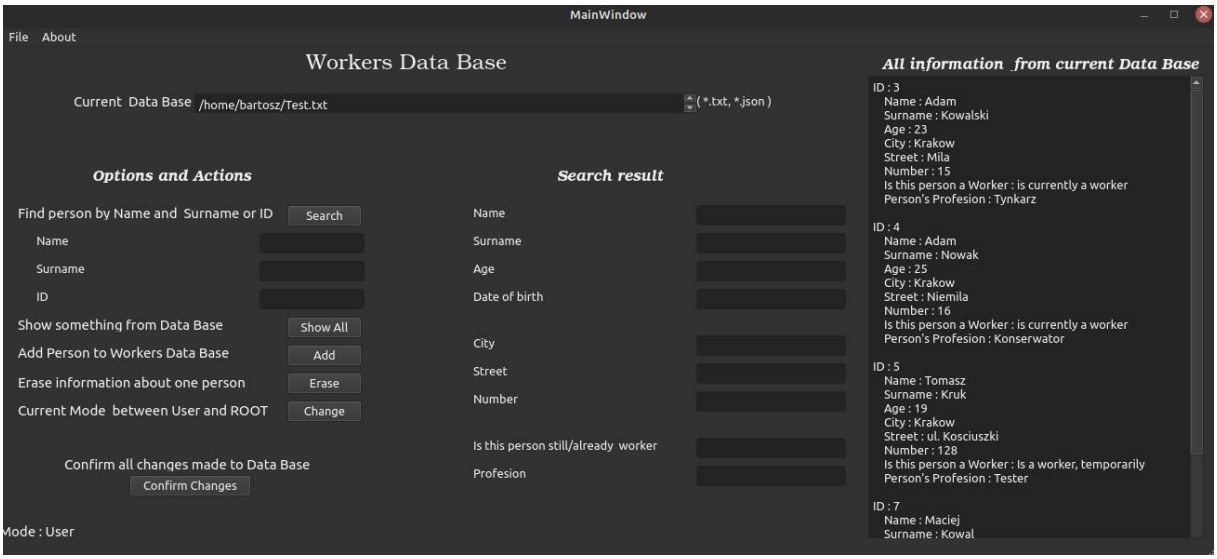


Aplikacja uruchamia się poprawnie.

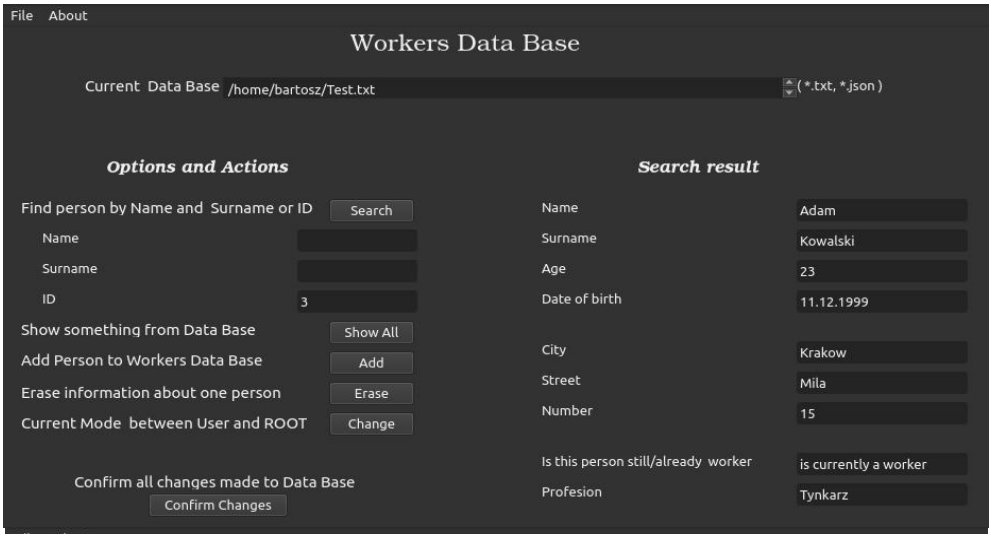
Test1 - Test bazy txt

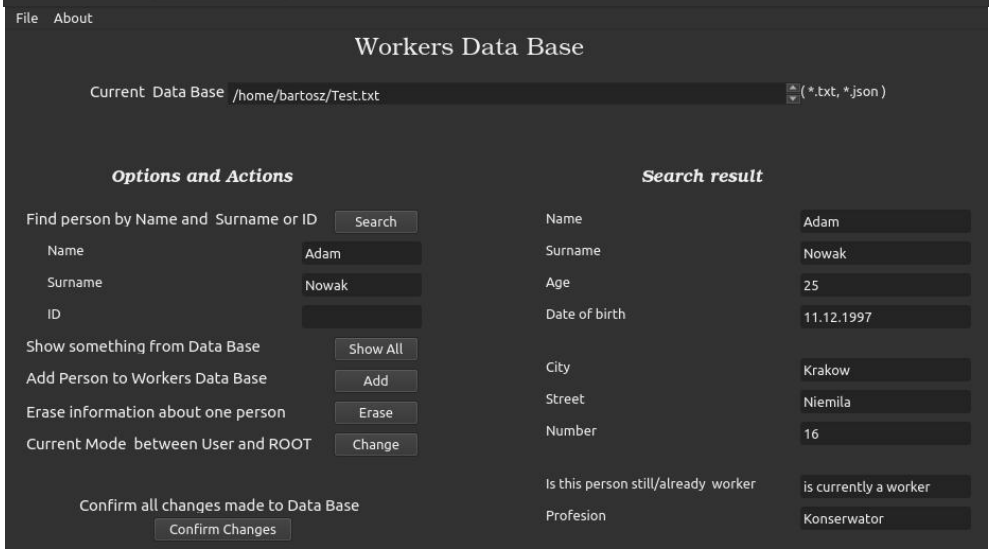


Po porawnym uruchomieniu aplikacji możliwe jest otworzenie pliku testowego w formacie txt.

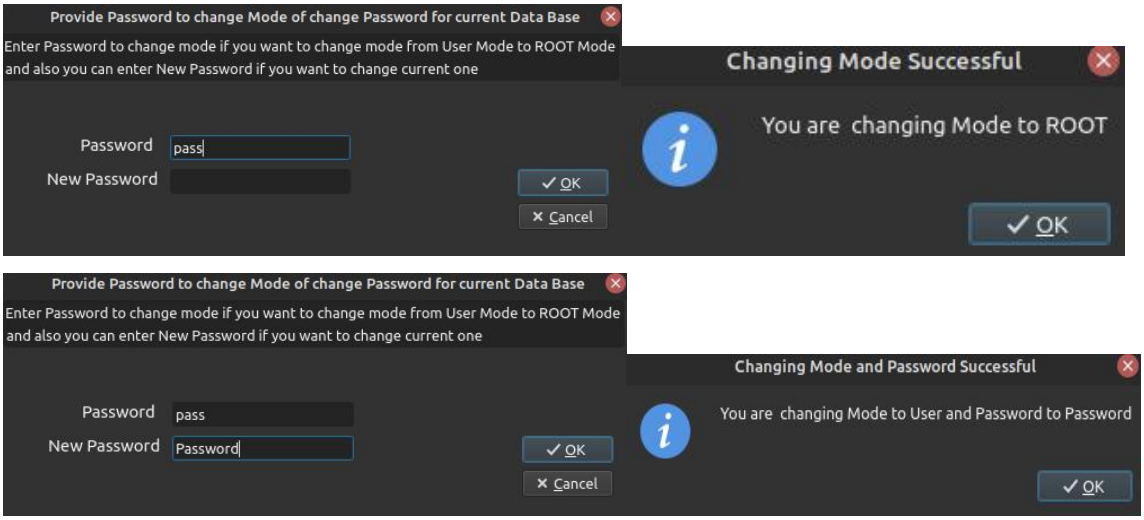


Zdjęcie pokazuje poprawność działania funkcji Show All która pokazuje całą bazę danych w wyznaczonym miejscu. Dzięki użyciu tej funkcji widzimy że baza danych została poprawnie odczytana z pliku txt. Lokalizacja danej bazy danych jest również poprawnie wyświetlana.

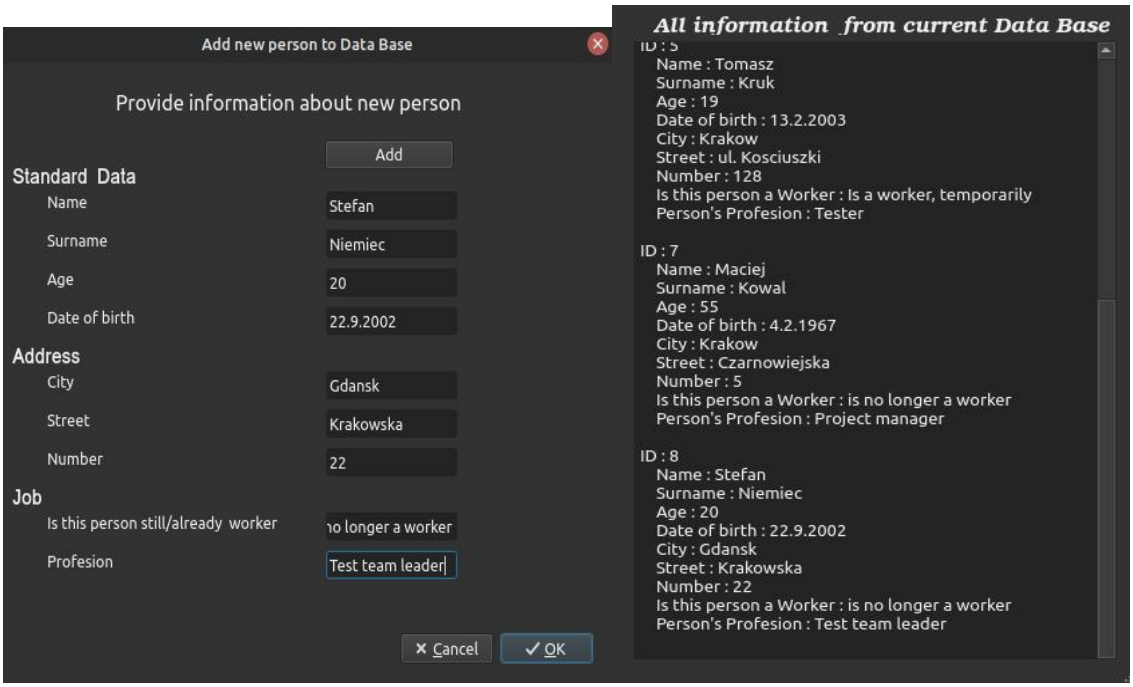




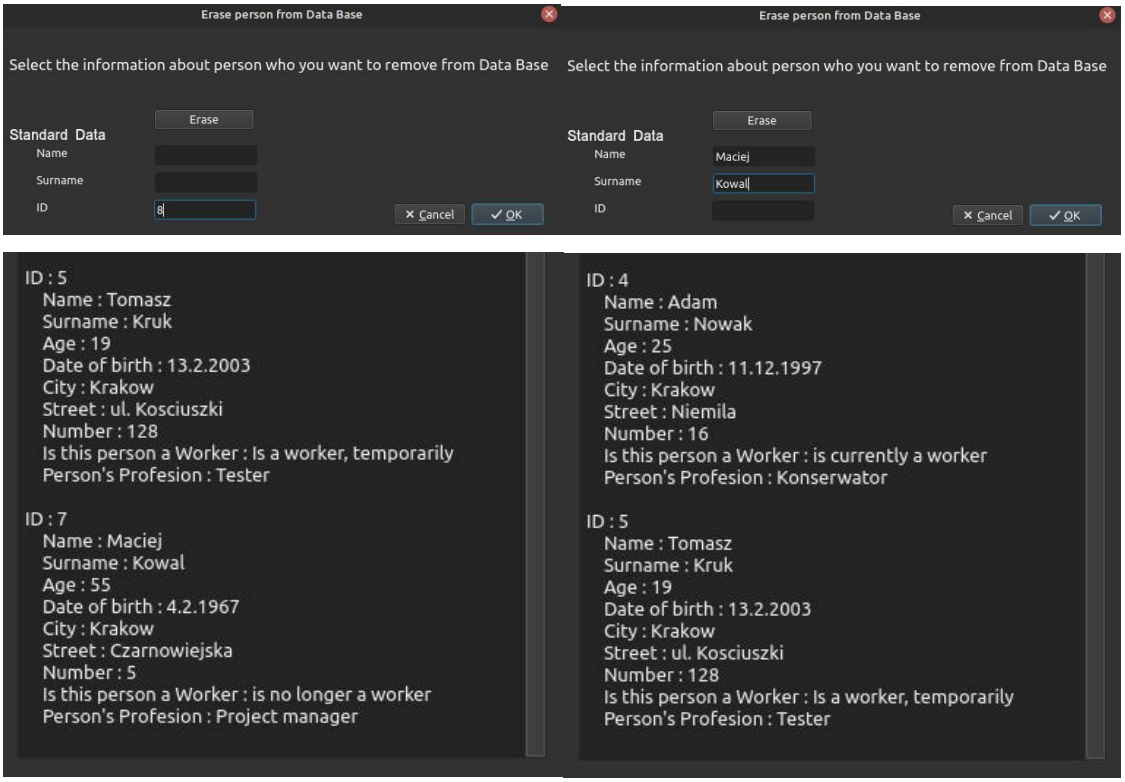
Powyższe zdjęcia pokazują poprawne fukcjonowanie funkcji Search na bazie odczytanej z formatu txt.



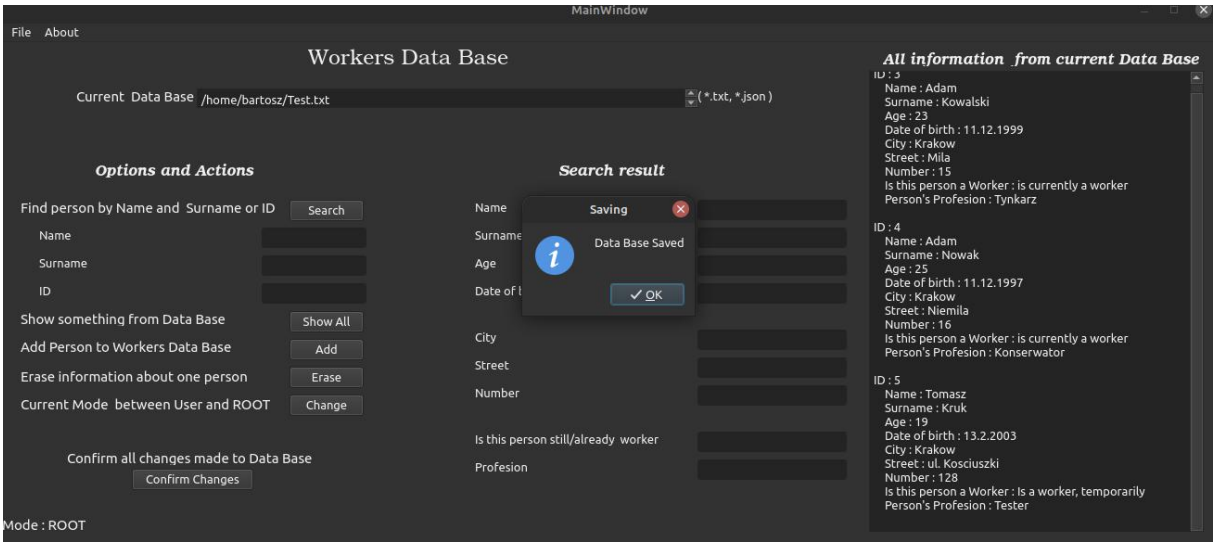
Na zdjęciach można zauważyć, że zmiana trybu w jakim pracujemy działa poprawnie, ponadto zmiana hasła przy podaniu poprzedniego hasła równierz działa poprawnie. Aplikacja jest w pełni przystosowana do wielekrotnego zmiany trybu pracy oraz hasła. Tryby odnoszą się do ograniczonych lub pełnych możliwości korzystania z aplikacji , dodawanie oraz usuwanie informacji o pracownikach jest dostępne tylko w trybie ROOT.



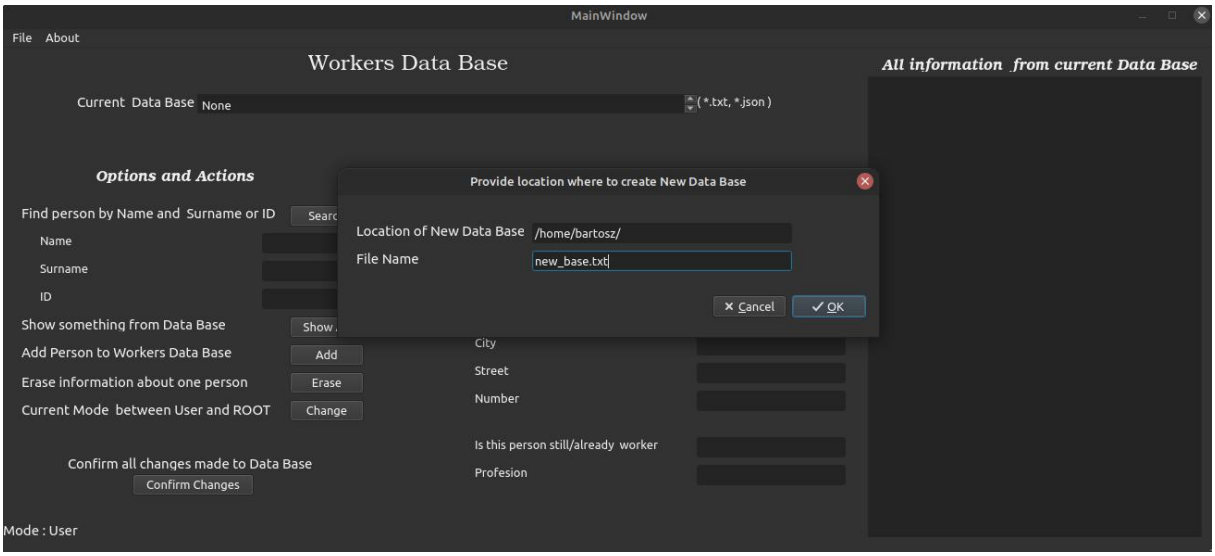
Powyżej pokazano przykładowe dodanie pracownika do testowej bazy danych. Dodawanie funkcjonuje w pełni poprawnie.

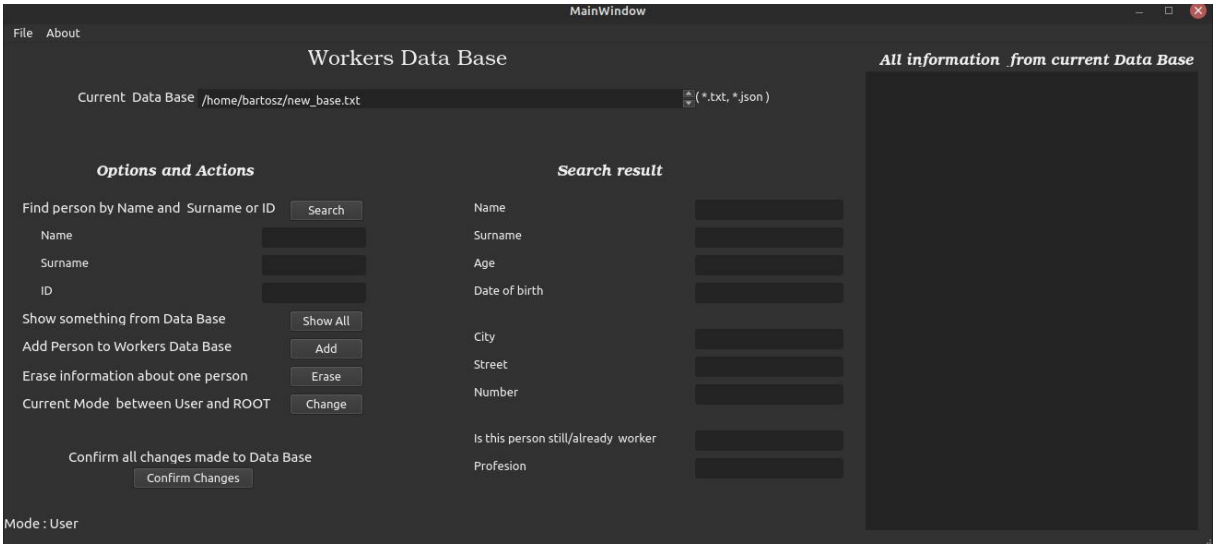


Następnie pokazano sprawdzono poprawność funkcji usuwania osób z bazy danych przy podawaniu numeru ID lub imienia i nazwiska , obie metody działają poprawnie.



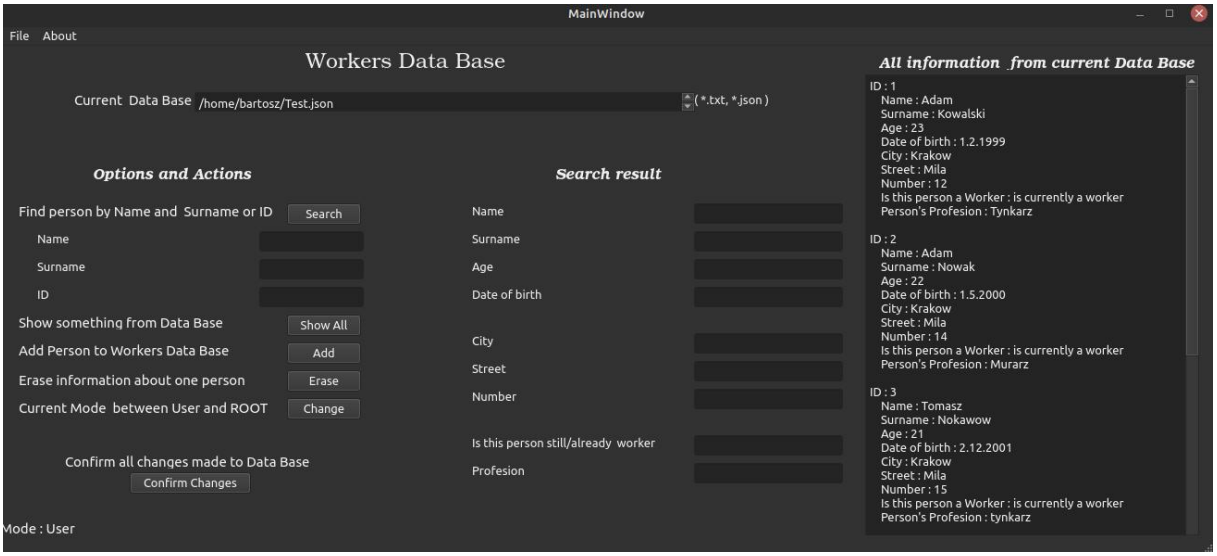
Kolejną sprawdzaną funkcją jest zapisywanie bazy danych i funkcja ta działa poprawnie.



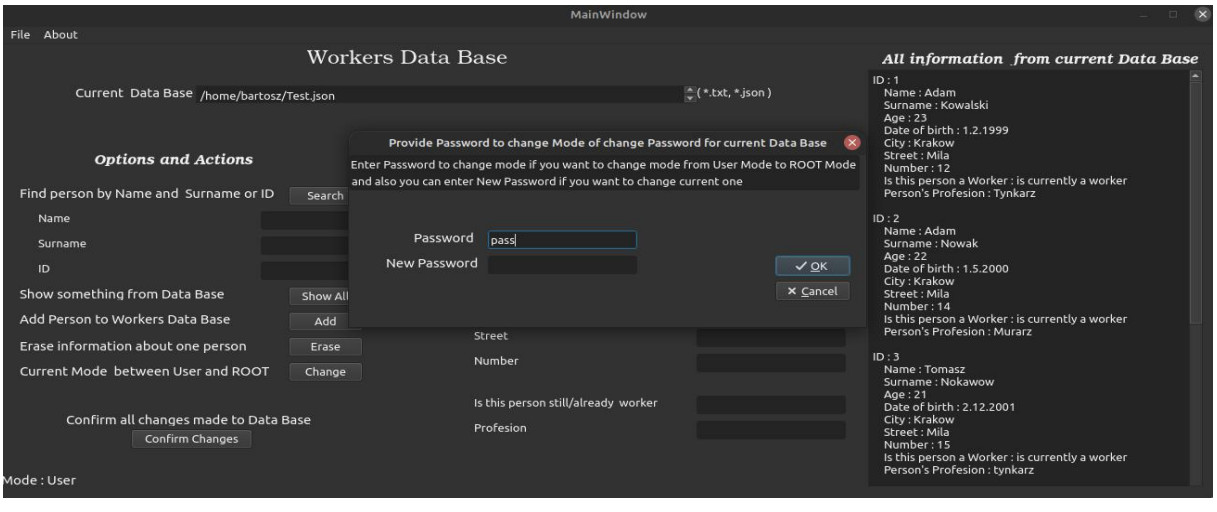


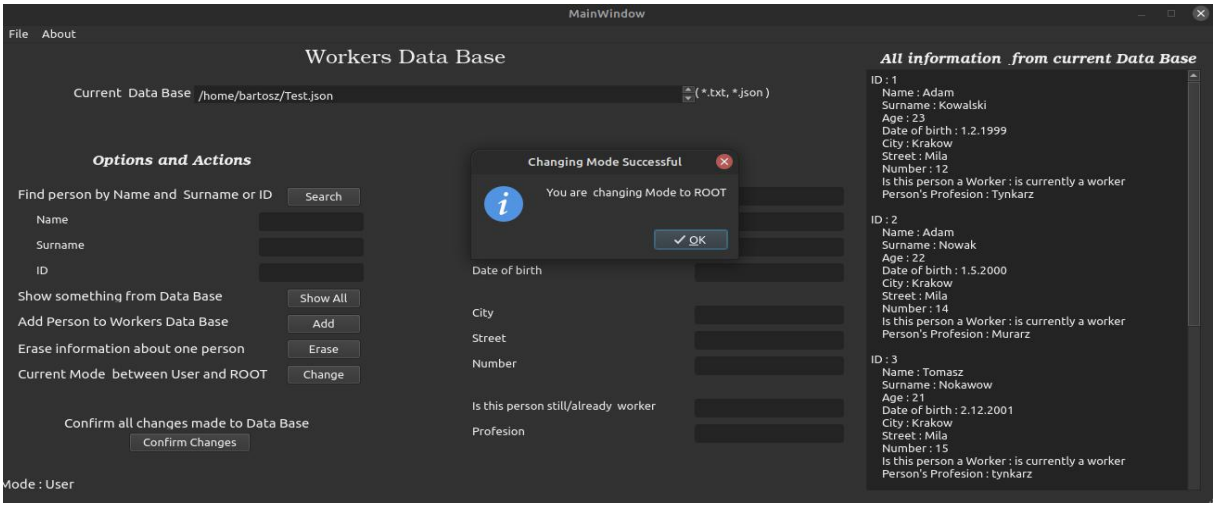
Ostatnią sprawdzaną funkcjonalnością jest tworzenie nowej bazy danych. Tworzenie nowej bazy w formacie txt działa poprawnie jednak wymagane jest odświerzenie/zatwierdzenie wykonanej operacji przez przycisk Confirm Changes aby aktualna lokalizacja została wyświetlona.

Test2 - Test bazy json

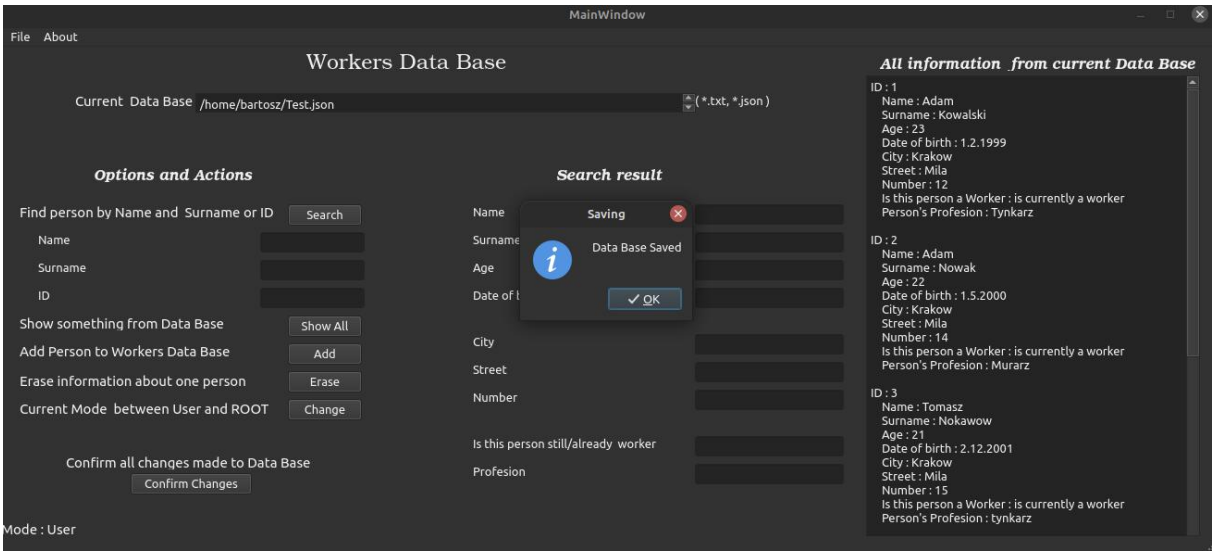


W przypadku bazy danych w formacie json fukcja odpowiadająca za otwieranie oraz wyświtlanie bazy równierz działa poprawnie. Wyświetlana lokalizacja pliku jest poprawna.





Powyższe zdjęcia pokazują, że hasło odczytane z bazy danych w formacie json również jest odczytywane poprawnie.

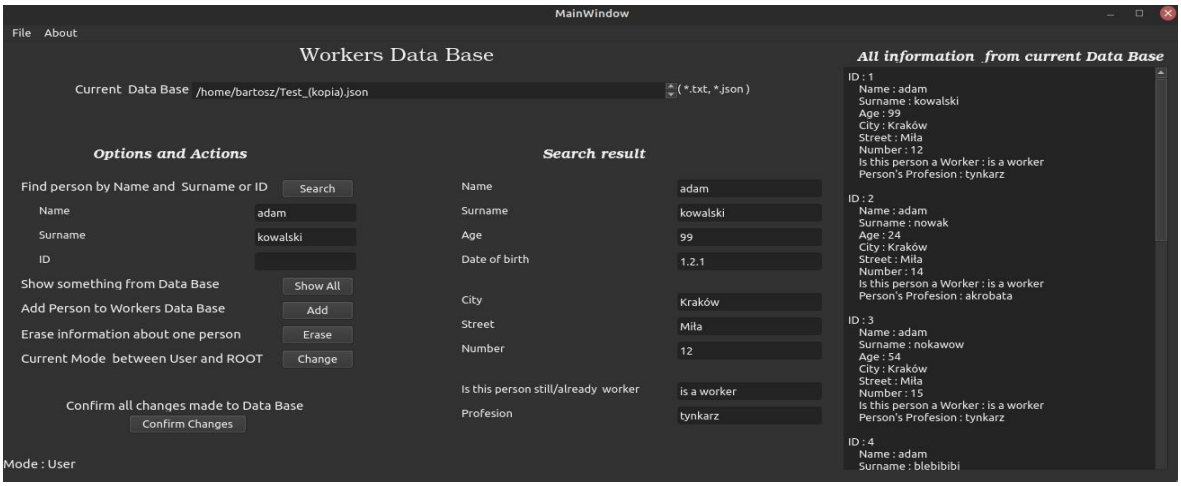


Ostatecznie zapisywanie bazy do pliku w formacie json również funkcjonuje poprawnie.

Pozostałe funkcje aplikacji operują już na wektorze obiektów klasy Person tak więc nie zależą od formatu pliku z jakiego zostały odczytane dane. Pozostałe funkcje działają tak samo więc funkcjonują poprawnie też dla pliku json.

6.Podręcznik użytkownika (*user's manual*)

Główne okno aplikacji



Rysunek 5-1. *Okno główne*

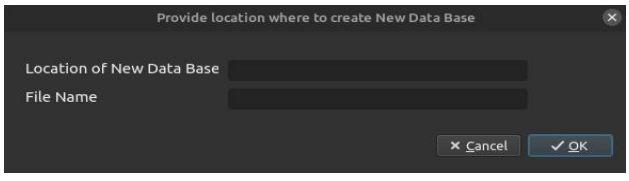
W powyższym oknie wykonywane są wszystkie operacje na bazie danych. Okno jest podzielone na 3 sekcje :

- Pierwsza sekcja “Options and Actions” zawiera przyciski odpowiadające funkcjom jakich możemu używać w aplikacji.
- Druga sekcja “Search result” to pole w którym wyświetlane są informacje na temat konkretnej osoby znalezione przez funkcję Search.
- Trzecia sekcja “All information from current Data Base” to miejsce w którym wyświetlane są wszystkie informacje z bazy danych. Wyświetlanie nastepuje po wciśnięciu przycisku Show All

W lewym dolnym rogu okna możliwe jest sprawdzenie aktualnego trybu pracy.

Current Data Base pokazuje ścieżkę do aktualnie otwartej bazy danych.

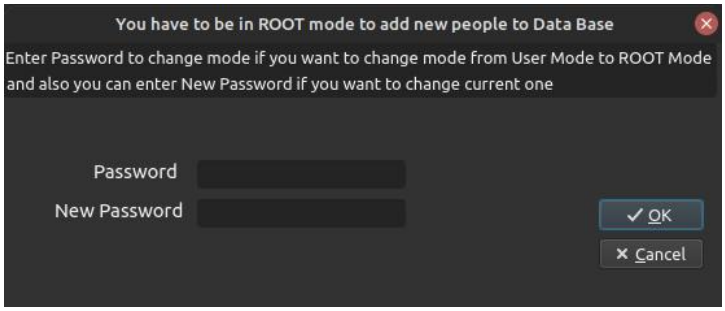
W zakładce File możliwe jest otworzenie, zapisanie oraz utworzenie bazy danych.



Rysunek 5-2. *Onko tworzenia nowej bazy*

W przypadku tworzenia nowej bazy danych należy podać nazwę nowej bazy wraz z rozszerzeniem oraz lokalizację w jakiej powinna zostać utworzona nowa baza.

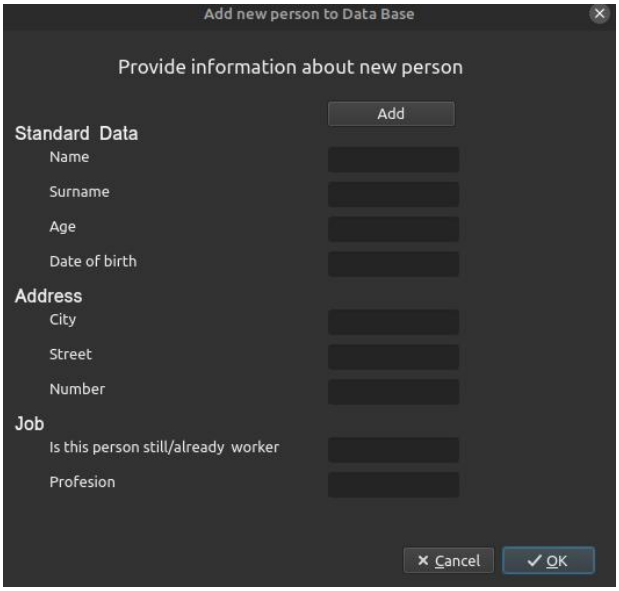
Przy próbie dodania / usunięcia informacji lub po wciśnięciu przycisku Change pojawi się okno w którym można zmienić aktualny tryb oraz jeśli zostanie coś wpisane do pola New Password dokonana zostanie zmiana hasła.



Rysunek 5-3. *Onko zmiany trybu i hasła*

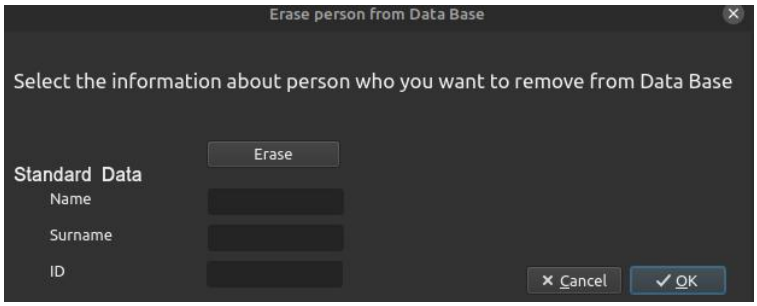
Po podaniu hasła pojawi się okno z komunikatem jaka operacja została ostatecznie wykonana.

Przy pracy w trybie ROOT możliwe jest dodawanie oraz usuwanie osób z bazy danych, poprzez podawanie danych w następujących oknach



Rysunek 5-4. Okno dodawania osób do bazy

Powyższe okno pojawia się po wciśnięciu przycisku Add, aby dodać osobę do bazy należy podać informacje na temat tej osoby w wyznaczonych polach oraz dodać przyciskiem Add.



Rysunek 5-5. *Okno usuwania osób z bazy*

Okno pojawia się po wciśnięciu przycisku Erase. W tym oknie należy podać imię i nazwisko lub numer ID osoby która ma zostać usunięta z bazy danych po czym zatwirdzić przyciskiem Erase.

Źródła oraz pomoce

[1] Cyganek B.: Introduction To Programing With C++ For Engineers.

[2] Grębosz J.: Symfonia C++ Standard.

[3] Qt Creator Manual - https://doc.qt.io/qtcreator/

[4] Qt Documentation - https://devdocs.io/qt/activeqt-server