WYŻSZA SZKOŁA
INFORMATYKI i ZARZĄDZANIA
z siedzibą w Rzeszowie

KOLEGIUM INFORMATYKI STOSOWANEJ

Kierunek: INFORMATYKA

Specjalność: -

Patryk Mikitka
Nr albumu studenta: w67163

Aplikacja do obsługi komisu samochodowego

Promotor:

PROJEKT PROGRAMOWANIE OBIEKTOWE

Rzeszów 2023

Spis treści

Rozdział 1

Opis projektu

1.1 Założenia projektu

Tematem projektu jest stworzenie aplikacji do zarządzania komisem samochodowym z wykorzystaniem języka C# w środowisku .NET oraz bazy danych SQL. Aplikacja będzie umożliwiać obsługę systemu ze strony sprzedawcy. Sprzedawca będzie mieć możliwość obsługi transakcji sprzedaży, dodawania oraz edycji ogłoszeń, jak również tworzenia raportów sprzedażowych czy finansowych. Z aplikacją będzie zintegrowana baza danych SQL, która będzie przechowywać wszelkie dane niezbędne do funkcjonowania komisu, m.in. dane klientów, pracowników, pojazdów, kontakty służbowe itd.

1.2 Wymagania funkcjonalne

1.2.1 Zarządzanie pojazdami w komisie

Pracownicy komisu będą mogli dodawać, edytować oraz usuwać wpisy dotyczące pojazdów będących na stanie komisu, w tym ich dane oraz historię przeglądów czy dokonanych napraw, aby baza danych była w jak największym stopniu aktualna.

1.2.2 Obsługa transakcji

System będzie umożliwiać obsługę transakcji sprzedażowych, takich sprzedaż, rezerwacje i wymiany samochodów. Będzie również zbierać informacje o warunkach umowy, cenach, datach transakcji i danych klienta. Dodatkowo, będzie możliwe automatyczne generowanie faktur dla dokonanych transakcji.

1.2.3 Generowanie raportów

Aplikacja umożliwi użytkownikom generowanie raportów, na przykład na temat przychodów czy najczęściej kupowanych modeli.

1.2.4 Zarządzanie klientami

System będzie przechowywać rozmaite dane na temat klientów. Dla każdego klienta będzie można przechowywać dane kontaktowe, historię transakcji czy też preferencje dotyczące marek samochodów.

1.2.5 Logowanie oraz uprawnienia użytkowników

Program będzie obsługiwać system logowania, jak również przydzielania użytkownikom uprawnień, aby uniknąć sytuacji, w której nieuprawniona osoba uzyskuje dostęp do poufnych danych.

1.3 Wymagania niefunkcjonalne

1.3.1 Wydajność

Aplikacja ma być responsywna i zapewniać szybki dostęp do danych nawet przy dużej ilości rekordów. Czas odpowiedzi interfejsu użytkownika powinien być minimalny, nawet podczas równoczesnej obsługi wielu użytkowników.

1.3.2 Bezpieczeństwo

System będzie zapewniać bezpieczne przechowywanie danych klientów, transakcji i informacji o samochodach w celu ich ochrony oraz zapobiegania ich wycieknięciu do sieci.

1.3.3 Interfejs użytkownika

Aplikacja będzie zaprojektowana w taki sposób, aby interfejs użytkownika był jak najbardziej czytelny, przejrzysty oraz prosty w obsłudze.

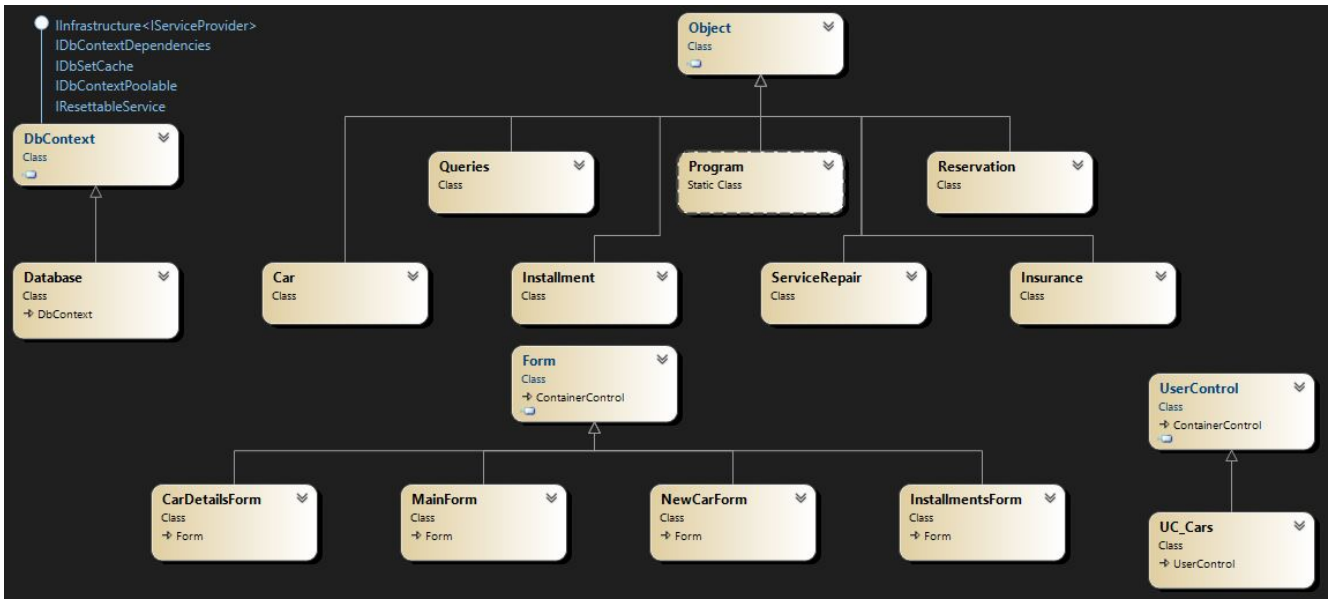
1.3.4 Skalowalność

System ma zapewniać możliwość skalowalności, aby obsłużyć rosnącą ilość danych i użytkowników w przyszłości.

Rozdział 2

Opis struktury projektu

2.1 Diagram klas



Rysunek 2.1: Diagram klas aplikacji

Rysunek powyżej przedstawia diagram klas aplikacji. Składa się on z czterech głównych sekcji, a największą i najbardziej liczną z nich jest widoczna u góry grafiki grupa klas dziedzicząca z klasy Object. Są to wszystkie klasy wykorzystane do połączenia z tabelami bazy danych, gdzie każda z nich odpowiada jednej tabeli w bazie (klasa Car to odpowiednik tabeli dbo.Cars itd.), jak również klasa Queries, przechowująca metody obsługujące zapytania do bazy danych.

Poniżej można zauważyć cztery klasy dziedziczące z klasy Form. Są to formularze WinForms, w których są prezentowane informacje z bazy danych. Do tej grupy można zaliczyć również klasę UC_Cars, która jest częścią formularza MainForm.

Ostatnią klasą jest dziedzicząca z typu DbContext klasa Database, której zadaniem jest obsługa połączenia z bazą danych.

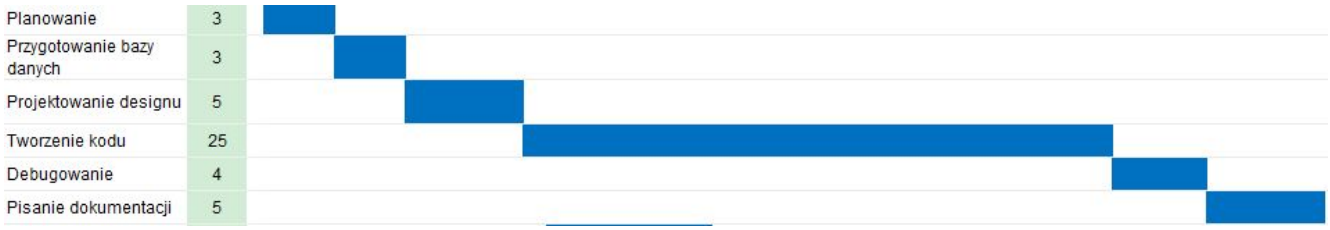
2.2 Szczegóły techniczne

Aplikacja "Car Dealership"została napisana w języku C# w wersji 12.0, z wykorzystaniem środowiska .NET 8.0, technologii WinForms oraz frameworku Entity Framework Core w wersji 8.0.1. Do obsługi plików CSV został także użyty pakiet CsvHelper w wersji 31.0.0. Do modyfikowania bazy danych SQL w trakcie tworzenia aplikacji wykorzystywano program SQL Server Management Studio w wersji 19.2.

2.3 Wymagania sprzętowe

- System Windows w wersji 10 lub 11
- Monitor o rozdzielczości minimum 1920x1080
- Minimum 50 MB wolnego miejsca na dysku

2.4 Diagram Gantta



Rysunek 2.2: Diagram Gantta procesu tworzenia aplikacji "Car Dealership"

2.5 Repozytorium i system kontroli wersji

W trakcie tworzenia aplikacji wykorzystywany był system kontroli wersji Git. Repozytorium projektu można znaleźć pod adresem: https://github.com/Patryk-MM/CarDealership.

Rozdział 3

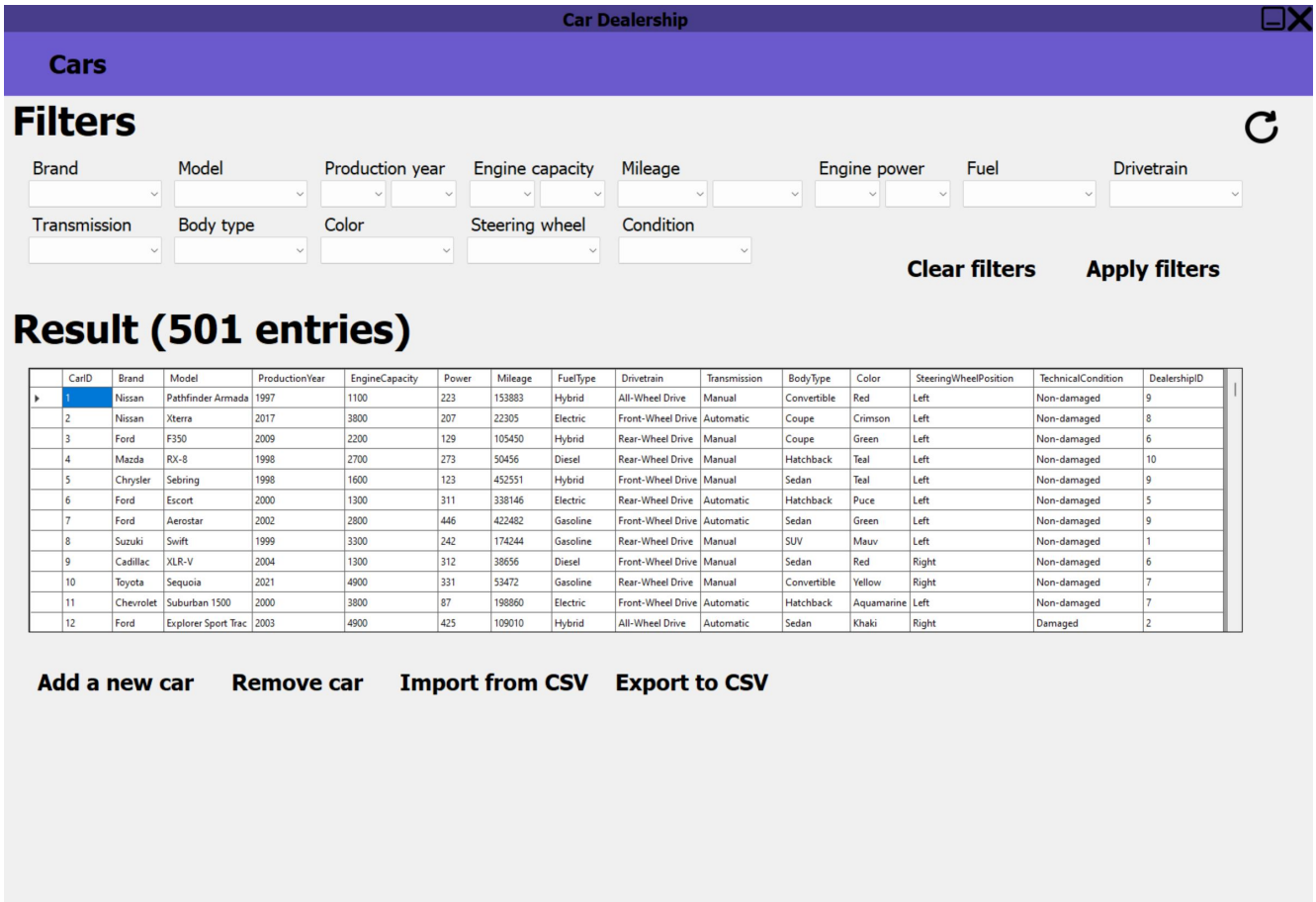
Warstwa użytkowa aplikacji

3.1 Okno główne

Po uruchomieniu aplikacji oczom użytkownika ukazuje się główne okno aplikacji. Składa się ono z trzech sekcji:

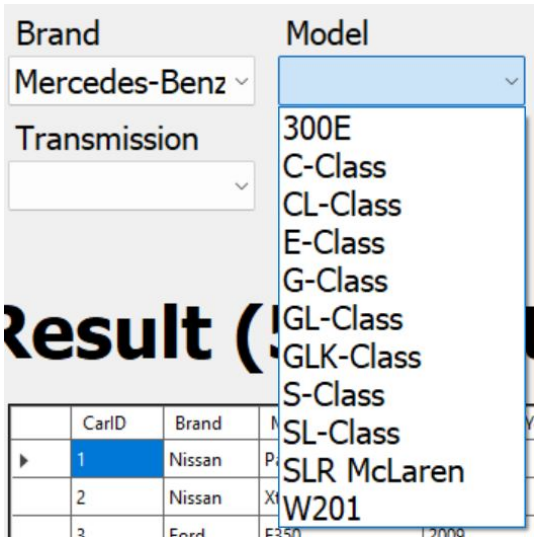
- sekcja **Filters** - odpowiada ona za definiowanie filtrów wyświetlania rekordów z bazy danych
- sekcja **Records** - tutaj wyświetlane są rekordy z bazy danych
- sekcja przycisków akcji poniżej tabeli wyników - zawiera ona przyciski odpowiadające za dodawanie oraz usuwanie rekordów z bazy danych, jak również za import oraz eksport do pliku CSV.

Po uruchomieniu aplikacja automatycznie wyświetla wszystkie pojazdy, które znajdują się w bazie danych, jak również aktualizuje ich ilość w etykiecie **Result**.



Rysunek 3.1: Główne okno aplikacji Car Dealership

Aby zdefiniować filtr, należy wybrać odpowiednie opcje z rozwijanych list w sekcji **Filters**, a następnie kliknąć przycisk **Apply filters**. W tabeli zostaną wyświetlone tylko i wyłącznie pojazdy spełniające podane kryteria. Opcje możliwe do wyboru w rozwijanych listach są pobierane z bazy danych, a zatem będą się automatycznie aktualizować w przypadku dodania do bazy pojazdu o nowych, niewystępujących wcześniej parametrach. Po wybraniu marki pojazdu na liście **Brand**, lista **Model** wyświetli jedynie modele pojazdów danej marki, co widać na rysunku poniżej:



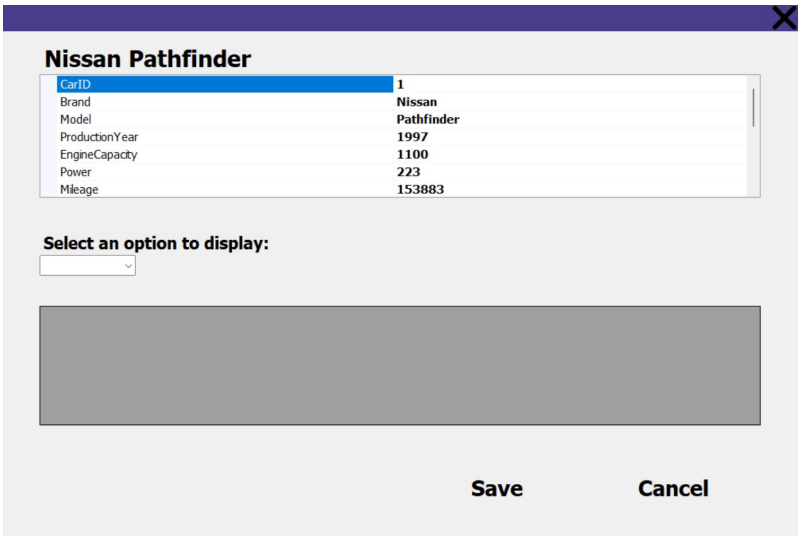
Rysunek 3.2: Opcje dostępne na liście Model po wybraniu konkretnej marki

Parametry liczbowe, czyli rok produkcji, pojemność i silnika oraz przebieg zostały zabezpieczone na wypadek, gdy wartość początkowa wybrana przez użytkownika będzie większa, niż wartość końcowa. Aby uniknąć błędów, oba parametry zostaną w takiej sytuacji zamienione miejscami.

Ostatnimi częścią sekcji **Filters** do omówienia są przycisk **Clear filters**, który służy do czyszczenia filtrów i wyświetlenia początkowego stanu tabeli rekordów, oraz przycisk odświeżenia widoku, umiejscowiony w górnym prawym rogu okna. On z kolei odświeża widok tabeli bez resetowania tabeli.

3.2 Formularz Car Details

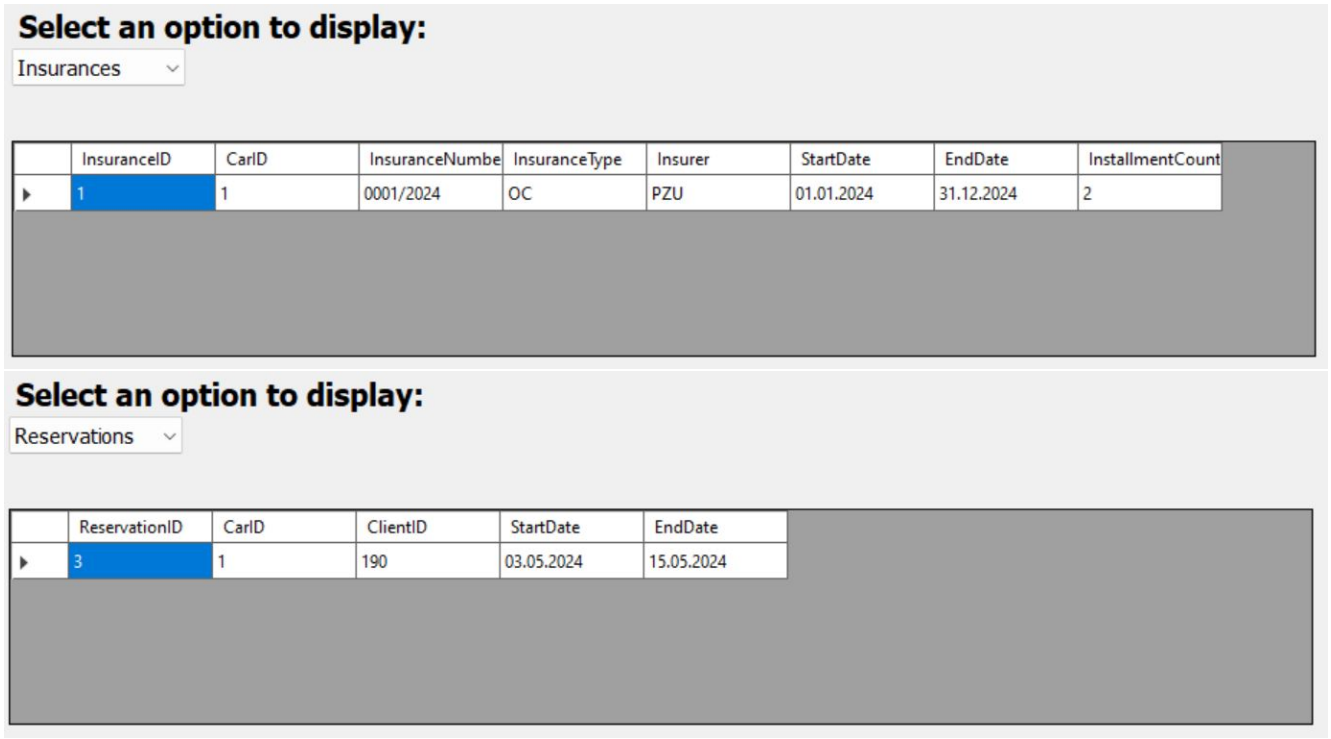
Po dwukrotnym kliknięciu w którąkolwiek z komórek wiersza danego samochodu wyświetli się formularz zawierający więcej szczegółów na jego temat. Umożliwia on również ich modyfikację.



Rysunek 3.3: Formularz Car Details ze szczegółami danego pojazdu

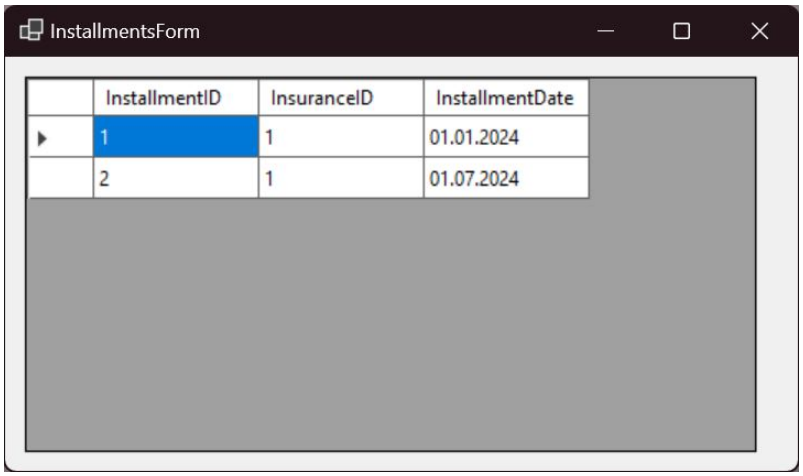
Składa się on z kontenera PropertyInfo u góry okna, kontenera DataGridView u dołu, rozwijanej listy służącej do wyboru opcji, oraz z dwóch przycisków: **Save** oraz **Cancel**. Kontener PropertyInfo zawiera wszystkie informacje na temat danego pojazdu, umożliwia również ich modyfikację - w tym celu należy zmienić wybraną wartość (lub wartości), a następnie nacisnąć przycisk **Save**. Wszelkie zmiany dokonane w kontenerze zostaną wtedy zapisane do bazy danych.

Dolna sekcja formularza odpowiada za wyświetlanie informacji dotyczących pojazdu znajdujących się w innych tabelach. Po wybraniu którejś z opcji z rozwijanej listy - jeśli takie dane są dostępne w bazie - zostaną wyświetlone w kontenerze DataGridView poniżej. W przypadku, jeśli takich danych w bazie nie ma, użytkownik zostanie o tym poinformowany stosownym komunikatem.



Rysunek 3.4: Widok opcji w formularzu "Car Details"

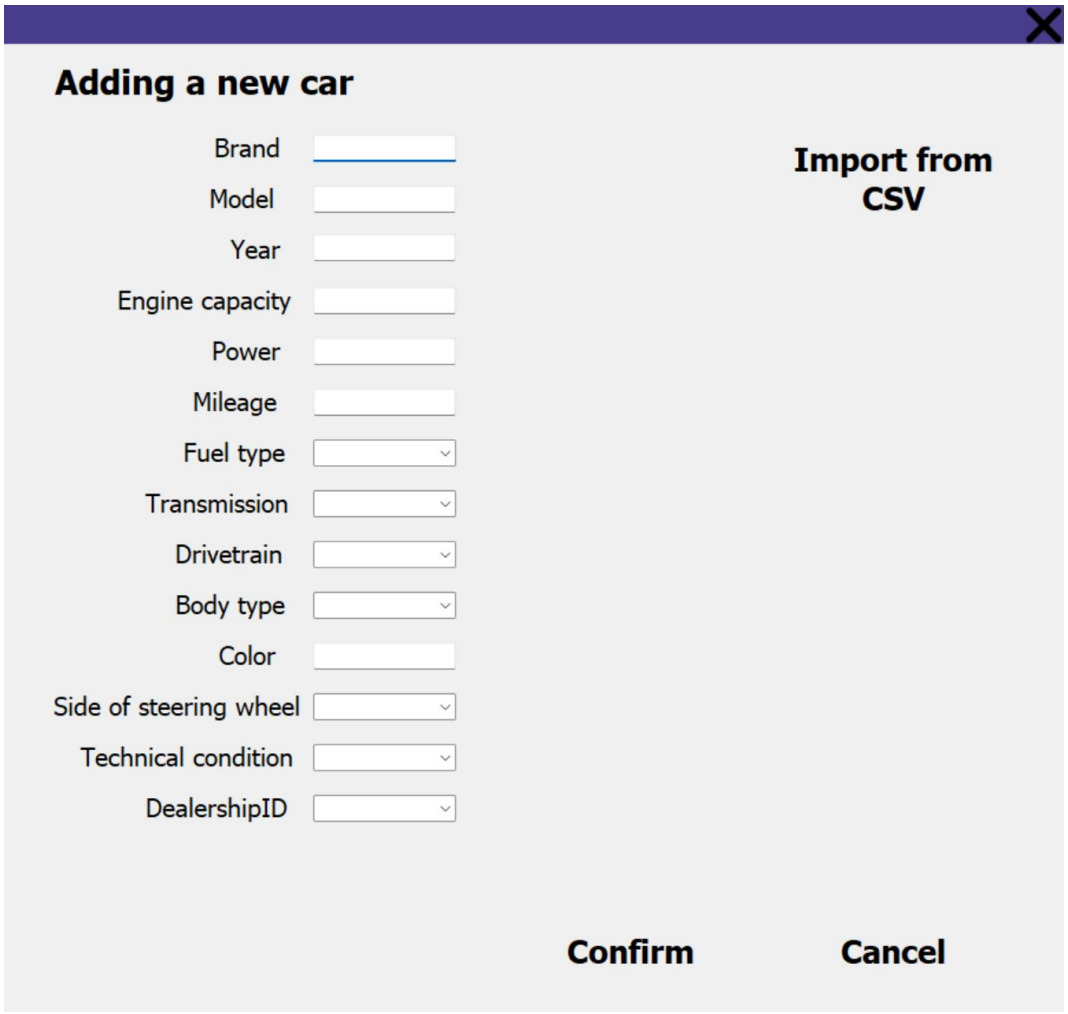
W przypadku opcji Insurances została zaimplementowana dodatkowa funkcjonalność - po kliknięciu w komórkę z ilością rat dla konkretnego ubezpieczenia otworzy się dodatkowe okno ze informacjami na temat dat płatności dla każdej z rat.



Rysunek 3.5: Okno szczegółów ubezpieczenia

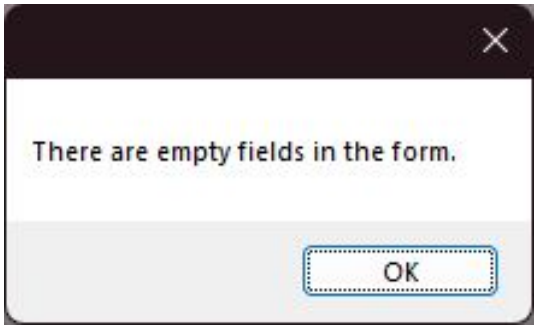
3.3 Formularz dodawania nowego pojazdu

Po kliknięciu w głównym menu przycisku **Add a new car**, przed oczami użytkownika ukaże się nowy formularz, służący do dodawania nowego pojazdu do bazy danych.



Rysunek 3.6: Formularz dodawania nowego pojazdu do bazy danych

Aby dodać pojazd do bazy danych, należy wypełnić wszystkie pola formularza i kliknąć przycisk **Confirm**. W przypadku, gdy któreś z pól pozostanie puste, program poinformuje o tym użytkownika komunikatem błędu. Ponadto, program w polach liczbowych nie pozwala na wprowadzanie liter.

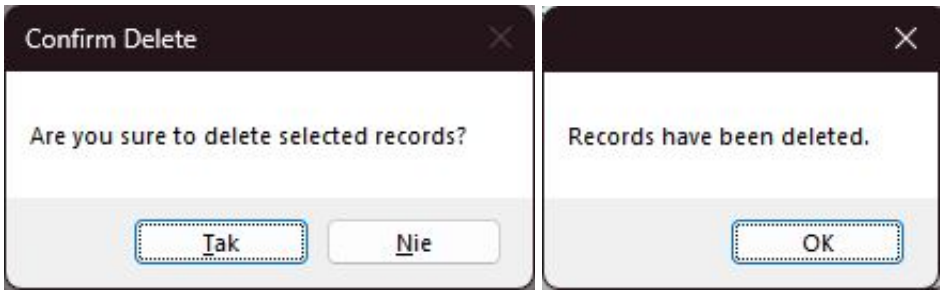


Rysunek 3.7: Komunikat informujący o pustym polu w formularzu

Można również skorzystać z opcji **Import from CSV**, wtedy program automatycznie uzupełni wartości pól z odpowiednio przygotowanego pliku CSV. Po kliknięciu przycisku **Confirm**, formularz dodawania pojazdu się zamknie, a wtedy w głównym oknie programu należy ręcznie odświeżyć tabelę rekordów.

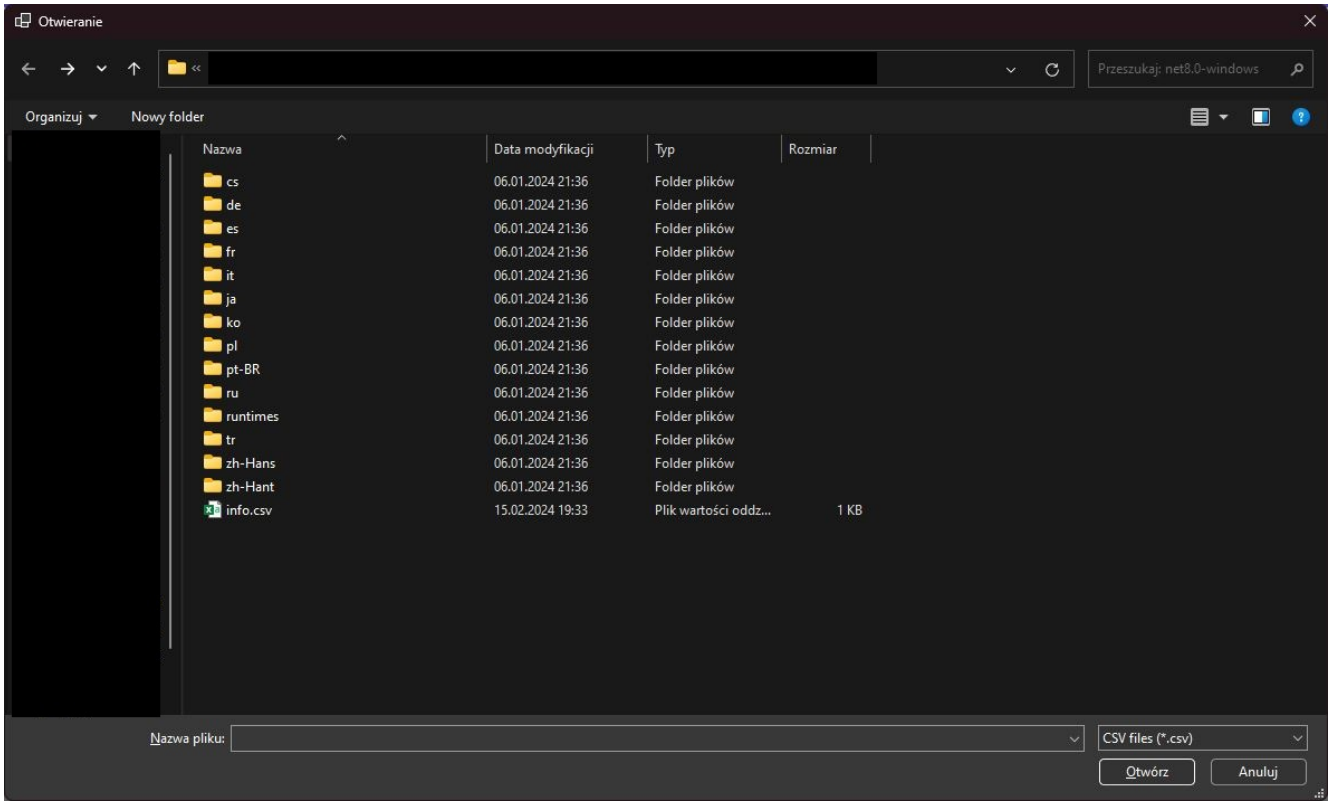
3.4 Pozostałe przyciski funkcyjne w menu głównym

W menu głównym można zauważyć jeszcze kilka nieomówionych wcześniej przycisków. Jednym z nich jest **Remove car**. Służy on, jak sama nazwa wskazuje, do usuwania rekordów z bazy danych. W tym celu należy zaznaczyć wiersz, który chcemy usunąć, klikając w komórkę w pierwszej, niepodpisanej kolumnie po lewej stronie tabeli (w celu usunięcia wielu wierszy, należy przytrzymać klawisz Ctrl i po kolei zaznaczać wiersze do usunięcia). Następnie należy kliknąć przycisk **Remove car**, a program zapyta, czy na pewno chcemy usunąć zaznaczone rekordy. Po potwierdzeniu komunikatu rekordy zostaną usunięte, a program wyświetli informację o powodzeniu operacji.



Rysunek 3.8: Komunikaty wyświetlające się w trakcie usuwania rekordu

Kolejnym przyciskiem jest **Import from CSV**, który różni się od tego występującego w formularzu dodawania tym, że pozwala na dodawanie wielu pojazdów na raz, ale nie umożliwia modyfikacji pól w trakcie importu. Po jego kliknięciu wyświetli się okno wyboru pliku, w którym należy wskazać plik CSV z rekordami.



Rysunek 3.9: Okno wyboru pliku CSV

Po otworzeniu i udanym przeanalizowaniu zawartości pliku CSV program zapyta, czy użytkownik na pewno chce zaimportować daną ilość rekordów. Po potwierdzeniu dane zostaną zaimportowane, wyświetli się informacja dla użytkownika, a tabela rekordów w menu głównym automatycznie się odświeży.

Ostatnim elementem w menu głównym do omówienia jest przycisk **Export to CSV**. Jego zadaniem jest eksport zaznaczonych wierszy do pliku CSV. Proces przebiega bardzo podobnie, jak w poprzednich przykładach. Najpierw należy zaznaczyć rekordy do eksportu, kliknąć przycisk, potwierdzić zamiar wyeksportowania danych, a następnie wybrać plik docelowy. Po udanym eksporcie użytkownik zostanie poinformowany o powodzeniu operacji.

Rozdział 4

Podsumowanie

4.1 Dalsze plany na rozwój aplikacji

Dalsze etapy rozwoju aplikacji Car Dealership obejmują dwa kluczowe obszary: interakcji z klientami i zarządzaniu pracownikami. Dodatkowo, w planach jest rozbudowanie opcji manipulacji danymi oraz umożliwienie użytkownikom bardziej elastycznego wyszukiwanie pojazdów poprzez ręczne wprowadzanie danych w filtry. Te kroki są niezbędne, aby zapewnić użytkownikom wygodną i intuicyjną obsługę aplikacji, a także zwiększyć jej funkcjonalność i przydatność w codziennej pracy.

4.2 Literatura

Jacek Matulewski, C#: lekcje programowania: praktyczna nauka programowania dla platform .NET i .NET Core, Helion, Gliwice 2021 lub nowsze

Mark J. Price, C# 10 i .NET 6 dla programistów aplikacji wieloplatformowych, Helion, 2023 lub nowsze