Zadanie 1

Konrad Strzechowski, nr indeksu: 305891

Data: 20.05.2021r.

1. Opis rozwiązania

Zadaną aplikację stworzyłem w języku C#. W tym celu wykorzystałem technologię ASP.NET Core MVC, która pozwala na łatwe stworzenie aplikacji WWW z wykorzystaniem bazy danych. ASP.Net Core MVC to rozbudowane struktury do tworzenia aplikacji internetowych oraz interfejsów API opartych na wzorcu Model-View-Controller. Wzorzec architektury MVC dzieli aplikację na trzy główne składniki: modele, widoki i kontrolery.

- Model: reprezentuje stan aplikacji, operacje oraz logikę biznesową, która powinna być wykonywana.
- Widok: odpowiada za prezentowanie zawartości za pomocą interfejsu użytkownika. Widoki powinny mieć minimalną logikę, a każda powinna odnosić się do prezentowania danych.
- Kontroler: obsługuje interakcję z użytkownikiem. Jest początkowym punktem wejścia. Odpowiada za wybór modeli do pracy oraz odpowiedniego widoku do wyświetlania informacji.

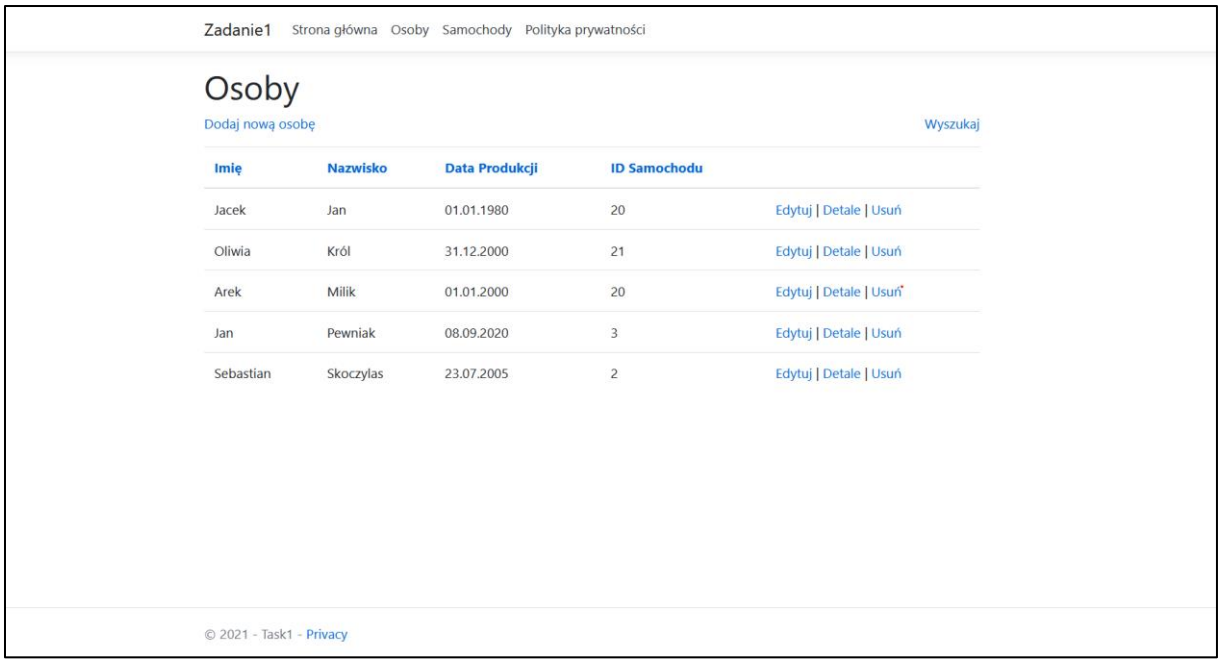
Na początku stworzyłem prostą stronę WWW na podstawie istniejącej juz bazy daych. W tym celu wykorzystałem komendę Scaffold-DbContext wywoływaną w menedżerze pakietów. W ten sposób automatycznie stworzyłem klasy modeli na podstawie tabel w wybranej bazie. Modele przedstawiają strukturę danych zamieszczanych w bazie. Następnie na ich podstawie możemy dodać odpowiednie kontrolery. Wybieramy kontrolery z widokiem, co przygotowuje nam całą strukturę do przechowywania oraz wyświetlania danych z bazy w aplikacji WWW. Automatycznie generowane są widoki do tworzenia nowych wierszy w tabeli, ich edytowania, pokazania detali oraz usuwania wierszy, razem z ich obsługą w kontrolerze . Ostatnim krokiem jest stworzenie odpowiedniej migracji (także za pomocą komendy), która umożliwi nam przesyłanie danych pomiędzy bazą, a aplikacją.

Dzięki wyżej opisanym krokom utworzyłem w pełni działającą aplikację WWW korzystającą z naszej bazy danych. Pozostało mi zmodyfikować ją do swoich celów. Dodałem niezbędną walidację, która sprawdza poprawność wpisywanych przez użytkownika danych, podczas tworzenia i edytowania wierszy. Dodałem możliwość sortowania tabeli osób oraz wyszukiwania osób po ich nazwisku lub dacie produkcji ich samochodu. Dodatkowo zmodyfikowałem interfejs użytkownika, tak aby był w całości w języku polskim.

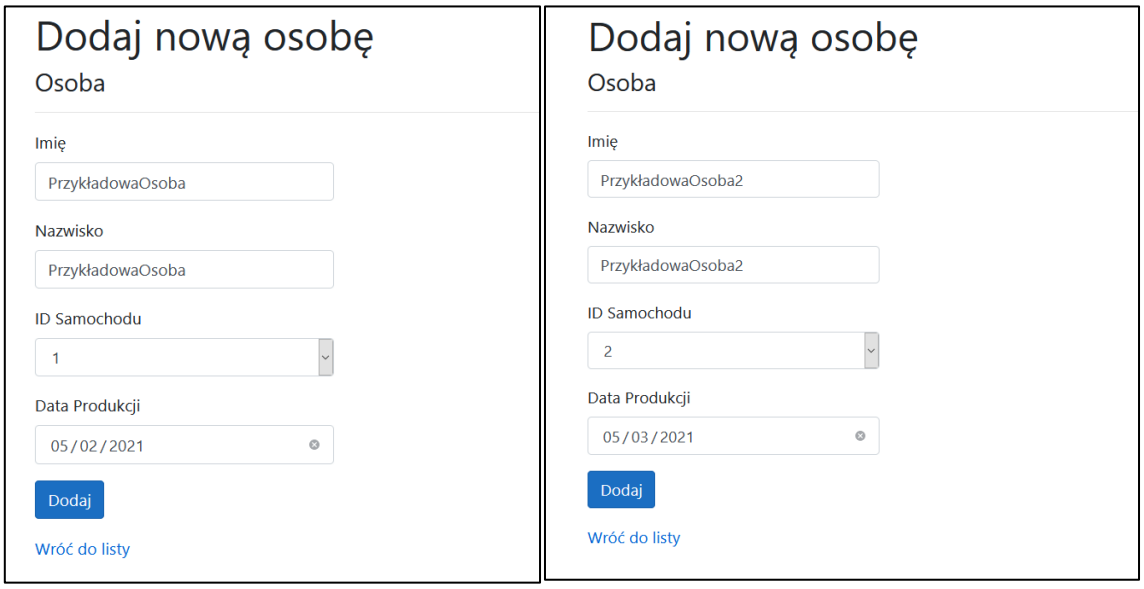
2. Wyniki testów symulacji

Wykonywanie operacji na tej samej osobie lub samochodzie przez kilka osób nie stanowi większych problemów dla naszego programu.

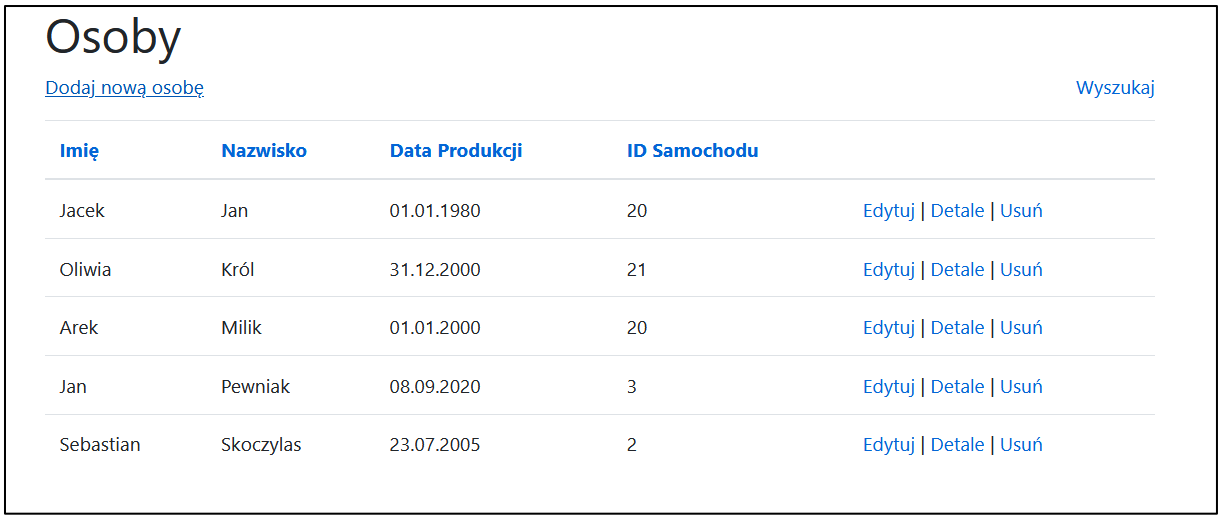
Mamy przykładowe dane w naszej bazie danych:

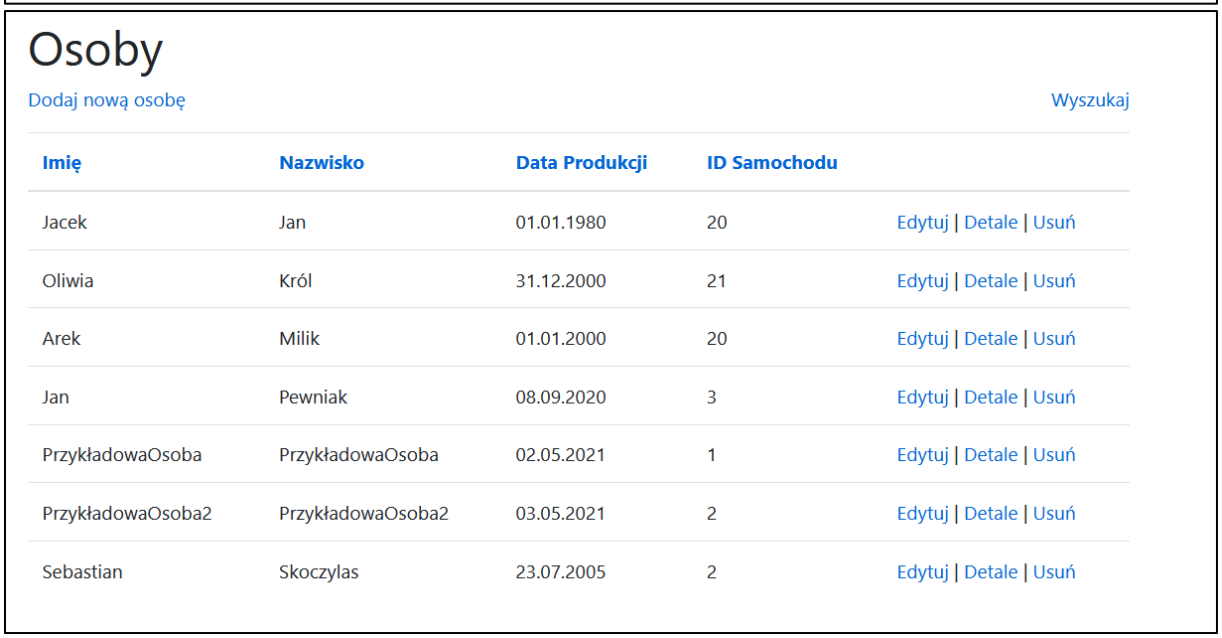


Otwieramy ją w kilku kartach i sprawdzamy co się dzieje, gdy na 2 z nich dodamy nowe osoby.



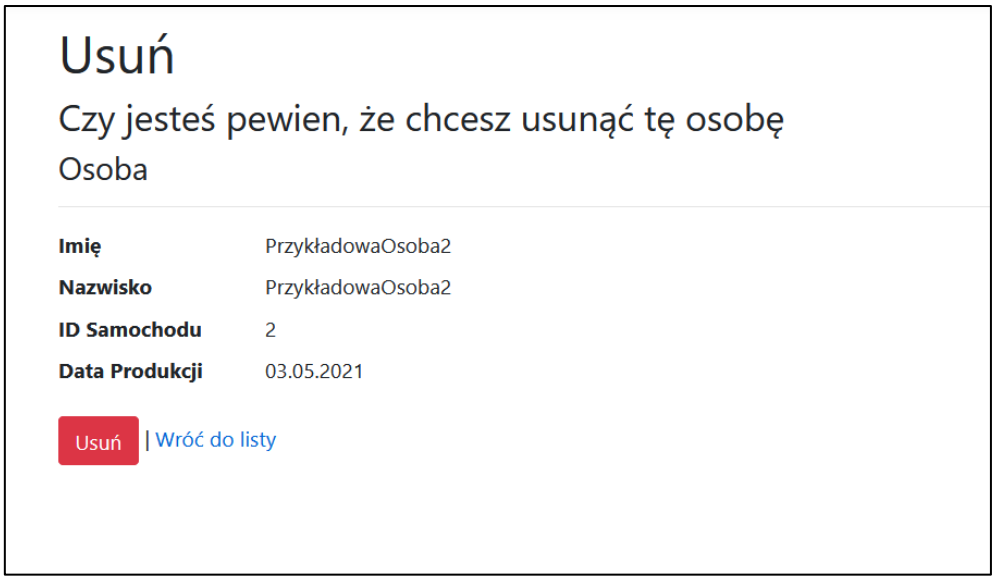
Na 1 stronie pojawiła się nowa osoba, na 2 pojawiły się obie. Na pozostałych nie pojawiły się od razu żadne osoby. Ale po odświeżeniu kart są już w naszej tabeli.





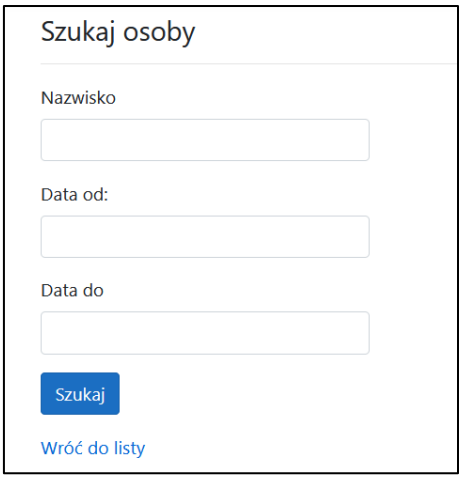
Dzieję się tak, ponieważ widok na naszej karcie nie odświeża się automatycznie przy zmianach zachodzących w bazie danych. Identyczna sytuacja zachodzi dla tabeli samochodów.

W sytuacji gdy próbujemy edytować lub wyświetlić detale osoby, która jest już usunięta lub usunąć ją drugi raz, wyświetala się komunikat, ponieważ dana osoba nie istnieje już w naszej bazie danych.



Co ciekawe po kliknięciu w przycisk „Wróć do listy” lub „Osoby” załaduje nam się poprawiona już lista osób, ale jeśli cofniemy do poprzedniej karty za pomocą narzędzia przeglądarki wyświetli nam się stara lista z widoczną, usuniętą osobą. Czyli cofniemy się do starego widoku, w którym osoba nie jest jeszcze usunięta.

Podczas tworzenia lub edytowania danych, aplikacja wymusza wpisanie obowiązkowych danych. Dodatkowo ID generowane jest losowo, tak aby było unikalne. W przypadku wyszukiwania błędne lub puste dane po prostu nic nie zwrócą.



Oczywiście sortowanie na jednej z kart nie ma wpływu na kolejność danych na pozostałych kartach (także po odświerżeniu strony).

Imię	Nazwisko	Data Produkcji	ID Samochodu	
Sebastian	Skoczylas	23.07.2005	2	Edytuj | Detale | Usuń
PrzykładowaOsoba	PrzykładowaOsoba	02.05.2021	1	Edytuj | Detale | Usuń
Jan	Pewniak	08.09.2020	3	Edytuj | Detale | Usuń
Arek	Milik	01.01.2000	20	Edytuj | Detale | Usuń
Oliwia	Król	31.12.2000	21	Edytuj | Detale | Usuń
Jacek	Jan	01.01.1980	20	Edytuj | Detale | Usuń

Imię	Nazwisko	Data Produkcji	ID Samochodu	
Oliwia	Król	31.12.2000	21	Edytuj | Detale | Usuń
Jacek	Jan	01.01.1980	20	Edytuj | Detale | Usuń
Arek	Milik	01.01.2000	20	Edytuj | Detale | Usuń
Jan	Pewniak	08.09.2020	3	Edytuj | Detale | Usuń
Sebastian	Skoczylas	23.07.2005	2	Edytuj | Detale | Usuń
PrzykładowaOsoba	PrzykładowaOsoba	02.05.2021	1	Edytuj | Detale | Usuń

3. Wykorzystanie transakcji

W tej aplikacji do zaimplementowania potrzebnych operacji skorzystałem z zapytań LINQ. Są one bardzo wygodne i zdecydowanie ułatwiają działania wykonywane na dużej ilości danych. Zamiast zapytań LINQ moglibyśmy korzystać ze standardowych komend wykonywanych w SQL poprzez nawiązywanie bezpośrednich połączeń z naszą bazą. W tym przypadku nie miało to jednak większego znaczenia. Przykładowe użycie sortujące dane:

```
Osoby = sortOrder switch
{
    "Imie" => Osoby.OrderBy(os => os.Imie),
    "Imie_desc" => Osoby.OrderByDescending(os => os.Imie),
    "Nazwisko" => Osoby.OrderBy(os => os.Nazwisko),
    "Nazwisko_desc" => Osoby.OrderByDescending(os => os.Nazwisko),
    "DataProd" => Osoby.OrderBy(os => os.DataProd),
    "DataProd_desc" => Osoby.OrderByDescending(os => os.DataProd),
    "SamochodId" => Osoby.OrderBy(os => os.SamochodId),
    "SamochodId_desc" => Osoby.OrderByDescending(os => os.SamochodId),
    _ => Osoby.OrderBy(os => os.Nazwisko),
};
return View(await Osoby.ToListAsync());
```

Dodatkowo podczas dodawania lub edytowania danych korzystamy z funkcji zapewnionych przez Microsoft.EntityFrameworkCore.dll:

```
if (ModelState.IsValid)
{
    _context.Add(samochod);
    await _context.SaveChangesAsync();
    return RedirectToAction(nameof(Index));
}
```

Dodatkowe uwagi:

Do prawidłowego działania programu potrzebowałem 3 pakietów NuGet:

- Microsoft.EntityFrameworkCore,
- Microsoft.EntityFrameworkCore.Sql,
- Microsoft.EntityFrameworkCore.Tools.

Podczas tworzenia aplikacji powstała także nowa tabela migracji w naszej bazie danych.

```
CREATE TABLE [dbo].[__EFMigrationsHistory] (
    [MigrationId]    NVARCHAR (150) NOT NULL,
    [ProductVersion] NVARCHAR (32)  NOT NULL,
    CONSTRAINT [PK___EFMigrationsHistory] PRIMARY KEY CLUSTERED ([MigrationId] ASC)
);
```

Oświadczenie

Oświadczam, że to zadanie jest w całości moją własną pracą, wykonaną samodzielnie i bez żadnej pomocy z niedozwolonych źródeł.
Konrad Strzechowski