

Aplikacja tworzona w ramach tego laboratorium ma za zadanie przybliżyć podstawy współpracy z bazą danych za pomocą EntityFramework. W zakres ćwiczeń wchodzi pobieranie danych z bazy, tworzenie nowych rekordów oraz edycja istniejących. Wyszukiwanie i filtrowanie powinno być realizowane po stronie bazy danych (LINQ-to-SQL), a nie po stronie aplikacji (LINQ-to-Objects).

Po stworzeniu aplikacji C# Windows Presentation Foundation (WPF) przechodzimy bezpośrednio do zainstalowania pakietu EntityFramework. Należy uruchomić Package Manager Console (wybrać z menu **(Tools > Nuget Package Manager > Package Manager Console)**). Następnie wpisać komendę Install-Package EntityFramework.

Następnym krokiem będzie utworzenie migracji. Na początek tworzymy klasy, których obiekty będą przetrzymywane w bazie danych.

Engine.cs

```
public class Engine
{
    public int Id { get; set; }
    public double Displacement { get; set; }
    public double HorsePower { get; set; }
    public String Model { get; set; }

    public Engine() { }
    public Engine(double displacement, double horsePower, string model)
    {
        this.Displacement = displacement;
        this.HorsePower = horsePower;
        this.Model = model;
    }

    public override string ToString()
    {
        return Model + " " + Displacement + " (" + HorsePower + " hp) ";
    }
}
```

Car.cs

```
public class Car
{
    public int Id { get; set; }
    public string Model { get; set; }
    public Engine Engine { get; set; }
    public int Year { get; set; }

    public Car() { }

    public Car(string model, Engine engine, int year)
    {
        Model = model;
        Engine = engine;
        Year = year;
    }
}
```

Następnie tworzymy kontekst, który będzie przedstawiał dane które zapisane będą w bazie danych.

```
public class MyDbContext : DbContext
{
    public DbSet<Car> Cars { get; set; }
    public DbSet<Engine> Engines { get; set; }
}
```

Aby zezwolić na tworzenie migracji należy wpisać w Package Manager Console następującą komendę: **Enable-Migrations**. Po tej czynności uzupełniamy plik Configuration.cs o dane początkowe, które zostaną dodane do bazy w trakcie jej uaktualniania.

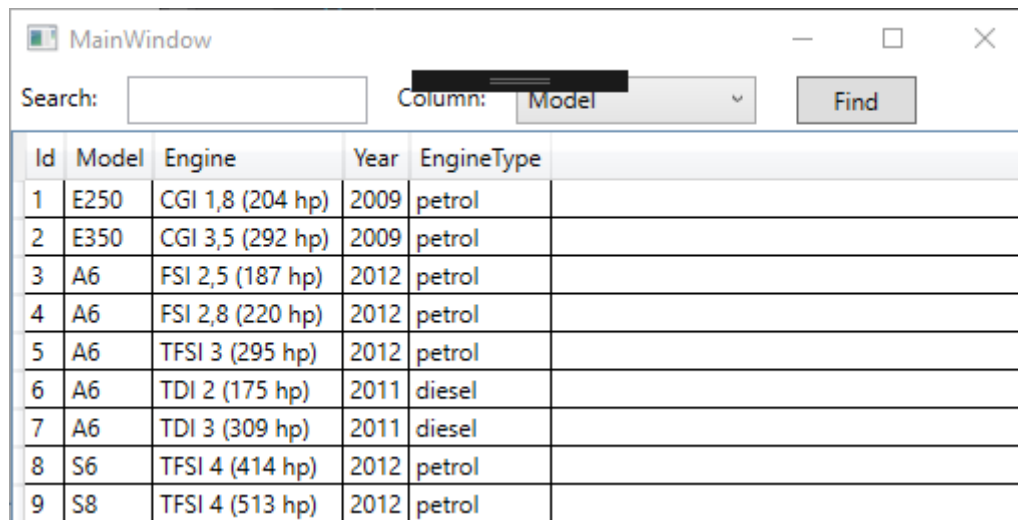
```
context.Cars.AddOrUpdate(
    new Car("E250", new Engine(1.8, 204, "CGI"), 2009),
    new Car("E350", new Engine(3.5, 292, "CGI"), 2009),
    new Car("A6", new Engine(2.5, 187, "FSI"), 2012),
    new Car("A6", new Engine(2.8, 220, "FSI"), 2012),
    new Car("A6", new Engine(3.0, 295, "TFSI"), 2012),
    new Car("A6", new Engine(2.0, 175, "TDI"), 2011),
    new Car("A6", new Engine(3.0, 309, "TDI"), 2011),
    new Car("S6", new Engine(4.0, 414, "TFSI"), 2012),
    new Car("S8", new Engine(4.0, 513, "TFSI"), 2012)
);
```

Przed stworzeniem/uaktualnieniem bazy konieczne jest dodanie migracji aby dane zapisane w klasach zostały uwzględnione w bazie, aby tego dokonać należy w Package Manager Console wpisać komendę: **Add-Migration nazwa**.

Następnie w celu stworzenia bazy uruchomić Package Manager Console i wpisać komendę **Update-Database**. Po tej operacji w bazie danych powinny być widoczne wstawione rekordy. Efekty tych czynności można zauważyć np. za pomocą programu Microsoft SQL Server Management Studio.

Zadania do wykonania:

**Wykonać wyświetlanie danych w elemencie DataGridView (1 pkt).**

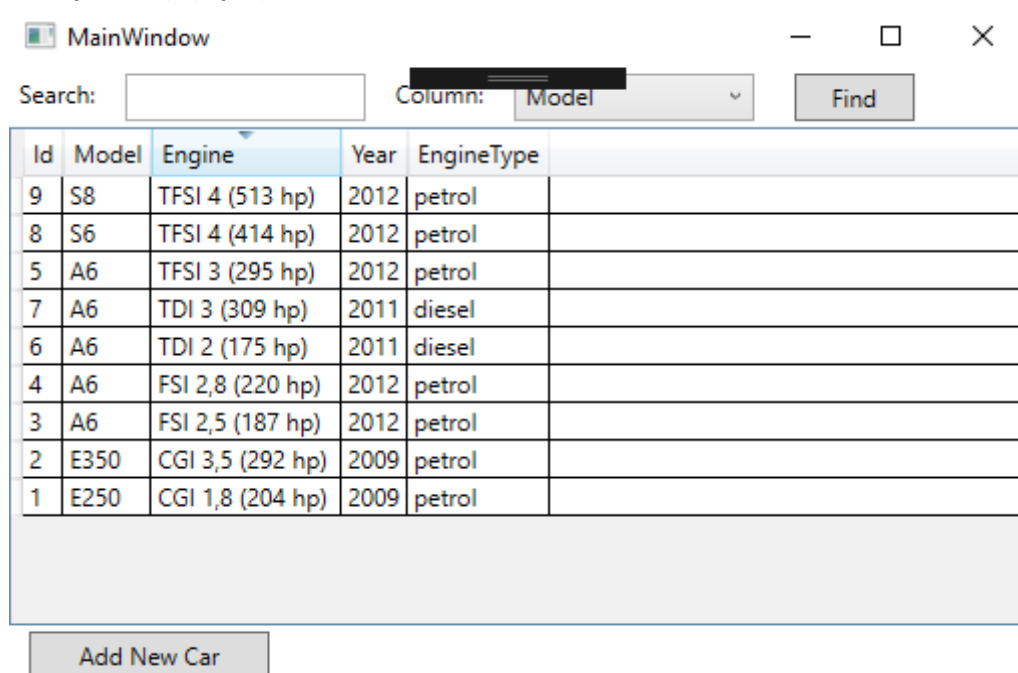


Id	Model	Engine	Year	EngineType
1	E250	CGI 1,8 (204 hp)	2009	petrol
2	E350	CGI 3,5 (292 hp)	2009	petrol
3	A6	FSI 2,5 (187 hp)	2012	petrol
4	A6	FSI 2,8 (220 hp)	2012	petrol
5	A6	TFSI 3 (295 hp)	2012	petrol
6	A6	TDI 2 (175 hp)	2011	diesel
7	A6	TDI 3 (309 hp)	2011	diesel
8	S6	TFSI 4 (414 hp)	2012	petrol
9	S8	TFSI 4 (513 hp)	2012	petrol

Pobieranie danych z bazy:

```
MyDbContext ctx = new MyDbContext();  
List<Car> carsList = ctx.Cars.Include("Engine").ToList();
```

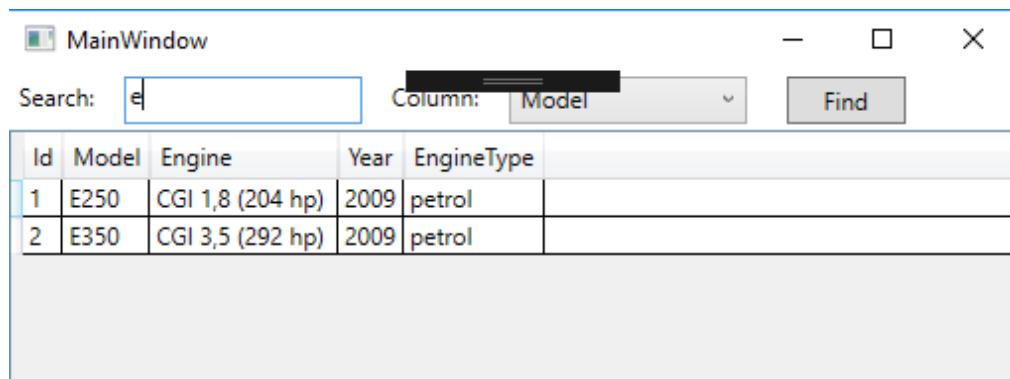
**Dodanie kolumny EngineType i umożliwienie sortowania za pomocą kolumny Engine (interfejs IComparable). (1 pkt)**



Id	Model	Engine	Year	EngineType
9	S8	TFSI 4 (513 hp)	2012	petrol
8	S6	TFSI 4 (414 hp)	2012	petrol
5	A6	TFSI 3 (295 hp)	2012	petrol
7	A6	TDI 3 (309 hp)	2011	diesel
6	A6	TDI 2 (175 hp)	2011	diesel
4	A6	FSI 2,8 (220 hp)	2012	petrol
3	A6	FSI 2,5 (187 hp)	2012	petrol
2	E350	CGI 3,5 (292 hp)	2009	petrol
1	E250	CGI 1,8 (204 hp)	2009	petrol

Add New Car

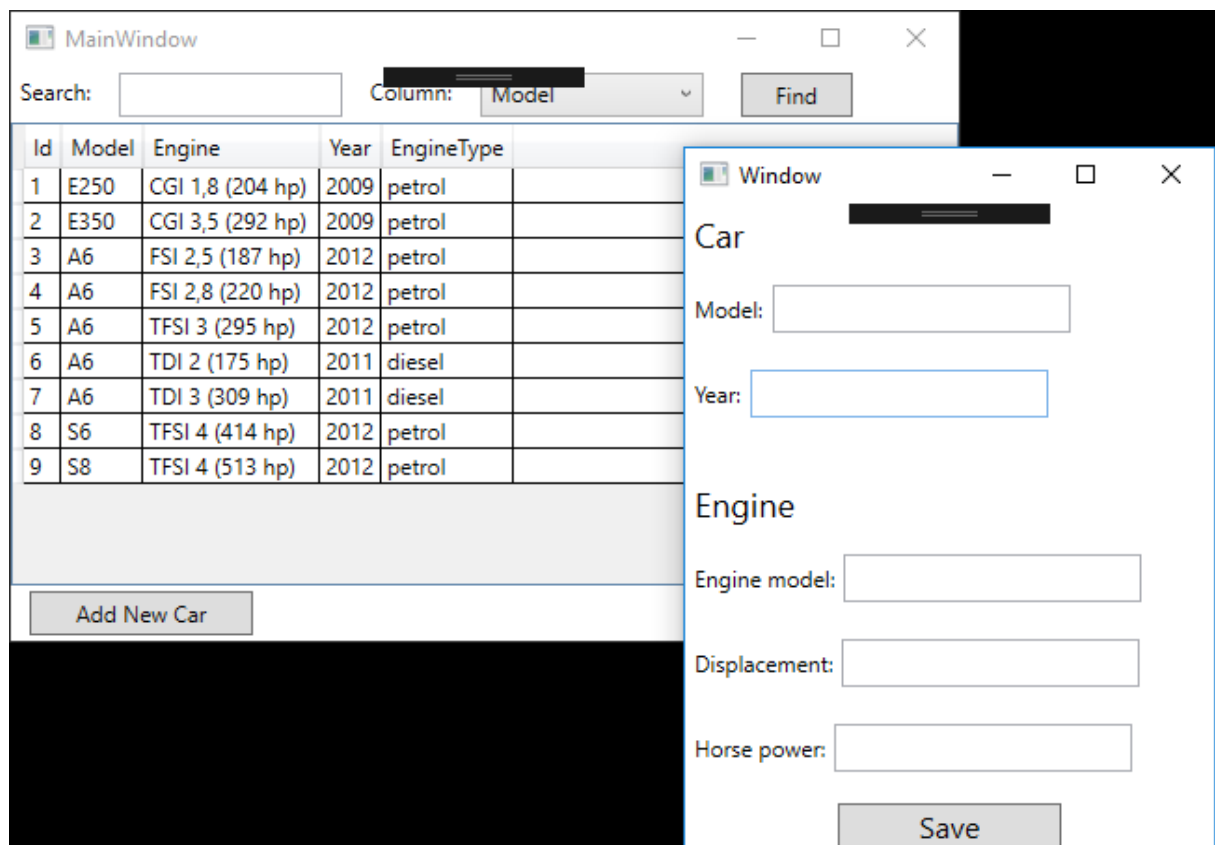
Wykonać wyszukiwanie odpowiadających wierszy dla kolumn Model i Year (1 pkt).



Id	Model	Engine	Year	EngineType	
1	E250	CGI 1,8 (204 hp)	2009	petrol	
2	E350	CGI 3,5 (292 hp)	2009	petrol	

Wyszukiwanie dla Year dla dokładnej wartości, dla Model użyć funkcji .Contains(). Naciśnięcie entera w polu TextBox lub przycisku Find powinno wykonać wyszukiwanie.

Zaimplementować możliwość dodawania nowych rekordów do bazy. (1 pkt)



Id	Model	Engine	Year	EngineType	
1	E250	CGI 1,8 (204 hp)	2009	petrol	
2	E350	CGI 3,5 (292 hp)	2009	petrol	
3	A6	FSI 2,5 (187 hp)	2012	petrol	
4	A6	FSI 2,8 (220 hp)	2012	petrol	
5	A6	TFSI 3 (295 hp)	2012	petrol	
6	A6	TDI 2 (175 hp)	2011	diesel	
7	A6	TDI 3 (309 hp)	2011	diesel	
8	S6	TFSI 4 (414 hp)	2012	petrol	
9	S8	TFSI 4 (513 hp)	2012	petrol	

Po naciśnięciu przycisku powinno pojawiać się nowe okno dialogowe, które umożliwi podanie właściwości nowego samochodu. Dodawanie danych do bazy

```
ctx.Cars.Add(car);  
ctx.SaveChanges();
```

**Edycja istniejących rekordów z bazy (1 pkt)**

Po 2-krotnym naciśnięciu wiersza powinno pojawić się okno dialogowe z możliwością edycji danych konkretnego obiektu (Można wykorzystać wcześniej utworzone okno dialogowe). Po zapisaniu danych konieczne jest uaktualnianie rekordów w bazie.

The screenshot shows a Windows application with a 'MainWindow' and a 'Window' dialog.

**MainWindow:**

- Search:
- Column:  Find
- Table:

Id	Model	Engine	Year	EngineType
1	E250	CGI 1,8 (204 hp)	2009	petrol
2	E350	CGI 3,5 (292 hp)	2009	petrol
3	A6	FSI 2,5 (187 hp)	2012	petrol
4	A6	FSI 2,8 (220 hp)	2012	petrol
5	A6	TFSI 3 (295 hp)	2012	petrol
6	A6	TDI 2 (175 hp)	2011	diesel
7	A6	TDI 3 (309 hp)	2011	diesel
8	S6	TFSI 4 (414 hp)	2012	petrol
9	S8	TFSI 4 (513 hp)	2012	petrol

**Window:**

Car

Model:

Year:

Engine

Engine model:

Displacement:

Horse power:

Pobranie konkretnego rekordu z bazy wygląda następująco:

```
var car = ctx.Cars.Where(x => x.Id == carId).FirstOrDefault();
```

Następnie po uaktualnieniu danych należy je zapisać.

```
ctx.SaveChanges();
```