

## Opgave: Uge 3 – Anton's Auto

*Fag: Udviklingsmiljøer*

*Udarbejdet af: Lars Larsen*

Vi har fået stillet følgende opgave

*Antons Auto ønsker en hjemmeside, hvorfra de kan administrere deres bilpark.*

*Tanken er at applikationen kun kører på deres interne netværk, så du behøver ikke tænke authentication, log in system mv. ind i app'en.*

*Antons Auto skal bruge applikationen internt, således at de forskellige medarbejdere hurtigt og nemt kan finde oplysninger om hvilke biler, de har i bilparken pt.*

*Oplysningerne kan for eksempel bruges i samtale med kunde.*

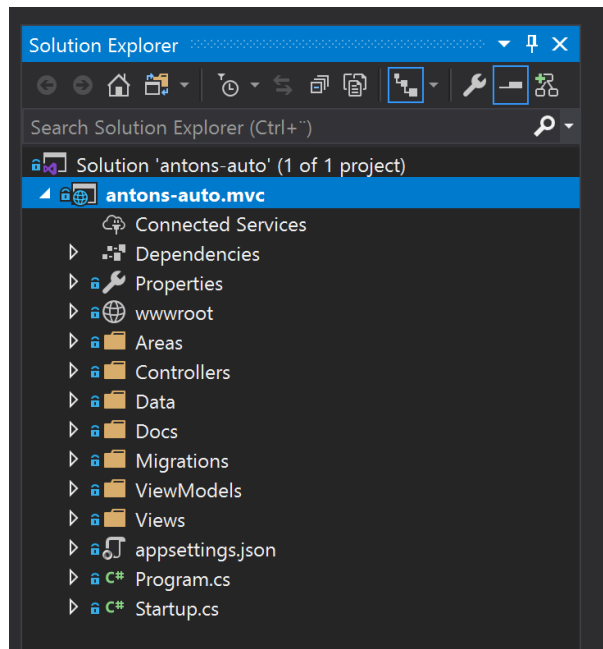
*Alle oplysninger, der ikke fremgår af ovenstående, står til fri fortolkning.*

### *Acceptance Criteria*

- *Du skal lave det som en EF Core MVC app med tilhørende database*
- *Fra denne skal man kunne oprette bilmærker (Audi, BMW, Volvo mv)*
- *Man skal også kunne oprette modeller baseret på det specifikke mærke.*
- *Man skal kunne se alle bilmærker og alle modeller.*
- *Man skal kunne se detalje visning for alle bilmærker og modeller.*

## Løsning

Anton's Auto er lavet som en ASP.Net core MVC løsning. I nedenstående er mappestrukturen

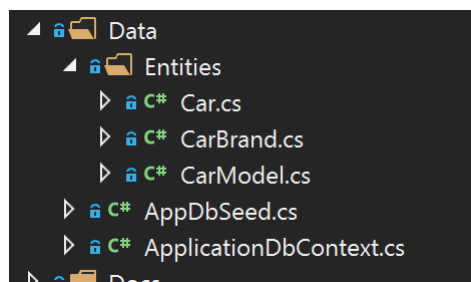


Følgende beskrives de mapper der afviger fra standard skabelonen som Visual Studio generer.

- Data/Entities => Indeholder klasser der beskriver database modellen
- ViewModels => Indeholder klasser der benyttes til views

## Data

Under data/entities er klasser som definerer databasen. Databasen er oprettet i Azure som en SQL Server database.



- CarBrand indeholder bilmærker
- CarModel indeholder bilmodeller
- Car indeholder bil

```

1  using System;
2  using System.ComponentModel.DataAnnotations;
3
4  namespace antons_auto.mvc.Data.Entities
5  {
6      public class Car
7      {
8          public int CarID { get; set; }
9          public int CarModelID { get; set; }
10
11         public DateTime CreationDate { get; set; }
12
13         [Required]
14         public int Year { get; set; }
15
16         [Required]
17         public double Price { get; set; }
18
19         [Required]
20         public int MileAge { get; set; }
21
22         [MaxLength(500)]
23         public string ImageUrl { get; set; }
24
25         public CarModel CarModel { get; set; }
26     }
27 }

```

Jeg har på de enkelte felter defineret hvilke egenskaber de skal have, ved brug af attributter. På ovenstående vises Car klassen. Her er Year, Price og MileAge obligatoriske felter og dermed not null i databasen. Endvidere er MaxLength på ImageUrl sat til maksimum 500 karakterer.

```

1  using System.Linq;
2  using antons_auto.mvc.Data.Entities;
3  using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
4  using Microsoft.EntityFrameworkCore;
5
6  namespace antons_auto.mvc.Data
7  {
8      public class ApplicationDbContext : IdentityDbContext
9      {
10         public DbSet<CarBrand> CarBrands { get; set; }
11         public DbSet<CarModel> CarModels { get; set; }
12         public DbSet<Car> Cars { get; set; }
13
14         public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
15             : base(options)
16         {
17         }
18
19         protected override void OnModelCreating(ModelBuilder modelBuilder)
20         {
21             modelBuilder.Seed();
22
23             modelBuilder.Entity<CarBrand>().ToTable("CarBrand");
24             modelBuilder.Entity<CarModel>().ToTable("CarModel");
25             modelBuilder.Entity<Car>().ToTable("Car");
26             modelBuilder.Entity<Car>(entity =>
27             {
28                 entity.Property(e => e.CreationDate).HasDefaultValueSql("CURRENT_TIMESTAMP");
29             });
30         }
31
32         base.OnModelCreating(modelBuilder);
33     }
34 }
35
36 }

```

I ApplicationDbContext konfigureres EF. Her angives hvad tabel navnene skal være. Som udgangspunkt vil de være i flertal. På Car tabellen angives en default værdi på CreationDate. Den udfyldes med dato og tid når en ny bil oprettes.

I linie 21 kaldes en funktion som opretter nogle predefinerede bilmærker og bilmodeller.

```
AppDbSeed.cs* x
antons-auto.mvc antons_auto.mvc.Data.Entities.AppDbSeed
1 using Microsoft.EntityFrameworkCore;
2
3 namespace antons_auto.mvc.Data.Entities
4 {
5     public static class AppDbSeed
6     {
7         public static void Seed(this ModelBuilder modelBuilder)
8         {
9             modelBuilder.Entity<CarBrand>().HasData(
10                 new CarBrand { CarBrandID = 1, Name = "BMW" },
11                 new CarBrand { CarBrandID = 2, Name = "VW" },
12                 new CarBrand { CarBrandID = 3, Name = "Seat" }
13             );
14
15             modelBuilder.Entity<CarModel>().HasData(
16                 new CarModel { CarModelID = 1, CarBrandID = 1, Name = "X1" },
17                 new CarModel { CarModelID = 2, CarBrandID = 1, Name = "X2" },
18                 new CarModel { CarModelID = 3, CarBrandID = 1, Name = "X3" },
19                 new CarModel { CarModelID = 4, CarBrandID = 2, Name = "Golf" },
20                 new CarModel { CarModelID = 5, CarBrandID = 3, Name = "Arona" }
21             );
22         }
23     }
24 }
```

## ViewModels

For at afkoble database klasserne fra View's benytter jeg mig af et MVVM pattern. På denne måde kan jeg ændre i database konfigurationen, uden at det har indflydelse på mine views.

```
CarViewModel.cs x
antons-auto.mvc antons_auto.mvc.ViewModels.CarViewMod
1 using System.ComponentModel;
2 using System.ComponentModel.DataAnnotations;
3
4 namespace antons_auto.mvc.ViewModels
5 {
6     public class CarViewModel
7     {
8         [Key]
9         public int CarID { get; set; }
10
11         public int CarBrandID { get; set; }
12
13         public string CarModelID { get; set; }
14
15         [DisplayName("Bilmærke")]
16         public string CarBrandName { get; set; }
17
18         [DisplayName("Model")]
19         public string CarModelName { get; set; }
20
21         [Required(ErrorMessage = "Feltet er påkrævet")]
22         [DisplayName("Årstal")]
23         public int Year { get; set; }
24
25         [Required(ErrorMessage = "Feltet er påkrævet")]
26         [DisplayName("Pris")]
27         public double Price { get; set; }
28
29         [Required(ErrorMessage = "Feltet er påkrævet")]
30         [DisplayName("Km")]
31         public int MileAge { get; set; }
32
33         [DisplayName("Billede")]
34         public string ImageUrl { get; set; }
35
36     }
37 }
38 }
```

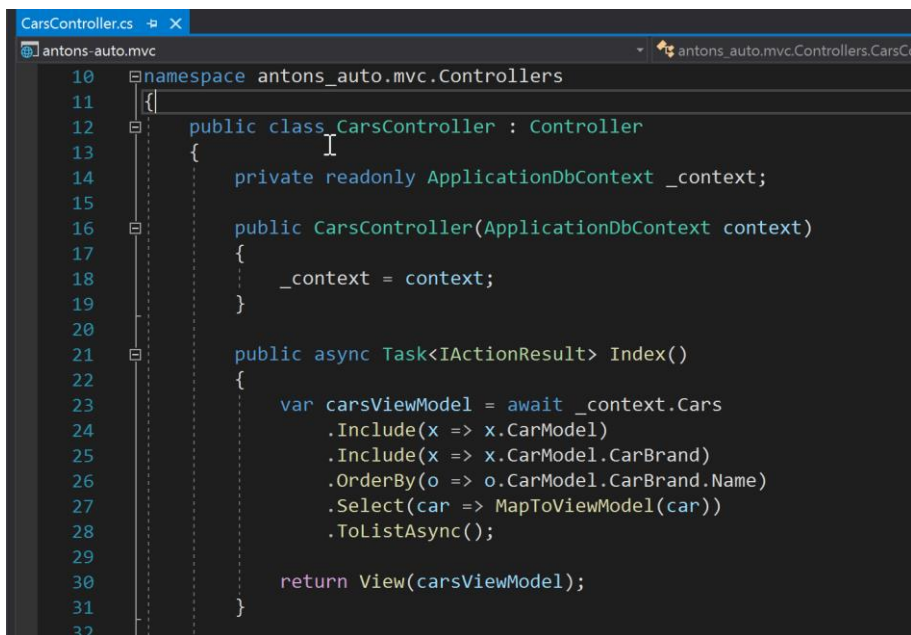
En anden fordel ved viewmodeller, er brugen af attributter til at konfigurere UI'en. Her er displayname konfigureret, så hvis det ændres slår det igennem på alle views.

Ved brug af required, får man klient validering out-of-the-box.

## Controllers

I Car controlleren har jeg taget udgangspunkt i den kode som VS genererer. Det er første gang jeg rigtig har prøvet det.

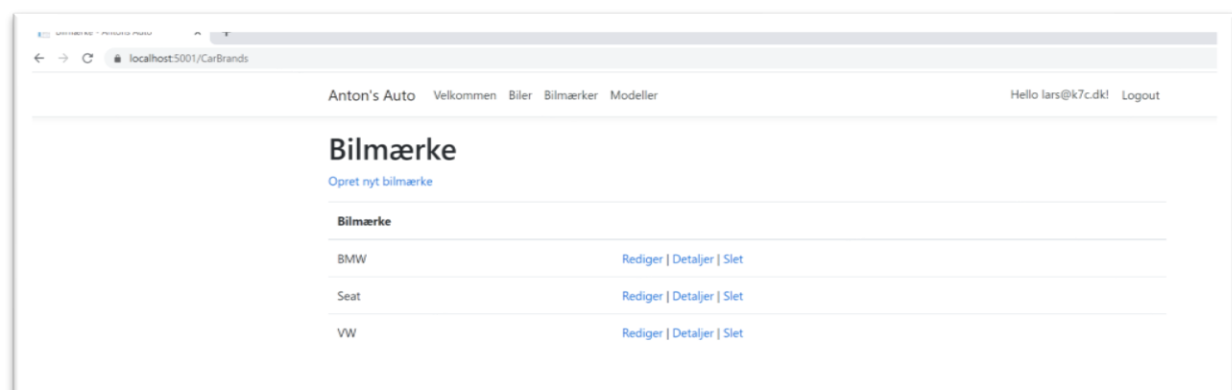
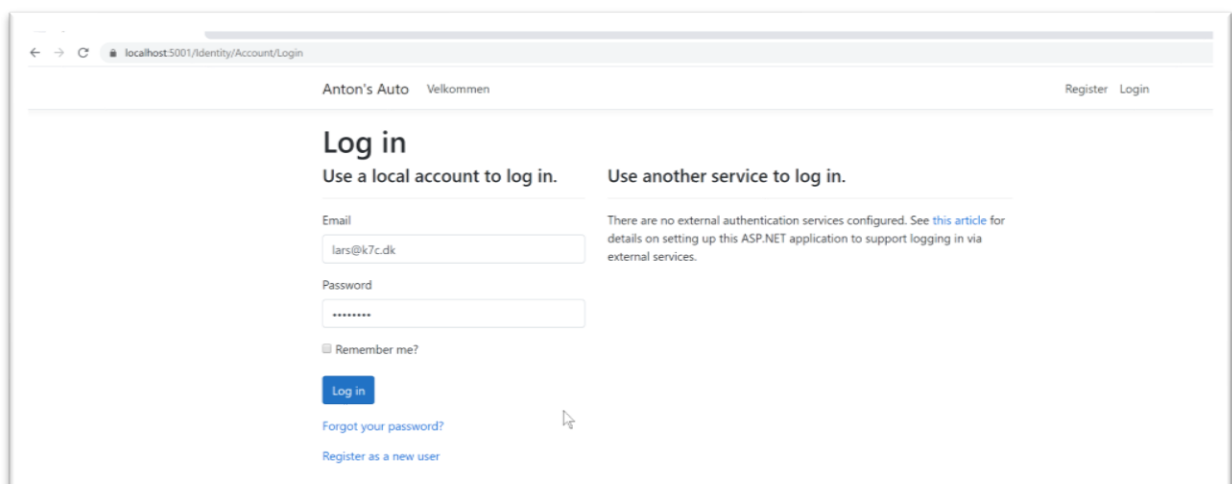
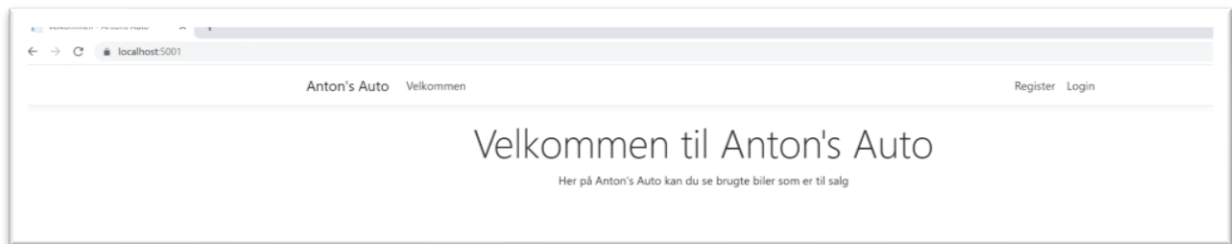
For at hente bilmærker og modeller, benytter jeg mig af Include. På denne måde er de hentet i samme forespørgsel.



```
10 namespace antons_auto.mvc.Controllers
11 {
12     public class CarsController : Controller
13     {
14         private readonly ApplicationDbContext _context;
15
16         public CarsController(ApplicationDbContext context)
17         {
18             _context = context;
19         }
20
21         public async Task<IActionResult> Index()
22         {
23             var carsViewModel = await _context.Cars
24                 .Include(x => x.CarModel)
25                 .Include(x => x.CarModel.CarBrand)
26                 .OrderBy(o => o.CarModel.CarBrand.Name)
27                 .Select(car => MapToViewModel(car))
28                 .ToListAsync();
29
30             return View(carsViewModel);
31         }
32     }
```

## Anton's Auto web

Følgende er skærbilleder fra web løsningen



BRØDER - ANTON'S AUTO

localhost:5001/CarModels

Anton's AutoVelkommenBilerBilmærkerModellerHello lars@k7c.dk!Logout

Modeller

[Opret ny model](#)


Bilmærke	Model	
BMW	X1	<a href="#">Rediger</a>   <a href="#">Detaljer</a>   <a href="#">Slet</a>
BMW	X2	<a href="#">Rediger</a>   <a href="#">Detaljer</a>   <a href="#">Slet</a>
BMW	X3	<a href="#">Rediger</a>   <a href="#">Detaljer</a>   <a href="#">Slet</a>
Seat	Arona	<a href="#">Rediger</a>   <a href="#">Detaljer</a>   <a href="#">Slet</a>
VW	Golf	<a href="#">Rediger</a>   <a href="#">Detaljer</a>   <a href="#">Slet</a>

localhost:5001/Cars

Anton's AutoVelkommenBilerBilmærkerModellerHello lars@k7c.dk!Logout

Biler

[Opret ny bil](#)

	Bilmærke	Model	Årstal	Pris	Km	
	BMW	X1	2018	525000	30000	<a href="#">Rediger</a>   <a href="#">Detaljer</a>   <a href="#">Slet</a>