

Opgave: Uge 5 – Anton's Auto

Fag: Udviklingsmiljøer

Udarbejdet af: Lars Larsen

Url: <https://sm-antons-auto.azurewebsites.net/>

Github repository: <https://github.com/larsk7cdk/antons-auto.git>

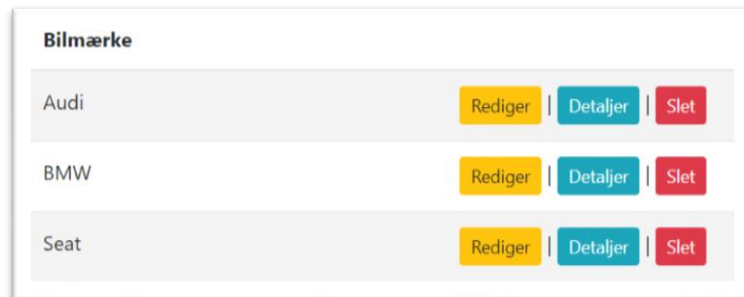
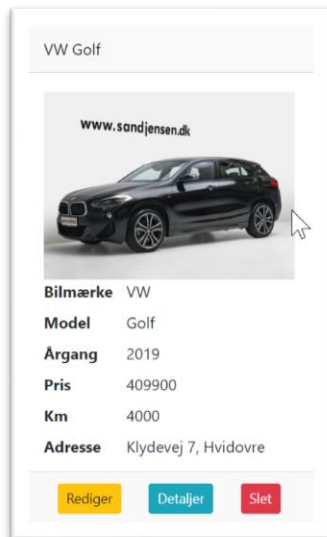
I denne uge har jeg arbejdet med

- Opdatering af HTML
 - o Knapper ændret til "sm" Bootstrap
- Opret ny bil
 - o Tilføjet adresse til en bil
- Google maps på detalje siden
 - o Dawa (Danmarks Adressers Web API)
 - o Google maps
- Sortering
 - o Sortering på bilmodel siden
 - o Sortering på bil siden
- Søgning
 - o Søgning på bilmærke og bilmodel på bil siden
- Sideinddeling på alle bil sider
 - o Forrige og Næste knapper på siden
- "Om" Side
 - o Karrusel til visning af biler

Opdatering af HTML

Knapper ændret til "sm" Bootstrap

Knapper på bil kort og tabeller er ændret til en "btn-sm" klasse. Dette gør knapperne virker mindre dominerende



Opret ny bil

Tilføjet adresse til en bil

Når man opretter en ny bil, er der nu felter hvor adressen kan angives. Oplysningerne gemmes i databasen og kan senere rettes hvis det er nødvendigt på redigerings siden

Bil

Bilmærke
Vælg et bilmærke

Model
Vælg en model

Årgang

Pris

Km

Adresse

Adresse nummer

Postnummer

Billede

[Opret](#)

[Tilbage til oversigt](#)

Rediger bil

Bilmærke
VW

Model
Golf

Årgang
2019

Pris
409900

Km
4000

Adresse
Klydevej

Adresse nummer
7

Postnummer
2650

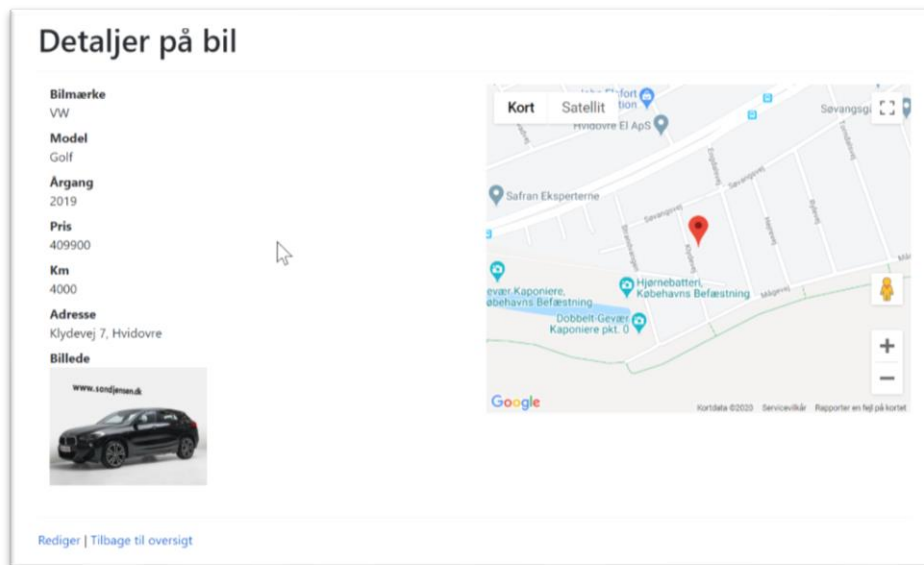
Billede
https://images.autounic.com/ida/car_images/

[Gem](#)

[Tilbage til oversigt](#)

Google maps på detalje siden

Hvis der er registreret en adresse på en bil, og man går ind på detalje siden, vises dette på et Google maps kort



Dawa (Danmarks Adressers Web API)

For at finde længde- og breddegrader for en angivet adresse, er der lavet en serviceproxy, som kan hente disse fra Dawa. Nedenstående viser et udsnit af DawaServiceProxy. Den har en metode der kaldes med vejnavn, vejnummer og postnummer. Ud fra de oplysninger hentes længde-og breddegrader til brug for Google maps

```
public LocationModel GetLocation(string streetName, string streetNo, int postalCode)
{
    var key:string = $"{streetName}#{streetNo}#{postalCode}";

    if (!_locationCache.ContainsKey(key))
    {
        var location = QueryLocation(key);
        _locationCache.Add(key, location);
    }

    return _locationCache[key];
}

internal LocationModel QueryLocation(string key)
{
    var keySplit:string[] = key.Split(separator: '#');
    var query:string = $"adgangsadresser?vejnavn={keySplit[0]}&husnr={keySplit[1]}&postnr={keySplit[2]}&struktur=mini";

    var response:string = GetData(query);

    if (!string.IsNullOrEmpty(response))
    {
        var jsonArray = JSONArray.Parse(response);
        if (jsonArray.Count > 0)
        {
            return new LocationModel
            {
                City = jsonArray[0]["postnrnavn"].ToString(),
                Longitude = float.Parse(jsonArray[0]["x"].ToString()),
                Latitude = float.Parse(jsonArray[0]["y"].ToString())
            };
        }
    }

    return new LocationModel { City = "Ukendt", Longitude = 0, Latitude = 0 };
}
```

Google maps

Google maps er lavet som et partial view og bliver vist ved brug af nedenstående javascript kode. Når siden med dette view renderes, tilføjes map'et til DOM'en og på de givne længde- og breddegrader sættes en markør

```

5 <style>
6   #map {
7     align-items: center;
8     display: flex;
9     justify-content: center;
10    min-height: 50vh;
11  }
12 </style>
13
14 <script>
15   let map;
16
17   document.addEventListener("DOMContentLoaded",
18     () => {
19       const s = document.createElement("script");
20       document.head.appendChild(s);
21       s.addEventListener("load",
22         () => {
23           var pos = {
24             lat: Model.Latitude.ToString().Replace(olddValue: ",", newValue: "."),
25             lng: Model.Longitude.ToString().Replace(olddValue: ",", newValue: ".");
26           };
27
28           map = new window.google.maps.Map(document.getElementById("map"),
29             {
30               center: pos,
31               zoom: 16,
32               mapTypeId: window.google.maps.MapTypeId.ROADMAP
33             });
34
35           var marker = new window.google.maps.Marker({ position: pos, map: map });
36         });
37       s.src = 'https://maps.googleapis.com/maps/api/js?key=ViewData["google-maps-key"]';
38     });
39 </script>
40
41 <div id="map"></div>

```

Sortering

Der er lavet sortering på bilmodel og bil siden.

Sortering på bilmodel siden

På bilmodel siden er sorteringen lavet ved klik på tabellens bilmærke overskrift. Ved klik skifter sortering i henholdsvis stigende og faldende sortering

Bilmodeller		
Opret ny bilmodel		
Bilmærke (Stigende)	Model	
Audi	A3	Rediger Detaljer Slet
BMW	X1	Rediger Detaljer Slet
BMW	X2	Rediger Detaljer Slet

Sortering foretages ved at udvide LINQ udtrykket, som henter bilmodeller med nedenstående kode

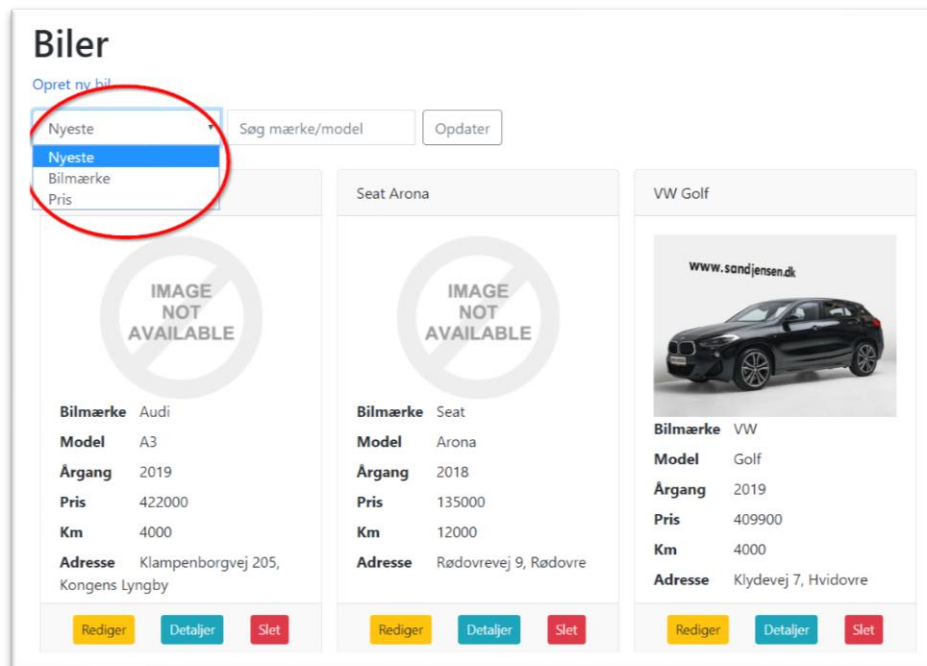
```

private static IQueryable<CarModel> SortCarModels(IQueryable<CarModel> model, string sortOrder)
{
    return sortOrder switch
    {
        "name_desc" => model.OrderByDescending(o => o.CarBrand.Name),
        _ => model.OrderBy(o => o.CarBrand.Name)
    };
}

```

Sortering på bil siden

På bil siden er sortering lavet ved brug af en dropdown liste. Når en given sortering er valgt opdateres ved klik på "Opdater" knappen



Sortering foretages ved at udvide LINQ udtrykket, som henter biler med nedenstående kode

```
private static IQueryable<Car> SortCars(IQueryable<Car> cars, string sortOrder)
{
    return sortOrder switch
    {
        "newest" => cars.OrderBy(o:Car => o.CreationDate),
        "price" => cars.OrderBy(o:Car => o.Price),
        _ => cars.OrderBy(o:Car => o.CarModel.CarBrand.Name)
    };
}
```

Søgning

Søgning på bilmærke og bilmodel


På bil siden er det muligt at indtaste en tekst, som laver en søgning på bilmærke og bilmodel. På nedenstående vises en søgning på audi

Biler

[Opret ny bil](#)


Nyeste

Audi A3



Bilmærke Audi
Model A3
Årgang 2019
Pris 422000
Km 4000
Adresse Klampenborgvej 205, Kongens Lyngby

Audi A3



Bilmærke Audi
Model A3
Årgang 2012
Pris 97900
Km 184000
Adresse Klydevej 7, Hvidovre

Søgning foretages ved at udvide LINQ udtrykket, som henter biler med nedenstående kode

```
private static IQueryable<Car> CarsFiltered(string searchString, IQueryable<Car> cars)
{
    var carsFiltered: IQueryable<Car> = cars.AsQueryable();
    if (!string.IsNullOrEmpty(searchString))
    {
        carsFiltered = cars.Where(s =>
            s.CarModel.CarBrand.Name.Contains(searchString) ||
            s.CarModel.Name.Contains(searchString));
    }
    return carsFiltered;
}
```

Sideinddeling på alle bil sider


På bilmærke og bilmodel siden er der lavet en sideinddeling så der vises 3 poster pr. side. På bil siden er antallet sat til 6. Her vises den sidste side hvor "Næste" knappen er disabled, fordi der ikke er flere sider at vise

Biler

[Opret ny bil](#)


Nyeste

VW Passat




Bilmærke VW
Model Passat
Årgang 2005
Pris 209800
Km 94000
Adresse Klydevej 7, Hvidovre

Volvo V60



Bilmærke Volvo
Model V60
Årgang 2015
Pris 174900
Km 150000
Adresse Klydevej 7, Hvidovre

Audi A3



Bilmærke Audi
Model A3
Årgang 2012
Pris 97900
Km 184000
Adresse Klydevej 7, Hvidovre

PaginatedList

Nedenstående hjælpeklasse bruges til at lave sideinddeling af poster

```

public class PaginatedList<T> : List<T>
{
    public int PageIndex { get; }
    public int TotalPages { get; }

    public PaginatedList(IEnumerable<T> items, int count, int pageIndex, int pageSize)
    {
        PageIndex = pageIndex;
        TotalPages = (int) Math.Ceiling(count / (double) pageSize);

        AddRange(items);
    }

    public static async Task<PaginatedList<T>> CreateAsync(IQueryable<T> source, int pageIndex, int pageSize)
    {
        var count = await source.CountAsync();
        var items = await source.Skip((pageIndex - 1) * pageSize).Take(pageSize).ToListAsync();
        return new PaginatedList<T>(items, count, pageIndex, pageSize);
    }
}

```

Her vises koden for index klassen af bil siden. ViewData bruges til at sende data mellem klient og server siden.

```

public async Task<ActionResult> Index(
    string sortOrder = "newest",
    string searchString = "",
    int pageNumber = 1)
{
    _noImage =
        $"{HttpContext.Request.Scheme}/{HttpContext.Request.Host.Value}/images/image_not_available.jpg";

    _sorting.ForEach(f | SelectListItem => f.Selected = f.Value == sortOrder);

    var cars = IQueryable<Car, CarBrand> = _context.Cars
        .Include(navigationPropertyPath: X: Car => X.CarModel) // IQueryable<Car, CarModel>
        .Include(navigationPropertyPath: X: Car => X.CarModel.CarBrand);

    var carsFiltered: IQueryable<Car> = CarsFiltered(searchString, cars);
    var carsSorted: IQueryable<Car> = SortCars(carsFiltered, sortOrder).AsNoTracking();
    var paginatedList = await PaginatedList<Car>.CreateAsync(carsSorted, pageIndex: pageNumber, PAGE_SIZE);
    var carsViewModel: IEnumerable<CarViewModel> = paginatedList.Select(MapToViewModel);

    ViewData["sorting"] = _sorting;
    ViewData["sortOrderSelect"] = sortOrder;
    ViewData["currentFilter"] = searchString;
    ViewData["pages"] = (int) Math.Ceiling((decimal) carsSorted.Count() / (decimal) PAGE_SIZE);
    ViewData["pageIndex"] = pageNumber;

    return View(carsViewModel);
}

```

Forrige og Næste knapper på siden

"Forrige" og "Næste" knapperne gør brug af et anchor tag. For at bibeholde de valgte værdier for sortering og søgning, sendes de med når der skiftes side

```

var prevDisabled: string = int.Parse(ViewData["pageIndex"].ToString()).Equals(1) ? "disabled" : "";
var nextDisabled: string =
    ViewData["pageIndex"].Equals(ViewData["pages"]) ||
    ViewData["pages"].Equals(0) ? "disabled" : "";

<a asp-action="Index"
    asp-route-sortOrder=@ViewData["sortOrderSelect"]
    asp-route-pageNumber=@(int.Parse(ViewData["pageIndex"].ToString()) - 1)
    asp-route-searchString=@ViewData["currentFilter"]
    class="btn btn-outline-secondary @prevDisabled">
    Forrige
</a>

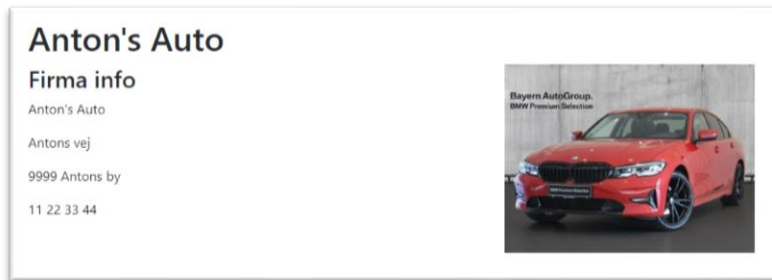
<a asp-action="Index"
    asp-route-sortOrder=@ViewData["sortOrderSelect"]
    asp-route-pageNumber=@(int.Parse(ViewData["pageIndex"].ToString()) + 1)
    asp-route-searchString=@ViewData["currentFilter"]
    class="btn btn-outline-secondary @nextDisabled">
    Næste
</a>

```

"Om" side

Karrusel til visning af biler

På "Om" siden har jeg lavet en karrusel, som viser de biler der har en imageUrl



Til dette formål har jeg benyttet Bootstrap. Ved første gennemløb skal "active" klassen sættes på det første element

