

BrugtMarket - Microservices

Helidon mircoservices og React

John Grandt Markvard Høeg | Microservices | 28/05/2021

eaajgmh@students.eaaa.dk

Contents

[Noter for seedning af database 2](#_Toc73090959)

[Arkitektur - Helidon 2](#_Toc73090960)

[Domain 2](#_Toc73090961)

[Entity 3](#_Toc73090962)

[Repository 3](#_Toc73090963)

[ressource 3](#_Toc73090964)

[DTO 3](#_Toc73090965)

[Service 3](#_Toc73090966)

[React 3](#_Toc73090967)

[Komponenttyper 3](#_Toc73090968)

[Header.js 4](#_Toc73090969)

[Forside.js 4](#_Toc73090970)

[Advertisements.js 4](#_Toc73090971)

[SpecificAdvertisement.js 4](#_Toc73090972)

[User.js 4](#_Toc73090973)

[Didmount lifecykle hook 4](#_Toc73090974)

[state mangement 5](#_Toc73090975)

[css 5](#_Toc73090976)

[Bilag 6](#_Toc73090977)

[Rest endpoint json for at fylde databasen. 6](#_Toc73090978)

[City 6](#_Toc73090979)

[User 6](#_Toc73090980)

[Category 7](#_Toc73090981)

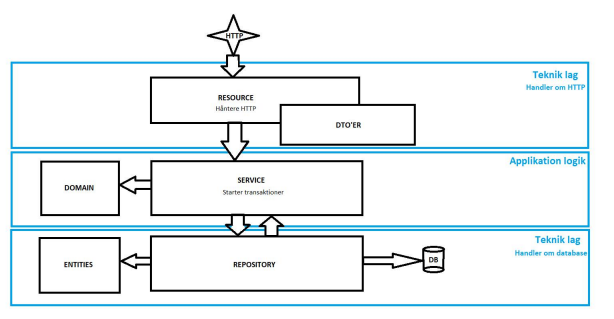
[Advertisements 8](#_Toc73090982)

# Noter for seedning af database

Venligst referer til [bilag](#_Billag) for mine endpoint cals til at seede databasen med data der kan fremvises på klienten.

# Arkitektur - Helidon

Jeg har valgt at tage udgangspunkt i Microservices arkitektur, jævnfør nedenstående billede, ligesom jeg gjorde i den opligatoriske aflevering.



Der er der lidt forskel mellem denne model og hvordan jeg har sat det op.

Jeg har valgt 4 lag i min opgave, de henholdsvis består af, Domain laget, Repository laget, Service laget og Ressource laget. Ikke nødvendigvis i den rækkefølge.

## Domain

Jeg har valgt at skabe domain klasse objekter for alle attributter (felter) når det gav mening. Eksempler på hvornår det ikke bliver brugt er i City klassen der har en attribut der hedder city. Det har gjort det lettere at arbejde med når vi kommer ned i min PO klasser. Det gør også jeg har mere kontrol over en klasse. Dog så kunne jeg godt have udskriftet mange af klasserne med ”Strings”, da det skaber en smule ”overflow” af klasser og kan virke uoverskueligt på første øjekast.

## Entity

I Repository laget har jeg en entity package hvor jeg har mine ”persistence objekter” eller PO-klasser. Disse objekter er til at kunne tilgå databasen, samt seede databasen med tabeller. Det er også her hvor mine relationer bliver smidt i spil.

## Repository

Der er skabt en Repository klasse til hver PO klasse, ligesom den obligatoriske opgave har jeg valgt at køre en Repository pattern med en generisk interface, det skal dog siges, at jeg kom i problemer med at bruge interfacet senere hen ad vejen, da der var metoder der blev overflødelige.

Jeg har valgt at tage en Data Mapper i brug i Repository laget der mapper mine entities til domain klasser og omvendt.

## ressource

Jeg har valgt at skabe en Ressource klasse for hver Repository. Dette er her hvor mine endpoints er defineret, jeg har også en Mapper der konverter mine Domain klasser om til DTO (Data Transfer Object) klasser, disse klasser bruger brugt i mine Resource metoder

## DTO

I min DTO-pakke har jeg for hver PO klasse en CreateDTO og en ReadDTO. Derudover har jeg specialiserede DTO-klasser til at opfylde bestemte parameter, såsom CategoryCountDTO der viser alle kategorier, og har en tæller ved siden af der fremviser, hvor mange af hver kategori der er.

## Service

I mit applikationslag har jeg en Service klasse for hver Repository klasse der er, servicelaget er det bindeled mellem Resource klasserne og Repositoriet.

# React

## Komponenttyper

I mit projekt har jeg taget brug af class’s og funktionelle komponenter.

Forside.js er min startside, det henter Header komponenten (Header.js) ind i toppen af sin render og sammen kobler dermed en fuldendt side. Hver af mine komponenter, Forside.js, Advertisements.js, SpecificAdvertisement.js og User.js har inkorporeret Header ind i deres komponent.

### Header.js

I min Header.js har vi et ikon der sender dig til stien: ”/” hvilket via min App.js er Forside.js. Derudover har den en ”dropdown menu” der henter alle mine kategorier via en fetch metode, og hvis en kategori bliver valgt, går den til den pågældende side. Derudover har den 3 knapper der ikke gør noget.

### Forside.js

Min Forside.js er ganske simpel den har en fetch metode der tager brug af min CategoryCountDTO, hvilket henter alle kategorier og hvor mange Advertisements der er tilknyttet hver kategori, den fremviser det på forsiden og der er en knap tilknyttet til hver kategori der render alle Advertisements der er tilknyttet den valgte kategori.

### Advertisements.js

Dette er siden man kommer til når du enten har valgt en kategori via Forside.js eller Header.js. Her vises en titel af den valgte kategori samt alle Advertisements til den valgte kategori. Der er givet et stok-billede til hver Advertisement, derudover så viser hver Advertisement hvilken type den er, samt en titel, tekst og pris for hver Advertisement.

Der er og en knap tilknyttet hver advertisement der sender dig videre til den bestemte advertisement.

### SpecificAdvertisement.js

Denne side render når man har valgt en specifik Advertisement i den forgående side.

Denne side viser en profil kort med kort information omkring den bruger der har skabt advertisement samt uddybende information omkring den valgte advertisement.

Nu kan du også se hvornår den blev skab og er formateret i en år-måned-dag format. Der er også angivet en knap der viser uddybende information omkring den specifikke bruger der har lavet den specifikke advertisement.

### User.js

Når du kommer ind på den sidste side af min klient User.js, kan du se uddybende information omkring den valgte bruger, samt en liste af alle de advertisements brugeren har.

## Didmount lifecykle hook

I alle mine components bortset fra User.js har jeg en componentdidmount metode der fetcher dataen og render den til fremvisning af den pågældende side, i User.js bliver den sent et objekt med i min render metode og så bliver der bare taget brug a this.props.object til at render de bestemte værdier.

## state mangement

I min fetch i didmount lifecyckle hook har jeg taget brug af setState til at gemme de værdier jeg får ud, hvorved jeg derefter kan tage dem i brug i min render page. Jeg har derudover lavet sådan at en page ikke kan render hvis der kommer en fejl eller hvis der ikke bliver hentet noget data. Dette vises i min render hvis dette er tilfældet.

## css



Jokes ved siden af, hver af mine komponenter har en css tilknyttet. Her skal jeg dog være ærlig at sige jeg har ingen forstand på design, og jeg er endnu dårligere til css. Jeg har fået hjælp til at lave meget af css, men resten er så grimt og dårligt designet at jeg aldrig ville aflevere det hvis ikke det var fordi jeg løb tør for tid.

# Bilag

## Rest endpoint json for at fylde databasen.

### City

http://localhost:8087/city

{

"zipCode": "8260",

"city": "Viby"

}

{

"zipCode": "8600",

"city": "Silkeborg"

}

### User

http://localhost:8087/users

{

"firstName": "Soren",

"lastName": "Hoeg",

"companyName": "Saphyron Industries",

"phoneNumber": "88888888",

"phoneCode": "+45",

"email": "test@test.dk",

"creation": 1621952561420,

"advertisements": [],

"city": {

"zipCode": "8260",

"city": "Viby"

},

"type": "køber"

}

{

"firstName": "John",

"lastName": "Hoeg",

"companyName": "Saphyron Industries",

"phoneNumber": "77777777",

"phoneCode": "+45",

"email": "test2@test.dk",

"creation": 1621952561420,

"advertisements": [],

"city": {

"zipCode": "8600",

"city": "Silkeborg"

},

"type": "sælger"

}

### Category

<http://localhost:8087/categories>

{

"category": "Bil",

"advertisements": []

}

{

"category": "MC",

"advertisements": []

}

{

"category": "Computer",

"advertisements": []

}

### Advertisements

[http://localhost:8087/advertisements/User/{userId}](http://localhost:8087/advertisements/User/%7buserId%7d)

{

"category": "Bil",

"type": "Salg",

"headline": "Toyota Corolla",

"text": "Toyota Corolla er modelbetegnelsen for en personbil i den lille mellemklasse, som siden 1966 er blevet fremstillet af det japanske Toyota Motor Corporation i flere forskellige udførelser.",

"price": 8000,

"creation": 1621952561420

}

{

"category": "Bil",

"type": "Lease",

"headline": "Ford Focus",

"text": "Ford Focus er en lille mellemklassebil bygget af Ford Motor Company. Den blev introduceret i Europa i efteråret 1998, i Nord- og Sydamerika i foråret 2000 og i Australien i september 2002. I de fleste lande efterfulgte den Ford Escort, i Asien og Australien Ford Laser. I 2000 og 2001 var Ford Focus verdens mest solgte bil.",

"price": 5000,

"creation": 1621952561420

}

{

"category": "MC",

"type": "Salg",

"headline": "Indian Challenger Dark Horse",

"text": "ABS bremser, corner-ABS, TCS (traction control), GPS-navigation, kørecomputer, gearindikator, infocenter, nøglefri betjening (keyless ride), indsprøjtning, justerbar kåbeglas, LED forlygte, LED baglygte, musik anlæg, USB tilslutning, bluetooth denne mc er mont med ekstraudstyr mod mer pris. Høj styr sissibar motorbøjle mm Gerne bytte billig finans",

"price": 514995,

"creation": 1621952561420

}

{

"category": "MC",

"type": "Lease",

"headline": "Knallert 45",

"text": "Knallert er et køretøj med 2 eller 3 hjul. Der findes omkring 240.000 knallerter i Danmark, svarende til at en person ud af 33 ejer en. Før 1976 brugte Færdselsloven betegnelsen cykel med hjælpemotor om knallerter. I Danmark er knallerter lovmæssigt inddelt i to grupper: Lille knallert (også kaldet knallert 30 eller almindelige knallert) som maksimalt må køre 30 km/t. Stor knallert (også kaldet knallert 45 eller EU-knallert) som maksimalt må køre 45 km/t. Det samlede antal er nogenlunde ligeligt fordelt på de to typer.",

"price": 2000,

"creation": 1621952561420

}

{

"category": "Computer",

"type": "Salg",

"headline": "Intel i7-7700k med GTX 1070",

"text": "En gamer pc med en gammel men god i7-7700k kan spille alle psil på markedet",

"price": 7500,

"creation": 1621952561420

}