

Afsluttende opgave – Dataservice

Her følger den afsluttende opgave, som er et 2-dages projekt i brug af Dataservice-teknologierne, hvor du skal prøve kræfter med forskellige elementer fra undervisningens øvelser og API-opgaver.

Opgaven rummer flere del-opgaver, men nåede du at gennemføre fagets løbende øvelser og opgaver, vil du opdage, at der er meget du kan genbruge, og i så fald når du måske det meste.

Men ellers udfør, det du kan nå, vis hvad du kan, og fokuser på API-funktionerne og knap så meget på stylingen. Styling er selvfølgelig ikke uden betydning, men hvis tiden er knap, så brug hellere tiden på at hente data, og style det færdig når/hvis der er tid.

Du forventes at bruge hvad der svarer til 2 hele "skoledage" (ca. 7 timer pr. dag) på opgaven – men ikke din fritid! Så løs opgaverne fra en ende af – løs én opgave ad gangen, og start med det, du synes er nemmest.

Du som nævnt må/kan (gen)bruge en stor del af koder mv. fra dine tidligere opgaver – men husk at tilpasse det til denne opgave, så koderne ikke fyldt med kommentarer, misvisende variabel- og funktions-navne mv. fra en helt anden opgave og kontekst.

AI: Du må bruge AI som inspiration og hjælp til at kode dele/elementer af opgaverne, men hent ikke hele store færdige klumper AI-genereret kode. Du skal (som til eksamen) kunne redegøre for, hvad din kode gør, hvordan den virker og hvorfor du har valgt at gøre det på "den måde".

Den afsluttende opgave indgår som en væsentlig del af den samlede karakter for faget, men i karakteren vil også indgå din generelle aktivitet i undervisningen – dvs. din deltagelse og fremmøde, interesse i emnerne samt dine præstationer/opgaveafleveringer (om alle opgaver er løst og færdighedsgraden/hvor gennemarbejdede løsningerne er).

Aflevering

Du skal samle din løsning i et (og kun 1) nyt React-/Next-projekt, som rummer det fra opgaven her, som du har nået.

Aflever senest fredag den 24. maj kl.9. Husk at notere, hvad du nåede/ikke nåede.

Aflever følgende:

1. **På din GitHub-konto:** Upload/publish dit React-projekt og dit API
2. **I VIDOnline (den afsluttende opgave):**

I afleverings-tekstboksen:

- Link til det uploadede projekt på GitHub
- Evt. link til projektet, hvis du har det kørende online et sted
- Skriv hvilke opgaver du **nåede at løse færdig**
- Skriv hvilket API du valgte i opgave 5, hvis du nåede denne opgave
- Evt. **kommentarer** til din aflevering

Upload/aflever dit react-projekt:

- som zip-fil (UDEN node_modules!!!)

Indhold

Aflevering.....	2
Opstart: Lav nyt react-projekt til opgaverne.....	4
Opgave 1 – Viborg Haveservice	5
Opgave 1.1 Viborg Haveservice - forside/velkomst	6
Hvis du har tid og kan klare det: Admin af about-teksten	7
Opgave 1.2 Viborg Haveservice - slider med kundeudtalelser/reviews	8
Hvis du har tid og kan klare det: Admin af sliderdata/reviews	8
Opgave 2 – Vejret/OpenWeather	9
DAWA og map – hvis du kan nå det og har løst det tidligere	9
Opgave 3 – Nyheder/NewsAPI.....	10
"Nyheds-valg"	10
Opgave 4 – Energidata/Elspot Prices.....	11
Hvis du har tid og kan: Valg af dato og pricearea	11
Opgave 5 – udtræk fra et eget valgt API	12

Opstart: Lav nyt react-projekt til opgaverne

1. Opret et nyt React-projekt

– kald det "**dataservice_afsluttendeopgave**" el.lign.

Installer nødvendige pakker – fx `react-router-dom`, `axios`, `react-icons` mv.

Installer efter eget valg det, du vil bruge til "**styling**"/**opsætning** – Tailwind, Bootstrap, Materialize, Sass, Pico.css el.lign.

Fjern overflødige filer og lav en **god mappe-struktur** – se evt. forslagene her:

<https://dev.to/chrisachard/tips-for-organizing-react-projects-191>

2. Opret nogle pages/components – indsæt i første omgang bare en overskrift fx `<h1>` i hver af dem (brug evt. snippet [rafce](#)) fx:

- **Home.jsx** (noget velkomst-tekst – eller hvad du nu vil fylde ud med)
- **ViborgHaveservice1.jsx** og **ViborgHaveservice2.jsx**
- **Vejret.jsx** (opgave 2)
- **Nyheder.jsx** (opgave 3)
- **Energidata.jsx** (opgave 4)
- Og hvis du har tid til overs: **Opgave 5 og 6** – hvis du kan nå det

3. Byg routing op og lav en fungerende navigation til alle siderne

4. Hent dine hooks (fx det til API-kald) og andet (evt. navbar) du kan genbruge fra dit tidligere projekt. Fx også **Error.jsx**, **Loading.jsx**, **NoMatch.jsx**, pagination mv.

5. Style det en smule nu (især navbaren), men fokuser på at hente/vise data fra API'er, men brug også lidt tid på at få det til at være responsivt og præsentabelt.

Opgave 1 – Viborg Haveservice

Viborg Haveservice er en tidligere eksamensopgave. I opgaven her skal du prøve kræfter med nogle få udvalgte dele af den oprindelige opgave.

Info til opgaven:

- Du får udleveret API'et og databasen (og alle request-kald i Insomnia), så du kan starte det op i VS Code (jeg guider).
- Herefter skal du lave **request til dette API**, på samme måde som et hvilket som helst andet API. Bare husk at have projektet/serveren startet op.
- **Prøv at løse en eller flere af følgende Viborg Haveservice-opgaver med udtræk fra API'et – og tilpas også stylingen, så det ligner lidt.**

Opgave 1.1 Viborg Haveservice - forside/velkomst

Udtræk data (og billeder) fra det udleverede Viborg Haveservice-API og opsæt en side, som ligner det, der er vist herunder.


Velkommen til **Viborg Haveservice**

Hos Viborg Haveservice er vi en virksomhed, som varetager en bred skare af forskellige arbejdsopgaver inden for havearbejde og kan blandt andet hjælpe til med:

- Hækkklipping
- Græsslåning
- Træfældning
- Belægning


Kort og godt er vi de rette at kontakte ved enhver type af opgave inden for haveservice, og med mange års erfaring i faget, kan du trygt lade dit valg falde på netop os.

[SE ALLE YDELSER](#)



Anlægsgartneri

Den fulde pakke for virksomheder, der ikke har tid til at passe sine udendørsarealer.



Plænepleje

For den private som ønsker en plan og flot græsplæne.

Prøv også at ramme stylingen – men fokuser på at hente/vise dataene.

Venstre kolonne (rød markering herunder)

- Overskriften "Velkommen til Viborg Haveservice" skal du selv tilføje (ikke en del af API'et)
- Data skal hentes fra endpoint "[aboutus](#)".
- **Knappen/linket** "Se alle ydelser" (se herunder) skal **linke til siden i opgave 1.2**.

Højre kolonne - "Anlægsgartneri" og "Plænepleje"

- Data skal hentes fra endpoint "[services](#)".

Der er 4 services i alt.

Vis **to tilfældige services**, hver gang siden loader.

Hvis det er for svært, så bare udtræk de 2 viste services (Anlægsgartneri og Plænepleje).


Velkommen til Viborg Haveservice

Hos Viborg Haveservice er vi en virksomhed, som varetager en bred skare af forskellige arbejdsopgaver inden for havearbejde og kan blandt andet hjælpe til med:

- Hækklipning
- Græsslåning
- Træfældning
- Belægning


Kort og godt er vi de rette at kontakte ved enhver type af opgave inden for haveservice, og med mange års erfaring i faget, kan du trygt lade dit valg falde på netop os.

[SE ALLE YDELSER](#)



Anlægsgardneri

Den fulde pakke for virksomheder, der ikke har tid til at passe sine udendørsarealer.



Plænepleje

For den private som ønsker en plan og flot græsplæne.

Hvis du har tid og kan håndtere det: Admin af about-teksten

Lav en (admin-)side, hvorfra man kan **rette (PUT) "aboutus"-teksten**. Dvs. teksten i den **røde ramme** på billedet herover.

Fra endpoint **"aboutus"**.

Data består af **"title"** og **"content"**.

Ignorer evt. at der er html-tags i den oprindelige tekst – det er fint nok bare med tekst uden formatering/html.

Opgave 1.2 Viborg Haveservice - slider med kundeudtalelser/reviews

Udtræk data (og billeder) fra Viborg Haveservice-API'et og opsæt en side, som ligner det, der er vist herunder.

Indholdet skal helst fungere som en "slider" – se [Bootstrap eksempel her](#). Og [Owl-eksempel her](#). Og Slideren behøver ikke køre automatisk.

Prøv at ramme stylingen så godt som du har tid til.

- **Endpoint** er "**reviews**", som rummer reviews/tekst, som skal vises i slideren.
- Brug evt. Bootstraps slider eller en anden pakke - eller kod det selv.
- 3 prikker = 3 slides/tekster. Og fremhæv prikken der markerer den aktive/synlige slide-tekst. Klik på prik nr. 2 "navigerer" til "slide 2".
- Det er kun teksten under overskriften som "slides" – ikke overskrift, baggrundsbillede mv.
- Overskriften "**Kundeudtalelser**" skal du skrive selv = ikke data fra API'et.



Hvis du har tid og kan håndtere det: Admin af sliderdata/reviews

Lav en "**admin-side**", hvorfra man kan oprette/rette/slette reviews (som "todos"!).

Fra **endpoint** "**reviews**".

API'et er sat op til det og kan allerede håndtere både POST, PUT og DELETE request, så prøv dig frem – og brug Insomnia som guide.

Opgave 2 – Vejret/OpenWeather

I undervisningen indgik øvelser og eksempler fra OpenWeathers endpoint "**current weather data**", så brug dine erfaringer og genbrug koden fra disse øvelser til at løse opgaven her:

- I en component (fx Vejret.jsx) - udtræk **vejrdata** fra [OpenWeather](#):
- Loop/map vejr-data ud fra endpoint'et "[Call 5 day / 3 hour forecast data](#)":
<https://openweathermap.org/forecast5>
- Det er vejrdata for 5 dage og for hver 3. time = **40 "vejr-klumper"** der skal map'es ud.
- **Udtræk vejret ud fra** zip/postnummer (fx 8000) og landekode (*da*) på samme måde som ved "current weather" – og tjek i dokumentationen, hvordan requestet skal opbygges.
- Udtræk og vis nogle vejrdata fx:
 - [bynavn](#)
 - [temperatur](#) (i [celcius](#))
 - tidspunkt for [solopgange og solnedgang](#)
 - [luftfugtighed](#) og [lufttryk](#)
 - [description](#)
 - og evt. et [vejr-ikon](#) ☀

Flere data (fx sol op og ned) bliver gentaget (er det samme) hver dag – prøv at samle det, så der ikke er en masse gentagelser.

- Det skal være vejrdata for et **postnummer** som indtastes i et **input**-felt.

DAWA og map – hvis du kan nå det og har løst det tidligere

Hvis du fik lavet øvelserne i klassen med DAWA-opslag og Leaflet-map, så har du mulighed for at nå at udvide med følgende:

1. [DAWA-opslag](#) på postnumre – fx som vi gjorde det med "current weather".
2. [Leaflet-kort](#) som skifter fokus til valgte by/postnummer – fx også som vi gjorde det med "current weather"
3. Udvid evt. med noget **ekstra** efter eget valg

Opgave 3 – Nyheder/NewsAPI

I undervisningen har vi også været omkring NewsAPI, hvor du har hentet data fra forskellige endpoints. Her får du virkelig mulighed for at bruge/genbruge det, du har lavet:

- I en component fx Nyheder.jsx - udtræk **nyheder** fra [NewsAPI](#)
- Brug endpointet **"everything"** - <https://newsapi.org/docs/endpoints/everything>
- Vis/udtræk minimum:
author, **title**, **description**, **billede** (nogle af nyhederne er uden billede), **link** til "læs mere".

OBS! Hvis du lægger dit projekt online, er det ikke sikkert, at du kan få det til at virke der – NewsAPI har vist en begrænsning på gratis-versionen, at den kun må køre på localhost!

"Nyheds-valg"

Hvis du løste alle NewsAPI-øvelserne, kan du nemt udvide med en eller flere af følgende:

- Valg af **language**-code i en select-dropdown el.lign.:

language The 2-letter ISO-639-1 code of the language you want to get headlines for. Possible options: **ar** **de**
en **es** **fr** **he** **it** **nl** **no** **pt** **ru** **sv** **ud** **zh** .

[Læs evt. dokumentationen her](#)

- Søgning/**søgefelt** – så brugeren kan fritekst-søge i nyhederne.
- Valg af **sortering** (fx igen i en select-dropdown) - udfyld med sorteringsmulighederne som select-options:

sortBy The order to sort the articles in. Possible options: **relevancy** , **popularity** , **publishedAt** .
relevancy = articles more closely related to **q** come first.
popularity = articles from popular sources and publishers come first.
publishedAt = newest articles come first.

[Læs evt. dokumentationen her](#)

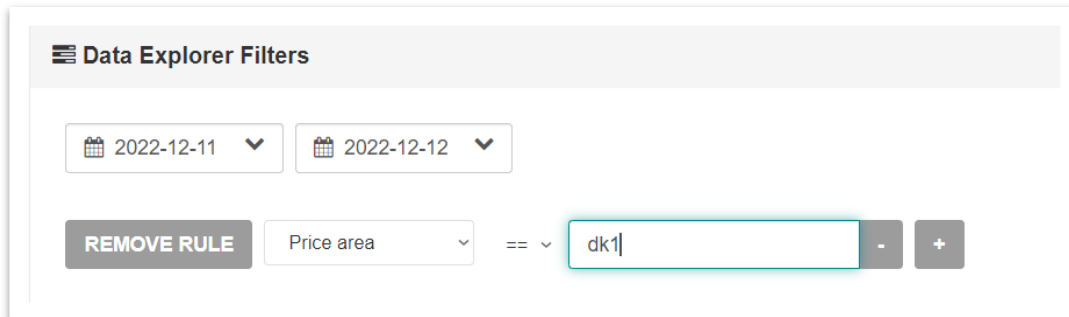
- Udvid evt. med flere muligheder – se API'et muligheder i dokumentationen. Og style det gerne lidt, så nyhederne præsenterer sig pænt.

Opgave 4 – Energidata/Elspot Prices

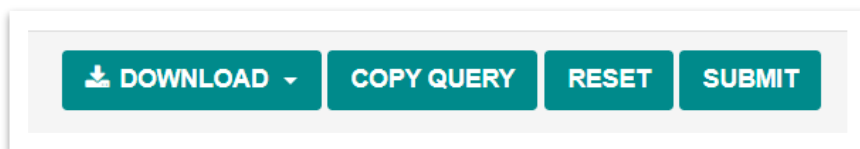
I en component (fx Energipriser.jsx) - udtræk energipriserne fra Elspot Prices:

Test først api'et og query:

1. Udfyld med en **dato** og vælg evt. **dk1** eller **dk2** (vest eller øst for Storebælt):



2. Klik på knappen **SUBMIT** for at se resultatet og herefter på **COPY QUERY** for at snuppe url'en til webservices/API'et:



3. I React – hent og vis (loop/map) priserne for strømmen for din udvalgte dato og opsæt det overskueligt fx i en table.

Hvis du har tid og kan: Valg af dato og pricearea

Vælg gerne en eller flere af disse udvidelser:

1. Tilføj et par input-felter el.lign. så brugeren selv kan **vælge datoer**
2. Tilføj fx en select-dropdown så man kan vælge mellem **"Price area" dk1 og dk2**
3. Vis gerne el-priserne i en **graf, søjler** (overvej [Chart.js](#)) eller på en anden **visuel måde**. Du kan se et ex her (fokuser på søjlerne – ignorer alt det udenom), men du må også gerne gøre det på en helt anden måde: <https://elspotpris.dk>. Brug evt. [gsap](#) el.lign. til at animere data-visningen, hvis du har erfaring med det.

Opgave 5 – udtræk fra et eget valgt API

I denne opgave skal du **udtrække data fra et API efter eget valg**. Dog **IKKE JSONPlaceholder, SWAPI, NewsAPI eller OpenWeather** eller af dem, der er gennemgået i **undervisningen** eller som du har afleveret/lavet tidligere.

- Find listen med med forslag til API'er/åbne data (ligger i kurset Dataservice) og vælg et.
- Eller lav noget helt nyt content i [PocketBase](#), [Airtable](#) eller [Contentful](#), som du så trækker ud i dit React-projekt med request til Airtables/Contentfuls API'er.

Det skal i så fald minimum rumme noget tekst og et billede. Og IKKE todos!

Hvis det giver mening i forhold til det API, du har valgt, må du meget gerne **fordele dataene fra API'et på 2/flere sider/components** (parent/child) – så der fx map'es overskrifter ud på den ene side/component – med link til detaljer, som så "linker til" visning af detaljer (det der er klikket på) i en anden side/component:

