D3.js Forårsprojekt

Af William Schultz

# Kravspecifikation

## Brugsmønstre (need-to-have)

### **Brugeren justerer deres kost**

**Formål:** Brugeren klikker på en søjle for at ændre mængden af kulhydrater, fedt eller protein for en bestemt dag.  
**Trin:**

1. Brugeren ser kostplanens visualisering.
2. Klikker på en søjle (f.eks. fedt for onsdag).
3. Et prompt-vindue åbner og beder om en ny værdi.
4. Brugeren indtaster en ny værdi.
5. Grafen opdateres, og ændringen gemmes i localStorage.

### **Brugeren modtager personlige kostråd**

**Formål:** Brugeren klikker på en knap og får anbefalinger til forbedring af kostplanen.  
**Trin:**

1. Brugeren klikker på knappen "Beregn dit personlige kostråd".
2. Programmet analyserer dataen.
3. Sammenligner totaler mod anbefalede niveauer (kostProcent).
4. Viser en alert med råd som:
   * "Overvej at reducere dit fedtindtag."
   * "Dit proteinindtag er i den anbefalede zone."

### **Brugerens data gemmes og gendannes automatisk**

**Formål:** Brugeren lukker eller genindlæser siden, og deres kostdata bevares.  
**Trin:**

1. Brugeren foretager ændringer i grafen.
2. Ændringer gemmes automatisk i localStorage.
3. Ved genindlæsning hentes dataen fra localStorage, ikke standardarrayet.

### **Målgrupper og brugsscenarier**

* **Privatpersoner**, der vil følge deres ernæringsindtag.
* **Personlige trænere eller diætister**, der tilpasser klienters kostplaner.
* **Undervisning**, hvor man lærer om næringsstoffer og datavisualisering.

### Interessant at udvide med (nice-to-have)

**Mulige trin fremover:**

* Tilføj login-funktionalitet så forskellige brugere kan gemme deres data.
* Gør det muligt at vælge forskellige kosttyper (f.eks. keto, vegetar).
* Integrer daglige kalorietal og mål (f.eks. “2000 kcal per dag”).

## CRUD

**Skab (Create):**

* Når diagrammet tegnes for første gang med renderChart(data), bliver SVG-elementerne (søjler, labels osv.) oprettet dynamisk.
* Når brugeren ændrer en værdi i grafen, gemmes den nye version af data i localStorage. Det opretter en ny lokal kopi, hvis den ikke allerede eksisterer

**Læs (Read):**

* Data bliver læst fra data-arrayet for at tegne diagrammet.
* Data bliver også læst fra localStorage ved hjælp af loadDataFromLocalStorage().

**Opdatér (Update):**

* Brugeren kan opdatere dataen i diagrammet ved at klikke på en søjle, hvilket udløser on('click')-eventet. De opdaterede data bruges derefter til at tegne diagrammet igen.
* De opdaterede data gemmes i localStorage med saveDataToLocalStorage(data).

**Slet (Delete):**

* Det gamle diagram bliver slettet med d3.select('#chart').select('svg').remove() før det opdaterede diagram tegnes igen.

# Programopbygning

## DOM-træ

html

└── head

├── meta (charset)

├── meta (viewport)

├── title ("D3.js Project")

└── script (D3.js bibliotek)

└── body

├── h1 ("Din kostplan")

├── div#chart

│ └── svg (tilføjes dynamisk af JavaScript)

│ ├── g (gruppe for hver "serie")

│ │ ├── rect (én for hver næringskategori pr. dag)

│ │ └── text (label for kcal pr. del)

│ └── text.bottom-label (dagsnavne under søjlerne)

├── script (JavaScript-koden med logik + renderChart-funktion)

├── button (onclick: kostRåd vises i alert)

## Json storage

Der bliver brugt en JSON fil til at opbevare næringsværdierne

Et billede, der indeholder tekst, skærmbillede, software

Indhold genereret af kunstig intelligens kan være forkert.

## Anonyme funktioner

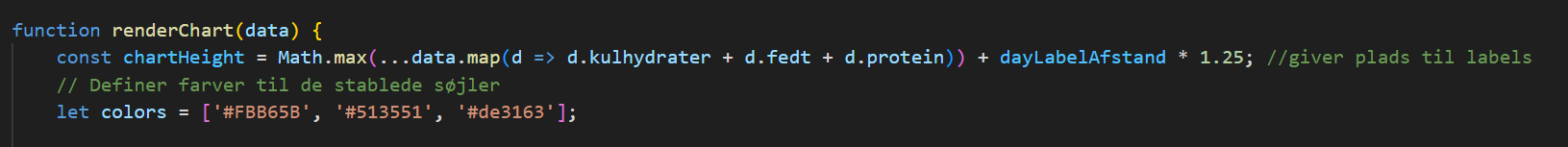
Et eksempel på en anonym funktion der er bliver brugt i programmet, er **function (d)** og **function (d,i)**. De bliver brugt til at referere arrayets indhold uden at bruge loops, sådan at ændringer påvirker de rette elementer.

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype

Indhold genereret af kunstig intelligens kan være forkert.

## Arrow syntax

Bliver brugt til at forkorte funktioner og hvis der kun er et statement og return, kan det forkortes til ugenkendelighed.



## Method chaining

Er når man (tro det eller ej) kæder metoder sammen, hvilket forbedrer læsbarheden yderligere og forkorter at man ikke skal lave en ny variable for hvert trin.

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype

Indhold genereret af kunstig intelligens kan være forkert.

## Event-håndtering

I stedet for konstant at køre noget kode for at tjekke om noget er sket, så kan man bare for det kodestykke til at selv at fortæller når noget er sket. Dette gør koden mere pålidelig og mindsker kraftigt proccesing kraft.

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype

Indhold genereret af kunstig intelligens kan være forkert.

## Indbyggede array-metoder

Det er de indbyggede metode til at manipulere arrayer der kommer med JavaScript. Fx her er .attr der ud fra en værdi i arrayet, ændre på parameteren for rektanglerne i grafen.

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype

Indhold genereret af kunstig intelligens kan være forkert.

## Data-binding

Er bare det at forbinde data til UI, fx med den indbyggede .join metode, der connector rektanglerne med deres korrespondere værdi i arrayet.

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype

Indhold genereret af kunstig intelligens kan være forkert.