



FILTRERING

• • •

Loop-view kontra single-view

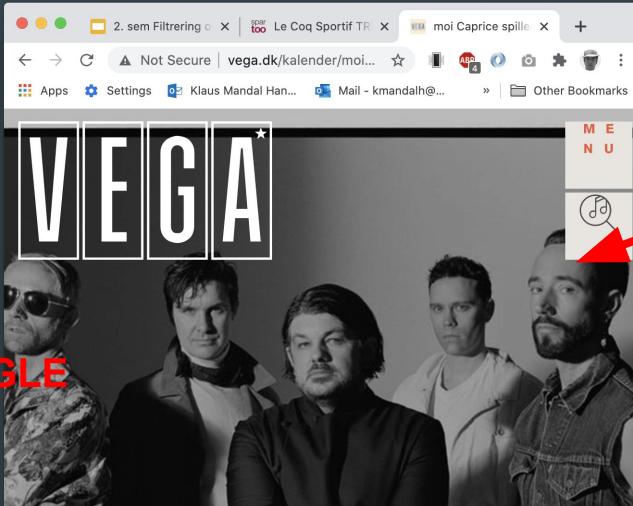
Loop-view er en oversigtsvisning af mange objekter
(en liste)

Single-view er en detaljeret visning af ét objekt

I dag arbejder vi
med loop-views

I morgen med
single-view

SINGLE



The browser window shows a navigation bar with tabs for 'Le Coq Sportif TR' and 'Koncertkalender'. Below the tabs are links for 'Apps', 'Settings', 'Mail', and 'Other Bookmarks'. The main content area displays a grid of concert listings:

- Alternativ pop**
Torsdag - Store VEGA
03.09.2020
MEKDES SUPPORT - JUWEHL
DREW SUPPORT - DOPHA
- Pop**
Fredag - Store VEGA
04.09.2020
JOY DIVISION
- Pop**
Fredag - Store VEGA
05.09.2020
MOI CAPRICE SUPPORT - ELOU ELAN
WIINSTON
- Alternativ pop**
Lørdag - Store VEGA
05.09.2020
Cookies?



Hvordan ser vores database ud?

persongalleri..html er et loop-view, der indeholder navn og billede fra restdb. Men i vores database på restdb er der også en kolonne der hedder troende. Troende kan have værdierne: *ja*, *nej* og *tvivler*.

Showing 1-11 of 11					
	fornavn	efternavn	fødselsdag	titel	troende
<input type="checkbox"/>	Jason	Walsh	07-06-2011	Et adipisci	nej
<input type="checkbox"/>	Shannon	Jaskolski	17-08-2006	Inventore sunt	ja
<input type="checkbox"/>	Santina	Kessler	22-01-1995	Vitae omnis nobis	tvivler
<input type="checkbox"/>	Danielle	Price	21-10-2014	Est accusamus et	nej
<input type="checkbox"/>	Quinn	McKenzie	18-02-2004	Vitae dolor	ja
<input type="checkbox"/>	Elias	Bechtelar	07-10-2005	A suscipit	ja
<input type="checkbox"/>	Wellington	Bins	29-06-1978	Id accusamus	tvivler
<input type="checkbox"/>	Citlalli	Smitham	08-06-1988	Rerum voluptatibus	nej
<input type="checkbox"/>	Orion	Reinger	27-03-2004	Voluptate dolore	tvivler
<input type="checkbox"/>	Melody	Kunde	02-02-2006	Tempora praesentium consectetur	nej
<input type="checkbox"/>	Verdie	Homenick	12-07-1979	quam	nej

troende
nej
ja
tvivler
nej
ja
tvivler
nej
ja
tvivler
nej
tvivler
nej
nej



Hvordan ser vores JSON ud?

The screenshot shows a browser's developer tools open to the 'Console' tab. A modal window is displayed with the text: "Vores troende property når vi logger JSON, her er den 'nej'" (Our 'troende' property when we log JSON, here it's 'nej'). Below the modal, the JavaScript code for loading JSON is visible:

```
async function loadJSON() {
  const JSONData = await fetch("https://persongalleri-5d3e.restdb.io/rest/persongalleri", {
    headers: myHeaders
  });
  personer = await JSONData.json();
  console.log("Personer", personer);
  visPersoner();
}
```

The JSON data object 'personer' is expanded to show its properties. One property, 'troende', is highlighted with a red box and has a red arrow pointing from the modal text to it. The value of 'troende' is 'nej'.

```
Personer
▼ (11) [{}]
  ▼ 0:
    ► billede: ["601b9d7fdc06ba22000364ca"]
      eternavn: "Walsh"
      fornavn: "Jason"
      fødselsdag: "2011-06-07T00:00:00.000Z"
      hobby: "cumque atque aperiam ad quos tempore odio"
      titel: "Et adipisci"
      troende: "nej" nej
      _id: "60116c1eef2e8a1b00036d3a"
      _mock: true
    ► __proto__: Object
  ▷ 1: {_id: "60116c1eef2e8a1b00036d39", eternavn: "Jaskolski", hobby: "est non qui omnis provident sed aspernatur", billed...}
  ▷ 2: {_id: "60116c1eef2e8a1b00036d3e", eternavn: "Kessler", hobby: "eligendi dolorem sed porro eveniet tenetur earum", bil...}
  ▷ 3: {_id: "60116c1eef2e8a1b00036d3b", eternavn: "Price", hobby: "qui ducimus autem cum nesciunt est ut", billede: Array(1...)}
  ▷ 4: {_id: "60116c1eef2e8a1b00036d3c", eternavn: "McKenzie", hobby: "deserunt dolorem quos quasi dolore quia cupiditate", ...}
  ▷ 5: {_id: "60116c1eef2e8a1b00036d3f", eternavn: "Bechtelar", hobby: "ea quasi rem doloremque facere ducimus aut", billede...}
  ▷ 6: {_id: "60116c1eef2e8a1b00036d3d", eternavn: "Bins", hobby: "error dolor illo consequuntur sit dolorem voluptates", bi...}
  ▷ 7: {_id: "60116c1eef2e8a1b00036d41", eternavn: "Reinger", hobby: "nihil quibusdam suscipit magni sed est est", billede: ...}
  ▷ 8: {_id: "60116c1eef2e8a1b00036d38", eternavn: "Kunde", hobby: "corrupti quisquam voluptatem sint quo enim molestiae", b...}
  ▷ 9: {_id: "60109134ef2e8a1b0002ec11", eternavn: "Homenick", hobby: "Qui quia dicta similiqe quidem molestias voluptas ip...", length: 11}
  ▷ __proto__: Array(0)
```



Loop-view med filtrering

Med udgangspunkt i *troende*-kolonnen skal vi lave en **filtrering** på siden, så vi ved at klikke på nogle radio-buttons har mulighed for kun at se troende, ikke troende og tvivlere, eller se alle:

Alle

All Troende Ikke troende Tvivler

Jason Walsh	Shannon Jaskolski	Santina Kessler	Danielle Price
Quinn McKenzie	Elias Bechtelar	Wellington Bins	Orion Reinger

Ikke troende

All Troende Ikke troende Tvivler

Jason Walsh	Danielle Price	Melody Kunde	Verdie Homenick
Citlalli Smitham			



Hvad er filtrering?

De enkelte objekter kan have egenskaber, hvor værdierne tilhører nogle få kategorier.

Disse kategorier kan man i loop-views **filtrere** på, så man kun ser objekter med en bestemt værdi.

Personerne i personlisten har egenskaben **troende**, som kan have værdierne **ja**, **nej** eller **tvivler**.

Derfor kan vi i loop-viewet filtrere på *troende*.



Hvad der skal gøres

1. **Scriptet skal kunne vise enten troende, ikke-troende eller tvivler.**
2. Der skal være filter-knapper til at vælge med.
3. Knapperne skal virke
4. Der skal være en overskrift, som viser det valgte
5. Den aktuelle knap skal være fremhævet.



Scriptet skal kunne vise enten troende, ikke-troende eller tvivler.

Vi kan indføre en **variabel** der hedder **filter**, som sættes til en af de tre værdier troende kan have fx. **ja**

I vise-funktionen vil vi nu kun vise objekter med denne værdi.

I `forEach`-loopen i vise-funktionen kan vi tjekke hvert objekt med `if`, om det har **troende == "ja"**. Vi viser så kun de troende.

The screenshot shows a web browser window with three tabs open, all pointing to the URL 127.0.0.1:50178/filtrering/index_filtrering_done.html. The browser interface includes standard controls like back, forward, and refresh, along with a search bar and a menu bar with 'Apps' and 'New folder'. Below the tabs, there are three cards, each containing a user's name, a small profile picture, and a background image. The first card for 'Shannon Jaskolski' shows a night scene at an outdoor ice rink. The second card for 'Quinn McKenzie' shows a woman smiling in a swimming pool. The third card for 'Elias B.' is mostly cut off but shows a person in a pool. The overall layout suggests a filtered list of users based on the 'troende' value set in the filter variable.



```
<script>
const header = document.querySelector("header h1");
const medieurl = "https://persongalleri-5d3e.restdb.io/media/";
const myHeaders = {
    "x-apikey": "600fe9211346a1524ff12e31"
}
document.addEventListener("DOMContentLoaded", start)
let personer;
let filter = "ja";
// første funktion der kaldes efter DOM er loaded
function start() {
    loadJSON();
}
async function loadJSON() {
    const JSONData = await fetch("https://persongalleri-5d3e.restdb.io/rest/persongalleri", {
        headers: myHeaders
    });
    personer = await JSONData.json();
    console.log("Personer", personer);
    visPersoner();
}
//funktion der viser personer i liste view
function visPersoner() {

    const dest = document.querySelector("#liste"); // container til articles med en person
    const skabelon = document.querySelector("template").content; // select indhold af html skabelon (article)
    personer.forEach(person => {
        // loop igennem json (personer)
        if (filter == person.troende) {
            const klon = skabelon.cloneNode(true);
            klon.querySelector(".navn").textContent = person.fornavn + " " + person.efternavn;
            klon.querySelector(".profil-billede").src = medieurl + person.billeder;
            dest.appendChild(klon);
        }
    })
}
</script>
```

Vi skal lave en global variabel, **personer**, der indeholder vores JSON

Vi skal lave en variabel, **filter**, der indeholder det vi vil filtrere på.
Vi sætter den til "ja" og sammenligner den med værdien af personen tro fra restdb



Øvelse 1

Gem en kopi af **02-personliste.html** fra i fredags i en ny undermappe til **tema7**-mappen, **filtreringSortering**. Omdøb filen til **01-troende.html**.

Lav om på scriptet i filen, så der kun vises troende.

The screenshot shows a web browser window with the URL `127.0.0.1:50178/filtrering/index_filtrering_slide_8.html`. The page displays three cards, each containing a person's name and a small image:

- Shannon Jaskolski**: An image of a person ice skating on an outdoor rink at night.
- Quinn McKenzie**: An image of a person wearing yellow sunglasses and a black bikini, sitting by a poolside.
- Elias Bechtelar**: An image of a person wearing yellow sunglasses and a black bikini, sitting by a poolside.



Hvad der skal gøres

1. Scriptet skal kunne vise enten troende, ikke-troende eller tvivler.
2. **Der skal være filter-knapper til at vælge med.**
3. Knapperne skal virke
4. Der skal være en overskrift, som viser det valgte
5. Den aktuelle knap skal være fremhævet.



2. Der skal være filter-knapper til at vælge med

Det skal være en menu med en knap for troende, ikke troende, tvivler og en knap for alle.

Man kan lave knapper med **button-tags** <button>...</button>

Knapperne skal have tilknyttet den værdi, som filter-variablen skal sættes lig med.

Til det kan vi bruge en **data-attribut**.

Vi skal fx have en **data-troende** attribut.

Den vil så kunne tilgås i javaScript med **dataset.troende**



Data attributten

```
<nav>
  <button data-troende="alle" class="valgt">Alle</button>
  <button data-troende="ja">Troende</button>
  <button data-troende="nej">Ikke troende</button>
  <button data-troende="tvivler">Tvivler</button>
</nav>
```

Læg mærke til **data-køn**.

data- er en ny attribut-type, som kan bruges til at tilknytte noget data til et element.
Værdien af attributten kan vi bruge i scriptet.



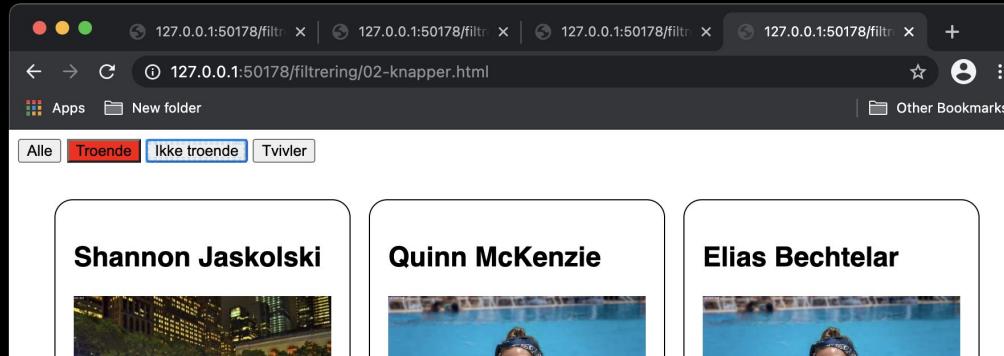
Øvelse 2: knapper til valg

Gem **01-troende..html** i en ny kopi, **02-knapper.html**

Tilføj knapper til filtrering.

Lav styling af knapperne, så de kommer til at stå centreret.

Lav data-attributter og tilføj en klasse, **valgt**, til Troende-knappen, og style den, så knappen træder tydeligt frem.





Hvad der skal gøres

1. Scriptet skal kunne vise enten troende, ikke-troende eller tvivler.
2. Der skal være filter-knapper til at vælge med.
3. **Knapperne skal virke**
4. Der skal være en overskrift, som viser det valgte
5. Den aktuelle knap skal være fremhævet.



3. Knapperne skal virke

1. I DOM'en er det **alle-knappen**, som skal have klassen valgt som udgangspunkt.
2. Variablen **filter** skal sættes til "**alle**", for alle personer skal vises først.
3. Vi skal lægge **eventListeners** på alle knapper - så der udføres en funktion ved klik.
4. **Eventhandler**-funktionen skal finde den **værdi**, der ligger i knappens **data-attribut**.
5. Denne **værdi** skal **filter-variablen** sættes lig med.
6. Så skal **visPersoner**-funktionen kaldes på ny.
7. **visPersoner** skal starte med at **fjerne det eksisterende indhold fra listen i html**.
8. Udbyg if-sætningen i **visPersoner** så den kan vise "**alle**"

```

<script>
const header = document.querySelector("header h1");
const medieurl = "https://persongalleri-5d3e.restdb.io/media/";
const myHeaders = {
  "x-apikey": "600fe9211346a1524ff12e31"
}
document.addEventListener("DOMContentLoaded", start)
let personer;
let filter = "alle";
// første funktion der kaldes efter DOM er loaded
function start() {
  const filterKnapper = document.querySelectorAll("nav button");
  filterKnapper.forEach(knap => knap.addEventListener("click", filtrerPersoner));
  loadJSON();
}
// eventlistener knyttet til knapperne der viger hvad for et filter der er aktivt
function filtrerPersoner() {
  filter = this.dataset.troende; // sæt variabel "filter" til værdien af data-troende på den knap der er klikket på
  visPersoner(); // kald funktionen visPersoner efter det nye filter er sat
}
async function loadJSON() {
  const JSONData = await fetch("https://persongalleri-5d3e.restdb.io/rest/persongalleri", {
    headers: myHeaders
  });
  personer = await JSONData.json();
  console.log("Personer", personer);
  visPersoner();
}
//funktion der viser personer i liste view
function visPersoner() {
  const dest = document.querySelector("#liste"); // container til articles med en person
  const skabelon = document.querySelector("template").content; // select indhold af html skabelon (article)
  dest.textContent = ""; // ryd container inden ny loop
  personer.forEach(person => {
    console.log("Troende", person.troende);
    // loop igennem json (personer)
    // tjek hvilken tro personen har og sammenligne med aktuelt filter eller vis alle, hvis filter har værdien "alle"
    if (filter == person.troende || filter == "alle") {
      const klon = skabelon.cloneNode(true);
      klon.querySelector(".navn").textContent = person.fornavn + " " + person.efternavn;
      klon.querySelector(".profil-billede").src = medieurl + person.billede;
      dest.appendChild(klon);
    }
  })
}
</script>

```

Eventlisteners på alle knapper



Eventhandler

Sæt filter-variablen lig med værdien af data-køn- attributten for den valgte knap. Når det er en data attribut kan den tilgås direkte i JS med dataset.

data-noget i html biver til **dataset.noget** i JS.

Kald visPersoner

Slet indholdet fra listen, altså det HTML element du appender template-kloner til

Udbyg if-sætningen:
Hvis filter er lig med valuen af person.troende (ja, nej tvivler) eller hvis filter er lig med "alle"



Øvelse 3: knapper til valg

Gem **02-knapper.html** i en ny kopi, **03-virker.html**

Få filtreringsknapperne til at virke!

The screenshot shows a browser window with six cards arranged in two rows of three. The first card, featuring Jason Walsh, has a red border around its title and photo, indicating it is the active selection. The other five cards are standard white cards.

Card 1 (Selected)	Card 2	Card 3
Jason Walsh 	Shannon Jaskolski 	Santina Kessler
Danielle Price 	Quinn McKenzie 	Elias Bechtelar

The screenshot shows the same browser window after applying a filter. Only three cards are now visible: Santina Kessler, Wellington Bins, and Orion Reinger. The cards for Jason Walsh, Shannon Jaskolski, Danielle Price, Quinn McKenzie, and Elias Bechtelar are no longer in view.

Card 1 (Selected)	Card 2	Card 3
Santina Kessler 	Wellington Bins 	Orion Reinger



Hvad der skal gøres

1. Scriptet skal kunne vise enten troende, ikke-troende eller tvivler.
2. Der skal være filter-knapper til at vælge med.
3. Knapperne skal virke
4. **Der skal være en overskrift, som viser det valgte**
5. Den aktuelle knap skal være fremhævet.

The screenshot shows a web browser window with two tabs open, both displaying the URL `127.0.0.1:50178/filtrering/index`. The active tab is titled `127.0.0.1:50178/filtrering/index_filtrering_done.html`. The page content is as follows:

Ikke troende

Alle **Troende** **Ikke troende** **Tvivler**

Jason Walsh	Danielle Price	Melody Kund
-------------	----------------	-------------

The word "Ikke troende" is displayed in large, bold black font above the filter buttons. The "Ikke troende" button is highlighted with a red border, indicating it is the selected filter. The other buttons ("Alle", "Troende", and "Tvivler") are in a standard grey font.



4. Det skal være en overskrift, som viser det valgte

Tilføj en **h1**-overskrift over filter-menuen med teksten “Alle”
Style den som du vil.

Ved knap-klik, skal **eventhandler-funktionen** sørge for, at overskriften bliver sat til at
være det samme, som det der står inde i knappen der er trykket på..



```
<h1>Alle</h1>
<nav>
  <button data-troende="alle" class="valgt">Alle</button>
  <button data-troende="ja">Troende</button>
  <button data-troende="nej">Ikke troende</button>
  <button data-troende="tvivler">Tvivler</button>
</nav>
const header = document.querySelector("header h1");

function filtrerPersoner() {
  filter = this.dataset.troende; // sæt variabel "filter" til
  visPersoner(); // kald funktionen visPersoner efter det nye
  header.textContent = this.textContent;
}
```

Tilføj h1-oeverskrift til at vise valgt filter

Når der klikkes på en knap, sørger eventhandler-funktionen for, at h1-overskriften indhold opdateres.

this henviser til den button der er klikket på...
evt. At *console.log(this)*



Øvelse 4: Overskrifter

Gem **03-virker.html** i en ny kopi, **04-overskrift.html**

Få overskrifter på filtreringerne:

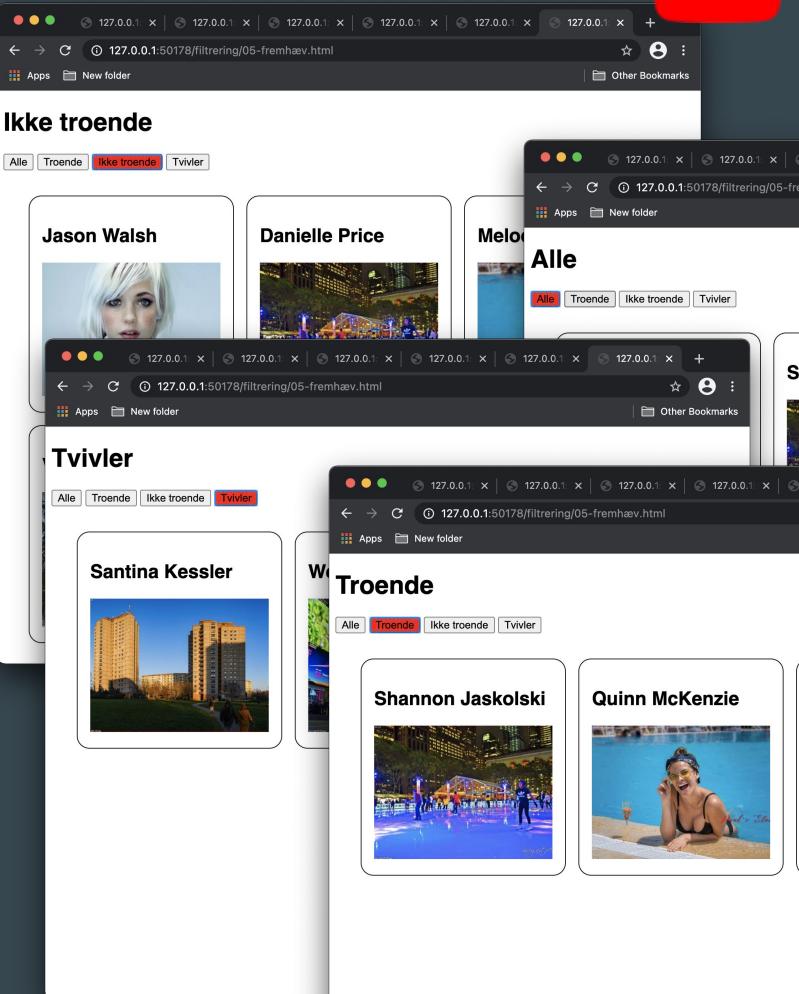
The image displays two side-by-side screenshots of a web browser window. Both screenshots show a URL bar with the address `127.0.0.1:50178/filtrering/04-overskrift.html`. The browser's title bar also shows multiple tabs, each with the same URL.

Left Screenshot (Alle tab): The title is "Alle". Below it is a navigation bar with four buttons: "Alle" (highlighted in blue), "Troende" (highlighted in red), "Ikke troende" (highlighted in blue), and "Tvivler". The main content area shows three cards with user profiles: Jason Walsh (woman with blonde hair), Shannon Jaskolski (ice skating rink), and Santina Kessler (city buildings).

Right Screenshot (Ikke troende tab): The title is "Ikke troende". Below it is a navigation bar with the same four buttons. The main content area shows three cards with user profiles: Jason Walsh (woman with blonde hair), Danielle Price (ice skating rink), and Melody Kunde (woman in a pool).

Hvad der skal gøres

1. Scriptet skal kunne vise et køn ad gangen.
2. Der skal være filter-knapper til at vælge med.
3. Knapperne skal virke
4. Der skal være en overskrift, som viser det valgt
5. **Den aktuelle knap skal være fremhævet.**





Den aktuelle knap skal være fremhævet

Ved at ændre udseende på den knap, der lige er klikket på, kan man gøre det nemmere for brugeren at se, hvilken filtrering, der er foretaget.

Når der klikkes på en knap, kan **evenhandler-funktionen**:

- **slette** .valgt klassen fra den valgte knap: `classList.remove()`
- **tilføje** .valgt klassen på den aktuelle knap: `classList.add()`



Fjern valgt og læg den på igen

```
function filtrerPersoner() {  
    filter = this.dataset.troende; // sæt variabel "filter" til værdien af data-troende på den knap  
    document.querySelector(".valgt").classList.remove("valgt"); // fjern klassen valgt fra den knap der  
    this.classList.add("valgt") // marker den knap der er klikket på  
    visPersoner(); // kald funktionen visPersoner efter det nye filter er sat  
    header.textContent = this.textContent;  
}  
// funktion loadJSON() {
```

Fjern klassen "valgt" fra den der var valgt

... og tilføj klassen "valgt" til den knap der er klikket på



Øvelse 5: Fremhæv aktuel knap

Gem **04-overskrift.html** i en ny kopi, **05-fremhæv.html**

Nu skal den valgte kategori også tydeliggøres med styling af den aktuelle knap

Husk at flytte class'en "valgt" fra ja-knappen til alle-knappen i HTML'en, hvis du ikke har gjort det endnu. Så passer det med at filter er sat til lig med "alle".

The diagram illustrates the state of a filter selection across four browser tabs:

- Tab 1 (Left):** Shows a grid of cards for all users. The "Alle" tab is selected, and the "Troende" tab is highlighted with a red border.
- Tab 2 (Middle Left):** Shows a grid of cards for users in the "Troende" category. The "Troende" tab is selected, and the "Alle" tab is highlighted with a red border.
- Tab 3 (Middle Right):** Shows a grid of cards for users not in the "Troende" category. The "Ikke troende" tab is selected, and the "Alle" tab is highlighted with a red border.
- Tab 4 (Right):** Shows a grid of cards for users in the "Tvivler" category. The "Tvivler" tab is selected, and the "Alle" tab is highlighted with a red border.