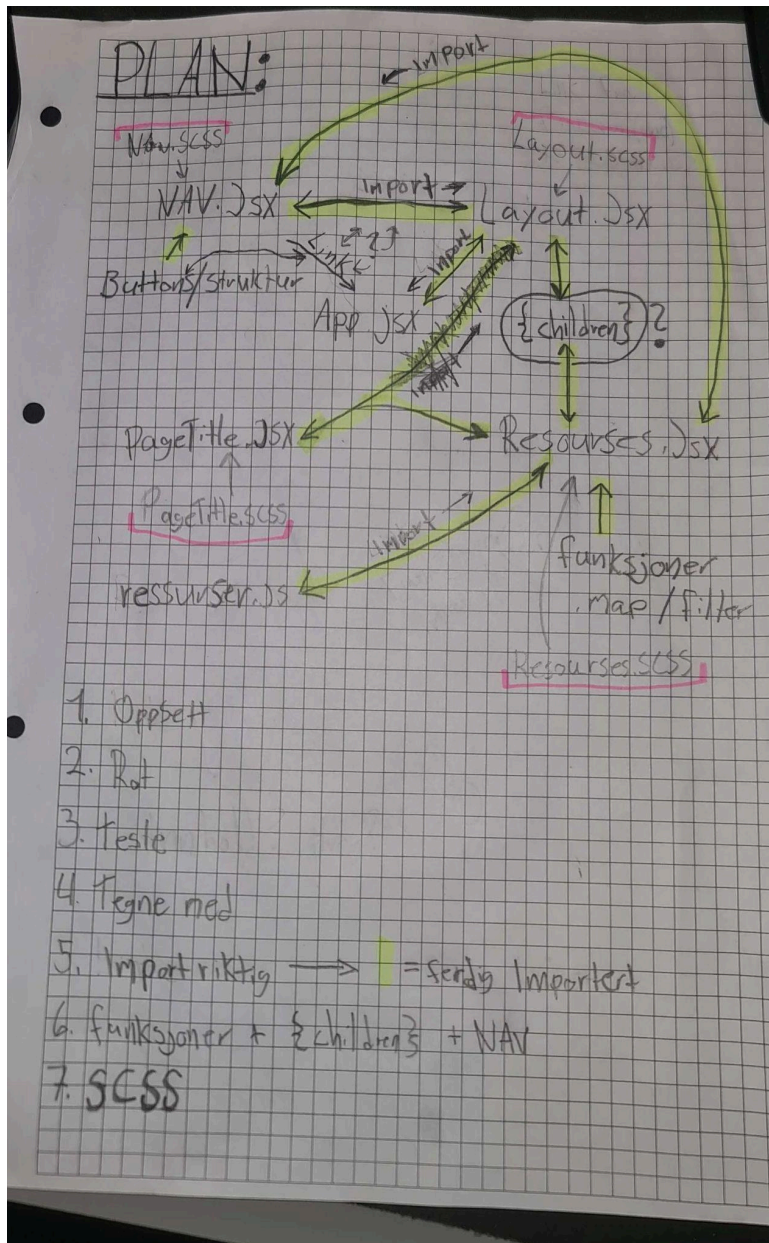


# Kilder

Hoved kilder:

- ChatGPT og
- VisualStudio-CoPilot.

Før jeg begynte, tegnet jeg en skisse i over hvordan det ca skulle se ut:



for oppgaven sier. Sett opp Routing til sidene /html, /css, /javascript, /react og /sanity, hvor hver av disse router til componenten "Resources", og sender med en prop til "Resources" med riktig ressurskategori ( eks: category="html" ). Bruk prop-en "category" og moderne javascript for å liste opp riktige ressurser fra ressurser.js. da er det vel best å bruke Link inni button?

Ja, for denne oppgaven er det best å bruke `<Link>` inni en `<button>`, siden du skal bruke routing og sende en prop ( `category` ) til `Resources`.

✓ Slik kan du gjøre det i `Nav.jsx`:

```
jsx
import { Link } from "react-router-dom";

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "sanity"];

  return (
    <nav>
      <h2>Kategorier</h2>
      {categories.map((cat) => (
        <button key={cat}>
          <Link to={`/${cat}`}>{cat.toUpperCase()}</Link>
        </button>
      ))}
    </nav>
  );
}
```

- Hva skjer her?
- Vi mapper gjennom `categories` og lager en `<button>` for hver.
- Inni knappen bruker vi `<Link>` til å endre URL-en (f.eks. `/html`, `/css`, osv).
- Når vi klikker på en knapp, navigerer vi til riktig rute.

✓ Slik setter du opp routing i `App.jsx`:

```
jsx
import React from 'react';
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import Layout from './components/Layout';
import Nav from './components/Nav';
import Resources from './components/Resources';
import './App.css';
import './style/style.scss';

function App() {
  return (
    <Router>
      <Layout>
```

```
function App() {
  return (
    <Router>
      <Layout>
        <Nav />
        <Routes>
          <Route path="/html" element=<Resources category="html" /> />
          <Route path="/css" element=<Resources category="css" /> />
          <Route path="/javascript" element=<Resources category="javascript" /> />
          <Route path="/react" element=<Resources category="react" /> />
          <Route path="/sanity" element=<Resources category="sanity" /> />
        </Routes>
      </Layout>
    </Router>
  );
}
```

- Hva skjer her?
- `Nav` har knapper som linker til `/html`, `/css`, osv.
- Når vi klikker en knapp, navigerer vi til riktig `Route`, og `Resources`-komponenten får riktig prop (`category`).

✓ Slik filtrerer du ressurser i `Resources.jsx`:

```
jsx
import React from 'react';
import { resources } from '../resources/ressurser';

export default function Resources({ category }) {
  const filteredResources = resources.filter(resource => resource.category === category);

  return (
    <>
      <h2>Ressurser for {category.toUpperCase()}</h2>
      <ul>
        {filteredResources.map((resource) => (
          <li key={resource.id}>
            <a href={resource.url} target="_blank">{resource.title}</a>
          </li>
        ))}
      </ul>
    </>
  );
}
```

- Hva skjer her?
- Vi filtrerer `resources` basert på `category`.
- Vi viser bare ressurser for den valgte kategorien.

🎯 Sluttresultat:

- ✓ Når du klikker på HTML-knappen, går du til `/html`, og `Resources` viser HTML-ressurser.
- ✓ Når du klikker på CSS-knappen, går du til `/css`, og `Resources` viser CSS-ressurser.
- ✓ osv.

🎯 Dette er den beste løsningen for oppgaven, fordi du bruker routing riktig!

Kode før og etter

Rød=før

Grønn=etter

```
uin25ak2_resourceapp_brukernavn/src/components/Nav.jsx

... @@ -1,18 +1,17 @@
1 - import React from 'react';
1 + import { Link } from "react-router-dom";
2
3 - export default function Nav({ setCategory }) {
4 -   const categories = ["html", "css", "javascript", "react", "sanity"];
3 + export default function Nav() {
4 +   const categories = ["html", "css", "javascript", "react", "headless-cms"];
5
6   return (
7     <nav>
8     <h2>Kategorier</h2>
9     {categories.map((cat) => (
10      <button key={cat} onClick={() => setCategory(cat)}>
11        {cat.toUpperCase()}
12      </button>
13    ))}
14    </nav>
7 +   <nav>
8 +     <h2>Kategorier</h2>
9 +     {categories.map((cat) => (
10 +       <button key={cat}>
11 +         /* Direkte kopiert fra ChatGPT: */
12 +         <Link to={`/${cat}`}>{cat.toUpperCase()}</Link>
13 +       </button>
14 +     ))}
15 +     </nav>
15 16   );
16 -
17 -
18 17 }
```

Her også før og etter:

```

  uin25ak2_resourceapp_brukernavn/src/components/Resources.jsx
  ... @@ -1,23 +1,19 @@
1  1  import React from 'react';
2  2  import { resources } from '../resources/ressurser';
3  - import Layout from './Layout';
4  -
5  3
6  4  export default function Resources({ category }) {
7  -
8  5      const filteredResources = resources.filter(resource => resource.category === category);
9  6  +
10  7      return (
11  8          <>
12  -          <h2>Ressurser for {category.toUpperCase()}</h2>
13  9  +          <h2>Ressurser for {category.toUpperCase()}</h2>
14  10         <ul>
15  11             {filteredResources.map((resource) => (
16  12                 <li key={resource.id}>
17  -                 <Layout>
18  -                     <a href={resource.url} target="_blank">{resource.title}</a>
19  -                 </Layout>
20  13 +                 <a href={resource.url} target="_blank">{resource.title}</a>
21  14                 </li>
22  15             ))}
23  16         </ul>
24  17     </>
25  18 );
26  19 }

```

```

lin25ak2_resourceapp_buknevavn > src > components > Resources.jsx > filteredResources.map() callback
1  import React from 'react';
2  import { resources } from '../resources/ressurser';
3
4  export default function Resources({ category }) {
5      const filteredResources = resources.filter(resource => resource.category === category);
6
7      return (
8          <>
9              <ul>
10                 {filteredResources.map((resource) => (
11                     <li key={resource.id} // resource.title>
12                     <a href={resource.url} target="_blank">{resource.title}</a>
13                     </li>
14                 ))}
15             </ul>
16         </>
17     );
18 }
19

```

CoPilot kom også med et auto input ( `|| resource.title` )  
Kopierte det inn

Nytt spørsmål til ChatGPT, kopierte en god del her

lettere metode?

Ja, her er en enda enklere metode:

♦ Superenkel React-kode

jsx

Kopier

Rediger

```
import { useState } from "react";
import { Link } from "react-router-dom";
import "../Nav.scss"; // Importer SCSS

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];
  const [active, setActive] = useState(null);

  return (
    <>
      {categories.map((cat) => (
        <link to={`/${cat}`} key={cat}>
          <button
            onClick={() => setActive(cat)}
            className={active === cat ? "active" : ""}
          >
            {cat}
          </button>
        </link>
      ))}
    </>
  );
}
```

♦ SCSS (Nav.scss)

scss

Kopier

Rediger

```
button {
  background-color: var(--whiteE);
  color: black;
  border: none;
  padding: 10px 20px;
  cursor: pointer;
  transition: background-color 0.3s;

  &.active {
    background-color: var(--blue2);
    color: white;
  }
}
```

✓ Hvorfor er dette den letteste metoden?

1. Bruker kun én state ( active ) i stedet for et objekt.

2. Ingen kompleks logikk – bare sjekker om active === cat .

3. Bruker SCSS riktig med .active -klasse.

4. Minimal kode og lett å forstå.

↓

4. Minimal kode og lett å forstå.

Perfekt balanse mellom enkelhet og funksjonalitet! 🎯 ⚡

🗑

👍

👎

🔍

🔗

🔄

📄

Kode før:

```
import { Link } from "react-router-dom";

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];
  const [buttonColor, setButtonColor] = useState("var(--whiteE)");

  const setNewButtonColor = () => {
    const newColor = buttonColor === "var(--whiteE)" ? "var(--blue2)" : "var(--whiteE)";
    setButtonColor(newColor);
  }

  return (
    <>
      {categories.map((cat) => (
        <Link to={`/${cat}`} key={cat}>
          <button onClick={setNewButtonColor}
            className="NavButtons">{cat}</button>
        </Link>
      ))}
    </>
  );
}
```

Kode etter:

```
import { useState } from "react";
import { Link } from "react-router-dom";
import "./Nav.scss"; // Importer SCSS

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];
  const [active, setActive] = useState(null);

  return (
    <>
      {categories.map((cat) => (
        <Link to={`/${cat}`} key={cat}>
          <button
            onClick={() => setActive(cat)}
            className={active === cat ? "active" : ""}
          >
            {cat}
          </button>
        </Link>
      ))}
    </>
  );
}
```

Spurte ChatGPT igjen heller om å gjøre det med min egen SCSS:

```
0 fikst det med min egen scss heller: .NavButtons{
padding-top: 25px;
padding-left: 6rem;
padding-right: 6rem;
padding-bottom: 25px;
background-color: var(--blue2);
color: var(--whiteF);
border: none;
cursor: pointer;
font-size: 14px;
font-weight: bold;
font-family: "roboto", sans-serif;
border-radius: 0;
text-transform: uppercase;
}
.active{
background-color: var(--blue5);
}
```

Her er den oppdaterte React-koden som fungerer med din eksisterende SCSS:

## ✓ Oppdatert React-komponent

```
jsx
import { useState } from "react";
import { Link } from "react-router-dom";
import "../Nav.scss"; // Importerer din SCSS-fil

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];
  const [active, setActive] = useState(null);

  return (
    <>
      {categories.map((cat) => (
        <Link to={`/${cat}`} key={cat}>
          <button
            onClick={() => setActive(cat)}
            className={`NavButtons ${active === cat ? "active" : ""}`}
            >
              {cat}
            </button>
          </Link>
        )
      )}
    </>
  )
}
```

## ✓ SCSS ( Nav.scss )

```
scss
.NavButtons {
padding-top: 25px;
padding-left: 6rem;
padding-right: 6rem;
padding-bottom: 25px;
background-color: var(--blue2);
color: var(--whiteF);
border: none;
cursor: pointer;
font-size: 14px;
font-weight: bold;
font-family: "roboto", sans-serif;
border-radius: 0;
text-transform: uppercase;
transition: background-color 0.3s ease; // Smooth overgang
}

.NavButtons.active {
background-color: var(--blue5);
}
```

## ✓ Hva er fikset?

1. Bruker din eksisterende `.NavButtons` -klasse uten å endre styling.
2. Legger til `.active` -klassen riktig for å endre bakgrunnsfargen.
3. Bruker en enkel `useState` for å holde styr på aktiv knapp.

3. Bruker en enkel `useState` for å holde styr på aktiv knapp.

4. Smooth fargeovergang lagt til med `transition: background-color 0.3s ease;`

🔥 Nå vil riktig knapp endre farge når den klikkes, og resten beholder standardstilen! 🚀

📄 🌟 🗨️ 🔊 🔄 🔄

Koden min før:

```
import { useState } from "react";
import { Link } from "react-router-dom";
import "./Nav.scss"; // Importer SCSS

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];
  const [active, setActive] = useState(null);

  return (
    <>
      {categories.map((cat) => (
        <Link to={`/${cat}`} key={cat}>
          <button
            onClick={() => setActive(cat)}
            className={active === cat ? "active" : ""}
          >
            {cat}
          </button>
        </Link>
      ))}
    </>
  );
}
```

Koden min etter:

```
import { useState } from "react";
import { Link } from "react-router-dom";
import "./Nav.scss";

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];
  const [active, setActive] = useState(null);

  return (
    <>
      {categories.map((cat) => (
        <Link to={`/${cat}`} key={cat}>
          <button
            onClick={() => setActive(cat)}
            className={`NavButtons ${active === cat ? "active" : ""}`}
          >
            {cat}
          </button>
        </Link>
      ))}
    </>
  );
}
```



Kopierte også ( .NavButtons ) foran (.active )

```
.NavButtons.active{
  background-color: var(--whiteF);
  color: black;
}
```

Etterpå la jeg inn hele koden og spurte ChatGPT:

```
LAG det også slik at ikke knappen restarter når jeg oppdaterer
siden: export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-
cms"];
  const [active, setActive] = useState(null);

  return (
    <>
      {categories.map((cat) => (
        <Link to={`/${cat}`} key={cat}>
          <button onClick={() => setActive(cat)} className=
{NavButtons ${active === cat ? "active" : ""}}>{cat}
        </button>
      </Link>
      )
    )}
  )
};
}
```

For å beholde den aktive knappen selv etter en sideoppdatering, kan vi lagre `active` -tilstanden i `localStorage`. Dette gjør at nettleseren husker hvilken knapp som var aktiv.

### ✅ Oppdatert React-kode med `localStorage`

```
jsx
import { useState, useEffect } from "react";
import { Link } from "react-router-dom";
import "../Nav.scss"; // Importerer SCSS

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];

  // Hent aktiv knapp fra localStorage, eller sett til null
  const [active, setActive] = useState(() => localStorage.getItem("activeCategory") || null);

  // Når en knapp trykkes, oppdater localStorage
  const handleSetActive = (cat) => {
    setActive(cat);
    localStorage.setItem("activeCategory", cat);
  };
}
```

```
};

return (
  <>
    {categories.map((cat) => (
      <Link to={`/${cat}`} key={cat}>
        <button
          onClick={() => handleSetActive(cat)}
          className={NavButtons ${active === cat ? "active" : ""}}
        >
          {cat}
        </button>
      </Link>
    )
  )}
  </>
);
}
```

### ✅ Hva er forbedret?

1. Lagrer aktiv knapp i `localStorage` → Overlever sideoppdateringer.
2. Bruker `useState` med en init-funksjon `(() => localStorage.getItem(...))` for å hente lagret verdi.
3. Enkel funksjon `handleSetActive()` → Oppdaterer både state og `localStorage`.

Nå vil den aktive knappen forbli aktiv selv etter refresh! 🚀

📄 📌 🔍 🔄 🔄 🔄

Koden min før:

```
import { useState } from "react";
import { Link } from "react-router-dom";
import "../Nav.scss";

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];
  const [active, setActive] = useState(null);

  return (
    <>
      {categories.map((cat) => (
        <Link to={`/${cat}`} key={cat}>
          <button
            onClick={() => setActive(cat)}
            className={`NavButtons ${active === cat ? "active" : ""}`}
          >
            {cat}
          </button>
        </Link>
      ))}
    </>
  );
}
```

Koden min etter:

```
import React, { useState } from "react";
import { Link } from "react-router-dom";
import "../Style/Nav.scss";
export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];

  const [active, setActive] = useState(() => localStorage.getItem("activeCategory") || null);

  const handleSetActive = (cat) => {
    setActive(cat);
    localStorage.setItem("activeCategory", cat);
  };

  return (
    <>
      {categories.map((cat) => (
        <Link to={`/${cat}`} key={cat}>
          <button
            onClick={() => handleSetActive(cat)}
            className={`NavButtons ${active === cat ? "active" : ""}`}>{cat}
          </button>
        </Link>
      ))}
    </>
  );
}
```

La merke til at når jeg navigerer manuelt i URL'en, så ville ikke knappene følge etter, med riktig kategori.

Da var det tilbake til ChatGPT og spurte følgende:

hvorfor vil ikke localStorage oppdater/skifte seg når jeg skifter URL'en? slik at buttonen følger med? =

### Problem

localStorage oppdateres riktig når du klikker på en knapp, men når du skifter URL, oppdateres ikke active -tilstanden fordi den kun settes når komponenten først lastes.

Dette betyr at knappen ikke endrer seg automatisk basert på URL-en.

### Løsning: Bruk useEffect + useLocation fra React Router

Vi må:

1. Lytte på URL-enderinger med useLocation().
2. Oppdatere active -tilstanden basert på URL-en når brukeren navigerer.

### Oppdatert React-kode

```
import { useState, useEffect } from "react";
import { Link, useLocation } from "react-router-dom";
import "../Nav.scss"; // Importerer SCSS

export default function Nav() {
  const categories = ["html", "css", "javascript", "react", "headless-cms"];
  const location = useLocation(); // Henter nåværende URL
  const [active, setActive] = useState(null);

  // Oppdaterer aktiv knapp basert på URL når siden lastes eller URL-en endres
  useEffect(() => {
    const path = location.pathname.substring(1); // Fjerner "/"
    if (categories.includes(path)) {
      setActive(path);
      localStorage.setItem("activeCategory", path); // Oppdater localStorage
    }
  }, [location.pathname]); // Kjør når URL endres

  // Oppdaterer aktiv knapp når en knapp trykkes
  const handleSetActive = (cat) => {
    setActive(cat);
    localStorage.setItem("activeCategory", cat);
  };

  return (
    <>
      {categories.map((cat) => (
        <Link to={`/${cat}`} key={cat}>
          <button
            onClick={() => handleSetActive(cat)}
            className={`NavButtons ${active === cat ? "active" : ""}`}
          >
            {cat}
          </button>
        )
      )}
    </>
  );
}
```

```
{cat}
</button>
</link>
)))
</>
);
}
```

### Hvordan fungerer dette?

1. Når URL-en endres, f.eks. "/react", vil useEffect() kjøre og oppdatere active basert på URL-en.
2. localStorage oppdateres også slik at det forblir riktig selv etter refresh.
3. Knappen får riktig "active"-klasse basert på URL-en, selv om du navigerer direkte eller oppdaterer siden.

### Resultat

- Knapper følger URL-en riktig (selv om du navigerer med Link eller skriver inn en ny URL manuelt).
- Beholder aktiv knapp etter refresh fordi verdien lagres i localStorage.
- Automatisk synkronisering mellom URL og aktiv knapp 🎯 ✨

Nå skal alt fungere perfekt! 🎉

👍 🗨️ 🔍 🔄 ⌂

Kode før:

```
1 import React, { useState } from "react";
2 import { Link, useLocation } from "react-router-dom";
3 import "../Style/Nav.scss";
4 export default function Nav() {
5   const categories = ["html", "css", "javascript", "react", "headless-cms"];
6
7   const [active, setActive] = useState(() => localStorage.getItem("activeCategory") || null);
8
9   const handleSetActive = (cat) => {
10     setActive(cat);
11     localStorage.setItem("activeCategory", cat);
12   };
13
14   return (
15     <>
16       {categories.map((cat) => (
17         <Link to={`/${cat}`} key={cat}>
18           <button
19             onClick={() => handleSetActive(cat)}
20             className={`NavButtons ${active === cat ? "active" : ""}`}>{cat}
21           </button>
22         </Link>
23       ))}
24     </>
25   );
26 }
```

Kode etter:

Er direkte kopiert fra  
ChatGPT

```
1 import { useState, useEffect } from "react";
2 import { Link, useLocation } from "react-router-dom";
3 import '../style/Nav.scss'
4
5 export default function Nav() {
6   const categories = ["html", "css", "javascript", "react", "headless-cms"];
7   const location = useLocation();
8   const [active, setActive] = useState(null);
9
10   useEffect(() => {
11     const path = location.pathname.substring(1);
12     if (categories.includes(path)) {
13       setActive(path);
14       localStorage.setItem("activeCategory", path);
15     }
16   }, [location.pathname]);
17
18   const handleSetActive = (cat) => {
19     setActive(cat);
20     localStorage.setItem("activeCategory", cat);
21   };
22
23   return (
24     <>
25       {categories.map((cat) => (
26         <Link to={`/${cat}`} key={cat}>
27           <button
28             onClick={() => handleSetActive(cat)}
29             className={`NavButtons ${active === cat ? "active" : ""}`}>{cat}
30           </button>
31         </Link>
32       ))}
33     </>
34   );
35 }
36
```

Så også når jeg skulle kommentere koden at

```
const [category] = useState("html");
```

var egentlig ikke nødvendig å ha med, men da jeg fjernet den så forsvant hele innholdet

Men hvis jeg fjerner: //Bruker useState for å sette en standard-kategori, slik at det vises html ved innlasting av siden første gang, const [category] = useState("html"); da forsvinner alt innholdet

Det høres ut som at `category` brukes til å bestemme hvilke ressurser som skal vises i `Resources.jsx`. Når du fjerner `useState("html")`, kan det hende at `category` blir `undefined`, og derfor vises ingenting.

### Løsning 1: Sørg for at `category` alltid har en verdi

I stedet for å bruke `useState` i `App.jsx`, kan du gi `Resources` en standardkategori i `Route` for `"/"`:

#### App.jsx (oppdatert)

```
jsx
<Routes>
  <Route path="/" element={<Resources category="html" />} /> {/* Standardkategori = HTML */}
  <Route path="/html" element={<Resources category="html" />} />
  <Route path="/css" element={<Resources category="css" />} />
  <Route path="/javascript" element={<Resources category="javascript" />} />
  <Route path="/react" element={<Resources category="react" />} />
  <Route path="/headless-cms" element={<Resources category="headless-cms" />} />
</Routes>
```

Nå vil `/` alltid vise HTML-kategorien, uten at du trenger `useState`.

### Løsning 2: Bruke URL i stedet for `useState`

En mer fleksibel måte er å hente `category` direkte fra URL-en med `useParams()` fra `react-router-dom`.

#### Resources.jsx (oppdatert)

```
jsx
import React from 'react';
import { useParams } from 'react-router-dom';
import { resources } from '../resources/ressurser';

export default function Resources() {
  const { category } = useParams(); // Henter kategori fra URL
  const filteredResources = resources.filter(source => source.category === category);
```

```
const filteredResources = resources.filter(resource => resource.category === category);

return (
  <>
    <h2>Ressurser for {category ? category.toUpperCase() : "HTML"}</h2>
    <ul>
      {filteredResources.map((resource) => (
        <li key={resource.id}>
          <a href={resource.url} target="_blank" rel="noopener noreferrer">{resource.name}</a>
        </li>
      ))}
    </ul>
  </>
);
```

- Nå vil `category` bli hentet fra URL-en (`/html`, `/css`, etc.).
- Hvis `category` ikke finnes, kan du gi en fallback (`"HTML"`).

#### App.jsx (bruker URL-parametere)

```
jsx
<Routes>
  <Route path="/:category" element={<Resources />} />
</Routes>
```

- Nå vil `/html`, `/css` osv. automatisk vise riktig kategori uten at du trenger å sende `category` som en prop.

### Hvilken metode bør du bruke?

- ✓ Løsning 1 er enkel og fungerer fint, men krever at du manuelt definerer alle kategorier i `App.jsx`.
- ✓ Løsning 2 er mer fleksibel fordi den automatisk henter kategorien fra URL-en.

Begge løsningene fungerer – det kommer an på hva du foretrekker! 🍌

📄 🍌 🗨️ 🔊 🎨 🔄

Kode før:

```
function App() {
  const [category] = useState("html");

  return (
    <>
      <Router>
        <Layout>
          <Routes>
            <Route path="/" element={<Resources category={category} />} />
            <Route path="/html" element={<Resources category="html" />} />
            <Route path="/css" element={<Resources category="css" />} />
            <Route path="/javascript" element={<Resources category="javascript" />} />
            <Route path="/react" element={<Resources category="react" />} />
            <Route path="/headless-cms" element={<Resources category="headless-cms" />} />
          </Routes>
        </Layout>
      </Router>
    </>
  )
}
export default App
```

Kode etter:

```
function App() {

  return (
    <>
      <Router>
        <Layout>
          <Routes>
            <Route path="/" element={<Resources category="html" />} />
            <Route path="/html" element={<Resources category="html" />} />
            <Route path="/css" element={<Resources category="css" />} />
            <Route path="/javascript" element={<Resources category="javascript" />} />
            <Route path="/react" element={<Resources category="react" />} />
            <Route path="/headless-cms" element={<Resources category="headless-cms" />} />
          </Routes>
        </Layout>
      </Router>
    </>
  )
}
export default App
```