

Case 1 – Mellemrum

Når man trykker på “Kontakt os” i footeren, er det bare en mailto-løsning. Jeg erstattede den eksterne mailto-løsningen i footeren med en intern React-komponent. På den måde håndteres kontaktfunktionen nu som en del af applikationens eksisterende komponent- og routingstruktur i stedet for at være afhængig af browserens eller brugerens lokale mailklient.

Jeg opdelte CSS for at koden er nemmere at vedligeholde. Selvom én stor css godt kan fungere, så vil den blive sværere og sværere at finde rundt i, når projektet vokser. Det kan gøre det svært at finde den CSS, der hører til en bestemt komponent. Det hjælper med at undgå at styles påvirker andre komponenter ved et uheld. Gør det nemmere at samarbejde med andre på samme fil og slette en komponent uden at skulle lede efter dens CSS.

Jeg lavede fejl-beskeder og loading på relevante sider for at forbedre UI-states og fejlhåndtering ved data fetch. Eksempler: Ingen besked ved tilmelding, bruger ved ikke om de er tilmeldt eller ej. Ingen besked ved ugyldig mail-adresse, brugeren kan dermed skrive en ugyldig mail og stadig gå igennem. Ingen fejlmelding ved søgning af ikke-eksisterende events. Alt dette har jeg forbedret, så brugeroplevelsen bliver bedre og mere intuitiv.

Jeg har lavet en log-ind prototype i både navigationen og via “Se tilmeldinger” linket i footeren. Dette er for at sikre brugersikkerhed, da man ellers ville kunne tilgå de tilmeldte brugeres e-mails og fulde navn via tilmeldingsoversigten. Dette er blot en prototype og havde det været mere relevant, ville jeg lave en mere intuitivt og detaljeret løsning. Hvis jeg havde haft mere tid, ville jeg også implementere en database til brugernes profiler og administratorernes profiler, så siden kan bruges efter det behov, man har derinde. På den måde kan man også sikre at kun relevante administratorer kan se brugernes private informationer i tilmeldingsoversigterne.

I supabase har jeg oprettet en kolonne med eventId i mit registrations table. Det gjorde jeg, fordi jeg kan bruge den som foreign key til at trække data fra mit events table og dermed få alle nødvendige event-informationer ind i registrations uden at skulle gentage data i begge tables. Dette øger vedligeholdelsesvenligheden i koden og supabase.

Jeg har tilføjet en ScrollToTop.jsx component, som jeg har importeret i app.jsx og har på den måde fået alle sider til at starte i toppen, når de åbnes. Jeg lavede det som component for at undgå at skrive samme kode på alle separate sider. Det skaber en konsistent brugeroplevelse gennem hele websitet.

Jeg lavede en supabase.js component i services, fordi den ellers var gentaget på flere sider. Så nu er den importeret på de relevante sider og dermed er den langt nemmere at ændre på senere hen. Det gør koden nemmere at vedligeholde.

Lighthouse-ændringer, på eventpage har jeg sat fetchPriority=“high” på event.image og gjort størrelsen på siden større, når den loader, så der ikke kommer et pludseligt, stort layout-skift, når siden har loadet alt indholdet. I samme forbindelse lavede jeg loading="lazy" på nogle billeder, så browseren ikke skal loade alle billeder samlet med det samme men kan loade dem lidt ad gangen som de vises. Dette giver en mere stabil brugeroplevelse og reducerer det indledende dataforbrug.

Jeg tilføjede en placeholder i signup-panel, fordi der var en på mail men ikke en på navn og designet skal være konsistent.

Jeg lavede footer til en component, fordi den ellers også var lagt ind på hver separat side, hvilket gør det mere besværligt at vedligeholde den del af koden og det giver meget ekstra arbejde, når/hvis man laver flere nye filer/sider.

Via min handleSubmit function i EventPage.jsx har jeg brugt en if-funktion til at fortælle browseren, at hvis der allerede eksisterer en tilmelding med samme eventId og e-mail, så skal den ikke lade tilmeldingen gå igennem. Hvis den ikke findes, så kan den lade tilmeldingen gå igennem. Jeg bruger også const normalizedEmail for at sikre, at samme e-mail med store eller små bogstaver ikke kan tilmeldes samme event flere gange. Det forbedrer datakvaliteten og tilmeldingsflowet mere pålideligt.

Jeg har fikset navigationen til 404-siden, så brugeren sendes til den side, når de søger efter et event som ikke eksisterer i databasen. Før kunne man fx søge på /events/99999 i URL'en og så troede browseren, at man stadig skulle sendes til eventpage selv om eventet ikke eksisterer. Hvis man leder efter et event nu som ikke ligger i databasen, så sendes man til 404-siden. Dette gør løsningen mere robust, brugervenlig og intuitiv.