SystemDesign_Webapp_gruppe14_Matbuddy - Oppgave 3.

Systemdesignet skal ta for seg følgende:

- Hvilken teknisk oppdeling/oppbygging skal applikasjonens deres skal ha?
  Frontend: React19, react router, TypeScript, React Query/SWR, Tailwind CSS

  Backend: Node.js + Express, alternativt RedwoodJS(Redwood SDK)

  Database: PostgreSQL(supabase eller dens egen hosting)

  ORM: Drizzle ORM

  Autentisering: JWT/Supabase/Auth0

  Lagring: Cloudinary/AWS S3

  Deploy: Vercel(frontend), Render/Heroku(backend), Supabase Postgre

  Observability: Sentry, logging via Loggly/Datadog(valgfritt)

  - ***Hvordan skal data lagres/hentes og håndteres?:***

    Lagring:

    - Data lagres i PostgreSQL i tabeller.

    Henting av data:

    - Brukeren gjør en handling i frontend (react), som sender en forespørsel til backend via rest/graphql. Backend bruker Drizzle til å hente/oppdatere data i databasen.

    Kobling:

    Database til backend – Drizzle

    Backend til frontend REST eller GraphQL

  - ***Er det behov for brukerhåndtering?:***

    Brukere må registrere seg og logge inn for å kunne lagre favoritter, lage og dele handlelister, kommentere og laste opp egne oppskrifter. Vi planlegger minst to roller user og admin. Admin kan redigere innhold og administrere oppskrifter.

  - ***Hvordan skal datamodellen se ut?:***

For at applikasjonen skal kunne administrere brukere, handlelister, oppskrifter og kommentarer, trengs en relasjonell datamodell. Databasen vi bruker er **MySQL**, og backend og databasen er koblet sammen via Drizzle ORM.

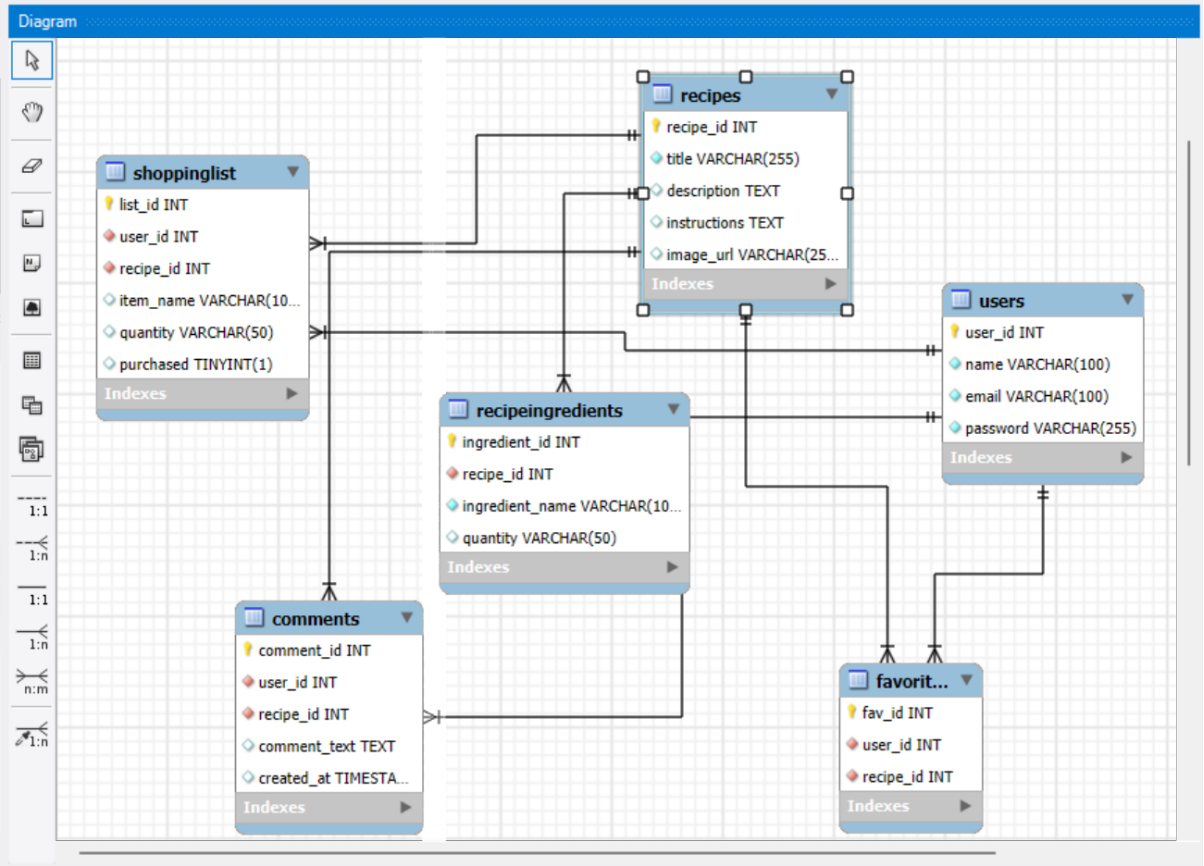
Hoved-tabellen

1. Users- lagrer informasjon om brukere(navn,e-post, passord e.t.c)
2. Oppskrifter- lagrer oppskrifter med navn, beskrivelse, ingredienser, steg for steg instruksjoner og bilder
3. Ingredienser – lagrer ingrediensene og mengden som trenges for hver oppskrift
4. Comment- lagrer det for folk skriver som kommentarer på oppskrifter
5. Favoritter – Det holder styr på hvilke oppskrifter brukerne liker best
6. Handleliste- lagrer ting folk legger i handlelisten og om de allerede er kjøpt eller ikke

Relasjoner:

- En bruker kan ha mange oppskrifter, kommentarer, favoritter og handlelister
- En oppskrifter kan ha mange ingredienser og kommentarer
- Favoritter og handleliste-elementer kobler bruker og oppskrifter sammen

For at appen effektivt skal kunne håndtere kommentarer, registrere favoritter, generere handlelister og vise oppskrifter, ordner datamodellen all informasjonen.



Hvilke hoveddeler består applikasjonen av, og hvordan skal disse fungere sammen?:

- ***Oppdater eller lag nye skjermbilder:***

  Logg inn/Register

  Oppskriftsfeed med søk og filter

  Oppskriftsdetalj med ingredienser, fremgangsmåte, knapp der det står "legg til i handleliste"

  Handleliste med avkryssing, deling og eksport

  Profilside med favoritter, og egne oppskrifter

  Admin-Dashboard

- ***Hvordan fungerer disse sammen/navigasjonen mellom de skje?:***

  Logg inn/Registrer > Feed

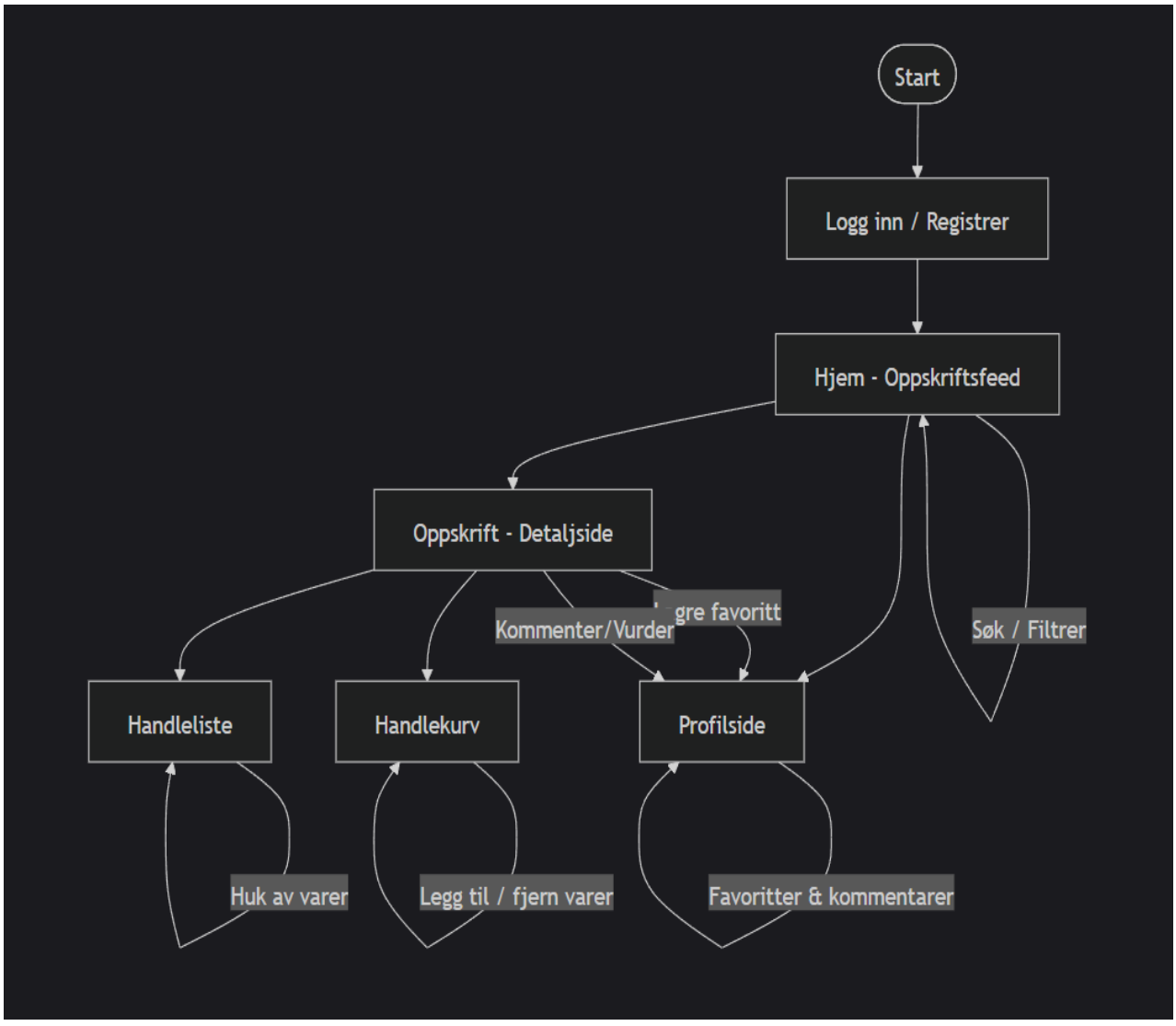
  Feed > Oppskriftsdetalj > Legg til handleliste

  Feed > Profil (Favoritter, egne oppskirfter)

  Profil > Admin-Dashboard (Kun Admin)

  Profil > Handleliste

- ***Det er lurt å tegne ned flyten i applikasjonen. F.eks. med flytdiagram (lite eksempel her).:***



- Henvis til aktuell dokumentasjon der det er hensiktsmessig
- etc.

- ***Hvilke ressurser (bilder/lyd etc.) skal dere lage selv, og hvilke skal dere eventuelt hente eksternt?:***

  Lage selv:

  Logo og Visuelle Profiler (Farger og Fonter)

  Enkle Skisser (Lo-fi og Hi-fi)

  Tekstinnhold: Tekster og Korte beskrivelser av funksjoner

  Enkle illustrasjoner knyttet til MatBuddy sitt tema f.eks. små ikoner

  Hentes eksternt:

  Ikoner fra etablerte bibloteker

  Oppskriftsbilder: Bruker generert innhold fra åpne API-er

  Matbilder fra gratis bildebanker om det er nødvendig

- ***Hva slags funksjoner / klasser / komponenter dere må lage og hva skal de gjøre?:***

  I frontend skal vi lage komponenter for de ulike og funksjonene i webappen. App-komponenten skal fungere som en hovedinngang til og hjelpe oss å håndtere navigasjonen mellom sidene. AuthProvider skal holde styr på innloggingsstatusen og sørger for at brukeren forblir autentisert. For oppskrifter, så lages en RecipeList som skal vise til flere oppskrifter, også en RecipeCard som viser en enkel oppskrift i et lite format. RecipeDetail viser til en hel oppskrift med ingredienser, instruksjoner, kommentarer og mulighet til å legge ingredienser til handleliste.

  For backend lages det funksjoner og klasser organisert inne på controller, service og respositories. AuthController skal håndtere registrering, innlogging og utlogging. RecipeController skal opprette, oppdatere og uthente oppskrifter. ShoppingListController jobb håndterer lagring og uthenting av handlelister. Servicen skal samle ingredienser fra flere oppskrifter til en handleliste.

Respository-klasser har ansvar for å hente og lagre data fra databasen via Drizzle -Orm. I tillegg så lages middleware for autentisering, validering av data og feilhåndtering.

I databasen vil det defineres Schema-klasser for brukere, oppskrift, ingredienser, handlelister, kommentarer og vurderinger. Disse bruker vi sammen med Drizzle Orm for å generere dataflyt og holde strukturen i databasen oppdatert.

- ***Oversikt over hvilke eksterne biblioteker / API'er dere skal benytte:***

Frontend: React 19, Typescript, Tailwind CSS, React Router, React Query.

Backend: Node.js/Express eller Redwood SDK.

Orm: Drizzle ORM.

Autentisering: Supabase

Lagring: Cloudinary eller AWS S3.

API: https://www.themealdb.com/

Dette api-et støtter funksjonaliteten vi trenger. Man kan bland annet søke opp retter basert på navn eller hoved-ingrediens (søk på mer enn en ingrediens er premium og koster penger). Det tilbyr bilder av både rett og ingredienser som behøves. Man kan også filtrere basert på kategori (sjømat, osv.) og område maten opprinnelig kommer fra.

- ***Hvilke funksjoner/deler av applikasjonen er dere usikre på hvordan skal lages/best løses (hva tror dere at dere vil trenge mest tid og hjelp til?):***

En del som virker krevende ut er autentisering og sikkerhet. Det er svært viktig å kunne opprettholde personelle og kritiske informasjoner lagret slik at brukerne kan forbli pålogget på en sikker måte, så riktige tokens og sesjoner kan være komplisert å implementere.

En annen del som vi antar er krevende er å lage et godt søk og filtrering av oppskrifter. Det er lett å starte med enkle søk, men det er vanskelig å utvide søk

til noe mer avanserte, som vil ta hensyn til flere filtre samtidig, f.eks: vegetar, prisnivå og tid.

Vi har kun jobbet med SQL databaser isolert i MYSQL tidligere, og har aldri brukt dette opp mot en applikasjon før. Det å finne “best practise” for integrasjon og bruk av database (drizzle i vårt tilfelle), vil være noe vi må bruke tid på å finne ut av.

Kilder:

API: https://www.themealdb.com/

Flowchart: https://mermaid.js.org/

MySQL Workbench Manual: https://dev.mysql.com/doc/workbench/en/