

Fagprøve IT-utviklerfaget

Kandidatens navn: Martin Kulvedrøsten Myhre

Lærebedrift: Inmeta Consulting AS

Sted og tid for prøven: 21-29 mai 2024, Inmeta hamarkontoret (parkgata 36, Hamar 2317)

Kontaktinfo prøvenemnda

Leder	Per Arne,
Medlem	Audun

<i>Oppgaven</i>	4
1.1 Den lokale matguiden	4
1.2 Registrering/innlogging	4
1.3 Opprette spisesteder	4
1.4 Legge inn anmeldelse	4
1.5 Se anmeldelser	4
1.6 Generelle krav til applikasjonen	4
2. Arbeidsform, prosess og generelt	5
2.1 Prosjektløpet	5
2.2 Generelt	5
<i>Planlegging</i>	5
Min tolkning av oppgaven	5
2. Krav til applikasjonen	6
3. Generelt	6
4. Generelle tanker om oppgaven	6
Techstack	7
Frontend	7
Back-end	7
Database	8
Devops og verktøy/applikasjoner	8
Autentisering	8
Arkitektur	10
1. Frontend	10
2. Backend	11
Database struktur	11
Restaurants Table (Restauranter)	11
Reviews Table (Anmeldelser)	12
Relasjoner	12
Design & UI Biblioteker	13
Dag-til-dag-plan	13
Onsdag og torsdag morgen	14
Torsdag etter lunsj	14
Fredag	14
Fredag+ / Helgen	14
Mandag	15
Tirsdag	15
Onsdag	16
<i>Gjennomføring</i>	16
Dag 1 (Tirsdag) - Planlegging	16
Dag 2 (onsdag) – Planlegging	16
Dag 3 (torsdag) – Planlegging og oppsett	17
Dag 4 (fredag) – Litt av hvert	18
Dag til dag plan	18
Arkitekturoversikt	18
Oppsett enkel: Database oppsett & Struktur	19
Microsoft Identity	20
Dag 5 og 6 (Fredag+ /helg)	21
Oppsett avansert: Services og controllers	21
Authentication for services og controllers	21
Next-auth og auth sider	21

Opprette restauranter	21
List og hent restauranter	22
List ut og opprett anmeldelser	22
Paginerings	22
Tilleggsendringer	22
Dag 7 (Mandag) – tredjepart & nytt design	23
Oauth2	23
Nytt design	23
Dag 8 (Tirsdag) – Dokumentere å fikse små feil	23
<i>Egenvurdering</i>	24
Planleggingen	24
Gjennomføringen	24
Design/visuelt	24
Læring	25
Hva ville jeg gjort annerledes?	25
Planlegging	25
Gjennomføringen	25
Plan for videre utvikling (hvis jeg hadde hatt mer tid)	26
1. Søkefunksjon	26
2. Filterfunksjon	26
3. Chatbot	26
4. Anonymisering av brukere	26
5. Refresh Tokens	26
Læremål	26
Ethiske vurderinger og lovverk	26
Sikkerhet og personvern	27
Design, interaksjon og brukerdialog	27
Demokrati og medborgerskap	27
Bærekraftig utvikling	27
Dokumentasjonen	27

Oppgaven

1.1 Den lokale matguiden

I takt med at Innlandetregionen blir et mer urbant sted popper det opp restauranter over en lav sko, spesielt i Innlandets hjerte på Hamar, men i det hele tatt er det i 2024 mulig å få i seg en god og ikke så god matbit overalt i regionen.

Din oppgave er å utvikle en enkel webapplikasjon hvor brukere kan legge inn og se anmeldelser av lokale restauranter, spisesteder og barer som tilbyr en form for servering. Applikasjonen skal være tilgjengelig fra internett, enkel å bruke og skal demonstrere grunnleggende funksjonalitet som kreves i moderne webapplikasjoner. Prøv å sette deg inn i en en anmelder eller en som skal lese en anmeldelse sine sko. Hvor og når gjør de det? Hvordan kan denne lettest gjøres tilgjengelig og spiselig for sluttbrukeren?

1.2 Registrering/innlogging

Det skal være mulig å logge inn på løsningen, og brukeren skal kunne opprette en konto og logge inn. Dette kan løses på flere måter, drøft gjerne litt vurderingene på hvordan du velger å løse dette.

1.3 Opprette spisesteder

For å kunne anmelde et spisested må disse ligge lagret et sted som anmeldelser kan knyttes opp mot. Denne delen av oppgven kan løses etter ønske på enklest vis, og det er ikke forventet at det skal være tilgjengelig et brukergrensesnitt for å opprette disse, men tek på hvordan informasjon og metadat som kan være nyttig å ha registrert på et spisested for å gi en best mulig brukeropplevelse for sluttbrukere av systemet.

1.4 Legge inn anmeldelse

Autentiserte brukere skal kunne legge inn en anmeldelse. Du bestemmer selv komponentene som inngår i en anmeldelse (rating, fritekst, betjening, service mv.). Dette skal vær et grensesnitt som er tilgjengelig fra internett.

1.5 Se anmeldelser

Alle besøkende av tjenesten skal kunne navigere seg i spisesteder og tilknyttede anmeldelser. I denne deloppgaven står du fritt til å velge teknologi. Her er det viktig å tekne brukerreisen, brukeropplevelse og tilgjengelighet. Som eier av applikasjonen tenker ide-haveren at dette er økkelen for å lykkes med tjenesten og få adopsjon i markedet. Her teller tanker rundt arkitektur og buisness case inn på dine valg.

1.6 Generelle krav til applikasjonen

Applikasjonen skal være responsiv og gi en god brukeropplevelse på tvers av ulike enheter. Tenk ellers over målkravene for faget. Pass på å ha en god og vedlikeholdbar kode. Du står fritt til å velge utviklerværktøy og utviklerspråk. Bruk gjerne de utviklertmiljøene som er tilgjengelige for deg på arbeidsplassen. Applikasjonen skal ha god dokumentasjon. Drøft

hvorfor dette er viktig. Sørg for å benytte god arbeidsmetodikk i utviklingen og argumenter kort for valget ditt (agile/føsefall? Hvordan jobbe med kodebasen daglig, mv.)
Kildekoden skal ligge tilgjengelig for de som ønsker tilganger under hele prosjektløpet

2. Arbeidsform, prosess og generelt

2.1 Prosjektløpet

Utviklingen skal gjøres som et lite prosjekt, der du planlegger brukerhistorier, epics og tasks. Arbeidsgiver har tilgang på Jira som kan være et godt alternativ.

2.2 Generelt

Oppgaven består av fire deler, planlegging, gjennomføring, dokumentasjon og egenvurdering. Stikkord for disse delene kan være estimering, sikkerhet, rettssikkerhet, bærekraft og etikk. Husk at det ikke kun er den teknologiske løsningen som teller mot en sluttevaluering. God selvrefleksjon og dokumentasjon teller også. Lykke til med oppgaven!

Planlegging

Min tolkning av oppgaven

Jeg skal lage en teknisk løsning på nett som viser en oversikt over restauranter i innlandsregionen og anmeldelser av dem. Brukere skal kunne anmelde restauranter selv.

Applikasjonen skal demonstrere grunnleggende funksjonalitet som kreves i moderne webapplikasjoner. Det jeg tror legges i dette er at jeg skal vise frem at jeg klarer å lage noe som går innenfor funksjonalitet som vi tar som en selvfølge i 2024 på en nettside, som responsivitet, universell utforming, sikkerhet, backend funksjonalitet og rask ytelse. Det er selvfølgelig mer enn dette, men dette er noen av de grunnleggende tingene i moderne webapplikasjoner.

Jeg skal bygge webapplikasjonen med anmeldere og brukere i tankene. Prøve å sette meg i deres sko for å se hva som er viktig i en anmeldelse, hvor lett det er å sjekke en anmeldelse og informasjon som burde være tilgjengelig om ulike restauranter.

Webapplikasjonen skal ha mulighet for innlogging. Du skal logge deg inn for å anmelde, men du trenger ikke å logge inn for å lese anmeldelser. Det er også åpent i oppgaven om administratorer skal eksistere og om de skal kunne legge inn restauranter. Her også nevnes det at jeg skal tenke på hvilken informasjon som er viktig for restaurantene.

Brukere skal kunne legge inn anmeldelse, men det er åpent for meg hva som skal være kriterier i anmeldelsen.

Brukere skal lett kunne finne og navigere seg til spisesteder og tilknyttede anmeldelser. Tenk business case.

Rammeverk og teknologi er optional.

2. Krav til applikasjonen

Applikasjonen skal være responsiv og gi en god brukeropplevelse på tvers av ulike enheter.

Koden skal være god og vedlikeholdbar.

Utvikler verktøy og språk er valgfritt.

Kan bruke utviklermiljøene som er tilgjengelig på jobb/arbeidsplassen

Applikasjonen skal være godt dokumentert

Hvorfor god dokumentasjon er viktig skal være svart på. Agile/fosefall?

Hvordan jobber man med kodebasen daglig?

Kildekoden skal være offentlig

Prosjektet skal planlegges, lage brukerhistorier, epics og tasks. Arbeidsgiver anbefaler Jira som jeg velger og bruke.

3. Generelt

Oppgaven består av 4 deler, planlegging, gjennomføring, dokumentasjon og egenvurdering.

Stikkord: estimering, sikkerhet, rettssikkerhet, bærekraft og etikk

4. Generelle tanker om oppgaven

Tenker først og fremst at oppgaven er veldig grei og skal kunne løses. Men tenker det er noen fallgruver som er viktig og være obs på. Spesielt rundt hvor mye jeg planlegger og lage og autentisering. Tenker jeg fort kan overestimere hvor mye jeg rekker og gjøre, og da er det lett og ikke bli ferdig med en "MVP".

Autentisering er også ett punkt hvor det er veldig lett å sette seg fast og bruke altfor mye tid. Tenker jeg må sette meg ned og tenkte godt igjennom hvordan jeg vil gjøre autentisering.

GDPR tenker jeg også med en gang, viktigheten av personvern på denne siden mtp at brukerinformasjon vil bli vist rundt anmeldelser.

Men er spent på og se hvordan jeg løser denne utfordringen.

Techstack

Sett opp en techstack som skal brukes. Både frontend og backend

Frontend

I frontend så skal jeg bruke Next.js med TypeScript. Dette er det jeg har mest erfaring med og det jeg kan best. Next.js er et kjent og velbrukt rammeverk som brukes i mange store prosjekter, så dokumentasjonen og eksemplene tilgjengelig er mange. TypeScript er veldig fint for å ha struktur og forutsigbarhet i prosjektet. Da vet du hva slags objekter du jobber med og hva du kan forvente av data. Samt at den sier fra hvis noe ikke matcher det du forventer.

Sammen med dette skal jeg bruke en rekke pakker som jeg er veldig kjent med og har god erfaring med å bruke:

- [Tailwind CSS](#) som er et class/CSS-bibliotek. Jeg bruker også SASS i stedet for CSS fordi den ekstra funksjonaliteten er veldig fin å ha.
- <https://ui.shadcn.com/> for ulike inputkomponenter (der det trengs, for eksempel å legge inn anmeldelse). Denne er veldig grei fordi det er stilfulle komponenter bygd på Tailwind med velfungerende funksjonalitet. Så sparer jeg tid på skjemakomponenter som ofte tar veldig mye unødvendig tid.
- [@tanstack/react-query](#) hvis det er behov for klientbasert datahenting, er denne veldig fin å bruke.
- [React Hook Form](#) og [Zod](#) for skjemahåndtering og validering.
- [Lucide React](#) for ikoner.
- [Axios](#) for datahenting.
- Også skal jeg bruke [BiomeJS](#) for linting og formatering.

Back-end

I backend har jeg valgt å gå for en API-løsning i .NET. Jeg tenker at jeg skal bruke et controller-basert oppsett. Jeg velger dette fremfor dotnet sin nye minimal API fordi det er et mer dokumentert mønster. Har mye erfaring med begge, men mest med controller-basert. Spesielt når det gjelder autentisering, er jeg mye sterkere på det.

Bruker ikke noe særlig med eksterne pakker i dette prosjektet. Eventuelt kan det være Identity, men dette er hvis jeg velger det som autentiseringsleverandør.

Kommer til å bruke pakker som EF Core for ORM.

Hvis jeg hadde hatt bedre tid, kunne jeg lagt til Fluent.SMTP for å sende ut e-poster til brukere. Så at pakka ikke vedlikeholdes lenger så ville ha lett etter en annen pakke da.

Autentisering kan du finne her: <https://notlimey.atlassian.net/browse/KAN-10>

Database

Til databasen kommer jeg til å bruke PostgreSQL. Hvis jeg velger å laste opp prosjektet offentlig, så kommer jeg til å bruke en gratis PgsqL-instans fra Vercel, ellers hoster jeg bare lokalt.

Devops og verktøy/applikasjoner

Devops avhenger veldig av tiden. Planen min er og ha hovedfokus på prosjektet så vil alt ekstra være hvis jeg har tid.

Men noen ting jeg skal bruke uavhengig er:

- Git og github
- Warp - For terminal på mac
- VSCode
- Slack, discord og teams for kommunikasjon
- Raycast for ulike LLM og LAM verktøy som chat og rettskrivning

Hvis jeg har tid, skal jeg legge ut prosjektet, og da kommer jeg til å bruke plattformen <https://vercel.com>. Jeg velger denne fordi den er best egnet for Next.js, og jeg har mye erfaring med Vercel. Jeg kommer også til å bruke PostgreSQL fra Vercel, noe jeg ikke har brukt før, men det kan være spennende å prøve. Jeg har ikke hostet en database gratis før, så jeg tenkte det kunne være kult å prøve det ut. Om jeg har tid eller ikke, er et annet spørsmål. Uansett om det blir en del av oppgaven eller ikke, vil jeg legge den ut etter oppgaven for ekstra læring. Da kan jeg også vise frem oppgaven til andre på GitHub.

Docker også kan være et verktøy jeg velger å benytte for at oppsettet av prosjektet skal være enklest mulig for de som vil sette det opp.

Figma benytter jeg kanskje hvis jeg velger å gå for et designsystem og ikke ferdige maler. Da er det fint for å sette opp komponenter der.

Autentisering

Jeg har tenkt mye på autentisering, og det er noen punkter jeg noterer meg:

1. Autentisering må fungere sømløst.

2. Adminroller:

1. Det må være mulig å legge til roller for admin-brukere.

3. OAuth-løsning:

1. Det kan være lurt å legge til Google, Facebook, etc. login for å gjøre det enklere for brukere å starte opp.
2. Viktig å huske på håndtering av tokens, fornyelse og god tilgangskontroll.

4. Plattformintegrasjon:

1. Autentisering må fungere både i Next.js og .NET. Begge må ha kontekst av autentisering.
2. Next.js-sesjoner må autentisere seg mot API-et.

5. Tilgangskontroll:

1. Begrenset tilgang hvis man ikke er logget inn i API-et.
2. Det kan være lurt å benytte regelen om at man må være autentisert, ellers må requesten komme fra server-side Next.js med API-token for å bli godkjent.

Dette er de jeg har vurdert som jeg har valgt og ikke gå for og hvorfor.

Clerk

Vurderte en tredjeparts tjeneste som heter Clerk men har i hovedsak ikke valgt denne pga lite tilgang på ressurser som sier noe om hvordan det integreres i .NET. Det er også verdt og merke at Clerk kan bli dyrt hvis det er ett stort prosjekt ettersom du betaler for antal aktive brukere over 100000. Men det er ett bra alternativ hvis jeg bare skulle bygd prosjektet i Next.js.

Firebase

Har brukt firebase før og synes det var tungvind/vanskelig og bruke i .NET. Så er det ikke lett og integrere roller i det på tvers av tjenester.

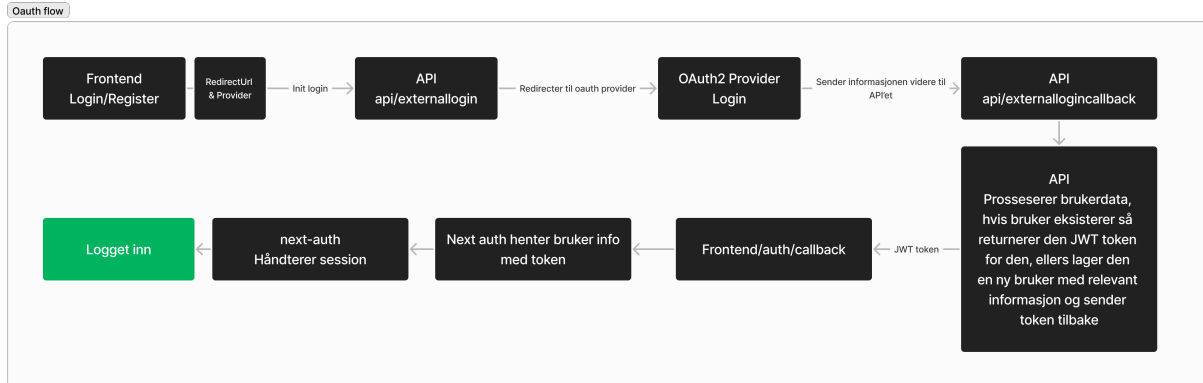
Jeg velger å gå for Identity i .NET som er bygd av Microsoft. Dette er en pakke som lar meg sette opp autentisering på egenhånd, håndterer roller veldig bra, og tillater OAuth-kobling med andre tjenester.

Fordeler:

- Har god erfaring med å jobbe med identity fra før
- Fungerer veldig godt med EF Core
- Får frem kompetansen min rundt autentisering og oppsett av avanserte strukturer i auth.
- Enkelt å sette opp rolle basert tilgangskontroll
- Rask utvikling og implementering inn i det allerede eksisterende .NET api'et

Ved å bruke Identity i .NET, sikrer jeg at autentiseringen i applikasjonen fungerer sømløst og sikkert, samtidig som jeg utnytter mine tidligere erfaringer og tekniske kunnskaper til å bygge en robust løsning.

Oauth2 flyt:



Bildet er hentet fra Figjam filene mine

Arkitektur

Legg en plan for både frontend og backend arkitektur. Hvordan skal oppsettet gjøres, og hva er viktig for at det skal fungere bra?

Arkitekturen bygger på techstacken som du finner over eller her <https://notlimey.atlassian.net/browse/KAN-8> på jira.

1. Frontend

- **Framework:** Next.js med TypeScript
- **UI Library:** Tailwind CSS og React Hook Form
- **Data Fetching:** Axios og @tanstack/react-query
- **State Management:** React Context eller Zustand (hvis nødvendig)
- **Ikoner:** Lucide React

Oppsett:

App router: Jeg kommer til å bruke app router til oppsettet av routing. Relativt nytt i Next.js men ikke for nytt så det er mye bugs. Veldig bra og enkelt og bruke for ISR og SSR av sider.

Common folder

- Components
- Hooks

- Providers
- ...etc

2. Backend

- **Framework:** .NET med et controller-basert oppsett
- **Autentisering:** Identity for brukeradministrasjon og autentisering
- **Database ORM:** EF Core

Arkitekturkomponenter

Controllers: Controllere for CRUD operasjoner på resturanter, anmeldelser og brukere

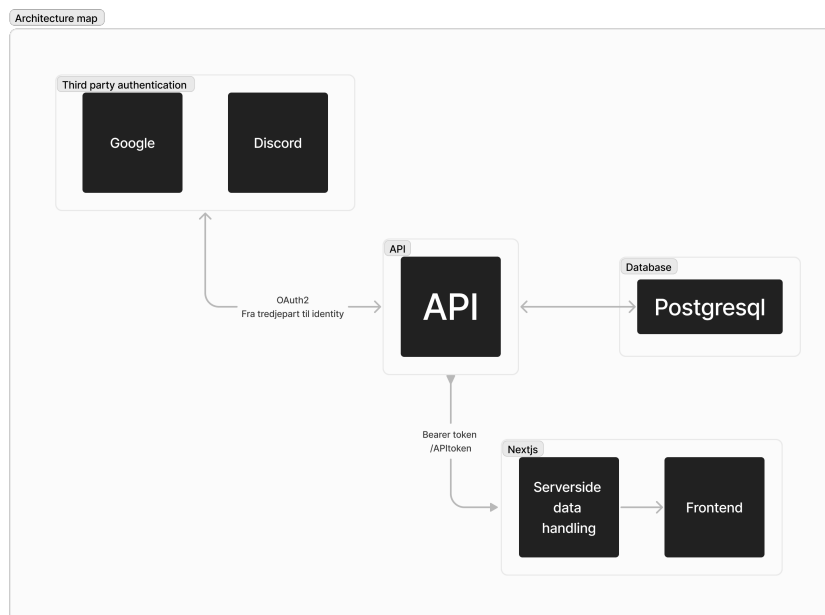
Services: For logikk mellom database og kontrollere

Models: Alt må ha modeller som representerer data i systemet

Authentication: Alle services og classer som håndterer authentication

<https://notlimey.atlassian.net/browse/KAN-13> (Database struktur) som du også ser under.

Generell arkitektur skisse:



Bildet er hentet fra Figjam skissene mine

Database struktur

Restaurants Table (Restauranter)

Kolonner:

- id (PK, GUID)
- name
- beskrivelse
- food_type
- address
- state (selvom oppgaven bare går på innlandet tar jeg med dette sånn at det kan brukes hvis prosjektet blir større)
- city
- zip_code
- phone_number
- website
- image_url - URL til restaurantens bilde
- opening_hours (JSON)
- latitude - Geolokasjon breddegrad
- longitude - Geolokasjon lengdegrad
- created_at (opprettet)
- updated_at (oppdatert)
- summary (ai generert summary av resturanten kan være kult og legge til hvis jeg har tid)

Reviews Table (Anmeldelser)

Kolonner:

- id (PK, UUID)
- user_id (FK til <http://ASP.NET> Identity)
- restaurant_id (FK til Restaurants)
- review_title
- visit_date (besøksdato)
- food_quality (matkvalitet) (Integer, rangering fra 1 til 5)
- service_quality (servicekvalitet) (Integer, rangering fra 1 til 5)
- ambiance (stemning) (Integer, rangering fra 1 til 5)
- value_for_money (valuta for pengene) (Integer, rangering fra 1 til 5)
- overall_rating (totalvurdering) (Integer, rangering fra 1 til 5) - Samlet vurdering
- comment - Detaljert anmeldelseskomentar
- created_at (opprettet)
- updated_at (oppdatert)

Relasjoner

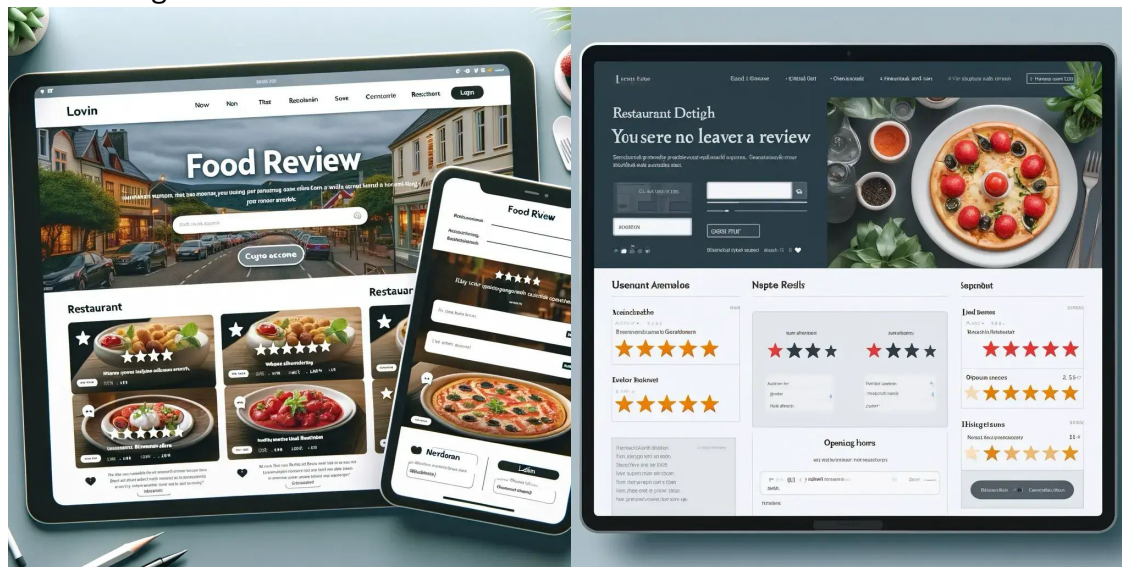
Restaurants til Reviews: En-til-mange (en restaurant kan ha mange anmeldelser).

Design & UI Biblioteker

Finn frem ett design eller design system som skal brukes.
Hvilke UI-biblioteker skal brukes?

Til alle forms komponenter skal jeg bruke: <https://ui.shadcn.com/> siden jeg har så mye kunnskap med det fra før og det er ett veldig stabilt og relativt pent bibliotek

Her er noen skjermbilder jeg planlegger og ta utgangspunkt i. De er AI genererte med DALL*E 3 og GPT 4o:



Tar utgangspunkt i de skissene over, ønsker ikke å bruke mer tid på denne delen av oppgaven enn det som er nødvendig ettersom design ikke er innenfor mitt fagområde.

Dag-til-dag-plan

<p>TORSdag 2</p> <p>Prosjekt oppsett ARKITEKTUR 23 MAY</p> <p>Oppsett enkel: Database oppsett & struktur BACKEND 23 MAY</p> <p>Oppsett avansert: Services og controllers BACKEND</p> <p>Authentication for Services og controllers BACKEND</p> <p>+ Create issue</p>	<p>FREDag 3</p> <p>MVP authentication: Microsoft Identity BACKEND 24 MAY</p> <p>Oppsett avansert: Services og controllers BACKEND</p> <p>Authentication for Services og controllers BACKEND</p> <p>+ Create issue</p>	<p>FREDag 4</p> <p>Sette opp next-auth og annen auth FRONTEND</p> <p>Opprett restauranter FRONTEND</p> <p>Restauranter FRONTEND</p> <p>Reviews FRONTEND</p>	<p>MANDag 4</p> <p>Design/UI FRONTEND</p> <p>Responsiv FRONTEND</p> <p>Tilleggsfunksjoner (mandag frontend) FRONTEND</p> <p>Tilleggsfunksjoner (mandag backend) BACKEND</p>	<p>TIRSdag 3</p> <p>Tilleggsfunksjoner (tirsdag frontend) FRONTEND</p> <p>Tilleggsfunksjoner (tirsdag backend) BACKEND</p> <p>Dokumenter DOKUMENTASJON</p>	<p>ONSDag 1</p> <p>Dokumenter pt2 og lag presentasjon DOKUMENTASJON</p>
--	---	---	---	--	---

Onsdag og torsdag morgen

Fullføre planen/planleggingen.

Torsdag etter lunsj

Sette opp prosjektet.

En arkitektur oppgave som innebærer å sette opp monorepoet, sette opp nextjs og dotnet prosjektet inni der.

Det innebærer også å sette opp repositoriet på github, og gi relevante parter tilganger som trengs.

Oppsett enkel: Database oppsett & struktur

Oppgaven innebærer å 1) sette opp en lokal postgres database. 2) Koble denne databasen opp til prosjektet/API'et. 3) Sette opp relevante modeller i henhold til database-struktur planen. 4) Sette opp entity framework core database contexten. 5) Sette opp en restaurant controller med statiske restaurant data. 6) Sørge for at swagger er satt opp og fungerer. 7) Teste at den statiske dataen kommer ut igjennom endepunktet (via swagger).

Fredag

MVP Microsoft Identity authentication

Overordnet så innebærer denne oppgaven å sette opp en MVP (Minimal viable product) versjon av Microsoft Identity. Stegene er som følger: 1) Installer Microsoft identity. 2) Sett opp IdentityDbContext. 3) Sett opp en service som extender UserService fra Identity og har funksjonene register og login. 4) Sett opp og håndter JWT tokens (Bearer tokens). 5) Sørg for at bearer tokens fungerer i swagger.

Oppsett avansert: Services og controllers

Oppgaven innebærer å sette opp de nødvendige servicene og controllerne for prosjektet. Stegene er: 1) Sett opp en service for restaurant (CRUD). 2) Sett opp en service for anmeldelser (CRUD). 3) Sett opp en controller for restaurant (CRUD). 4) Sett opp en controller for anmeldelser (CRUD)

Authentication controller med login & register

Sett opp en controller for authentication. Stegene er: 1) Sett opp register funksjonen. 2) Sett opp login funksjonen

Fredag+ / Helgen

Pga litt utgått tid på onsdag har jeg satt opp plan for å ta igjen noen oppgaver i helga.

Sette opp next-auth og annen auth

Sett opp next-auth i frontenden og sett opp generelle funksjoner for å håndtere sessions server og client side. Stegene er: 1) Sett opp next-auth. 2) Lag en logg inn side. 3) Lag en

registrer side. 4) Koble opp sider og layout til session. 5) Sett opp funksjonalitet for å hente ut session client og server side.

Opprett resturanter

Lag en side for å opprette resturanter /admin/resturants/new

Senere skal vi beskytte denne siden med roller

Restauranter

Stegene er: 1) Hent ut alle restauranter (paginert). 2) Hent ut restaurant på id ved å gå til /restauranter/slug

Anmeldelser

Stegene er: 1) Hent ut alle anmeldelser på restaurant. 2) Opprett en anmeldelse på restaurant.

Mandag

Design

Implementer ett generelt design hvor alle sider bruker samme stil. Stegene er: 1) Desing layout. 2) Design landingssiden. 3) Design individuelle restaurant sider

Responsiv

Denne oppgaven går ut på å gå igjennom løsningen på mobil resolusjon og endre elementer som ikke passer på mobil til å passe på mobil.

Tilleggsfunksjoner frontend

Dette er en samling av tilleggsfunksjoner jeg ser for meg at jeg kanskje har tid til å implementere på mandag. Funksjonene er følgene: 1) Avstand til resturant fra deg. Kan sorteres på

Tilleggsfunksjoner backend

Dette er en samling av tilleggsfunksjoner jeg ser for meg at jeg kanskje har tid til å implementere på mandag. Funksjonene er følgene: 1) oauth2 kobling med google og evt andre tredjepartsaktører. 2) Rollebaserte tilganger

Tirsdag

Tilleggsfunksjoner frontend

Dette er en samling av tilleggsfunksjoner jeg ser for meg at jeg kanskje har tid til å implementere på mandag. Funksjonene er følgene: 1) Beskytt /admin/* for alle uten rollen Admin. 2) Valgfrie rating skalaer, stjerner, emojis, memes.

Tilleggsfunksjoner backend

Dette er en samling av tilleggsfunksjoner jeg ser for meg at jeg kanskje har tid til å implementere på mandag. Funksjonene er følgende:

1) Tredjeparts tilgang via api nøkkler.

Dette krever at man har litt kontroll over hvor mye det brukes etc. Men i min løsning med begrenset tid gir jeg bare tilgang via API nøkkler.

Noter at i prod så ville jeg ha begrenset antall forespørrelser noen kan gjøre. Samt hatt oversikt over bruk

Ønsker å legge til muligheten for å fjerne api nøkkler fra ett admin dashboard eller databasen.

2) GPT Summary av restauranten når den opprettes og for hver 5 review, eller oppdatering. + etter 2 reviews

Begrunner tankegangen, 0, 2 og 5->..+5 med at dette var logiske punkter i hjernen min at summary burde oppdateres. For at den ikke oppdateres for ofte.

Tenker også at etter 50 reviews kan det være for hver 10ende review

Dokumentering

Resten av tirsdagen skal brukes til dokumentering

Onsdag

Hele onsdagen skal brukes til dokumentering, lagring av en demo/presentasjon og eventuelle rettelser i applikasjonen.

Gjennomføring

Dag 1 (Tirsdag) - Planlegging

Dag 1 så hadde jeg møte med prøvenemda for å få og gå igjennom oppgaven. Møtet gikk veldig fint, og jeg tenkte positivt på mulighetene for å gjennomføre oppgaven og gjøre det bra. Møtet var litt sent så hadde ikke mye tid igjen av dagen.

Etter møtet startet jeg med en gang med planleggingen av prosjektet. Jeg starta med å definere min egen forståelse av prosjektet. Hvordan jeg oppfattet oppgaven og hva jeg mente jeg skulle gjøre. Dette så jeg kunne vise det tilbake til prøvenemda og sjefen min og få min egen oppfattelse bekreftet. Jeg satte opp planleggingen i Jira som var anbefalt i oppgaven.

Jeg rakk ikke så mye mer denne dagen.

Dag 2 (onsdag) – Planlegging

På onsdagen var dagen min litt redusert, men jeg brukte den tiden jeg hadde godt utover dagen.

Jeg starta med å sette opp resterende oppgaver i Jira: Techstack, Autentisering, Arkitektur, Database struktur, Design og UI bibliotek og dag til dag plan.

Første jeg starta med var å planlegge techstacken. Jeg hadde en ganske god ide om hva jeg skulle bruke men det var fint å få det definert ned. Jeg bruker teknologi som jeg i utgangspunktet har ganske god peiling på og som jeg har jobbet mye med. Overordnet sett så bestemte jeg meg for å bruke Next.js som jeg bruker hver dag på Inmeta og .NET som vi bruker, men som jeg også har veldig mye egen erfaring med. [Her er hele techstacken](#).

Videre jobbet jeg med å planlegge autentisering. Her hadde jeg ikke så veldig føringer på hva jeg skulle bruke. Det som var viktig for meg var 1) at jeg fikk satt det opp relativt raskt og at det ikke kom i vegen for resten av oppgaven, 2) at jeg fikk frem kompetansenivået mitt på ett så avansert tema som autentisering. Jeg valgte og ha mest fokus på punkt nr 2 men å gå for en løsning som jeg har gjort før. Det ble å sette opp Microsoft Identity som kommer ganske ut av boksen, men du må implementere det selv, og du kommer ingen vei uten å skjønne hvordan det fungerer. Det er veldig GDPR vennlig med tanke på at det bare er 1 (meg) som har informasjonen og det ligger ikke host noen tredjepartstjenester ut av boksen. Jeg får full kontroll på informasjonen selv og det er ikke altfor vanskelig og koble til tredjepartstjenester som Google igjennom Oauth2, noe som jeg ønsker å gjøre.

Så fikk jeg satt opp en overordna arkitektur, denne baserer seg jo egentlig bare på techstacken men det er fortsatt greit å definere. Jeg skal tegne ett overordnet arkitektur kart senere.

Til slutt så genererte jeg noen skisser ved hjelp av DALL-E og GPT-4o til jeg var fornøyd. Ønsker ikke å bruke lang tid på å lete etter skisser og jeg fant ikke noen jeg likte umiddelbart.

Så planen er å ta utgangspunkt i skissene jeg genererte og lage noe ut av de.

Dag 3 (torsdag) – Planlegging og oppsett

Jeg starta dagen med å definere ned en database struktur og bestemme meg for hvilke properties ulike modeller trenger. Kom frem til at jeg trenger ganske få attributter men at jeg kan gjøre veldig mye mer ut av prosjektet med ett par ekstra. Som åpningstider, det vil være en litt mer avansert datastruktur, men det er noe som er veldig relevant å ha med på siden. Geolokasjon er veldig kult å ha med for da kan jeg vise ett kart over hvor restauranten ligger. Mattype er fint å vite. Også tenkte jeg at det hadde vært kult å legge til en oppsummering gjort av GPT-4o hvis jeg har tid til det, så jeg legger på attributten.

På review bestemte jeg meg for å legge til mange fler rating punkter enn jeg orginalt hadde tenkt. Fant ut at det blir mye kulere om man rangerer maten på mye mer enn bare 1-5, la til matkvalitet, servicekvalitet, stemningsnivå, verdi for pengene og generell rangering. Det var noen som nevnte at jeg bare kan kalkulere generell rangering basert på de andre men jeg mener at brukere kanskje vil tippe litt ene eller andre veien basert på egen oppfattning og ikke tallene på de andre. La også til besøksdato.

Etter dette hadde jeg ett møte med prøvenemda, vi diskuterte hvor langt jeg hadde kommet og jeg synes møtet gikk fint. Vi diskuterte at jeg var litt bak skjema etter min egen

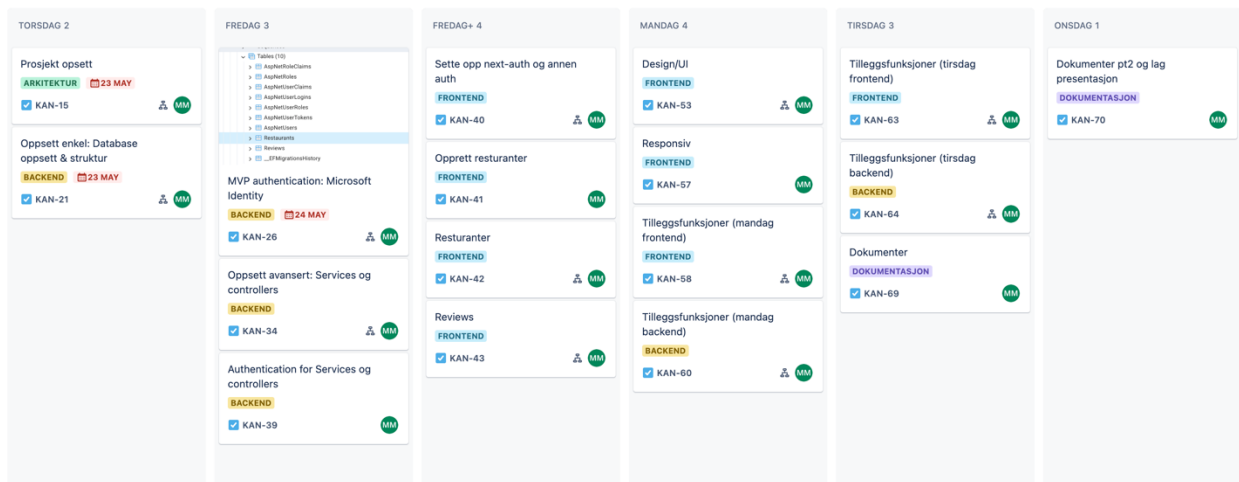
oppfatning. Per Arne nevnte at jeg burde lage en dag til dag plan og inkludere dette i dokumentasjonen min samt planleggingen på Jira. Så det skal jeg gjøre i morgen. Fredrik kommenterte på at jeg måtte legge til en overordnet arkitekturskisse som jeg også planlegger å sette opp i morgen.

På slutten av dagen installerte pakker jeg og satte opp prosjektet. Jeg går for strukturen monorepo for at det skal være mest lesbart for prøvenemda. I ett ordentlig prosjekt så ville jeg kanskje ha lagt API og frontend i hver sine repositories for å gjøre det enklere med deployment og vedlikehold, men det er ikke ett problem i denne oppgaven.

Dag 4 (fredag) – Litt av hvert

Dag til dag plan

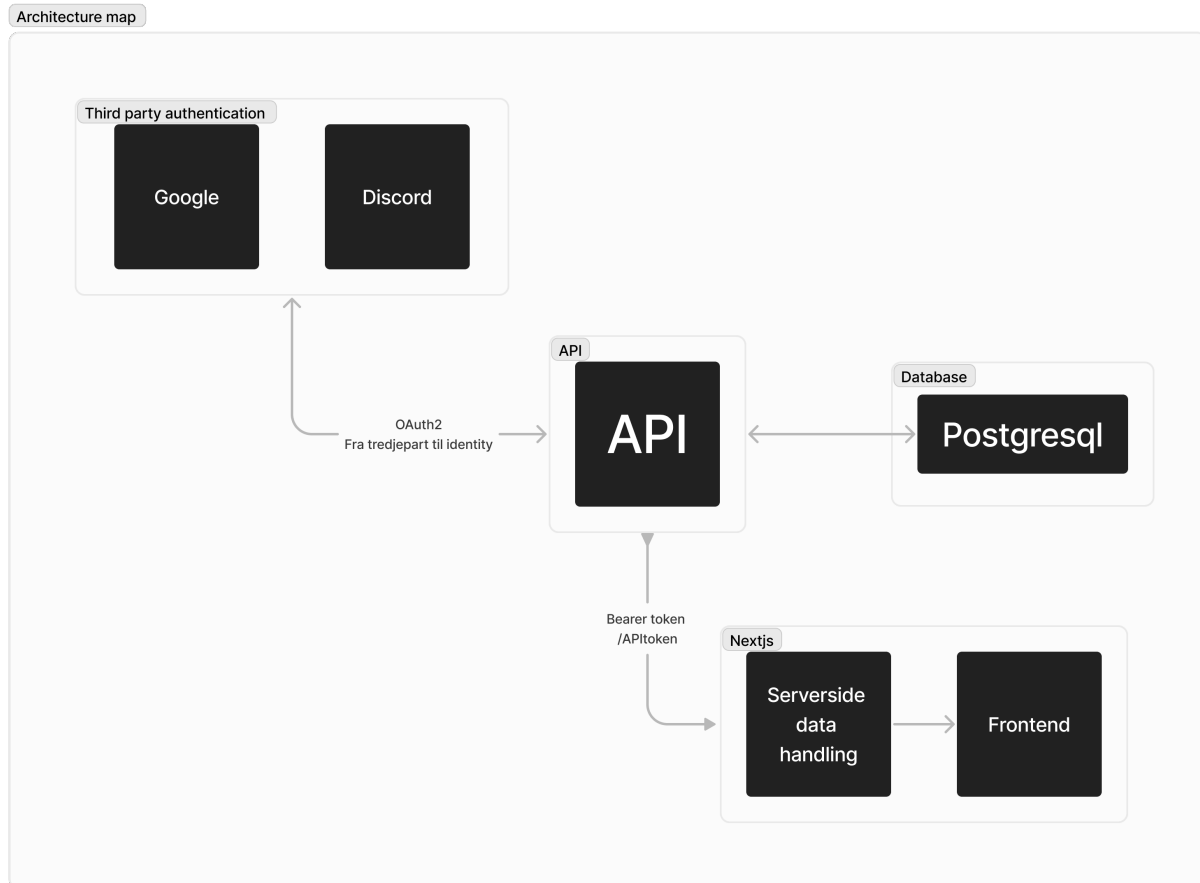
Første jeg gjorde denne dagen var å sette opp en dag til dag plan. Jeg satt den opp i jira.



Fra mitt eget perspektiv så tenker jeg at jeg kanskje har vært litt over optimistisk eller tenkt veldig teoretisk maks jeg får gjort. Men det er en plan jeg planlegger å holde. Jeg tenker at alt som ligger etter responsiv på mandag er ting som bare gjøres om jeg har ekstra tid. Ellers fokuserer jeg på dokumentasjon.

Arkitekturoversikt

Her er en oversikt over arkitekturen jeg tegna i figjam:



Oppsett enkel: Database oppsett & Struktur

Oppsettet gikk ganske rett frem. Litt rusten på EF Core men jeg brukte ett tidligere prosjekt som utgangspunkt og kom raskt frem til en løsning som funket. Sto litt fast på hvordan jeg skulle gjøre JSON i databasen, men så fant jeg ut at Dotnet 8 og postgres har fått native støtte for JSON data så lenge man spesifiserer det i dbcontexten. Fulgte denne:

<https://www.npgsql.org/efcore/mapping/json.html?tabs=data-annotations%2Cpoco>

Swagger var satt opp ut av boksen så slapp å sette opp det.

Det som tok lengst tid var modellene og relations imellom de. Var ikke sånn at jeg ikke skjønte hvordan jeg gjorde det men mer det at jeg ikke var helt sikker på om jeg gjorde det riktig, og det er veldig greit å gjøre det riktig så dobbelt sjekket det og leste mye dokumentasjon fra microsoft for å freshe opp huet.

Når det gjelder database så brukte jeg bare en kommando for å spinne opp en postgres database i docker på macen min:

```
docker run --name my-postgres -e POSTGRES_USER=postgres -e POSTGRES_PASSWORD=postgres -e POSTGRES_DB=db -p 5432:5432 -d postgres
```

Microsoft Identity

Neste jeg starta på var Microsoft Identity.

Installeringen gikk veldig rett fram, så satte jeg opp DbContexten min til å inherite fra IdentityDbContext.

Deretter satte jeg opp min egen definisjon av rolle og bruker. Disse modellene inheriter fra modellene til identity men lar meg overskrive og endre definisjonen av ulike properties, får også lagt på flere properties som var det viktigste.

I planen min hadde jeg ikke skrevet noen ekstra verdier på bruker, men jeg fant her ut at jeg trenger noen fler:

- Fornavn og etternavn: Tenker det vil gjøre reviews mer personlig
- AvatarURL: for egne avatarer til alle brukere generert med dicebear
- IsAnonymous: Muligheten for at brukere får gjøre profilen sin anonym i all data.

Hadde satt opp en oppgave om at jeg måtte sette opp en UserService men hadde glemt at Identity UserManager fungerer helt fint på egenhånd så bare brukte det. Var ingen bruk for en egen service.

Next up så satte jeg opp JWT authentication, litt utenfor hva jeg husket hvordan man gjorde. De har endret litt på strukturen i API siden siste jeg jobbet med det så var nødt til å lese meg opp på litt dokumentasjon rundt hvordan du behandler Claims i .NET 8 webapi men gikk relativt greit. Satte først expire til 60 min, men endret det til 7 dager etter at jeg ble nødt til å hente ny token hele tiden. Planen er å sette den tilbake til 60 og sette opp refresh tokens hvis jeg får tid til det.

Hadde satt opp en egen oppgave på fredag+ til å sette opp login og register endepunkter, men merka det var bare lettere og sette opp de samtidig som jeg satte opp identity så jeg fikk testa det jeg lagde.

I denne prosessen av å sette opp Microsoft Identity lærte jeg om en feature som jeg ikke hadde hørt om før i C#, method extensions eller bare extensions. Som lar deg legge på funksjoner på allerede eksisterende klasser. Uten å måtte gjøre noe mer enn og bare lage extensions filene. Så jeg flytta hele startup Auth og swagger logikken inn i extensions på builder.Services, på denne måten ble Program.cs fila mi mye mer ryddig.

Kom også frem til at jeg måtte finne ett bibliotek for å mappe modeller til dtos før de går ut til brukeren. Dette er best practice for å vite at man alltid har kontroll på hvilken data som går ut igjennom API'et og hva som sendes med. Når du mapper det om til DTOs så er du klar over hva som sendes ut. Og hvis hovedmodellen endres så sendes ikke automatisk det nye ut til brukeren så du slipper og med ett uhell expose hemmelig eller sensitiv informasjon. Brukte Riok.Mapperly

Dag 5 og 6 (Fredag+ /helg)

Oppsett avansert: Services og controllers

Sette opp services og controllers gikk ganske rett frem, har gjort det mange ganger og var ikke noe nytt til prosessen. Var bare vanlige CRUD funksjoner mot databasen.

Når jeg var ferdig med dette og begynte å teste det så fikk jeg en error, «cycle loop». Som betyr at entitetene referer til hverandre og .NET vet ikke hvor det stopper så den thrower en error. Sto fast på det en stund men etter å ha googlet litt fant jeg ut at det fiksa seg selv om jeg mappa entitetene til en dto som definerer hvor dypt det går. Det funka.

Her fant jeg ut at det var lurt å legge på en overallrating på restaurant entiteten sånn at vi slipper å kalkulere den hver gang vi henter ut anmeldelser. Men kan få den rett på restaurant. La på dette på restaurant entiteten.

Authentication for services og controllers

Denne er gjort ettersom jeg gjorde det i går. Men jeg satte opp API tokens for prosjektet. Sånn at tredjepartstjenester kan hente ut restauranter og anmeldelser fra mitt system.

Dette er en løsning på det rundt at jeg skal tenke business case.

Hvis ApiToken er sendt med i api'et så kan hente alle Get requests per nå. Men man kan ikke opprette anmeldelser f.eks dette er fortsatt begrenset til bruker.

Next-auth og auth sider

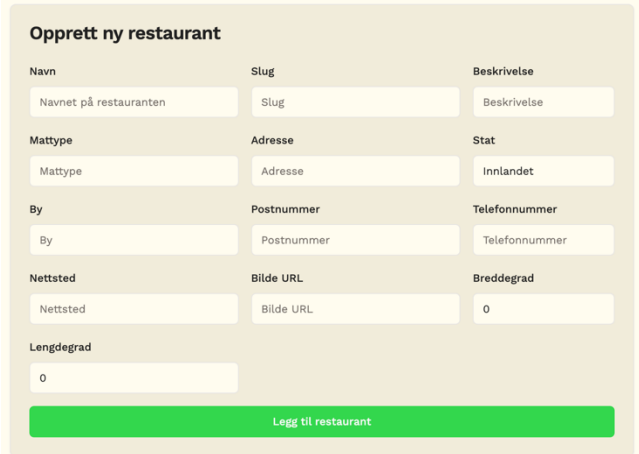
Next-auth gikk veldig rett frem, hentet frem kode fra et tidligere prosjekt hvor jeg brukte et liknende oppsett. Brukte Shadcn til login og register sidene og synes de ble veldig bra. Til slutt koblet jeg opp next-auth til api'et mitt, og det funka med en gang. Satte opp en liten /profile page hvor du kan se informasjon om den nåværende sesjonen du har i next-auth. Resten av oppgavene jeg hadde satt opp falt på plass bare next-auth var satt opp. Glemte hvor mye de har ut av boksen. Når jeg satte opp disse skjemaene hadde jeg fokus på tilgjengelighet og universell utforming.

Opprette restauranter

Selv om det ikke var nødvendig i oppgaven så planla jeg å legge til så admins kan opprette restauranter. Så slipper jeg legge det inn manuelt i databasen.

Satte opp skjemaet under for å legge inn data.

Funka helt fint sammen med endepunktet. Satte også opp så bare de med rollen Admin har mulighet til å se denne siden og opprette restaurant.



The image shows a web form titled "Opprett ny restaurant". It contains several input fields organized in a grid. At the bottom is a green button labeled "Legg til restaurant".

Navn	Slug	Beskrivelse
<input type="text" value="Navnet på restauranten"/>	<input type="text" value="Slug"/>	<input type="text" value="Beskrivelse"/>
Matttype	Adresse	Stat
<input type="text" value="Matttype"/>	<input type="text" value="Adresse"/>	<input type="text" value="Innlandet"/>
By	Postnummer	Telefonnummer
<input type="text" value="By"/>	<input type="text" value="Postnummer"/>	<input type="text" value="Telefonnummer"/>
Nettsted	Bilde URL	Breddegrad
<input type="text" value="Nettsted"/>	<input type="text" value="Bilde URL"/>	<input type="text" value="0"/>
Lengdegrad		
<input type="text" value="0"/>		

Legg til restaurant

List og hent restauranter

Hent ut og se spesifikt fungerte veldig fint. Brukte litt tid på å prøve og få det til å se nogen lunde greit ut. Har avtalt med en designer på jobb at han skal skisse opp to sider til meg så jeg får de på mandag. Men jeg bruker litt tid på utseende nå bare så jeg har ett bedre startpunkt på mandag, og om det ikke er ferdig eller jeg ikke har tid så har jeg i det minste noe som er greit å vise frem.

Fikk lista ut alt veldig greit og spesifikke sider fungerte fint. La på en slug attributt på restaurant som er unik. Det gir ett bedre inntrykk for brukere enn en GUID.

List ut og opprett anmeldelser

Liste anmeldelser gikk veldig greit, igjen ganske rett frem og fulgte bare controllerene jeg har satt opp.

Neste jeg gjorde var å sette opp opprett anmeldelse. Det var ganske rett frem men hadde noen issues med at når du oppretta en anmeldelse så dukka den ikke opp og du kunne lage en til. Fiksa det ganske raskt både frontend og backend.

Så fikk jeg litt issues når du prøvde og slette den. Relatert til det samme.

Paginerings

Jeg skrev i planleggingen min at jeg skulle sette opp paginering, men dette er noe jeg utsetter. Var litt usikker på strukturen som er best practice og brukte litt mye tid på det så bestemte meg for at det ikke var nødvendig for denne MVP'en. Noe som absolutt burde legges til senere, men ikke nødvendig for levering av oppgaven.

Tilleggsendringer

Gjennomsnitts rangeringer for alt

La på gjennomsnitts rangering for alle attributter relatert til rangering på anmeldelse. Dette gjør at jeg kan hente ut en gjennomsnitts rangering på alle attributter på restaurant uten og kalkulere det hver gang. Hadde noen issues med at når du slettet og oppretta rangeringer så prøvde den å oppdatere 2 entiteter i forskjellige calls, først anmeldelse også restaurant. Men flyttet de i samme database operasjon inne i ReviewService.

Toasts

La til popups/toasts når du gjør ulike handlinger på siden.

Leverer roller til frontend

Satte opp så frontend får med rollene til brukeren sånn at den kan skjule og vise innhold basert på rollene om det ønskes.

Hva ble spist?

La til et nytt attributt for hva som den som anmelder spise på restauranten. Tenkte det kan være veldig relevant for noen og lese når de ser en anmeldelse.

Skjule innhold basert på handlinger

For eksempel så skjuler jeg opprett anmeldelse hvis du allerede har anmeldt restauranten.

Dag 7 (Mandag) – tredjepart & nytt design

Denne dagen hadde jeg hovedfokus på tilleggsfunksjoner. Hadde allerede blitt ferdig med oppgavene jeg hadde estimert for mandag så da var det tilleggsfunksjoner som var på planen.

Oauth2

Har hatt lyst til å sette opp oauth2 med google hele tiden så denne tilleggsfunksjonen jeg legger hovedfokus på. Flyten på å sette det opp gikk ganske greit, men møtte på ett par utfordringer.

Hadde mye issues med returnUrl, hadde spesifisert min egen men det viste seg at identity hadde presets så du måtte legge inn de på oauth.

Så hadde noen problemer med at returnUrl var cacha av google så sto litt fast der. Men det som løste det var bare å vente.

Hadde samme issues når jeg prøvde å koble til discord.

Hadde litt problemer med flyten på return fra oauth hvor jeg skulle sette opp og returnere ett identity token koblet til riktig bruker. Fikk det til å funke til slutt. Var nødvendig og encode tokenet før jeg sendte det med tilbake til next-auth. Når jeg gjorde det så funka det.

Til slutt så funka google og discord.

Nytt design

Andreas på kontoret lagde ett nytt design til fagprøven min som jeg brukte resten av dagen på å implementere. Hadde ett godt grunnlag fra før så tok ikke så lang tid. Ble ferdig ila dagen.

Men veldig fornøyd med det jeg fikk av Andreas. Nå er min innlevering mest teknisk og mest backend/next logikk, men med ett ordentlig design så får jeg mye bedre frem hva jeg har laget og alt ser mye mer profesjonelt ut.

Spesielt for de utekniske så er design ekstremt viktig i en slik oppgave som det her.

Det tok ikke lang tid å implementere, så gjorde jeg designet responsivt selv.

Dag 8 (Tirsdag) – Dokumentere å fikse små feil

Tirsdag har fokuset vært å fikse opp i småfeil rundt designet og jobbe med dokumentasjon.

Har også lastet opp prosjektet til vercel og azure. Hoster backenden min hos Azure også har jeg databasen og nextjs applikasjonen i vercel. Database oppsettet var veldig lett,

samme med nextjs. Brukte litt tid på azure for hadde litt problemer med connectionstring men fikk fiksa det til slutt.

Andre små endringer jeg har gjort

- Oppdatert så produksjon bruker API nøkkler
- Catcher errorer som skjer i frontend
- Fjernet whitespace fra tredjepartsaktørers brukernavn (ex Martin Myhre -> MartinMyhre)
- Lasta opp planleggings dokumenter i repo
- Mobil justeringer

Egenvurdering

Overordnet synes jeg oppgaven gikk veldig bra. Jeg er stolt av det jeg har produsert på den tiden jeg har hatt. Både teknisk, men også med tanke på planlegging og dokumentasjon. Veldig fornøyd med å ha fått til så mye forskjellig som jeg ikke har gjort før. For eksempel OAuth2 integrering i .NET.

Planleggingen

Planleggingen av prosjektet gikk bra, men jeg har absolutt forbedringspotensialer. Her er noen ting jeg noterer meg med planleggingen.

- Fikk ikke alt jeg tenkte ned i planen. Som det jeg tenkte med at tredjeparter skal kunne koble seg på med APITokens.
- Hadde ikke en dag-til-dag-plan før det ble påpekt at dette var noe jeg burde ha (fra prøvenemda)
- Litt for optimistisk på hva jeg får til på så kort tid
- Kunne vært bedre på å dra inn læringsmålene inn i planen min

Planen min var å koble til OpenAI til å summere anmeldelser og restaurant inn i en bolk med tekst. Rakk ikke å gjøre dette. Men jeg føler det var helt greit å ikke få implementert. Det hadde nokk ikke tatt meg så lang tid, men jeg bestemte meg for at det hadde vært en fun to have feature bare. Alt i alt så er anmeldelser uansett folks ord og det er noe man ikke vil manipulere.

Kan være noe som implementeres senere med det krever litt mer baktanke.

Gjennomføringen

Gjennomføringen er jeg veldig fornøyd med, har gjort det jeg kan på den tiden jeg har hatt. Har ikke stått fast på så mye og flyten på arbeidet har gått veldig bra.

Design/visuelt

Veldig fornøyd med det visuelle på siden. Jeg fikk ett generelt design av Andreas Rasmussen for hovedsiden og restaurant siden, men det visuelle rundt skjemaer,

innlogging, registrer synes jeg ble veldig bra til tross for at jeg ikke har hatt det som hovedfokus. Synes også jeg har vært god på å passe på UU rundt skjemaer, noe som er veldig lett og droppe når det er en oppgave som det her med så lite tid.

Læring

Har lært mye på dette prosjektet, fått utfordret meg selv godt. Jeg kunne tatt den lette veien og bare gjort ting som jeg har gjort før men når jeg velger og utfordre meg sånn som jeg har gjort i denne oppgaven så sitter jeg igjen med ett resultat jeg er stolt av og med ny kompetanse.

Hva ville jeg gjort annerledes?

Planlegging

Det er mye jeg ville gjort annerledes nå som jeg har vært igjennom det en gang, men er fornøyd med det jeg har gjort utafra det jeg kunne når jeg starta forrige uke.

Med den kunnskapen jeg har nå så ville jeg ha gjort ett par ting annerledes:

- Jeg ville definert brukergrupper i planen min
- Vært flinkere til å notere meg tanker frem til planen, så jeg husker å få med meg alt
- Satt opp en dag-til-dag-plan tidligere

Gjennomføringen

Ikke så mye å notere meg på gjennomføringen, veldig fornøyd med det jeg har fått til på den tiden jeg har hatt.

Skulle ønske jeg spurte kollegaen min litt tidligere om han kunne laget en design skisse, så jeg kunne startet og utvikle den med en gang. Da hadde jeg hatt tid til å diskutere endringer med han og kanskje også implementert litt fler features i appen min.

Savner en tredje oauth2 integrering, fikk beskjed om at det ikke var nødvendig med fler enn 2 men i retrospekt skulle jeg ønske jeg allikevel la til meta/facebook innlogging ettersom dette er så populært. Det ville passet bra inn i argumentasjonen min om business case.

Savner funksjonen for å anonymisere brukere. Har lagt inn attributtene på bruker, men per nå blir ikke brukerdatabene anonymisert på vei ut fra API'et om denne er satt. Brukte litt tid på å prøve og få det til men bestemte meg jo for at det ikke var nødvendig og bruke så mye tid på. Mener det var riktig beslutning av meg, men skulle ønske denne featuren var på plass i innlevert oppgave. Det ville også passet bra inn for læringsmålet GDPR, tanken ligger der.

Skulle ønske refresh token var på plass. En ting som bør være på plass for sikkerhetsgrunner, men som ikke ble lagt til på grunn av tidsgrunner. Litt mer komplisert, men tror jeg skulle fått lagt det inn med 4 timer ekstra på oppgaven.

I planen min nevnte jeg også ulike måter å representere rangeringen på. Dette tror jeg hadde vært en kul ting å implementere og er noe jeg skulle ønske jeg hadde implementert.

Plan for videre utvikling (hvis jeg hadde hatt mer tid)

Selv om jeg har oppnådd mange av målene i oppgaven, er det flere funksjoner jeg planlegger å implementere i fremtidige versjoner av applikasjonen for å gjøre den mer robust og funksjonell. Her er planen for videre utvikling:

1. Søkefunksjon

Implementer en søkefunksjon som lar brukere søke etter restauranter basert på navn. Dette vil gjøre det lettere å finne spesifikke restauranter raskt.

2. Filterfunksjon

Legg til filtre som brukere kan benytte for å sortere og finne restauranter basert på ulike kriterier som rangering, prisnivå, matkjøkken, åpningstider, og andre preferanser. Dette vil forbedre brukeropplevelsen ved å tillate mer målrettet leting.

3. Chatbot

Utvikle en chatbot som kan foreslå restauranter basert på brukernes samtaler og preferanser. Chatboten vil kunne stille spørsmål som "Hva slags mat liker du?" eller "Hvilket område befinner du deg i?" og deretter gi forslag. Brukerne vil også kunne si nei til forslagene og få nye basert på utdypede preferanser.

4. Anonymisering av brukere

Videreutvikle funksjonen for anonymisering av brukere i anmeldelser. Sørg for at brukerne kan velge å anonymisere sin identitet, og implementer logikk for å håndtere dette på både frontend og backend.

5. Refresh Tokens

Implementer en mekanisme for refresh tokens slik at sikkerhet og brukeropplevelse forbedres ved at brukerne slipper å logge inn ofte. Dette øker sikkerheten ved å tillate tokens å utløpe og fornye seg automatisk.

Læremål

Etiske vurderinger og lovverk

For å sikre at løsningen min oppfyller GDPR, har jeg implementert solide autentiseringsmetoder og sørget for at brukerdata lagres trygt. Jeg har tenkt nøye gjennom etiske spørsmål, som for eksempel anonymisering av brukere i anmeldelsene, og sikret at dataene bare brukes til det de er ment for. Systemet er designet med tanke på å beskytte brukernes personvern og overholde lovkrav.

Sikkerhet og personvern

Jeg har prioritert sikkerhet og personvern høyt i applikasjonen. Brukerdata beskyttes gjennom kryptert lagring og sikker autentisering ved hjelp av Microsoft Identity og OAuth2. For å ivareta GDPR har jeg inkludert funksjoner for anonymisering av anmeldelser. Alt er designet for å sikre at personopplysninger håndteres forsvarlig og at brukernes data er trygge.

Design, interaksjon og brukerdialog

Ved bruk av Next.js, Shadcn og Tailwind CSS har jeg laget et responsivt og brukervennlig design. Skjemaer og interaksjoner er universelt utformet for å sikre en god brukeropplevelse på alle enheter.

Demokrati og medborgerskap

Applikasjonen gir brukerne en plattform for å dele sine meninger om lokale restauranter, og bidrar til å fremme deltakelse og fellesskap. Med anonymisering og sikker oppbevaring av data, sikrer jeg at alle kan delta trygt og åpent. Dette styrker lokalt engasjement og medborgerskap.

Bærekraftig utvikling

Ved å promotere lokale restauranter og matopplevelser, støtter applikasjonen bærekraftig utvikling i lokalsamfunnet. Ved å velge effektive teknologier som Next.js og skybaserte løsninger, sørger jeg for optimal ressursbruk, noe som bidrar til å redusere påvirkningen på miljøet. Ved å gjøre all dokumentasjon og kildekode tilgjengelig for andre, oppfordrer jeg til gjenbruk og deling av kunnskap, som også er viktige aspekter av bærekraftig utvikling.

Dokumentasjonen

Eneste 2 tingene jeg har å påpeke på dokumentasjonen er at jeg kanskje kunne startet litt før med den og at jeg kunne dokumentert mer underveis. Det hadde nok resultert i litt mindre som blir levert, men det er en stor del av oppgaven så viktig å gjøre det riktig. Men jeg er fornøyd med dokumentasjonen jeg har levert.