Norges teknisk-naturvitenskapelige universitet

TDT4140 - Programvareutvikling

Iterasjonsplanlegging L2

Forstudie - Gruppe 40

Ivar Carlsen
Lars Tønder
Marius Arhaug
Karen Hompland
Eskild G. Hansen
Joakim N. S. Schäffer
Torunn Elisabeth Seyffarth
Veileder:
Sindre Langaard

February 5, 2021

1 Problembeskrivelse

Det sies at å spise med andre kan være både mer økonomisk samt bærekraftig. Produkteier ønsker en tjeneste der man kan gjøre det mulig å dele middager, med både kjente og ukjente. Målet er at tjenesten skal gjøre den ellers ensomme middagen til noe sosialt, der tjenesten lar folk publisere en planlagt middag som andre kan melde seg på. Produktet vil bestå av administratorer, brukere og fremtidig støtte for annonsører. Som bruker vil man blant annet kunne poste og avlyse egne middager, melde seg på andres middager, filtrere middagene, rate og kommunisere med andre brukere. Som administrator vil man kunne slette og moderere både brukere og innhold. Vi har løpende dialog med produkteier, som har gitt oss frie tøyler når det kommer til farger og design, logo og navn.

2 Tech-stack

Fordi gruppen har mye ulik erfaring hadde medlemmene et møte hvor de mest erfarne medlemmene presenterte ulike alternativer og brukte mye tid på å diskutere fordeler og ulemper. I denne seksjonen vil en kort oversikt over valgene gruppen har tatt foreligge.

2.1 Programmeringsspråk

Ettersom uansett valg vil føre til at medlemmer må sette seg inn i ny teknologi, har gruppen bestemt seg for å bruke kun ett programmeringsspråk, **TypeScript**. Ønsket om å bruke et "strongly-typed" språk stod sterkt i hele gruppen, og ble valgt over JavaScript og Python. Det er mange likheter, da TypeScript er et supersett av JavaScript. Selv om Typescript har en litt brattere læringskurve, bestemte gruppen at tiden spart på blant annet enklere debugging og en mer selv-dokumenterende kode er fordeler som ønskes.

2.2 Front-end

Gruppen brukte en del tid på å diskutere front-end rammeverk og endte på **React** da dette ser ut til å være mest brukt i norsk arbeidsliv (Finn.no, 29/01-2021). I diskusjonen rundt et front-end rammeverk kontra vanilla HTML/CSS var det to konsiderasjoner som ble tatt, produksjonshastighet og enkelt bruk. Det er ikke alle medlemmer som har god erfaring med webutvikling, men vi endte likevel til slutt på **React**, da det er greit å sette seg inn i og reduserer boiler-plate code.

2.3 Back-end

Gruppen tok videre for seg valget av ulike backend teknologier. For å spare gruppen mer tid på opplæring i enda et språk ble **Feathers.js** og **Node.js** valgt. Dette vil drastisk senke implementeringstiden.

2.4 Databaser & lagring

Diskusjonen rundt databasen dreide seg om implementasjonshastighet, men også tverrfaglighet. TDT4140 Datamodellering og Databasesystemer tar for seg relasjonsdatabaser og bruker **MySQL**, noe som gruppen syntes kunne være nyttig å eksperimentere med. Gruppen har enda ikke helt tatt avgjørelsen om API-implementasjon, da scopet for dette enda ikke er klart.

3 Standarder

For å sikre at koden er lett å lese og inneholder færrest mulig feil, er det hensiktsmessig å følge visse konvensjoner. Gruppa ble enige om å bruke **AirBnB styleguide**. For å best mulig korte ned på tidsbruken av å følge denne, og for å hindre opprydningsarbeid i etterkant, legges det opp til bruk av verktøy som **Prettier** og **ESLint**, til henholdsvis stilkorreksjoner og feilretting. AirBnB sin styleguide er en veldig utbredt standard i industrien, og et ønske om en bedre koordinert kodebase valgte gruppen denne.

3.1 Smidige metoder

Gruppen har valgt å følge anbefalte møterutiner i henhold til Scrum, med noen unnvikelser. Rollen som Scrum-master går på rundgang, der ett medlem tar rollen for en sprint. Gruppen har "daglige" standup-møter [5], hver mandag og torsdag kl 12.15, samt et sprintplanleggingsmøte ved sprintstart, og en retrospektiv ved sprintslutt. Gruppen velger dermed å gå bort fra Knibergs anbefaling om å ha bakloggssortering og sprintplanlegging som separate møter, siden ett lengre møte passer bedre med gruppemedlemmenes timeplaner. (Kniberg, s. 20).

3.2 GitLab

For at sprintplanlegging/oppgavefordeling skal gå så effektivt som mulig anbefales det å bruke en form for oppgavetavle (Kniberg, s.57). Gruppen landet dermed på å bruke de innebygde verktøyene i **GitLab**, ettersom dette samler alle verktøyene på ett sted. Hver brukerhistorie blir lagt inn som ett issue (Wu, 2018), med beskrivelse som inneholder konkrete arbeidsoppgaver på punktlisteformat. Sprinter representert av **milepæler**. Hver brukerhistorie blir gitt et tidsestimat og tilegnet en milepæl. Ved sprintstart tildeles oppgavene til gruppemedlemmene, brukerhistorier flyttes til "doing", og det opprettes en ny gren for hver brukerhistorie. Gruppen anvender muligheten til å knytte **commits** og **merge requests** til de relevante issuene. Før et issue kan lukkes skal det ligge en beskrivelse om hva som har blitt gjort for å løse det, eller eventuelt en årsak til hvorfor det ikke løses. For å sikre at koden følger standarder og oppfyller tester kjøres **CI pipelines** i GitLab før den merges til hovedgrenen. I tillegg skal koden gjennomgås og merges av noen

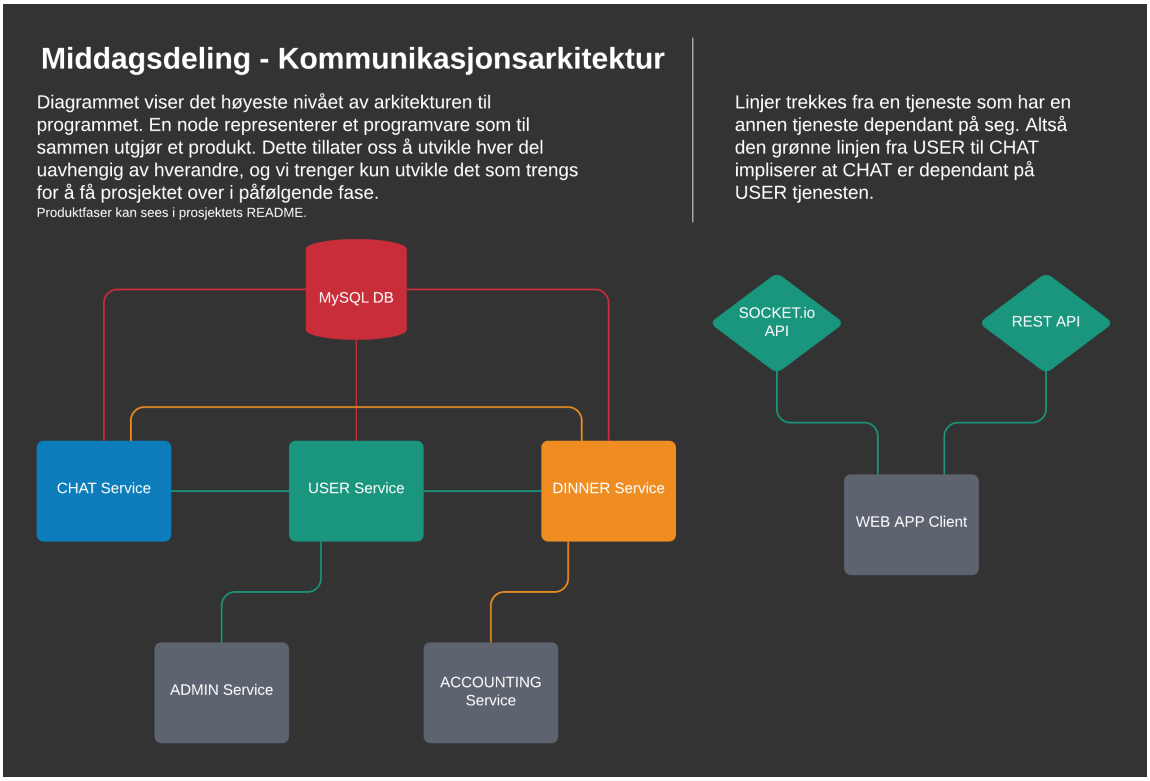

Figure 1: Kommunikasjonsarkitektur

andre enn forfatteren. Alle rutinene rundt bruk av GitLab står videre beskrevet i filen **CONTRIBUTING.md** i rotmappen på repoet.

3.3 Strukturering av kodebase

Som vist i figur 1 ønsker gruppen å utvikle fem ulike programvare som sys sammen til et helhetlig produkt. Dette vil gjøre det lettere å jobbe på ulike moduler samtidig og sørge for at vi ikke trenger å utvikle mer enn det absolutte nødvendige. Dette tror vi vil føre til en mer smidig utvikling, fordi de ulike tjenestene fungerer som en "black-box". Om det da blir klart at noe må endres kan dette trygt gjøres uten at det påvirker helheten. Strukturen på kodebasen vises i figur 2.

4 Testplan

Gruppen var raskt enig at testing spiller en viktig del av produktutviklingen, og valgte derfor å ha fokus på gode og fullstendige tester fra begynnelsen av prosjektet. Ifølge Kniberg [1] (s. 117) bør ingen funksjonalitet vurderes som "ferdig" før den har bestått alle sine tester. For at testingen ikke skulle ha mindre prioritet enn utvikling av ny funksjonalitet, bestemte gruppen seg for å prøve **Test Driven Development**. Tester blir dermed skrevet på forhånd av utvikling av funksjonalitet, som fører til at testene til en viss grad "styrer" utviklingen. Dersom gruppen klarer å planlegge prosjektet godt nok før utvikling starter, vil testene også beskrive tenkt funksjonalitet i den grad at de kan brukes som dokumentasjon til videre utvikling om det skulle være nødvendig.

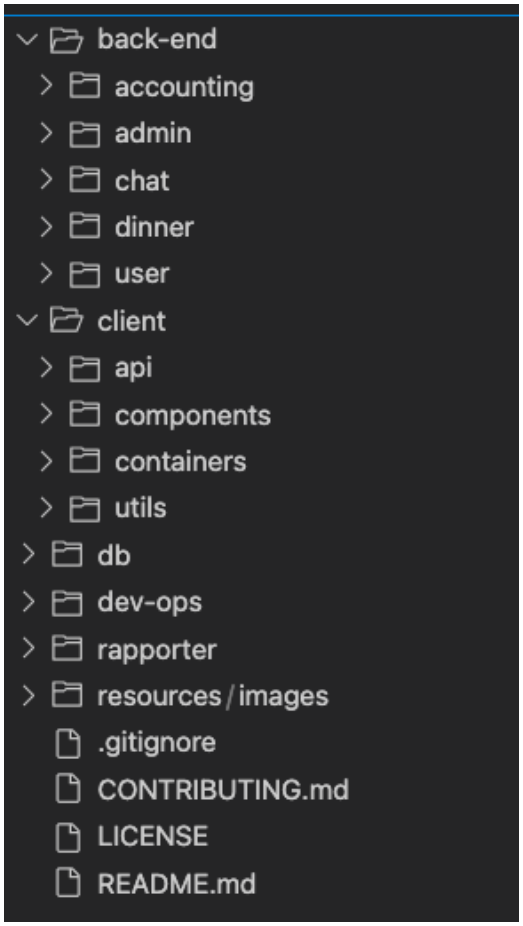

Figure 2: Mappestruktur

4.1 Verktøy

Ettersom hele produktet skal skrives i TypeScript og React, ble det naturlig for gruppen å se etter populære testrammeverk for disse teknologiene. Et gruppemedlem hadde erfaring i å bruke **Jest** til enhetstesting, og det ble derfor naturlig å velge dette rammeverket. Av samme grunn valgte gruppen å bruke **Enzyme** som rammeverk for å utføre **integrasjonstester**.

4.2 Testdekningsgrad

Som et mål på hvor mye av funksjonaliteten vi har testet, ble det nødvendig å se på en testdekningsgrad. Testrammeverkene vi har valgt kan konfigureres slik at etter tester er gjennomført, blir testdekningsgrad i de forskjellige modulene rapportert til gruppen. Ifølge Steve Cornett [3] (2013) har empiriske studier vist at å høyne testdekningsgraden til mer enn 70-80% tar lang tid. Ettersom dette prosjektet foregår over et relativt kort tidsrom, har gruppen valgt at en testdekningsgrad på minst 60% på alle moduler er ønskelig.

5 Releaseplan

Vi har valgt å ha to sprinter på rundt tre uker, i samsvar med Knibergs anbefalinger (Kniberg 2015, s. 23). På denne måten sammenfaller sprintene med henholdsvis Gjennomgang 1 og påskeferie. Vi opprettholder dermed momentum gjennom utviklingsprosessen samtidig som vi får korreksjoner gjennom møter med produkteier, fortrinnsvis ved Gjennomgang 1. Sprintene vil være fra 8. februar til 25. februar og fra 8. mars til 26. mars. Dette medfører at vi har en uke etter påske til å ferdigstille prosjektet før Gjennomgang 2, siden ingen ønsket å ha en sprint i påskeferien. Vi har lagt inn en uke til refleksjon for å vurdere hva som fungerte bra i den første sprinten og hva vi skal endre på til neste (Kniberg 2015, s. 84). En videre diskusjon rundt utviklingsstrategier finnes i **README.md** dokumentet på GitLab.

5.1 Prioritert produktkø

Som bruker ønsker jeg å:

1. Opprette profil med personopplysninger, som navn, adresse og allergier, slik at dette er lagret til neste gang. (Opprette bruker)
2. [Administrator] Muligheten for å slette brukere, slik at jeg har kontroll over nettsamfunnet. (Admin - slette)
3. [Administrator] Moderere innhold. (Admin - moderere)
4. Få oversikt over hvilke middager tilgjengelig. (Middagsoversikt)
5. Melde meg på middagsarrangementer. (Middagspåmelding)
6. Opprette middagsarrangementer, slik at jeg kan få folk på middag. (Opprette middag)
7. Endre og avlyse middagsarrangementer jeg har opprettet.
8. Filtrere alle middagsinvitasjoner som ligger ute, slik at jeg får en tilpasset middagsopplevelse.
9. Ha en chatfunksjon, slik at jeg kan kommunisere med andrebrukere.
10. Gi rating og tilbakemelding av andre brukere.
11. Legge til utgifter for middagen jeg har arrangert, slik at deltakere kan dele utgiftene.
12. [Annonsør] Ta kontakt med administrator, slik at jeg kan få publisert annonser på tjenesten.

5.2 Sprinter

Sprint 1 - Tromsø

Mål: Kunne opprette brukere og administratorer, samt melde seg på middager

For å fastsette tidsestimatene har vi benyttet oss av mageføelelsen siden vi ikke har noe tidligere arbeid å basere oss på (Kniberg 2015, s. 33).

ID	Forkortelse	Hvordan demonstrere	Timer
1	Opprette bruker	Trykk på profilikon i menybar, trykk "ny bruker", opprett ny bruker, trykk "logg inn".	45
2	Admin - slette	Trykk på profilikon i menybar, trykk "adminfunksjoner". trykk "vis brukere", trykk på bruker, trykk "slett bruker", trykk "bekreft".	10
3	Admin - moderere	Trykk på profilikon i menybar, trykk "adminfunksjoner", trykk "vis brukere", trykk på bruker, trykk "slett bruker", trykk "bekreft", trykk "moderer innhold", trykk "legg til oppslag", fyll inn felter, trykk "publiser".	15
4	Middagsoversikt	Naviger til hovedsiden ved å trykke på husikon i menybar, scroll nedover.	15
5	Middagspåmelding	Fra hovedsiden: trykk på et middagsarrangement, trykk "bli med".	15
6	Opprette middag	Trykk "+" i menybar, fyll ut felt, trykk "publiser".	20

Sprint 2 - Paris

For å imøtekomme vårt ønske om å drive smidig programmering, har vi valgt å ikke lage en detaljert plan for sprint 2. På denne måten kan vi tilpasse oss ønsker fra produkteier og best mulig vurdere veien videre basert på product backlog (Sommerville 2015, s. 86). Vi vil bruke "Yesterday weather" for å prøve å fastslå tidsestimatene for sprint 2, der vi legger til grunn Sprint 1 (Kniberg 2015, s. 30). Likevel vil vi også måtte justere tidsestimatene siden folk blir flinkere og flinkere til å benytte de nye teknologiene.

References

[1] Kniberg, H. (2015). *Scrum and XP from the Trenches 2nd edition.* C4Media.

[2] Sammenligning av antal ledige stillinger med stikkordet Vue, React og Svelt, 29. januar, 2021. *https://www.finn.no/job/browse.html*

[3] Steve Cornett. *Minimum Acceptable Code Coverage* Bullseye Testing Technology, 2006-2013. *Hentet fra https://www.bullseye.com/minimum.html 03. februar, 2021*

[4] Wu, Victor (2018). *How to use GitLab for Agile software development*, 3. februar 2021. *https://about.gitlab.com/blog/2018/03/05/gitlab-for-agile-software-development/*

[5] Sommerville, I. (2015). *Software Engineering.* Pearson Education Limited.

[6] AirBnB.io. *Airbnb JavaScript Style Guide.* 3. februar, 2021 *https://airbnb.io/javascript/react/*