Bacheloroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

John Fredrik Bendvold
Magnus Lutro Allison
Pedro Pablo Cardona

Digitalisering av landbruksmarkedet

Bacheloroppgave i Dataingeniør
Veileder: Alexander Holt

Mai 2024

NTNU
Kunnskap for en bedre verden

John Fredrik Bendvold
Magnus Lutro Allison
Pedro Pablo Cardona

Digitalisering av landbruksmarkedet

Bacheloroppgave i Dataingeniør
Veileder: Alexander Holt
Mai 2024

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

NTNU
Kunnskap for en bedre verden

Abstract

The bachelor's thesis addresses the development of Lokal, a digital marketplace aimed at bringing consumers and local producers closer together. In today's society, increasing problems with the supply of locally produced food are being observed, raising questions about current regulations of the agricultural market and supply channels. There is a clear trend of a growing desire to buy directly from producers, bypassing all the intermediaries involved in the current supply chain.

Therefore, the team developed a platform to reduce the distance between consumer and producer, to explore the following issue: ”What functionalities make a digital platform valuable for local sellers?”. The platform consists of three applications: an informative website with administrative functions for sales outlets, a mobile application for sellers, and a mobile application for consumers. These include comprehensive functionalities necessary for such a web platform, inspired by existing similar solutions such as Foodora, Wolt, and TooGoodToGo.

By conducting extensive user tests and interviews with people connected to the agricultural market, the team explored the issue by developing a platform that connects consumers directly with producers.

Sammendrag

Bacheloroppgaven omhandler utviklingen av Lokal, en digital markedsplass som har som mål å koble forbrukere og lokale produsenter tettere sammen. I dagens samfunn observeres det økende problemer med forsyningen av lokalprodusert mat, noe som har reist spørsmål ved dagens reguleringer av landbruksmarkedet og forsyningskanalene. Det er en tydelig trend der det er et økende ønske om å handle mer direkte fra produsentene, uten alle mellomleddene som inngår i forsyningskjeden.

Derfor har teamet utviklet en plattform for å redusere avstanden mellom forbruker og produsent, for å utforske følgende problemstilling: ”Hvilke funksjonaliteter gjør en digital plattform verdifull for lokale utselgere?”. Plattformen består av tre applikasjoner: en informativ nettside med administrative funksjoner for utsalg, en mobilapplikasjon for utselgere, og en mobilapplikasjon for forbrukere. Disse inneholder omfattende funksjonaliteter som er nødvendige for en slik nettplattform, inspirert av eksisterende lignende løsninger som Foodora, Wolt og TooGoodToGo.

Ved å gjennomføre omfattende brukertester og intervjuer med personer tilknyttet landbruksmarkedet, har teamet utforsket problemstillingen ved å utvikle en plattform som kobler forbruker direkte med produsent.

Forord

Bacheloroppgaven er skrevet i forbindelse med emnet ”IDATT2900 Bacheloroppgave” våren 2024 og utgjør den avsluttende oppgaven for dataingeniørstudiet ved NTNU i Trondheim.

Oppgaven er utarbeidet av teamet selv og ble til etter egne observasjoner av et potensielt behov i samfunnet for en løsning som effektivt kobler forbrukere direkte med gårdsutsalg og mindre lokale utsalg. I emnet ”INGT2300 Ingeniørfaglig systemtenkning” ble det utarbeidet en rapport med samme tematikk, som støtter påstanden om at en slik løsning potensielt ville være et verdifullt tilskudd i samfunnet, spesielt som en tilleggsnæring for bønder. Ved å utføre en rekke intervjuer og brukertester med personer tilknyttet gårdsdrift, vil teamet identifisere hvilke elementer som er best egnet i et system designet for å redusere avstanden mellom produsent og forbruker.

Vi ønsker å takke vår veileder, Alexander Holt, for hans kontinuerlige veiledning og rådgivning gjennom hele bachelorperioden. Vi ønsker også å gi en spesiell takk til Sissel Langørgen fra Høstadsand gård og Christel Ossletten fra Voll gård for deres gode samarbeid og innsikt i gårdsmarkedet. Vi vil også takke alle andre som har bidratt med brukertester og tilbakemeldinger under utviklingen av Lokal.

Trondheim, 20. mai 2024.

John Fredrik Bendvold | Magnus Lutro Allison | Pedro Cardona

Oppgavebeskrivelse

Oppgaven ”Digitalisering av landbruksmarkedet” utforsker hvordan et nyskapende konsept kan digitalisere utvalgte aspekter ved landbruksmarkedet. Konseptet og systemet, kalt ”Lokal”, er en digital markedsplass for handel av lokalproduserte varer. Målet med systemet er å øke synligheten for mindre produsenter, og samtidig skape et aktivt handelsnettverk som fasiliterer direkte-salg.

Problemstillingen for oppgaven, opprinnelig formulert som i vedlegg A, ble omformulert under oppstarten av prosjektet. Den endelige formuleringen er presentert i avsnitt 1.1 Problemstilling.

Oppgavebeskrivelsen, som vist i vedlegg A, trekker frem overordnede krav til systemet. For det første skal utselgere kunne opprette et utsalg og lage en digital katalog over tilbudte produkter. Videre skal forbrukere kunne oppdage utsalg i nærheten. For å være et fullstendig system, skal også forbrukeren kunne bruke Lokal for å gjennomføre kjøp av produkter fra et lokalt utsalg. Selve oppgaveteksten er lik for sluttproduktet, der eneste endring er at det refereres til systemet Lokal som én enkelt ”applikasjon”, hvor det heller ble bestemt at systemet skulle bestå av tre applikasjoner i samspill etter utviklet visjonsdokument (se vedlegg C).

Spesifiserte krav, både funksjonelle og ikke-funksjonelle, er vedlagt med opprinnelig formulering i visjonsdokumentet (se vedlegg C). Under prosessen ble imidlertid nye funksjonelle krav identifisert, og andre forkastet. Resultatene av dette fremkommer i kapittel 4.2.5.

Oppdragsgiveren for oppgaven er den ene utvikleren på teamet, John Fredrik Bendvold. Selve oppgaveteksten ble utarbeidet av teamet i samarbeid høsten 2023, og sendt inn av John Fredrik Bendvold.

Innhold

Abstract i

Sammendrag ii

Forord iii

Oppgavebeskrivelse iv

Innhold v

Figurer x

Tabeller xii

Ordliste xiii

Akronymer xv

1 Introduksjon og relevans 1
- 1.1 Problemstilling . . . 1
- 1.2 Rapportstruktur . . . 2

2 Teori og relevant litteratur 3
- 2.1 Vitenskapsteori . . . 3
  - 2.1.1 Kvantitativ metode . . . 3
  - 2.1.2 Kvalitativ metode . . . 3
  - 2.1.3 Observasjon . . . 3
  - 2.1.4 Intervju . . . 3
    - 2.1.4.1 Strukturert intervju - spørreundersøkelse . . . 4
    - 2.1.4.2 Dybdeintervju . . . 4
- 2.2 Design . . . 4
  - 2.2.1 Brukergrensesnitt og brukeropplevelse . . . 4
  - 2.2.2 WCAG . . . 5
  - 2.2.3 Passordløst design . . . 5
- 2.3 Teknologi . . . 6
  - 2.3.1 BaaS . . . 6
  - 2.3.2 Database . . . 7
  - 2.3.3 SQL . . . 7
    - 2.3.3.1 Transaksjoner i SQL . . . 8

2.3.3.2 Databaseutløser 8
2.3.4 RLS 8
2.3.5 Webhooks 9
2.4 Utviklingsprosess 9
2.4.1 Versjonskontrollsystem 9
2.4.2 Kontinuerlig integrasjon og leveranse 10
2.4.3 Smidig programvareutvikling 10
2.4.3.1 Scrum 11
2.4.4 Metodikker for designprosess 12
2.4.4.1 Design Sprint 12

3 Metode **14**
3.1 Forskningsmetode 14
3.1.1 Design and Creation 14
3.2 Utviklingsmetodikk 15
3.2.1 Forprosjektplan og visjonsdokument 15
3.2.2 Smidig programvareutvikling og Scrum 16
3.2.2.1 Scrum-teamet 16
3.2.2.2 Inndeling i Scrum-sprinter 16
3.2.2.3 Fokus på brukerhistorier 16
3.2.2.4 Product Backlog 17
3.2.2.5 Arbeid mot spesifikke sprintmål 17
3.2.2.6 Sprint Backlog 18
3.2.2.7 Sluttfasen av Scrum-sprinten 19
3.2.2.8 Issues 19
3.2.2.9 Brukertesting og MVP 20
3.2.3 Designsprint 20
3.3 Teknologivalg 22
3.3.1 Nettsiden 23
3.3.1.1 React - Next.js 23
3.3.1.2 Cypress 23
3.3.2 Mobilapplikasjonene 24
3.3.2.1 Valg av kryssplattform-løsning 24
3.3.2.2 React Native - Expo 25
3.3.2.3 Jest 25
3.3.3 Backend as a Service med Supabase 26
3.3.3.1 PostgreSQL 26
3.3.3.2 RLS 27
3.3.3.3 Edge Function 27
3.3.3.4 Webhooks 27
3.3.3.5 Sanntidskommunikasjon 27
3.3.3.6 OAuth 28

3.3.3.7 E-posttjenester 28
3.3.3.8 Lagring 29
3.3.3.9 Sammenligning 29
3.3.4 Versjonskontrollsystem og Git 29
3.3.4.1 GitLab 29
3.3.5 Eksterne tjenester 30
3.3.5.1 GitHub Pages 30
3.3.5.2 Google-API 30

4 Resultater **31**
4.1 Forskningsresultater 31
4.1.1 Markedsundersøkelse 31
4.1.2 Brukertester 33
4.2 Utviklingsresultater 35
4.2.1 Overordnet systemarkitektur 35
4.2.2 Mobilapplikasjon for forbrukere 38
4.2.2.1 Innlogging 38
4.2.2.2 Utforsk 39
4.2.2.3 Kart 40
4.2.2.4 Profil 41
4.2.2.5 Hjelp 41
4.2.2.6 Utsalg 42
4.2.2.7 Kasse 43
4.2.2.8 Ordre 44
4.2.3 Nettside og mobilapplikasjon for utselgere 45
4.2.3.1 Nettsidens forside 45
4.2.3.2 Innlogging som utselger 46
4.2.3.3 Registrering av utsalg 47
4.2.3.4 Kontrollpanel 48
4.2.3.5 Administrering av utsalg 49
4.2.3.6 Økonomi 51
4.2.3.7 Ordreoversikt 52
4.2.3.8 Produkter 54
4.2.3.9 Ansatte 56
4.2.4 Nettside for administratorer 57
4.2.5 Funksjonelle krav 57
4.2.6 Ikke-funksjonelle krav 60
4.2.6.1 Funksjonalitet (F) 60
4.2.6.2 Brukervennlighet (U) 60
4.2.6.3 Pålitelighet (R) 60
4.2.6.4 Ytelse (P) 60
4.2.6.5 Støtte (S) 61

4.2.6.6 Andre krav (+) 61
4.3 Administrative resultater 61
4.3.1 Smidig utvikling 61
4.3.2 Prosess 63
4.3.3 Arbeidsfordeling 63
4.3.4 Dokumentasjon 63

5 Diskusjon **64**
5.1 Forskningsresultater 64
5.1.1 Markedsundersøkelse 64
5.1.2 Brukertester 65
5.2 Utviklingsresultater 66
5.2.1 Mobilapplikasjon for forbrukere 66
5.2.1.1 Design 66
5.2.1.2 Funksjonalitet 66
5.2.2 Nettside og mobilapplikasjon for utselgere 67
5.2.2.1 Design 67
5.2.2.2 Funksjonalitet 67
5.2.3 Nettside for administratorer 68
5.2.4 Begrensninger ved systemet 69
5.2.4.1 Kostnader 69
5.2.4.2 Ansvarsfraskrivelse 69
5.2.4.3 Utviklingstid 69
5.2.4.4 Krav om bruk av mobilenheter 70
5.3 Administrative resultater 70
5.3.1 Smidig utivkling 70
5.3.2 Prosess 70
5.3.3 Arbeidsfordeling 71

6 Konklusjon og videre arbeid **72**
6.1 Problemstilling og utviklingsfokus 72
6.2 Videre arbeid 73
6.2.1 Allergener og oppbevaringsinformasjon 73
6.2.2 Betalingsløsning 73
6.2.3 Hentetidspunkt 73
6.2.4 Tilgjengeliggjøring av brukerdata 73
6.2.5 Skalering og optimalisering 74

Samfunnspåvirkning **75**

Bibliografi **76**

Vedlegg A Opprinnelig oppgavebeskrivelse **i**

Vedlegg B Forprosjektsplan iv

Vedlegg C Visjonsdokument v

Vedlegg D Brukertest med Sissel vi

Vedlegg E Brukertest med Christel vii

Vedlegg F WCAG-rapport for nettsiden viii

Vedlegg G WCAG-rapport for mobilapplikasjonene ix

Vedlegg H Bærekraftig matforsyning: En direkte forsyningskanal mellom bonde og forbruker x

Vedlegg I Prosjekthåndbok xi

Vedlegg J Systemdokumentasjon xii

Vedlegg K Kravdokumentasjon xiii

Figurer

2.1 Representasjon av et passordløst system som utnytter epost-innlogging. Bildet er hentet fra frontegg.com den 13.05.2024. 6
2.2 Illustrasjonen viser mulig struktur for funksjonaliteter i BaaS. Bildet er hentet fra back4app.com den 30.04.2024. 7
2.3 Definisjon med oversikt over de tre hovedkomponentene som bygger opp en databaseutløser i SQL-3. Bildet er hentet fra springer.com den 05.05.2024 (Lee mfl., 2005). 8
2.4 Illustrasjon av essensielle nøkkelsteg i en prosess som følger smidig utviklingmetodikk. Bildet er hentet fra krasamo.com den 01.05.2024. . . . 10
2.5 Oversikt over teamet og prosessen i Scrum. Bildet er hentet fra linkedin.com den 01.05.2024. 12
2.6 Overordnet innhold for de fem dagene i en Design Sprint. Hentet fra thesprintbook.com den 16.04.2024. (Knapp og Zeratsky, 2024) 12

3.1 Metodemodell basert på Oates (Oates, 2006). 14
3.2 Figur 8.1 fra Oates, 2006 som representerer en mulig fremgangsmåte for bruk av ”Design and Creation”. Hentet den 04.05.2024. 15
3.3 Bildet er fra GitLab *IssueBoards* som viser et utsnitt av Product Backlog brukt i utviklingen. 17
3.4 Endelig versjon av Gantt-diagrammet brukt for planlegging av milepæler i prosjektet, lagd ved hjelp av onlinegantt.com. 18
3.5 Bilde av GitLab milepæl 6 som representerer Sprint Backlog for sprint 6. 18
3.6 Eksempel-bilde av et issue som brukt i prosjektet. 20
3.7 Skisse som viser et forslag til hvordan kjøp kan foregå i applikasjonen, se vedlegg K, seksjon 7 for flere eksempler. Skissen ble tegnet under designaktiviteten i fasen ”Dag 4: å prototype” i designsprinten. 21
3.8 Wireframe som viser et prototypeforslag til kjøp av produkter i endelig Forbrukerapplikasjon. Skissen ble tegnet for å oppsummere ”Dag 5: valider” i Design Sprinten. 22
3.9 Overordnet oversikt over kompilering av Flutter- og React Native-kode. . 25
3.10 Illustrasjon av Row-Level Security / Radnivåsikkerhet (RLS) for databasetabellen ”Arbeider”. 27
3.11 Samtlige integrerte autentiseringssystemer tilgjengelig i Supabase. . . . 28

4.1 Stolpediagram med hvilke grunner folk oppgir for at de ikke handler mer lokale råvarer. Fra Google Forms-meningsmåling som inngikk i markedsundersøkelsen. Flere svaralternativer var mulig å velge. 32

4.2 Sektordiagram med oversikt over om folk er bosatt i et område med tilgang til lokale utsalg. Fra Google Forms-meningsmåling som inngikk i markedsundersøkelsen. 33
4.3 Systemarkitektur-diagram for brukere og komponenter som inngår i Lokal som system. 35
4.4 Klient-tjener-diagram for kommunikasjonsarkitekturen i Lokal. 36
4.5 Representasjon av ulike autorisasjonsnivåer for forskjellige brukertyper i Lokal. 37
4.6 Forsiden for innlogging i forbrukerapplikasjonen, der bakgrunnen er en gjentakende video-snutt. 38
4.7 Utforsk-siden i forbrukerapplikasjonen. 39
4.8 Kart-siden i forbrukerapplikasjonen. 40
4.9 Brukerhjelp i forbrukerapplikasjonen. 41
4.10 Utsalg i forbrukerapplikasjonen. 42
4.11 Til venstre kassa i forbrukerapplikasjonen, til høyre kvittering på e-post. 43
4.12 Ordre i forbrukerapplikasjonen: til venstre en ny uhentet ordre midt i en konfetti-animasjon, til høyre en eldre hentet ordre. 44
4.13 Forsiden til nettsiden. 45
4.14 Innlogging som utselger på nettsiden. 46
4.15 Registrering av nytt utsalg på nettsiden. 47
4.16 Kontrollpanelet på nettsiden. 48
4.17 Brukerveiledning for kontrollpanelet på nettsiden. 48
4.18 Administrering av eget utsalg på nettsiden. 49
4.19 Endring av åpningsstatus i utselgerapplikasjonen. 50
4.20 Økonomioversikt på nettsiden 51
4.21 Regnskapet for eget utsalg på nettsiden. 51
4.22 Ordreoversikt på eget utsalg på nettsiden. 52
4.23 Ordre i utselgerapplikasjonen: til venstre er uhentet ordre, til høyre er eldre ordre. 53
4.24 Produktoversikt på nettsiden. 54
4.25 Redigering av et produkt på nettsiden. 54
4.26 Produkter i utselgerapplikasjonen: til venstre er en produktoversikt, til høyre er registreringen av et nytt produkt. 55
4.27 Oversikt over ansatte på nettsiden. 56
4.28 Administrator-kontrollpanelet på nettsiden. 57
4.29 Bildet fra Supabase viser gjennomsnittlig lastetid fra 13.04.2024 til 13.05.2024. 61
4.30 Representasjon av issues-distribusjon per kategori som et sektordiagram. 62
4.31 Antall issues per sprint. 62
4.32 Teamets arbeidstimer gruppert etter kategori fra timelisten. 63
4.33 Strukturen til Wikien. 63

Tabeller

3.1 Gruppering av kategoriene for issues brukt i prosjektplanleggingen. . . . 19

4.1 Oversikt over funksjonaliteter identifisert som ønsket i systemet under de totalt ti brukertestene som ble utført. 35
4.2 Resultat over alle funksjonelle krav, og om de ble implementert i systemet. 59

Ordliste

ACID er et sett med egenskaper som er avgjørende for databasetransaksjonssystemer, som sikrer pålitelig prosessering. ACID står for Atomisitet, Konsistens, Isolasjon, og Varighet:

- **Atomisitet** sikrer at en transaksjon utføres som en udelelig enhet; den fullføres helt, eller ikke i det hele tatt.
- **Konsistens** opprettholder databasens korrekthet ved å sikre at kun gyldige data lagres etter transaksjonens avslutning.
- **Isolasjon** bestemmer hvordan transaksjoner vises for andre brukere og systemer, slik at dataendringer under en enkelt transaksjon ikke er synlige før transaksjonen er fullført.
- **Varighet** garanterer at en gang en transaksjon er fullført, vil endringene være permanente, selv i tilfelle av systemfeil.

Disse prinsippene er fundamentale for å sikre databasenes integritet og stabilitet (Hansen og Mallaug, 2008). 8, 26

databaseutløser er en sekvens av instruksjoner som blir utført etter en spesifikk hendelse i en tabell i databasen. (Lee mfl., 2005). x, 8, 9, 27, 28, 43, 53, 60

Design Sprint er en design-metodikk med fem gitte faser introdusert av Google Ventures. Målet er å produsere idéer og prototyper raskt. (Araújo mfl., 2019). x, 12, 13, 20–22, 42, 63, 65, 72

Design Thinking er et design-konsept med mål om å reflektere kontinuerlig og tilpasse et design til brukeren av et system. Konseptet går ut på forsøke å besvare spørsmål som 'Hvilke individer er brukere av produktet?' (Araújo mfl., 2019, Waidelich mfl., 2018). 12, 13, 22

Edge Function er autonome funksjoner som utføres i små, distribuerte maskiner lokalisert geografisk nærmere brukerne. Disse funksjonene blir instansiert i et virtuel miljø i en generisk server; denne instansen avsluttes etter at funksjonene er fullført. (Hall og Ramachandran, 2019). 27, 36, 43, 53, 60

inkrement er ett enkelt element eller en funksjonalitet som gir verdi til produktet. Hvert enkelt inkrement er bygd opp slik at det legger til ytterligere verdi til det forrige inkrementet av produktet (Schwaber og Sutherland, 2020). xiv, 17, 33

issue omfatter en beskrivelse av den enkleste formen for en oppgave eller arbeid som må utføres knyttet til et prosjektproblem. (GitHub, 2024). x–xii, 16–20, 61, 62

magic-link En midlertidig, unik URL som brukes for autentisering. En magic-link sendes vanligvis til brukerens e-postadresse, og ved å klikke på den kan brukeren logge inn uten å måtte oppgi et passord. (Matiushin og Korkhov, 2021). 28, 38, 46

milepæl refererer til et betydelig steg i en utviklingsprosess, som markerer fremgang og inneholder inkrementer for produktet. (Rolstadås, 2019). x, 11, 17–20, 70

native refererer til programvare eller applikasjoner som er spesifikt utviklet for å kjøre på en bestemt plattform, eller i et bestemt operativsystem. Ved å utnytte plattformens egne API og optimaliseringer, vil ytelsen og integrasjonen ofte forbedres. (Gillis, 2024). 24

Product Backlog er en logg over alle funksjonalitetene som vil gi verdi til ferdig produkt. Product Backlog endres dynamisk dersom teamet legger til eller fjerner identifiserte problemer eller inkrementer (Schwaber og Sutherland, 2020). x, 11, 16, 17, 19

Product Owner er en viktig rolle i Scrum som tar kontakt med kunden og identifiserer ønskede funksjonaliteter for et produkt. Fra informasjonen utvikles en spesifikk Product Backlog (Schwaber og Sutherland, 2020). 11, 16

Scrum master er en lederrolle i Scrum som sørger for at Scrum-metodikken overholdes gjennom utviklingsprosessen (Schwaber og Sutherland, 2020). 11, 16

Sprint Backlog er en spesifikk type Product Backlog som brukes for å organisere og loggføre endringer mot sprintmålet for en bestemt sprint (Schwaber og Sutherland, 2020). x, 11, 18, 19

universell utforming innebærer design av produkter og tjenester som er brukbare av alle mennesker, uavhengig av alder, funksjonshemming eller andre faktorer. Målet er å maksimere tilgjengeligheten og brukervennligheten, og fjerne unødvendige barrierer. (Lid, 2024). 4, 5, 60

webhook er et hendelsebasert kommunikasjonssytem, hvor tilbyder sender nyttelast til en forhåndsdefinert destinasjon etter en bestem hendelse har inntuffet. (Varmey mfl., 2023). 9, 27, 43, 53, 60

wireframe er en skisse av forutsett design for et produkt eller system. x, 21, 22

Akronymer

API	Application Programming Interface / Programmeringsgrensesnitt.
BaaS	Backend as a Service / Backend som en tjeneste.
CI/CD	Continuous Integration and Continuous Delivery / Kontinuerlig integrasjon og kontinuerlig leveranse.
MVP	Minimum viable product / Enkleste brukbare produkt.
RLS	Row-Level Security / Radnivåsikkerhet.
SMTP	Simple Mail Transfer Protocol.
SQL	Structured Query Language / Strukturert spørrespråk.

1. Introduksjon og relevans

Med utgangspunkt i dagens markedssituasjon i Norge, undersøker prosjektet hvordan et system kan effektivisere handelen mellom forbrukere og lokale produsenter. Dette har blitt gjort ved å utvikle en digital plattform, kalt ”Lokal”. Basert på landbruksmarkedet, hvor flere bønder har gårdsutsalg, identifiserte teamet et potensielt samfunnsbehov for en digital plattform som fjerner alle ledd i forsyningskjeden mellom bonden og forbrukeren. Den digitale plattformen ble foreslått som en tilleggsnæring for bøndene. Idéen var at mindre produsenter ville få muligheten til å skape en ekstra inntektskilde. Deretter ble omfanget økt til at også andre utselgere, som antikvitetsutsalg og loppemarked, skulle kunne selge varer i systemet, derav bruken av ordet ”utselger”.

Overordnet sett har prosjektet hatt to fokusområder; utforskning av markedet, og utvikling av et system som møter behovene identifisert. Metodegrunnlaget for å kartlegge markedet ble basert på teamets prosjektarbeid i emnet ”Ingeniørfaglig Systemtenkning” (INGT2300) våren 2024, som resulterte i rapporten *Bærekraftig matforsyning: En direkte forsyningskanal mellom bonde og forbruker* (se vedlegg H). Rapporten i INGT2300 vinklet problemstillingen mot å undersøke matsikkerhetsfordelene ved et system som Lokal. Likevel ble flere elementer fra prosjektet i INGT2300 også relevante for bachelorprosjektet, og dermed indirekte for prosessen.

For bachelorrapporten er imidlertid utviklingen av selve systemet Lokal i fokus. Målet har vært å utvikle et helhetlig system, med bred funksjonalitet. Rask utvikling av funksjonaliteter har derfor vært en essensiell faktor ved valg av teknologi og metode.

1.1 Problemstilling

Problemstillingen for bacheloroppgaven lyder dermed som følger: *Hvilke funksjonaliteter gjør en digital plattform verdifull for lokale utselgere?*

Problemstillingen forsøker å danne et siktepunkt for utviklingen, ved å presisere at utforskningen underveis skal implementere verdifulle funksjonaliteter rettet mot utselgere som målgruppe.

1.2 Rapportstruktur

Bachelorrapportens innhold er fordelt over følgende kapitler:

1. Introduksjon og relevans presenterer bakgrunn og relevans for bacheloroppgaven, samt problemstillingen for rapporten.

2. Teori og relevant litteratur omhandler teori som er essensiell for å forstå teknologien og konsepter som ligger til grunn for utviklingen av systemet.

3. Metode forklarer hvilke metodiske fremgangsmåter som er nyttet underveis, samt konkrete teknologivalg for prosjektet.

4. Resultater er lagt frem som en oversikt over faktiske resultater fra utviklingen. Kapitlet er delt inn i forskningsresultater, utviklingsresultater og administrative resultater.

5. Diskusjon tar for seg hvorfor resultatet ble slik det ble, og ser på potensielle begrensninger ved systemet.

6. Konklusjon og videre arbeid ser på hvordan oppgaven er løst i forhold til problemstillingen, og presenterer hvilke funksjonaliteter og løsninger som er naturlige videreutviklinger for det utviklede systemet.

Samfunnspåvirkning begrunner hvorfor konseptet og systemet er relevant i samfunnet.

2. Teori og relevant litteratur

Dette kapittelet presenterer det teoretiske grunnlaget for bachelorprosjektet. Fokuset er på å forklare konsepter og teknologi som er relevant for utforskningen, utviklingen og selve produktet.

2.1 Vitenskapsteori

2.1.1 Kvantitativ metode

Kvantitativ metode er en forskningsmetode som brukes for innsamling og analyse av kvantitative data. Kvantitative data er alle typer observasjoner som kan representeres i tallform (Grønmo, 2023). Innsamlingsmetoder for kvantitativ data inkluderer spørreundersøkelser, eksperimenter og strukturerte observasjoner. Dataene som samles inn gjennom disse metodene, kan analyseres ved bruk av ulike statistiske teknikker. (Aanesen, 2020)

2.1.2 Kvalitativ metode

Kvalitativ metode er en forskningsmetode som anvendes for innsamling og analyse av kvalitative data (Grønmo, 2023). Kvalitative data karakteriseres av observasjoner som ikke kan kvantifiseres direkte i tallform. Innsamlingsmetoder for kvalitative data inkluderer typisk teknikker som dybdeintervjuer med åpne spørsmål og detaljerte observasjoner. Disse teknikkene tillater forskere å oppnå innsikt på en systematisk måte. (Creswell, 2012)

2.1.3 Observasjon

Observasjon som undersøkelsesmetode innebærer direkte eller indirekte overvåkning av brukerens adferd. Målet er å avdekke autentiske reaksjoner på produkter, tjenester, eller markedsføringstiltak. Observasjon kan gjennomføres enten åpent, hvor deltakerne er informert om at de blir observert, eller skjult hvor deltakerne er uvitende. Skjult observasjon forsøker å hente adferdsdata om naturlig bruksmønster uten ekstern påvirkning. (Elisabeth Thoresen og Sundbye, 2021)

2.1.4 Intervju

Intervju er en forskningsmetode som brukes til å samle inn informasjon ved å stille spørsmål til deltakere. Denne metoden kan tilpasses både kvalitative og kvantitative

forskningsdesign, og brukes ofte for å få innsikt i menneskelige opplevelser, holdninger og atferd (Dalen, 2013).

2.1.4.1 Strukturert intervju - spørreundersøkelse

Et strukturert intervju eller spørreundersøkelse er en intervjumetode som brukes til å samle inn kvantitative data. Denne typen intervju er basert på et spørreskjema, hvor innholdet, formen og rekkefølgen av spørsmålene er fastsatt. De fleste spørsmål har tydelige svaralternativer. Resultatene fra spørreundersøkelsen kan normaliseres, analyseres og presenteres ved hjelp av ulike diagrammer, som sektordiagrammer eller søylediagrammer. (Malt og Grønmo, 2023)

2.1.4.2 Dybdeintervju

Et dybdeintervju er en semistrukturert intervjumetode som samler inn kvalitativ data, med mål om å oppnå dypere innsikt i individers holdninger, tanker og følelser. Dette kan gjøres gjennom individuelle intervjuer. Under intervjuene kan en moderator ha ansvar for å lede samtalen og sikre relevant informasjon. (Elisabeth Thoresen og Sundbye, 2021)

2.2 Design

2.2.1 Brukergrensesnitt og brukeropplevelse

Hensikten med et godt brukergrensesnitt (UI) er å effektivt formidle informasjon mellom brukeren og systemet, og dermed forenkle informasjonsflyten. Dersom en overveldende mengde informasjon presenteres til brukeren på én gang, kan dette gjøre informasjonen og brukergrensesnittet uoversiktlig. En måte å forenkle informasjonsflyten er ved å introdusere et modulært design, som presenterer informasjon på en mer selektiv og oversiktlig måte for brukeren (Suto mfl., 2007). I faglitteraturen er det bredt anerkjent at et godt brukergrensesnitt er essensielt for å sikre en intuitiv brukeropplevelse. Et effektivt brukergrensesnitt bør dermed være estetisk tiltalende samtidig som det opprettholder klarhet og simpelhet, slik at brukeren kan navigere og forstå interaksjonsmulighetene som tilbys. (McKay, 2013)

Brukeropplevelsen (UX) er viktig for et godt utformet brukergrensesnitt. Brukeropplevelse omfatter det helhetlige inntrykket en bruker får ved interaksjon med et system. Ifølge boken ”Don’t make me think” av Steve Krug, er en nøkkelegenskap ved en god brukeropplevelse at systemet er intuitivt, selv for en utenforstående part. Dette kan oppnås ved å implementere et logisk og brukervennlig design. Velkjente elementer som standardsymboler og at systemet følger universell utforming, gjør det mer gjenkjennelig og lettlært. Et design som skal gi en god brukeropplevelse må også ta hensyn til brukerens behov og preferanser, og tilrettelegge for en problemfri og tilfredsstillende interaksjon. Dette inkluderer alt fra oppsettet av brukergrensesnittet til

den emosjonelle responsen brukeren opplever. Målet er å skape en opplevelse som er både intuitiv og engasjerende. (Krug, 2014,Coelho, 2022)

Brukergrensesnittet og brukeropplevelsen, har en sentral rolle i design av multiplattformsystemer. I slike systemer der brukere får tilgang til produkter eller tjenester ved å utnytte forskjellige teknologiske enheter, må både brukergrensesnitt- og brukeropplevelse-design tilpasses for å sikre en konsistent og sømløs opplevelse på tvers av alle plattformene. Designet av brukergrensesnittet bør være fleksibelt og responsivt for å tilpasse seg ulike skjermstørrelser og orienteringer, mens design med hensyn på brukeropplevelse må vurderes nøye for hver plattform, og ta høyde for ulike brukervaner. (McKay, 2013, Krug, 2014)

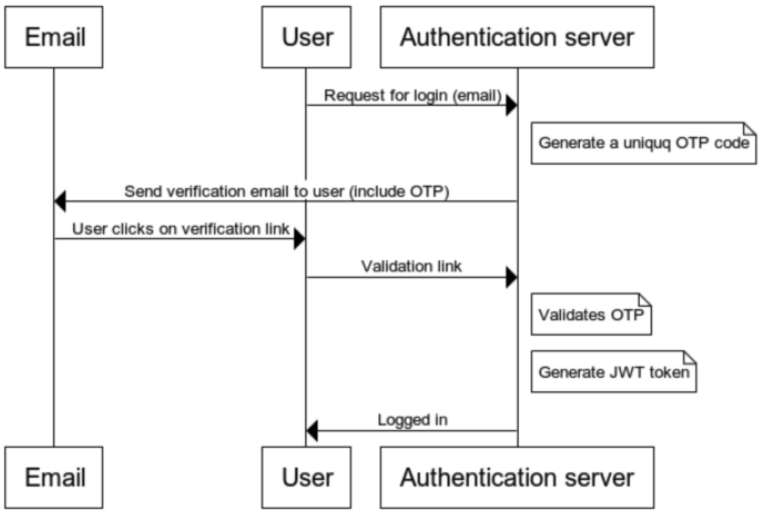
2.2.2 WCAG

Web Content Accessibility Guidelines (WCAG) er et sett med standarder som gjelder for nettsider og mobilapplikasjoner. Retningslinjene er definert av standardiseringsorganisasjonen World Wide Web Consortium (W3C). Målet med WCAG er å tilgjengeliggjøre webinnhold for alle, med spesielt fokus på personer med ulike funksjonsnedsettelser. Eksempler er visuelle, auditive, verbale og kognitive funksjonsnedsettelser. WCAG skal sørge for at alle kan få tilgang til og bruke internett på en rettferdig måte, og forsøker med andre ord sikre universell utforming av webinnhold. (Initiative (WAI), 2024)

I Norge er det obligatorisk for privat og offentlig sektor å følge minst 35 WCAG-krav ved utvikling av webtjenester. Ved å følge disse retningslinjene bidrar organisasjoner til et mer inkluderende digitalt miljø. Samtidig kan den generelle brukeropplevelsen for alle som besøker deres nettsteder forbedres. Noen av WCAG-kravene som må oppfylles inkluderer: god kontraststyrke, bildetekster, minimert bruk av blinkende animasjoner, og uavhengighet av lyd og video for funksjonalitet. («WCAG-standarden | Tilsynet for universell utforming av ikt», 2024)

2.2.3 Passordløst design

Passordløst design er en systemtilnærming der brukerne autentiserer seg uten å bruke passord. Passordløse systemer fungerer ved å generere et nøkkelpar, bestående av offentlig og privat nøkkel. Offentlig nøkkel utstedes av autentiseringstjenesten, som kan være en ekstern server (Vipps), en applikasjon (Gmail) eller via en nettside (Facebook), under registrering og lagres sammen med privat nøkkel, som kun er tilgjengelig ved bruk av biometrisk signatur, automatisk generert token eller andre autentiseringmetoder som ikke involverer passord. Brukere blir bedt om å oppgi sin offentlige identifikasjon (brukernavn, telefonnummer, e-postadresse eller annen registrert identifikasjon) i de mest populære implementasjonene, og deretter fullføre autentiseringsprosessen ved å bevise sin identitet gjennom en akseptert type autentiseringsmetode. (Parmar mfl., 2022)



Figur 2.1: Representasjon av et passordløst system som utnytter epost-innlogging. Bildet er hentet fra frontegg.com den 13.05.2024.

Studier indikerer at sikrere og lengre passord, gjør det vanskeligere for brukere å huske det. Derfor kan et passordløst system forbedre brukeropplevelsen samtidig som det opprettholder systemets sikkerhet. (Gao mfl., 2018)

2.3 Teknologi

2.3.1 BaaS

Backend as a Service (BaaS) er en tjenestemodell hvor alle backend-funksjonaliteter i en web- eller mobilapplikasjon håndteres av ekstern leverandør. Dette inkluderer for eksempel databehandling, lagring, brukerautentisering, server-side logikk, push-varslinger, og integrasjon med sosiale nettverk. (Carranza-García mfl., 2016). Slik funksjonalitet er utformet for å være universelt anvendelig, med det formål om å etablere et backend-system som kan samhandle sømløst med diverse frontend-arkitekturer. Dette oppnås ved bruk av generiske APIer for kommunikasjon med BaaS-løsningen. (Saldamli mfl., 2021)

Figur 2.2: Illustrasjonen viser mulig struktur for funksjonaliteter i BaaS. Bildet er hentet fra back4app.com den 30.04.2024.

2.3.2 Database

En database er en organisert samling av lagrede data som følger en spesifikk modell, kjent som en databasemodell. Databasemodellen er en struktur som definerer hvordan data lagres i databasen. For å opprette og administrere databaser brukes ulike applikasjoner, kalt databasesystemer. (Kapittel 1 i Hansen og Mallaug, 2008)

En av de mest brukte databasemodellene er relasjonsmodellen, hvor data lagres i rader, også kalt poster, med attributter i todimensjonale tabeller. I motsetning til relasjonsmodellen finnes NoSQL-modeller, som omfatter databasemodeller som ikke har relasjoner, og som gir en høy grad av fleksibilitet i lagringen av data. (Bratbergsengen og Mallaug, 2024)

2.3.3 SQL

Structured Query Language (SQL) er et standardisert programmeringsspråk som brukes til å administrere og endre data i relasjonelle databaser. SQL tilbyr grunnleggende operasjoner på data lagret i en database, kjent som CRUD. CRUD innebærer å kunne opprette (CREATE), lese (READ), oppdatere (UPDATE) og slette (DELETE) data. SQL tillater også oppretting og modifisering av databasestruktur, samt kontroll av tilgang til data. (Alan, 2020, Sharif, 2022)

Utover grunnleggende operasjoner, implementerer SQL konsepter fra relasjonsalgebra som et teoretisk grunnlag for relasjonelle databaser. Dette inkluderer muligheten til å konstruere nye relasjoner gjennom operasjoner som forening, union, snitt, og differanse. Disse operasjonene lar brukere kombinere og manipulere datasett fra ulike tabeller basert på forskjellige kriterier. Relasjonsalgebra i SQL tilbyr dermed en mekanisme for dataabstraksjon og spørringsoptimalisering. (Hansen og Mallaug, 2008)

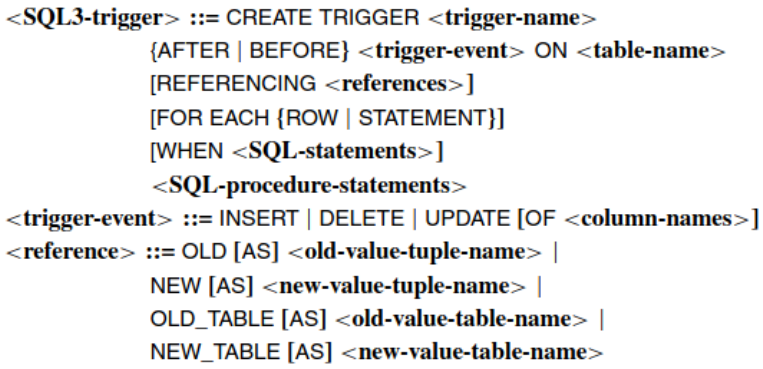
2.3.3.1 Transaksjoner i SQL

I SQL representerer en transaksjon en sekvens av databasemanipulasjoner som behandles som en enkelt, udelelig operasjon. Dette prinsippet, kjent som "alt eller ingenting", innebærer at enten blir alle operasjonene i transaksjonen gjennomført suksessfullt, eller så blir alle tilbakestilt til databasens tilstand før transaksjonen ble initiert. Dette sikrer at ingen delvise endringer lagres, noe som er essensielt for å opprettholde databasens integritet. Transaksjoner er også kritiske for å håndtere samtidige tilganger til databasen. (Kapittel 12 i Alan, 2020)

Transaksjoner i SQL er basert på ACID-egenskapene. ACID er utformet for å garantere at databasetransaksjoner utføres på en pålitelig og effektiv måte. Disse prinsippene er avgjørende for å sikre at databehandlingen er robust mot feil og uregelmessigheter. (Haerder og Reuter, 1983)

2.3.3.2 Databaseutløser

Databaseutløsere, også kjent som *event-condition-action*s, er delt inn i tre deler slik navnet antyder: hendelsen som utløser databaseutløseren, betingelsen som må oppfylles for at databaseutløseren skal aktiveres, og handlingen som utføres når betingelsen er oppfylt, som vanligvis er en databasefunksjon. (Lee mfl., 2005)



Figur 2.3: Definisjon med oversikt over de tre hovedkomponentene som bygger opp en databaseutløser i SQL-3. Bildet er hentet fra springer.com den 05.05.2024 (Lee mfl., 2005).

2.3.4 RLS

RLS er en sikkerhetsmekanisme som muliggjør kontroll over tilgang til data på radnivå innenfor en SQL-database. Dette oppnås ved å definere sikkerhetsregler som bestemmer hvilke data brukere eller grupper kan se eller manipulere basert på deres identitet, rolle eller andre selvdefinert krav. Målet med RLS er å sikre data på en pålitelig

måte, ved å begrense tilgangen direkte i databasen. Dermed vil applikasjonslogikken og brukergrensesnittet være støttet av dette ekstra laget med sikkerhet. (Randolph, 2024)

2.3.5 Webhooks

Webhooks er et hendelsesbasert kommunikasjonssystem som består av to deler: en Webhooks-tilbyder og en Webhooks-konsument. Tilbyder sender nyttelast til en forhåndsdefinert destinasjon etter en bestemt hendelse har inntruffet, ofte som følge av en databaseutløser. Konsument mottar deretter nyttelast og behandler informasjonen videre. (Varmey mfl., 2023)

Webhooks er en videreutvikling av web-polling. I motsetning til polling, som krever at konsument regelmessig sender forespørsler for å sjekke etter endringer, sender Webhooks automatisk informasjon når en hendelse oppstår. Dette fører til mer effektiv bruk av ressurser og reduserer forsinkelser i kommunikasjonen. Webhooks muliggjør dermed sanntidskommunikasjon mellom systemer, og kan dermed utnyttes av applikasjoner som krever umiddelbar oppdatering og respons. (Biehl, 2017)

2.4 Utviklingsprosess

2.4.1 Versjonskontrollsystem

Et versjonskontrollsystem er et verktøy som brukes for å administrere og holde orden på ulike versjoner av kildekode. Bruk av versjonskontroll muliggjør lagring, sporing og sammenligning av endringer over tid, og tillater ulike utviklere å arbeide på forskjellige deler av prosjektet samtidig uten å overskrive hverandres bidrag. Ved å bruke et versjonskontrollsystem kan utviklere også gå tilbake til tidligere versjoner av kodebasen hvis det oppstår problemer. På dette vis er versjonskontrollsystemet et verktøy for å sikre integritet og effektivt samarbeid i programvareprosjekter over tid. (Zolkifli mfl., 2018).

Overordnet sett kan versjonskontrollsystem kategoriseres som én av to hovedtyper. De to vanligste typene det skilles mellom er sentraliserte versjonskontrollsystemer og distribuerte versjonskontrollsystemer. (Brindescu mfl., 2014)

Sentraliserte versjonskontrollsystemer fungerer ved at en enkelt server inneholder alle versjonene av et prosjekt. Dette betyr at endringer og versjonshistorikk er lagret på en sentral lokasjon, noe som gir enkel tilgang og oversikt for alle teammedlemmer. Ulempen med dette systemet er at det krever tilgang til serveren for å kunne arbeide effektivt, noe som kan være en begrensning i situasjoner uten internettforbindelse eller ved servernedetid.

På den andre siden tillater distribuerte versjonskontrollsystemer at hver utvikler har

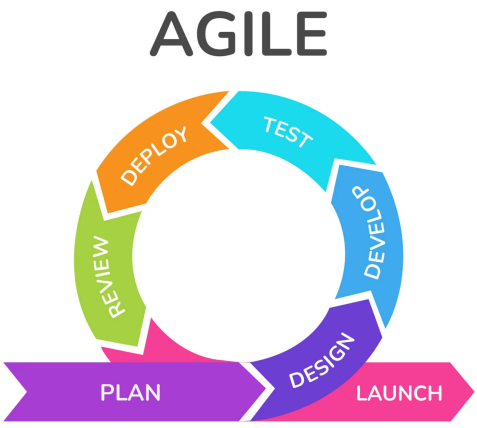
en lokal kopi av hele kode-repositoriet, inkludert hele endringshistorikken. Dette betyr at utviklere kan arbeide uavhengig av hverandre, gjøre commits, reversere endringer, og utforske prosjektets historikk uten å være avhengig av tilgang til en sentral server. Endringer mellom utviklere kan synkroniseres, noe som gir en robust struktur for samarbeid og versjonskontroll selv under nettverksbrudd. Et distribuert oppsett fremmer dermed en mer desentralisert arbeidsflyt. (Khleel og Nehéz, 2020)

2.4.2 Kontinuerlig integrasjon og leveranse

I programvareutvikling er kontinuerlig integrasjon og kontinuerlig leveranse (CI/CD) en praksis for å automatisk verifisere integrasjon av nye funksjonaliteter og utføre leveranse av et produkt. Kontinuerlig integrasjon (CI) innebærer ifølge IEEE som regel både automatisering og rapportering av bygging, testing og deteksjon av feil og sikkerhetshull i kildekoden til et produkt. Kontinuerlig leveranse (CD) publiserer produktet til produksjon automatisk etter CI har bekreftet at kildekoden er i henhold til kravene satt opp av utviklingsteamet. (Shahin mfl., 2017)

2.4.3 Smidig programvareutvikling

Smidig programvareutvikling er en metodikk som omfatter et sett av prinsipper og praksiser for utvikling av digitale løsninger. Hovedmålet ved smidig utvikling er å maksimere kundeverdi samtidig som man effektiviserer utviklingsprosessen for å redusere både arbeidsmengden og tidsforbruket. Tilnærmingen vektlegger fleksibilitet og kontinuerlig forbedring. I praksis benyttes smidig programvareutvikling ofte for å sikre effektivt samarbeid mellom tverrfunksjonelle team og kunder for å produsere programvare. («Principles behind the Agile Manifesto», 2024)



Figur 2.4: Illustrasjon av essensielle nøkkelsteg i en prosess som følger smidig utviklingmetodikk. Bildet er hentet fra krasamo.com den 01.05.2024.

Smidig metodikk innebærer blant annet planlegging, utvikling, testing, og leveranse av programvare. At metodikken er ”smidig” vil si at det legges vekt på at det skal kunne gjøres tilpasninger etter kundens behov, utviklernes kapasitet eller marked fortløpende. Ved å dele opp prosessen i mindre perioder kalt iterasjoner eller sprinter, med veldefinerte milepæler, sikrer smidig utvikling at produktet kontinuerlig utvikler seg og forbedres gjennom hele utviklingsprosessen. (Alsaqqa mfl., 2020)

2.4.3.1 Scrum

Scrum et et spesifikt rammeverk innenfor smidig programvareutvikling. Fokuset i Scrum er å produsere verdi så raskt som mulig mens utviklingen kan gjøre smidige tilpasninger underveis. Metodikken er et resultat av konsolidert erfaring basert på empiri, som forklart av ScrumGuides.org:

> *Scrum is founded on empiricism and lean thinking. Empiricism asserts that knowledge comes from experience and making decisions based on what is observed. Lean thinking reduces waste and focuses on the essentials.* (Schwaber og Sutherland, 2020)

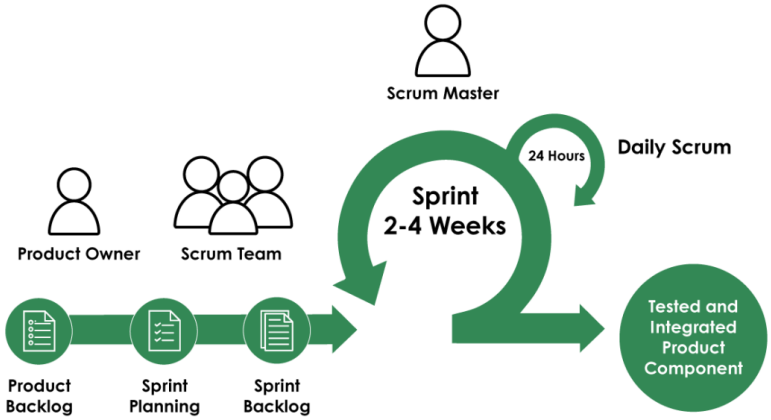
Dermed er Scrum et veiledende rammeverk med én til flere iterative arbeidsperioder som ofte omtales som sprinter. Et viktig aspekt ved rammeverket er at hver sprint skal resultere i konkret verdiskaping, i henhold til milepæler satt for sprinten. I tillegg består Scrum-rammeverket av flere distinkte eventer, inkludert sprintplanlegging ved starten av en sprint, en Daily Standup for å kommunisere status og hindringer på daglig basis, og en retrospektiv samt en Sprint Review ved slutten av sprinten. (Schwaber og Sutherland, 2020)

Sprintplanleggingen innebærer å diskutere hvorfor sprinten er viktig, å velge ut hva som skal gjøres, og å bestemme hvordan teamet skal få dette til. I planleggingen er det vanlig å velge ut elementer fra Product Backlog som skal utføres i sprinten ved å overføre dem til Sprint Backlog.(Schwaber og Sutherland, 2020)

Sprint-retrospektiv vil si at teamet vurderer hva som gikk bra og hva som kunne gått bedre, mens Sprint Review går ut på at teamet har en gjennomgang av hva som ble oppnådd i sprinten. I Sprint Review skal teamet sørge for at det gjøres tilpasninger i forhold til hva som ble oppnådd for å forbedre utførelsen av neste sprint. (Schwaber og Sutherland, 2020)

Målet med eventene er å sikre en strukturert fremgangsmåte med gitte gjøremål. Ifølge ScrumGuides.org skal eventene føre til smidige tilpasninger underveis gjennom inspeksjon av eget arbeid, som gir innsikt i nødvendige endringer. (Schwaber og Sutherland, 2020)

Oppbygningen av Scrum-teamet består av en Product Owner, en Scrum master og utviklere som samlet sett skal jobbe mot å oppnå målet om et tilfredsstillende sluttprodukt for kunden. (Schwaber og Sutherland, 2020 og Sachdeva, 2016)

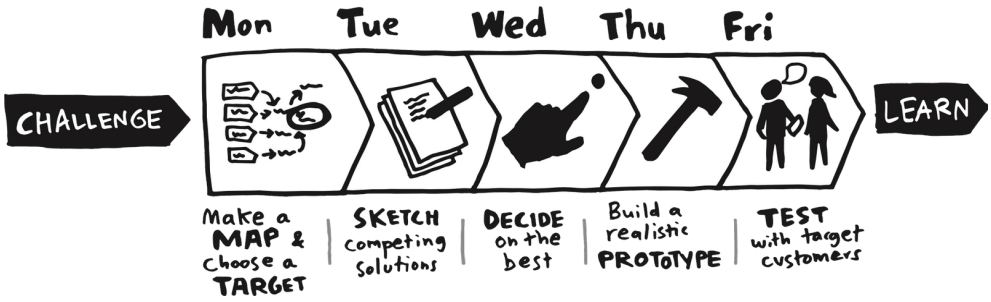


Figur 2.5: Oversikt over teamet og prosessen i Scrum. Bildet er hentet fra linkedin.com den 01.05.2024.

2.4.4 Metodikker for designprosess

Med bakgrunn i Design Thinking har det oppstått en rekke metodemodeller for programvaredesign. Felles for metodemodellene er at de er bygd opp av definerte *prosessteg* gitt som faser med spesifikke mål og aktiviteter. Et annet kjennetegn er at modellene forsøker å designe et produkt ved hjelp av alt fra idéer til håndfaste prototyper. "Designet" som refereres til innebærer imidlertid mer enn det visuelle aspektet ved produktet. Like viktig er for eksempel forespeilet bruksmønster og planlegging. På dette vis er metodikken et virkemiddel for å reflektere rundt det helhetlige sluttproduktet. (Araújo mfl., 2019, Higuchi og Nakano, 2017)

2.4.4.1 Design Sprint



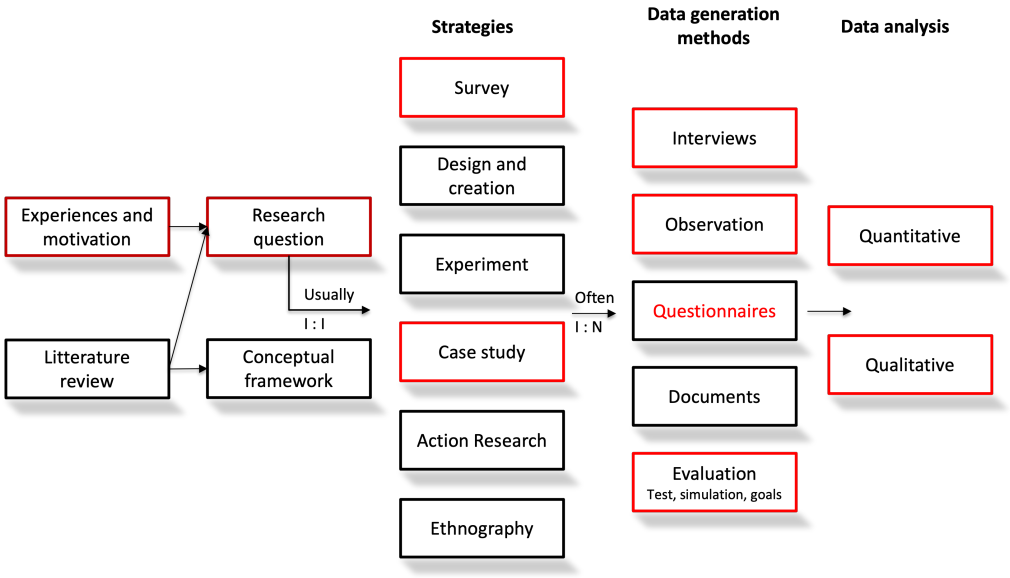
Figur 2.6: Overordnet innhold for de fem dagene i en Design Sprint. Hentet fra thesprintbook.com den 16.04.2024. (Knapp og Zeratsky, 2024)

En moderne metodemodell for utvikling av design er Design Sprint. Design Sprint kombinerer sprintoppsettet samt smidig utviklingsmetodikk fra Scrum, med Design Thinking (Higuchi og Nakano, 2017). Som prosess forsøker Design Sprint å skape forslag til løsninger på samfunnskritiske problemer over et kort tidsrom. Målet med de fem fasene er derfor å generere en rekke prototyper og utføre idémyldring med kunden, for å så bestemme etter dag fem om produktdesignet skal videreutvikles eller ikke. De fem fasene i Design Sprint er satt opp slik som vist av illustrasjonen over. Hovedfaktoren som skiller Design Sprint fra andre alternativer er fokuset på å fullføre fasene over kun fem dager, og dermed generere et designforslag raskt. (Araújo mfl., 2019, Ventures, 2024)

3. Metode

Metode-kapittelet legger frem de metodiske tilnærmingene teamet har gjort underveis i prosessen. Kapittelet trekker inn metoder fra ulike faser i prosessen, som metodikk for utviklingen av selve produktet, og teknologivalg. Målet med kapittelet er å gi innsyn i grunnlaget for valgene som ble tatt.

3.1 Forskningsmetode



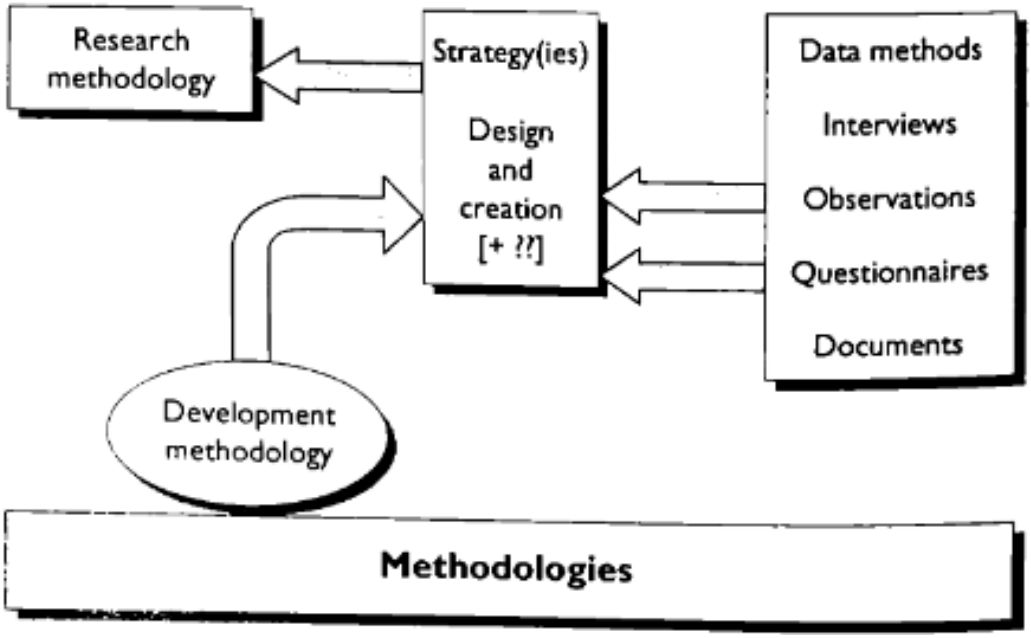
Figur 3.1: Metodemodell basert på Oates (Oates, 2006).

3.1.1 Design and Creation

Hovedforskningsmetoden i prosjektet har vært *Design and Creation*. Metodemodellen kan brukes for utvikling av teknologiske systemer, slik som Lokal. *Design and Creation* formidler en prosess med innsamling, analyse og presentasjon av informasjon i en akademisk kontekst, og fokuserer dermed på å demonstrere at produktet er verdifullt i praksis. Teamet har praktisert *Design and Creation* ved å utføre brukertester underveis i prosessen. (Oates, 2006)

Hovedtypen av data som samles inn gjennom metoden *Design and Creation* er kvalitative data. Dataene hentes fra observasjoner, dybdeintervjuer, dokumentasjon og

praktisk erfaring. Informasjonen har blitt brukt til å både forbedre systemet Lokal og for å rapportere akademiske resultater. (Oates, 2006)



Figur 3.2: Figur 8.1 fra Oates, 2006 som representerer en mulig fremgangsmåte for bruk av "Design and Creation". Hentet den 04.05.2024.

En potensiell ulempe med *Design and Creation* sammenlignet med andre metoder som *Survey*, som er en forskningsmetode som fokuserer på å hente representative data fra store deler av befolkningen, er at akademiske resultater avhenger av et spesifikt produkt. Dette medfører at resultatene er betinget av egenskapene og ferdighetene til teamet. (Oates, 2006)

3.2 Utviklingsmetodikk

3.2.1 Forprosjektplan og visjonsdokument

I henhold til prosessmalen for bachelorprojektet utviklet teamet en forprosjektplan. Teamet utnyttet denne for å kartlegge mål og rammer for prosjektet, samt generell organisering og gjennomføring. Se vedlegg B for å lese forprosjektplanen. Parallelt utarbeidet teamet et omfattende visjonsdokument. Visjonsdokumentet spesifiserer tiltenkt funksjonalitet for sluttproduktet i dybden. Målet med å komponere et visjonsdokument var å definere målgruppene til Lokal. Ved å gjøre dette tidlig, hadde hele teamet en samlet forståelse av hvem som skulle bruke det tiltenkte systemet. Hensikten var altså å skape enighet om sluttproduktet. Prosessen bygde på idémyldring og diskusjon av de foreslåtte funksjonalitetene. Videre ble funksjonalitetene dokumentert

som brukerhistorier i visjonsdokumentet. Visjonsdokumentet er lagt ved som vedlegg C.

3.2.2 Smidig programvareutvikling og Scrum

Det smidige utviklingsrammeverket Scrum har hatt en sentral rolle i prosjektet. Teamet valgte Scrum ettersom intensjonen bak metodikken er å effektivt utvikle komplekse systemer ved å utnytte veldefinert og godt utprøvd metodikk. I tillegg konkluderte teamet at Scrum passet godt med prosjektets fokus på rask leveranse av mange funksjonaliteter.

3.2.2.1 Scrum-teamet

Teamet bak Lokal har bestått av tett samarbeid mellom utviklere med spissede roller. Teammedlemmene har alle vært utviklere i hele prosjektperioden. Innenfor utvikling, har teammedlemmenes ekspertise vært et grunnlag for ansvarsinndelingen, der én utvikler hadde overordnet ansvar for backend-funksjonalitet, en annen for frontend og design-implementasjon og den siste personen sørget for frontend-integrasjon mot backend samt testing. I tillegg var én person på teamet Scrum master og en annen Product Owner.

3.2.2.2 Inndeling i Scrum-sprinter

I samsvar med Scrum ble prosessen delt inn i syv sprinter på to til fire uker hver. Med grunnlag i identifiserte behov og funksjonaliteter fra visjonsdokumentet, komponerte teamet en omfattende Product Backlog som inneholdt alle planlagte brukerhistorier for systemet Lokal. En Product Backlog ble utarbeidet ved bruk av GitLab IssueBoards i repositoriet til nettsiden (se Lokal IssueBoard).

3.2.2.3 Fokus på brukerhistorier

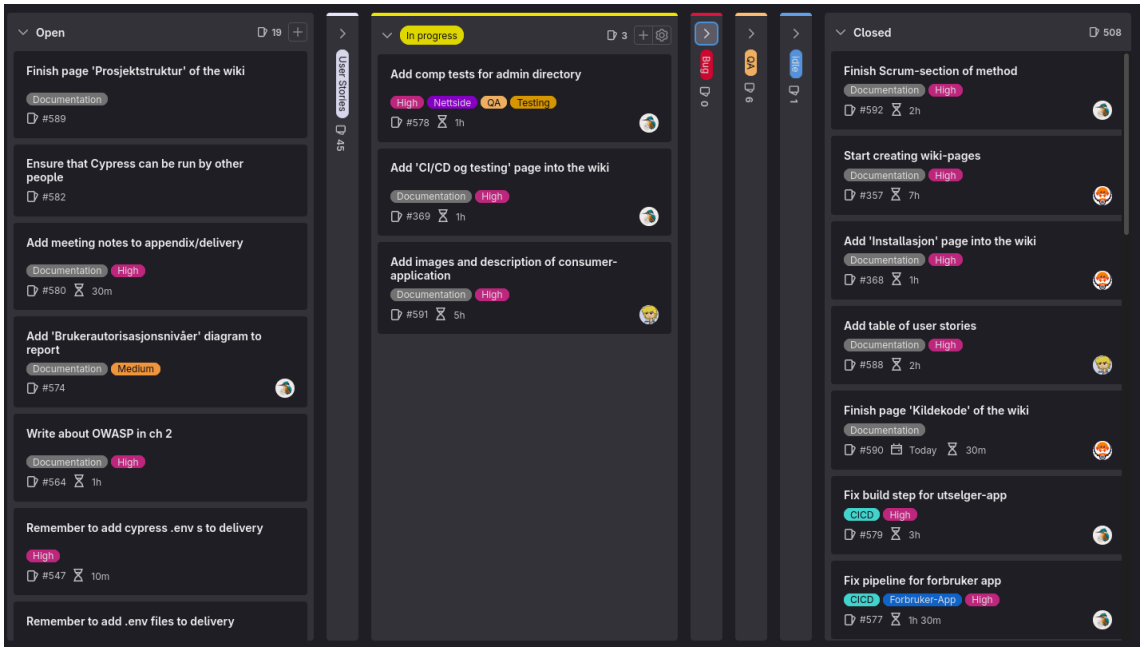
En brukerhistorie slik brukt i prosjektet er en presist definert funksjonalitet som et produkt tilbyr sluttbrukerene. Formålet med å definere denne funksjonaliteten er å danne et bilde av hvilke verdier systemet skulle tilby sluttbrukeren (Kapittel 1 i Cohn, 2013). Brukerhistoriene ble definert som issues, med en egen kategori-merkelapp i GitLab kalt ”User Stories”. Dermed hadde teamet en oversiktlig måte å overse progresjonen mot funksjonelle elementer satt opp som krav for det fullførte systemet.

Teamet identifiserte de ulike brukergruppene i sluttsystemet som utselgere, forbrukere og administratorer. Gruppene dannet grunnlaget for vidt forskjellige behov ved sluttproduktet. Fremgangsmåten gikk dermed ut på å nøye planlegge hvilken type bruker som en brukerhistorie skulle skape verdi for. Slike målrettede funksjonaliteter satte fokus på å utvikle en mer tilpasset brukeropplevelse for hver gruppe. I tillegg var dette en logisk tilnærming ettersom Lokal skulle bestå av tre separate applikasjoner

i et system. Målet baserte seg også på å isolere brukergruppenes interaksjon til ulike applikasjoner (Kapittel 2 i Cohn, 2013).

3.2.2.4 Product Backlog

En Product Backlog ble utnyttet for å administrere issues på overorndet nivå for hele prosjektet. I Product Backlog ble ønsket funksjonalitet ved sluttproduktet definert ved brukerhistorier gitt som issues. Disse ble lukket eller forkastet ved Sprint Reviews. Brukerhistoriene i Product Backlog ble forkastet ved å sette merkelappen "Abandoned", eller markert fullført med merkelappen "Completed User Story". Dermed ble brukerhistoriene oppdatert i henhold til smidig utvikling ved nye identifiserte behov og endring av prioritering for hva sluttproduktet skulle tilby av funksjonalitet.



Figur 3.3: Bildet er fra GitLab *IssueBoards* som viser et utsnitt av Product Backlog brukt i utviklingen.

3.2.2.5 Arbeid mot spesifikke sprintmål

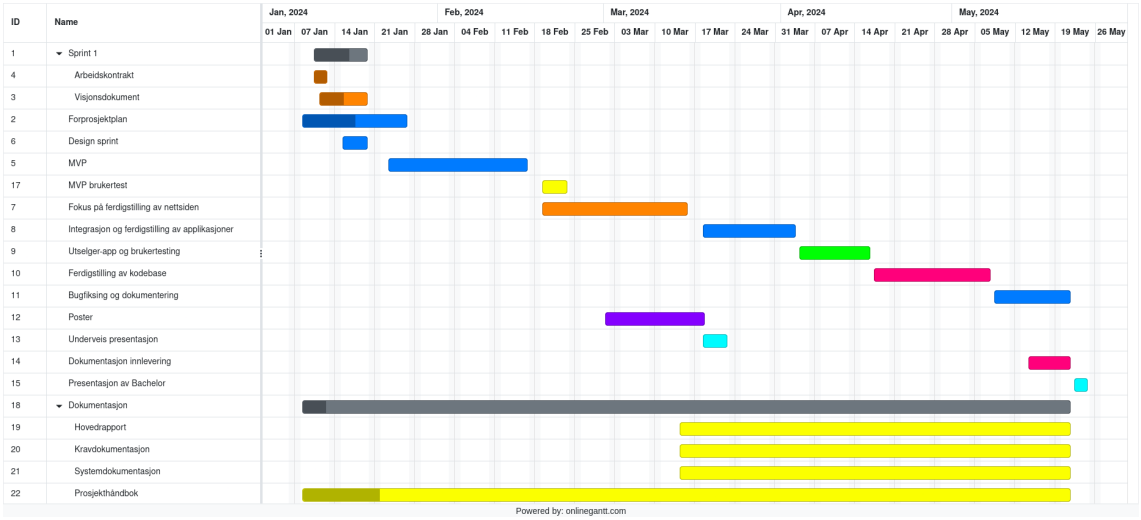
For hver sprint ble et sprintmål, kalt milepæl i GitLab, definert. Målet var å skape en presis beskrivelse av hvilke "elementer av verdi", eller inkrementer, sprinten skulle sikte på å implementere.

Hver morgen holdt teamet daglige statusmøter kalt Daily Standups. På Daily Standup gikk hvert medlem gjennom hva de hadde gjort den foregående dagen, eventuelle problemer, og hva som skulle gjøres fremover. Målet med Daily Standup var å forsikre kommunikasjon ved et bestemt tidspunkt på dagen. På denne måten kunne eventuelle

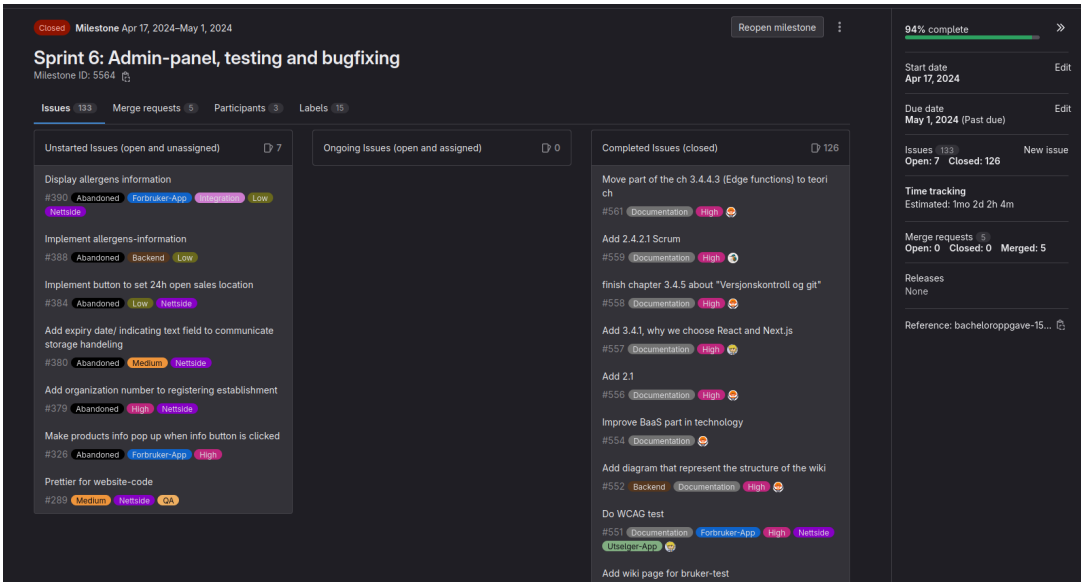
problemene diskuteres, og bli forsøkt løst ved hjelp av kollektiv kunnskap.

3.2.2.6 Sprint Backlog

En Sprint Backlog ble brukt som grunnlag for å definere sprintmålet og overvåke fremdriften av arbeidet i hver sprint. GitLab IssueBoard ble brukt i praksis som en Sprint Backlog ved å angi hvilken milepæl hver issue tilhørte. Under er en oversikt over milepælene for de ulike sprintene.



Figur 3.4: Endelig versjon av Gantt-diagrammet brukt for planlegging av milepæler i prosjektet, lagd ved hjelp av onlinegantt.com.



Figur 3.5: Bilde av GitLab milepæl 6 som representerer Sprint Backlog for sprint 6.

3.2.2.7 Sluttfasen av Scrum-sprinten

I slutten av hver enkelt sprint hadde teamet et arbeidsmøte, der en Sprint Review, samt retrospektiv ble utført. Sprint Review gikk ut på at teamet metodisk evaluerte hvordan forrige sprint gikk i forhold til sprintmålet (milepæl) ved å reflektere over Burndown-diagrammet, og oppdatere status for Product Backlog. På dette vis tilpasset teamet systemet med smidige endringer underveis. I retrospektiv-delen av arbeidsmøtet ble teammedlemmenes innsats og arbeid vurdert, for å forbedre arbeidet i de kommende sprintene (se oversikten over sprintplaner og Sprint Reviews i vedlegg I, seksjon 5).

3.2.2.8 Issues

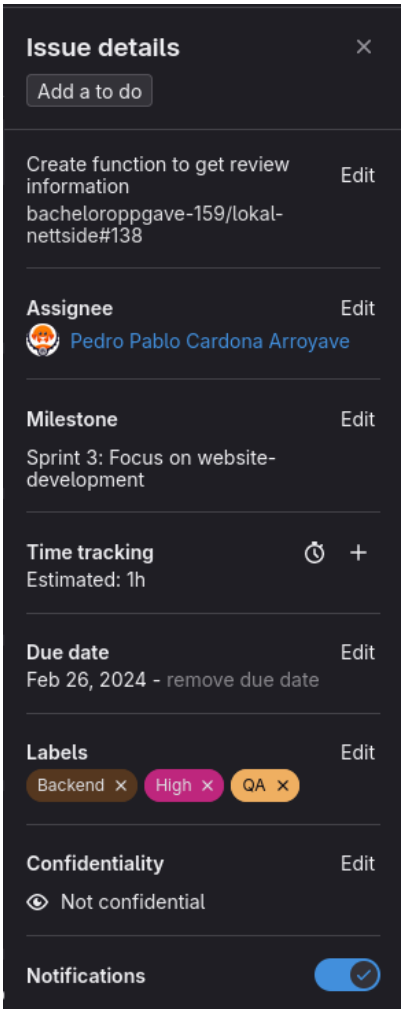
Teamet opprettet en rekke issues som skulle ta for seg mindre eller større delmål i utviklingsprosessen. På denne måten sørget teamet for oppfølging av bestemte arbeidsoppgaver (GitHub, 2024). Issues ble delt inn i ulike kategorier basert på problemområde og prioritet. Kategoriseringen ble utført aktivt, noe som sikret teamet oversikt i organiseringen av arbeidet. Kategoriene som ble bestemt i starten, og lagt til underveis var:

Gruppe	**Kategorier**
Prosjektstatus	Abandoned, In progress, Completed User Stories, Idle
Prioritering	High, Medium, Low
System-komponent	Backend, Nettside, Forbruker-App, Utselger-App
Kvalitetssikring	Bug, CICD, Integration, QA
Dokumentasjon	Documentation, User Stories, User test

Tabell 3.1: Gruppering av kategoriene for issues brukt i prosjektplanleggingen.

Grunnen til at disse kategoriene ble valgt, var for å støtte bruken av GitLab IssueBoards som både en overordnet Product Backlog, og som hver enkelt Sprint Backlog. Dermed ble ”User Stories” og ”Completed User Stories” brukt for å overvåke progresjonen mot sluttproduktet, for å skille disse issues fra vanlige, mindre, issues brukt innad i sprintene.

Issues ble satt opp ved å estimere antall timer som en bestemt oppgave ville kreve, samt å sette en datofrist. Dermed skapte issues en mer håndfast plan som var med på å drive prosjektet fremover. Samtidig utnyttet teamet issues for å kommunisere og lettere kunne prioritere og fordele ressursene til teamet. Følgende bilde viser et eksempel på et issue slik de ble opprettet i GitLab IssueBoards.



Figur 3.6: Eksempel-bilde av et issue som brukt i prosjektet.

3.2.2.9 Brukertesting og MVP

Metoden for å sikre tilbakemelding under utviklingen var å produsere funksjonalitet som kunne testes fortløpende ved brukertester. Brukertestene ble satt som dybdeintervjuer for innsamling av kvalitative data. På grunn av dette var milepæl for sprint 2 å ha fungerende MVP for applikasjonene. Deretter ble funksjonalitene i MVP-versjonen av Lokal testet. Se resultatene for brukertestingen for hvordan teamet endret ønskede funksjonaliteter ved systemet i kapittel 4.1.2.

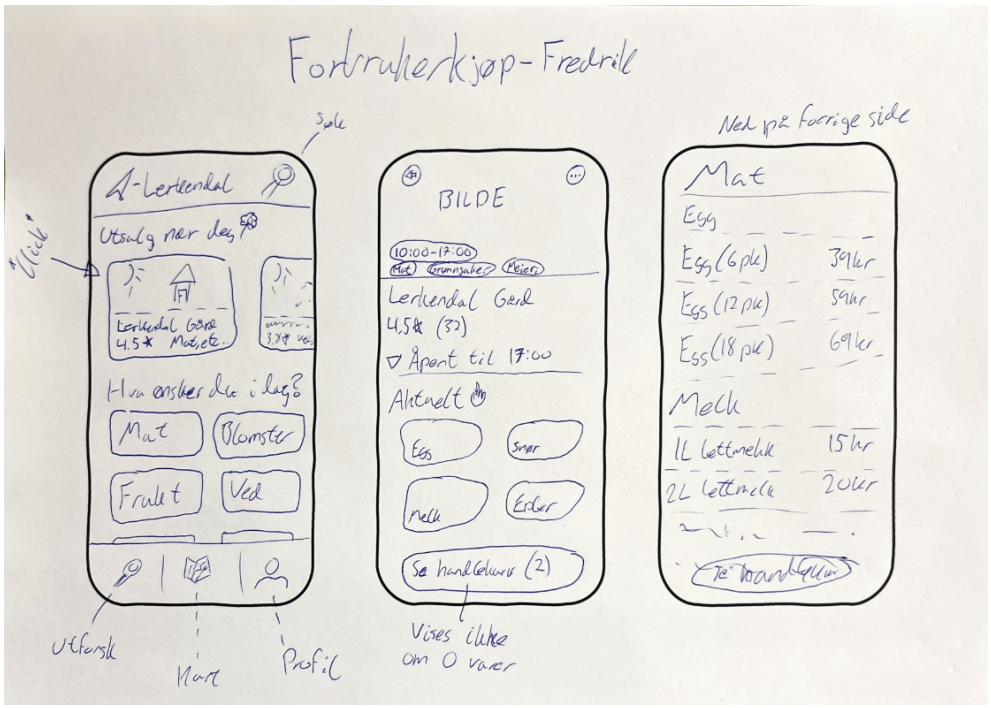
3.2.3 Designsprint

For å skape et solid grunnlag for designet fulgte teamet en prosess inspirert av Google Ventures' Design Sprint (Araújo mfl., 2019). Fordelen for teamet ved å følge Design Sprint-metodikk var å sikre grundig refleksjon rundt sluttsystemet via en strukturert

prosess, som vist av figur 2.6. Dette ble gjennomført for å unngå å starte utviklingen uten enighet om et mål for designet.

Rask utvikling av funksjonaliteter som videre kunne testes av reelle brukere, ble identifisert som et av de viktigste punktene for å danne et virkelighetstilpasset og relevant produkt. Som følge av dette, ble det bestemt at designsprinten skulle inneholde skissering av wireframes. Disse skulle resultere i et forslag til designet av MVPen. I tillegg ble validering av teamets idéer et viktig punkt. Alt i alt var planen å følge Design Sprint-oppsettet, men fordele fasene for ulike applikasjoner i systemet Lokal over flere dager.

Hovedsakelig, ble fasen "Dag 1: forstå og definer" utført fra oppstarten av semesteret, ved at hvert teammedlem noterte egne tanker om hvilke applikasjonssider som måtte designes for å møte problemomfanget til oppgaven. Dermed, ved starten av designsprinten, startet teamet den første dagen med å utføre fasene "Dag 2: diverger" og "Dag 3: bestem". Teamet idémyldret sammen, og bestemte så hvilke applikasjonssider som inngå i systemet. De valgte sidene ble deretter brukt som utgangspunkt i en designaktivitet for idémyldring som vist på bildet under.

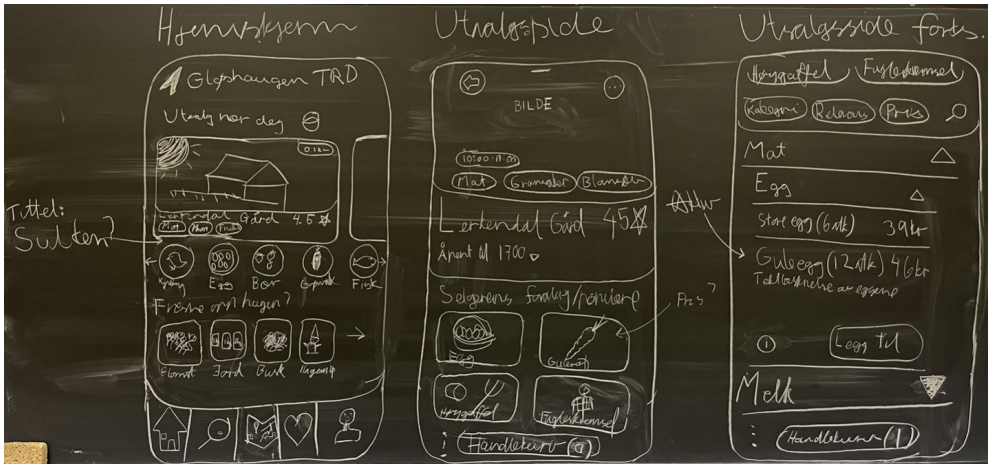


Figur 3.7: Skisse som viser et forslag til hvordan kjøp kan foregå i applikasjonen, se vedlegg K, seksjon 7 for flere eksempler. Skissen ble tegnet under designaktiviteten i fasen "Dag 4: å prototype" i designsprinten.

Designaktivitetene dannet grunnlaget for fasen "Dag 4: å prototype". Denne fasen ble spredt over tre dager, én for hver av de tre applikasjonene: nettsiden, forbrukerap-

plikasjonen og utselgerapplikasjonen. Designaktiviteten ble utført individuelt av alle medlemmene i teamet. En enkel mal ble brukt og utfylt, slik som vist på figur 3.7, for å lage wireframes av design og bruksflyt for applikasjonssidene valgt ut i fasen "Dag 3: bestem".

Den siste fasen "Dag 5: valider" ble utført på den siste dagen av designsprinten. Her presenterte hvert teammedlem sine prototyper, i form av wireframes. Deretter ble det i flere runder stemt på spesifikke elementer som skulle med på MVPen. Valideringen koordinerte et samlet syn i teamet på hvilke applikasjonssider som skulle designes. På dette viset utnyttet teamet en tilpasset Design Sprint for å danne et bilde på de viktigste designaspektene for MVPen.



Figur 3.8: Wireframe som viser et prototypeforslag til kjøp av produkter i endelig Forbruker-applikasjon. Skissen ble tegnet for å oppsummere "Dag 5: valider" i Design Sprinten.

Teamet prioriterte Design Sprint fremfor andre design-metoder på grunnlag av målet om å utvikle testbar funksjonalitet tidlig i prosessen. En Design Sprint kunne dermed lett tilpasses for å fokusere på rask utvikling av en MVP som kunne brukes videre, mens andre former for Design Thinking legger mer vekt på å forstå konteksten over lengre tid. Som nevnt var rask utvikling et viktig fokus for prosjektet, noe teamet konkluderte at ville bli støttet på en god måte ved å følge en Design Sprint for å starte utviklingen. (Araújo mfl., 2019, Waidelich mfl., 2018)

3.3 Teknologivalg

Lokal er et omfattende system bestående av to mobilapplikasjoner og en nettside. Flere ulike teknologier har derfor blitt anvendt i utviklingsprosessen. Dette kapittelet beskriver og argumenterer for valget av teknologiene som bygger opp Lokal som helhet.

3.3.1 Nettsiden

Lokal Nettside er hovedsakelig tilrettelagt for å være et kontrollpanel for utselgere. Kontrollpanelet krever en dynamisk visning av informasjon og må tilby en brukervennlig administrasjon av utsalget. Ved å implementere et modulært design, som nevnt i kapittel 2.2.1, sikres det at informasjonsflyten blir så oversiktlig som mulig. Et rammeverk som støttet dette formålet måtte dermed velges.

3.3.1.1 React - Next.js

Lokal Nettside er dermed utviklet ved bruk av React-rammeverket Next.js. React tilbyr en modulær og effektiv tilnærming til webutvikling, noe som passer for en flersideapplikasjon som Lokal. Next.js tilbyr nyttige funksjoner som server-side rendering og statisk sidegenerering, som forbedrer lastetider, søkemotoroptimalisering, og brukeropplevelsen på tvers av enheter. I tillegg tilbyr Next.js en intuitiv og innebygd rutingmekanisme som forenkler navigasjonen mellom ulike sider. Med verktøy som dynamisk ruting og nyttige ferdiglagde komponenter som ”Link”, gir Next.js flersideapplikasjoner en responsivitet og dynamikk som ligner den man finner i enkelte enkeltsideapplikasjoner (SPA). Slik funksjonalitet er verdifull for å utvikle en flersideapplikasjon som Lokal. (Thakkar, 2020, Dinku, 2022)

Det finnes flere konkurrerende rammeverk innen webutvikling. I tillegg til React, er Vue og Angular blant de største og mest brukte («Most used web frameworks among developers 2023», 2023). Både Vue og Angular tilbyr robuste løsninger for utvikling av webapplikasjoner. Vue er spesielt tilpasset for å utvikle enkeltsideapplikasjoner, noe som kan gjøre det mindre ideelt for mer komplekse flersideapplikasjoner som Lokal («Vue.js», 2024). React skiller seg ut med en effektive og modulær tilnærming i utviklingen av webapplikasjoner. Med sitt store økosystem av komponenter og dokumentasjon, kan det argumenteres for at React er et mer egnet rammeverk for utvikling av et komplekst system som Lokal. (Joshi, 2023)

3.3.1.2 Cypress

For å forsikre at Lokal Nettside tilbyr forretningskritisk funksjonalitet over tid, har Cypress blitt brukt som test-rammeverk. Grunnlaget for Cypress-testing i prosjektet er at det muliggjør testing av brukerhistorier, samt at komponenter vises på riktig måte. (Kapittel 2 i Mwaura, 2021)

Ende-til-ende-testing har blitt brukt som metode for å teste brukerhistoriene. Grunnlaget for å bruke ende-til-ende-testing er å tilnærme brukermiljøet til en ekte bruker i testene ved at Cypress faktisk besøker nettsiden i browser. Komponent-testing sørger for at komponentene vises på riktig måte ved endringer. Dette sikrer at kodeendringer ikke bryter med tidligere definert logikk for hvordan komponentene skal oppføre seg. Komponent-testing i Cypress er satt opp til å kjøres ved merge-requester og manuelt

av utvikleren (se vedlegg I, seksjon 10). (Mwaura, 2021)

3.3.2 Mobilapplikasjonene

Lokal består av to mobilapplikasjoner: Lokal Forbrukerapplikasjon og Lokal Utselgerapplikasjon. Applikasjonene har lignende kodebaser, der utselger-applikasjonen gjenbruker elementer av kode fra forbruker-applikasjonen. Dermed benyttes samme teknologi i begge applikasjoner.

Det var avgjørende at applikasjonene fungerte på både iOS- og Android-enheter for å øke tilgjengeligheten for brukerne. Derfor var det viktig å velge en effektiv kryssplattform-løsning for utviklingen av applikasjonene. På dette vis kunne brukere uavhengig av operativsystem få en lik brukeropplevelse.

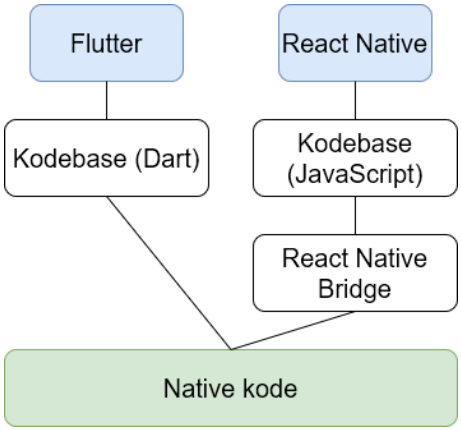
3.3.2.1 Valg av kryssplattform-løsning

For valg av kryssplattform-løsning, er det flere alternativer. De to mest utbredte løsningene er Metas React Native og Googles Flutter (JetBrains, 2024). Begge rammeverkene har flere likhetstrekk, men de har også vesentlige forskjeller som kan være avgjørende.

Den største forskjellen på Flutter og React Native er at Flutter bruker programmeringsspråket Dart, mens React Native benytter JavaScript og TypeScript. For utviklere med erfaring fra webutvikling vil React Native ofte være enklere å bruke, ettersom JavaScript og TypeScript er mye brukt i denne sammenhengen. Dart er et objektorientert programmeringsspråk som har flere likhetstrekk med Java. Basert på kompetansenivået hos utvikleren, kan Flutter ha en vesentlig brattere læringskurve sammenlignet med React Native. (Fentaw, 2020)

En annen betydelig forskjell mellom React Native og Flutter ligger i måten de kompilerer og kjører kode på, noe som har direkte innvirkning på applikasjonens ytelse. React Native bruker en prosess kalt React Native Bridge for å kommunisere mellom JavaScript-koden og enhetens native-kode. Denne broen oversetter JavaScript-koden til plattformens native-kode når den kjøres, noe som kan føre til forsinkelser og redusert ytelse, spesielt i ressurskrevende applikasjoner. (Paul og Nalwaya, 2019)

Flutter kompilerer isteden koden direkte til native-kode. Dette gjør at Flutter kan integrere mer direkte og effektivt med plattformene, noe som ofte fører til høyere ytelse og en mer sømløs brukeropplevelse. Denne forskjellen i kompileringsteknikker gjør Flutter til et særlig attraktivt rammeverk for mer grafikkintensive applikasjoner. (Fentaw, 2020)



Figur 3.9: Overordnet oversikt over kompilering av Flutter- og React Native-kode.

En betydelig fordel med React Native er et omfattende økosystem av dokumentasjon og ferdiglagde komponenter. Dette store økosystemet kan delvis forklares med bruken av JavaScript, et programmeringsspråk som er svært utbredt i utviklermiljøet. I React Native finnes det flere mulige løsninger for å takle utfordringer man møter på, i Flutter derimot er utvalget av mulige løsninger ofte mer begrenset. Kombinasjonen av et rikt økosystem og det faktum at det ikke er markante ytelsesforskjeller mellom de to rammeverkene på mindre grafikkintensive applikasjoner som mobilapplikasjonene i Lokal, gjør React Native til et foretrukket alternativ for utviklingen. (Fentaw, 2020)

3.3.2.2 React Native - Expo

I tillegg til React Native benytter teamet Expo, et rammeverk bygget på React Native. Expo har en rekke nyttige ressurser for mobilutvikling. Det mest essensielle er at Expo muliggjør utvikling og bygging av applikasjoner for både iOS og Android fra hvilken som helst type utviklingsmaskin. Nedenfor er noen andre funksjonaliteter Expo tilbyr:

- Live-utvikling direkte på egen mobilenhet
- API for push-varslinger
- Mulighet for online kompilering av applikasjoner
- Tilgang til enhetens sensorer og kamera uten ekstra konfigurasjon

(Expo, 2024)

3.3.2.3 Jest

Jest-testing ble brukt for å sikre at kodebasene ansvarlige for mobilapplikasjonene fungerte som forventet over tid. Grunnlaget for å bruke Jest er at rammeverket er veletablert, og støtter render-testing og enhetstester av React Native-komponenter, samt applikasjonssider. I tillegg lar Jest testeren mocke ønsket funksjonalitet på en

intuitiv måte. En rekke andre test-rammeverk ble forsøkt satt opp, som Appium og Detox, men Jest ble lettest konfigurert i forhold til oppsettet av kildekoden til mobil-applikasjonene, og ble dermed et naturlig valg for testing. (Salohonka, 2020, Meta Platforms, 2024)

3.3.3 Backend as a Service med Supabase

Lokal implementerer BaaS som backend-system. Dette er hovedsakelig på grunn av et bredt spekter av funksjonaliteter som gjerne tilbys av BaaS-løsninger, som forklart i kapittel 2.3.1. Ettersom BaaS-modellen gjerne tilbyr forhåndslaget funksjonalitet, var det et naturlig valg å implementere, ettersom det ville støtte teamets fokus på rask utvikling av testbare funksjonaliteter. Utover dette, var en fordel at BaaS ville bidra til standardisering av kodebase og kostnadseffektivitet ved eventuell oppskalering. (Nguyen, 2016, Carranza-García mfl., 2016)

Som alternativ til BaaS kunne teamet satt opp en server ved å bruke Java med Spring Boot, Rust med Rocket, eller Python med Django. Dette ville gitt teamet større fleksibilitet og kontroll til å tilpasse systemet etter behov, men ville kreve mer tid og ressurser for opprinnelig design, utvikling og skalering av systemet. (Nguyen, 2016)

Supabase bygger opp backend-systemet for de tre applikasjonene i Lokal. Supabase er et BaaS-system med åpen kildekode som tilbyr en rask måte å sette opp og administrere skalerbare backend-tjenester. Plattformen har enkel databaseadministrasjon, autentisering, sanntidssynkronisering, og automatisk genererte API, som gjør det enklere for utviklere å bygge komplekse applikasjoner raskt (Amanuel, 2022).

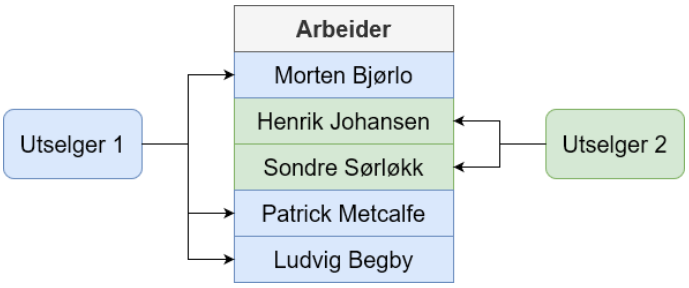
3.3.3.1 PostgreSQL

I motsetning til andre BaaS-systemer som Firebase, som primært inneholder en NoSQL-database, benytter Supabase seg av en PostgreSQL-database. Dette gir tilgang til PostgreSQL-funksjoner for robust dataintegritet, komplekse spørringer, og støtte for både strukturerte og ustrukturerte data. PostgreSQL er kjent for pålitelighet, ytelse, og fleksibilitet, noe som gjør Supabase til et kraftfullt verktøy for applikasjonsutviklere som krever mer avanserte databasefunksjoner og større kontroll over data. (Smith, 2010)

Databasefunksjonene er definert som transaksjoner («42.8. Transaction Management», 2024). Dette forenkler arbeidet for utvikleren ved å automatisere prosessen som sikrer dataintegritet og konsistens ettersom ACID-egenskapene implementert i PostgreSQL. Ved å ha transaksjoner som en del av grunnoppsettet, kan utviklere enkelt rulle tilbake endringer hvis en feil oppstår eller hvis en operasjon ikke fullføres som forventet, uten å risikere korrupte data eller inkonsistens i databasen. Disse egenskapene ved transaksjoner presenteres mer i dybden av kapittel 2.3.3.1. (Smith, 2010, Hansen og Mallaug, 2008)

3.3.3.2 RLS

Alle PostgreSQL-databaser implementerer RLS, som nevnt i kapittel 2.3.4 er en god sikkerhetsmekanisme. Supabase tilbyr et intuitivt grensesnitt for administrasjon av RLS. Supabase-grensesnittet gjør det enkelt å administrere og tilpasse tilgangsreglene for tabeller i databasen, noe som bidrar til å forenkle kompleksiteten og gjøre administrasjonen mer tilgjengelig. Under er et eksempel fra databasetabellen "arbeider" på hvordan RLS-regler kan sikre at spesifiserte roller kun har tilgang til egne data. (Randolph, 2024)



Figur 3.10: Illustrasjon av RLS for databasetabellen "Arbeider".

3.3.3.3 Edge Function

Supabase, som BaaS, tilbyr integrasjon av databasetjenester. Imidlertid, for mer komplekse prosesser eller integrasjon med tredjepartstjenester, kan implementasjon av Edge Functions være en løsning. Edge Functions i Supabase er serverløse TypeScript-endepunkter som fungerer på lignende måte som standard API-endepunkter, men kjører i et eget Deno-miljø. Deno er en enkel, moderne og sikker kjøretid for JavaScript og TypeScript som er bygget i Rust (Inc., 2024). Dette gir fleksibilitet og effektivitet slik at funksjonaliteten til en applikasjon kan gå utover grunnleggende databasetilbud ved for eksempel integrasjon av BaaS. Teamet måtte benytte TypeScript som programmeringsspråk og Deno som kjøretidsmiljø, siden dette for øyeblikket er den eneste teknologikombinasjonen som støttes av Edge Functions i Supabase. (Supabase, 2024c)

3.3.3.4 Webhooks

Supabase tilbyr en enkel implementasjon av webhooks, som kan koble serveren med eksterne systemer via et standard API eller ved bruk av en Edge Function. Webhooks i Supabase kun utføres etter en databaseutløser i systemet. (Supabase, 2024b)

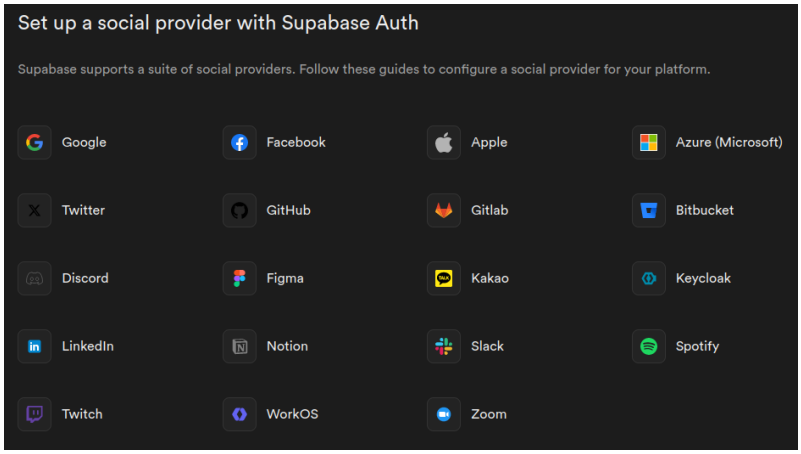
3.3.3.5 Sanntidskommunikasjon

Klienter kan etablere sanntidskommunikasjon mot Supabase på ulike måter. En implementasjon er *Postgres-Changes*. Denne implementasjonen er satt opp til å kunne

sende informasjon om endringer fra tjener til bruker ved gitte databasemanipulasjoner. *Postgres-Changes* kunne dermed bli brukt av teamet for å overvåke spesifikke endringer i databasen, som muliggjorde sanntidskommunikasjon i systemet. Dette inntreffer dermed ved at en databaseutløser aktiveres ved hendelsen. (Supabase, 2024e)

3.3.3.6 OAuth

OAuth 2.0-standarden kan brukes i Supabase for å delegere autentiseringen av brukerne til eksterne systemer. Dersom dette settes opp, med for eksempel Facebook eller Google, kan brukeren logge inn med sin eksisterende konto fra tredjepartstilbyderen, for minimal konfigurering ved innlogging. Ved bruk av OAuth-integrasjonen i Supabase, kan man enkelt sette opp et passordløst system som omtalt i kapittel 2.3.2 Passordløst design. (Supabase, 2024f)



Figur 3.11: Samtlige integrerte autentiseringssystemer tilgjengelig i Supabase.

3.3.3.7 E-posttjenester

I Supabase er det mulig å autentisere brukere ved hjelp av engangspassord eller magic-link som sendes til e-post, som nevnt i kapittel 2.2.3 kan være et mer brukervennlig alternativ til tradisjonelle innloggingssystemer. Supabase har integrert en Simple Mail Transfer Protocol (SMTP)-server som kan sende opptil tre e-poster per time, men den har også muligheten til å koble til eksterne SMTP-tilbydere som Twilio SendGrid, AWS SES og Resend for å øke antall eposter og redusere ventetid. (Supabase, 2024a)

Lokal kobler Supabase-serveren til en SMTP-server levert av Resend. Resend er en SMTP-tilbyder som kan brukes gjennom SDK-funksjoner eller API-er. Resend tilbyr en gratis versjon som tillater opptil 100 e-poster per dag og 3000 per måned. Resend kan også kobles til et domene for å definere et alias for SMTP-server. Aliaset til serveren

er *lokalapp.org*. (Resend, 2024c, Resend, 2024a, Resend, 2024b)

Teamet valgte Resend som SMTP-leverandør på grunn av enkel integrasjon med Supabase. Etter undersøkelser, fant teamet ingen bemerkelsesverdige fordeler ved bruk av de andre tjenestene. I tillegg anbefaler Supabase å bruke Resend, som har mest dokumentasjon og tilgjengelige eksempler sammenlignet med Twilio SendGrid og AWS SES (Rocha, 2023).

3.3.3.8 Lagring

Som andre webservicer, tilbyr Supabase et lagringssystem for statiske filer. Dette lar brukere lagre og håndtere filer som bilder, videoer eller dokumenter direkte fra applikasjonen. Supabase tilbyr sikker og pålitelig lagring med muligheten til å skalere etter behov. (Supabase, 2024g)

3.3.3.9 Sammenligning

Teamet kunne ha utforsket alternative BaaS-plattformer som AWS Amplify, Netlify og Google Firebase. Disse plattformene tilbyr et variert utvalg av nøkkelfunksjoner og har lignende ytelse som Supabase. Likevel, er den primære grunnen til å foretrekke Supabase fremfor disse alternativene at Supabase er åpen kildekode. Dette medfører enkelhet ved behov for å eksportere data for lokal drift eller for bruk hos andre leverandører (Kolesnikov mfl., 2024).

Dette gir backend en uavhengighet fra eksterne leverandører som Google eller Amazon. Det åpner også for muligheten til å migrere systemet fra Supabases egne tjener, til en lokal versjon dersom nødvendig. I tillegg gir det muligheten til å verifisere hvordan de ulike funksjonalitetene fungerer, og sikre håndteringen av lagret brukerdata.

3.3.4 Versjonskontrollsystem og Git

Versjonskontrollsystemet som har blitt brukt i utviklingen av systemet er Git. Egenskapene til versjonskontrollsystemet, som presentert i kapittel 2.4.1, er fordelaktige for prosjektet, både på grunn av begrenset utviklingstid og antallet medlemmer i teamet.

3.3.4.1 GitLab

I prosjektet har GitLab blitt benyttet som Git-løsning. GitLab er en plattform for kildekodehåndtering som tilbyr en rekke funksjonaliteter for effektiv versjonskontroll og samarbeid. Løsningen støtter systemutviklingsprosessen ved å tilrettelegge for kontinuerlig integrasjon og leveranse, noe som gjør det mulig å dele, utvikle og integrere kode effektivt i fellesprosjekter. (Hethey, 2013)

Flere Git-løsninger kunne blitt vurdert for lagring av kodebasen, inkludert GitHub, Azure Repos, eller en dedikert server. Imidlertid ble GitLab valgt som løsning fordi det

var et spesifikt krav fra universitetet. Dermed ble også bruk av Git obligatorisk.

Teamet gikk inn for å sette opp kontinuerlig integrasjon og kontinuerlig leveranse med GitLab-CI/CD fra tidlig i prosessen. GitLab-CI/CD har blitt brukt for å sikre funksjonalitet og verifisere kodebasen. Overordnet sett var metoden å sette opp CI/CD-miljøet på GitLab til å automatisk installere og bygge prosjektene, samt teste koden. I tillegg, for å lagre informasjonen over tid, ble CI/CD konfigurert til å produsere loggfiler for testing og sikkerhetssjekking. (Shahin mfl., 2017)

3.3.5 Eksterne tjenester

3.3.5.1 GitHub Pages

GitHub har blitt benyttet for kontinuerlig leveranse. Plattformen ble brukt for lansering av nettsiden via GitHub Pages, som muliggjør hosting av nettsider direkte fra et GitHub-repositorie. En klar fordel med GitHub Pages er at hostingen er gratis og tilbyr eget domenenavn. I teamets pipeline for nettsiden blir bygget kodebase lastet opp til GitHub-repositoriet, som automatisk publiserer nettsiden. (GitHub, 2024)

Ettersom teamet bruker Next.js, ville Vercels egen hostingtjeneste vært et godt alternativ til GitHub Pages. Vercel, som eier Next.js, tilbyr en hostingløsning som er tett integrert med rammeverket. Imidlertid er en stor fordel med GitHub Pages at man direkte kan sende artefakt-filer fra GitLab-pipeline, som allerede er bygget for testing. Dette eliminerer behovet for ekstra trinn og integrasjon som ville vært nødvendig med Vercel, og forenkler dermed publiseringsprosessen betydelig. (Vercel, 2024)

Bruken av GitHub Pages over alternativer som GitLab Pages, begrunnes med at GitLab-repositoriet for prosjektet er begrenset til intern bruk av NTNU, noe som forhindrer muligheten til å distribuere nettsiden til eksterne enheter, samt koble nettsiden opp mot et eget domene. I kontrast er GitHub Pages siden tilgjengelig på samtlige enheter.

3.3.5.2 Google-API

For avansert kartfunksjonalitet har teamet implementert Google-APIer. Disse tilbyr essensielle tjenester for Lokal som system. For det første, ble kartfunksjonalitet identifisert som et verdifult aspekt ved Lokal. Dermed var Google Maps en tjeneste fra Google-API som kunne brukes. Videre medførte kartet at utselgere skulle kunne plassere hvor utsalget var. Adressesøk med Google Autocomplete og adressesøk fra koordinater var dermed essensielt for å bygge opp utselgers administrasjon av utsalget på kartet.

Selv om alternativer som *Mapbox*, *Open Street Map*, og *OpenLayers* kunne vært vurdert, tilbyr Google en unik kombinasjon av enkel implementasjon for nettlesere, Android, og iOS, samt spesifikke funksjoner som er nødvendige for Lokal, som kartvisning, autofullfør i søkefunksjoner, og adressevalidering. Derfor falt valget på å bruke Google-API.

4. Resultater

I dette kapittelet presenteres de ulike resultatene som ble oppnådd i bacheloroppgaven. Forskningsresultatene er hovedsakelig basert på brukertester og intervjuer utført av teamet underveis i utviklingsprosessen.

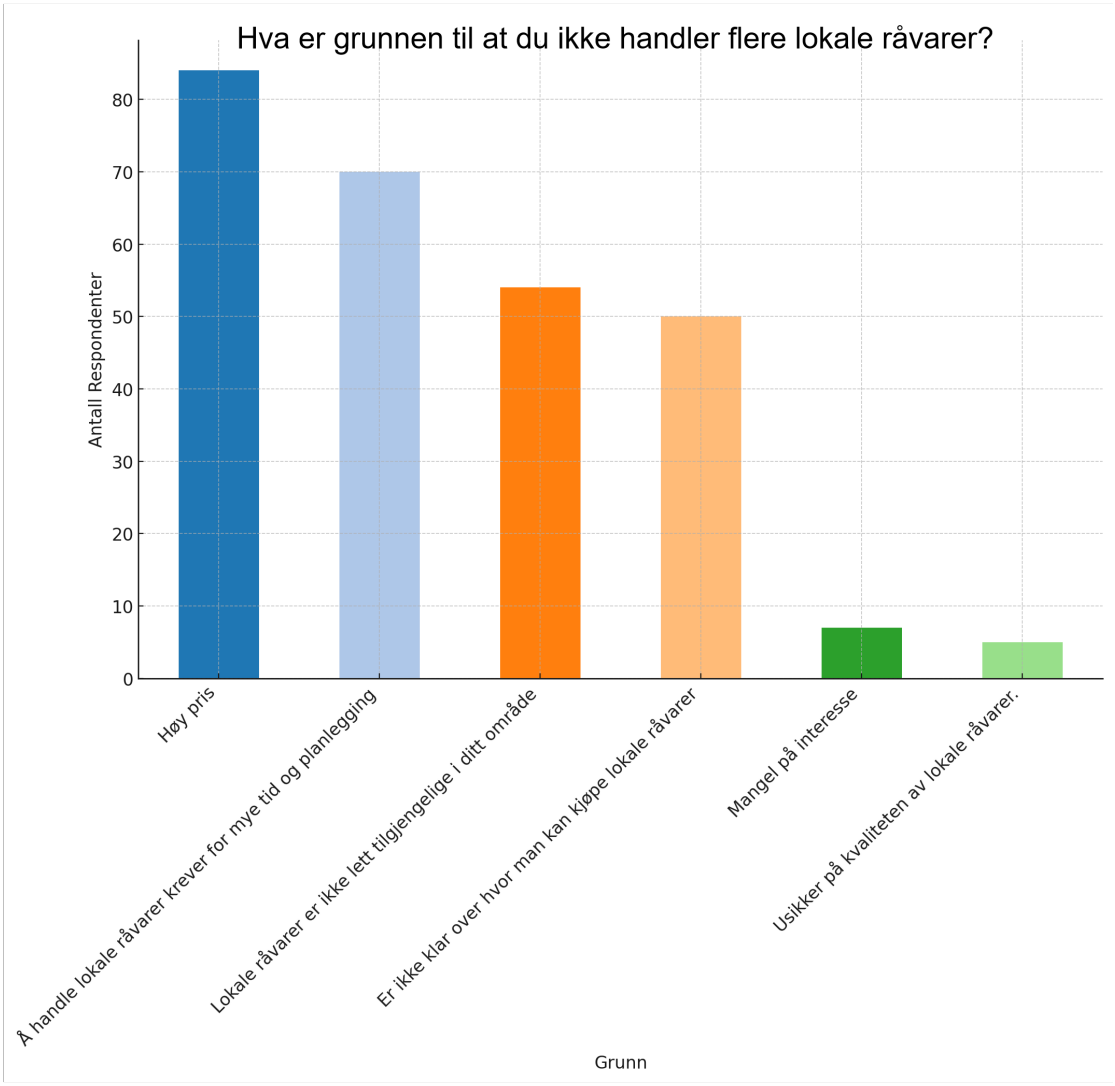
Det helhetlige systemet til Lokal består av tre deler: en nettside med kontrollpanel for utselgere, en applikasjon for utselgere, og en applikasjon for forbrukere. I presentasjonen av utviklingsresultatene vil disse tre delene av systemet bli inndelt i to grupper, henholdsvis for utselgere og forbrukere, for å forenkle redegjørelsen av de ulike funksjonelle kravene som er oppfylt.

4.1 Forskningsresultater

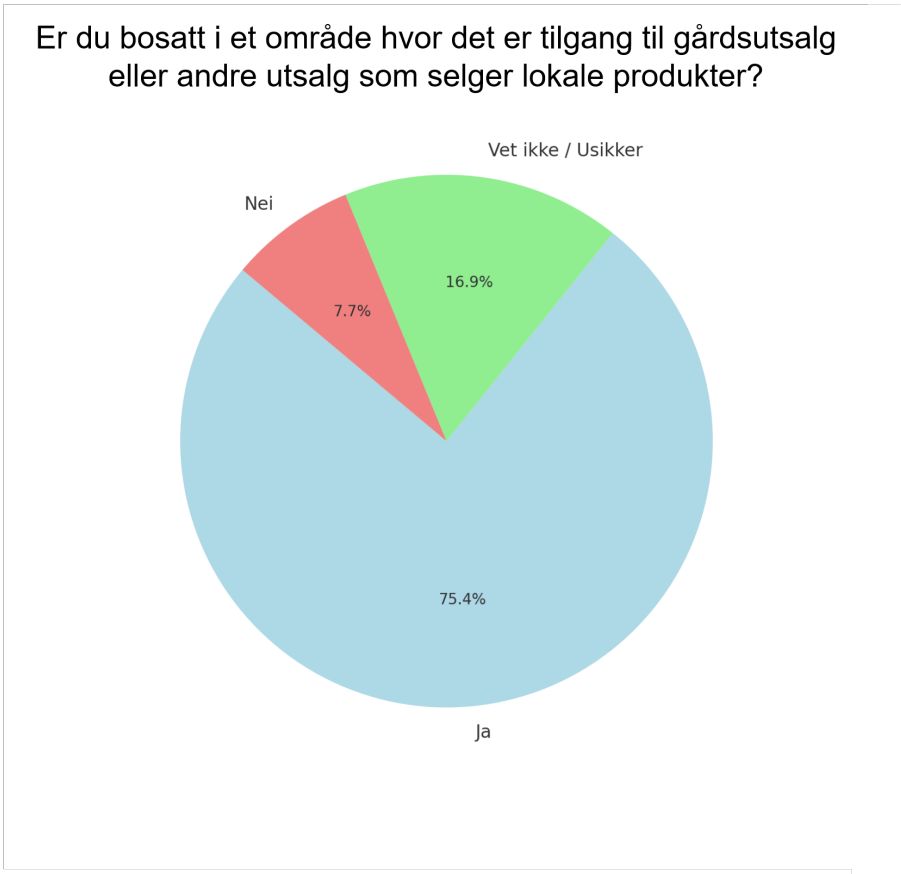
I forbindelse med utviklingen av Lokal, ble det utført en rekke brukerundersøkelser for å sikre at løsningen møter faktiske behov i markedet.

4.1.1 Markedsundersøkelse

En markedsundersøkelse ble derfor utført i emnet Ingeniørfaglig Systemtenkning (INGT2300). I den anledning ble en Google Forms-meningsmåling sendt ut på Facebook. Under vises noen av de viktigste resultatene fra spørreundersøkelsen. Spørreundersøkelsen fikk totalt 195 svar (se vedlegg H for relaterte resultater).



Figur 4.1: Stolpediagram med hvilke grunner folk oppgir for at de ikke handler mer lokale råvarer. Fra Google Forms-meningsmåling som inngikk i markedsundersøkelsen. Flere svaralternativer var mulig å velge.



Figur 4.2: Sektordiagram med oversikt over om folk er bosatt i et område med tilgang til lokale utsalg. Fra Google Forms-meningsmåling som inngikk i markedsundersøkelsen.

Videre ble det utført ti telefon-intervjuer, to epost-samtaler, og et lengre fysisk intervju med Jostein Vik, professor ved NTNU, fakultet for sosiologi og statsvitenskap. Disse dybdeintervjuene inngikk i kontaktloggen i INGT2300 (se vedlegg H). Oppsummert sett, var respondentene positive til Lokal, samtidig som det ble identifisert visse faktorer i samfunnet som kunne være hindringer for bruk av systemet. De viktigste resultatene var at det finnes alternativer som matfra.no, og at det kan være vanskelig å etablere en solid brukergruppe ved oppstart.

4.1.2 Brukertester

I selve utviklingsprosessen var dybdeintervjuer i form av brukertester for MVPen og videre inkrementer av produktet, til stor grad ansvarlige for smidige endringer fortløpende. Under følger en tabelloversikt med et sammendrag av ønskede funksjonaliteter basert på tilbakemeldinger fra brukertestene.

App formidler hvilke applikasjoner i systemet den ønskede funksjonaliteten gjelder:

U tilsvarer Utselgerapplikasjon, mens ***F*** står for Forbrukerapplikasjon og ***N*** gjelder for Nettsiden.
Fra brukertest (nummer) viser til i hvilke brukertester ønsket funksjonalitet ble nevnt. Fullstendige rapporter fra de ti brukertestene er dokumentert i prosjekthåndboken. Der nevnes oppdagede bugs, ønskede småendringer og positive tilbakemeldinger fra brukertestene (se vedlegg I, seksjon 6).
Prioritetsgrad viser til hvilken grad teamet så på ønsket funksjonalitet som viktig for sluttproduktet.

Funksjonalitet	**App**	**Fra brukertest (nummer)**	**Prioritetsgrad**
Enkel administrasjon av ordre	U	0, 7, 8	Høy
Administrasjon av åpningstid	U, F	1, 2, 3, 6, 8, 9	Høy
Animasjon ved fullført kjøp av produkt	F	2	Middels
Knapp for å tømme handlekurv	F	2	Middels
Oversikt over aktivt steg i registrering	N	3	Middels
Bruk av kuponger	F	4	Høy
Knapp for å sende review for utført ordre	F	4, 7	Lav
Slippe å oppdatere lagerstatus for varer	N, U	5	Lav
PDF-generering for meny av tilbudte produkter	N	5	Lav – mulig videreutvikling
Henteskap for bestilte varer	F	6	Lav – videreutvikling på lang sikt
Mulighet for bildegalleri for utsalg	N, F	6, 8	Høy
Mulighet for å kommunisere hentetidspunkt	N, U, F	6, 9	Middels – mulig videreutvikling
Tutorial for bruk av nettsiden	N	6, 8	Høy
Regnskap for utselgere	N	6, 8	Høy
Visuell tilbakemelding av reserverte ordre	N	6, 9	Middels

Skjule/vise oppføringer av produkter	N	6	Høy
Felter for allergener, produksjonsmetode, holdbarhet	N, U, F	6, 8	Middels - mulig videreutvikling

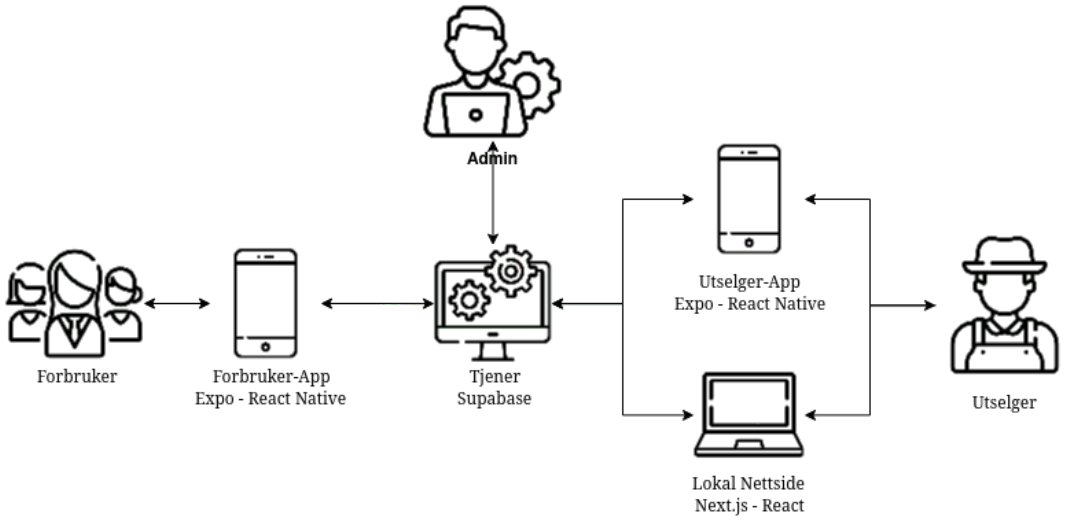
Tabell 4.1: Oversikt over funksjonaliteter identifisert som ønsket i systemet under de totalt ti brukertestene som ble utført.

4.2 Utviklingsresultater

Denne delen presenterer de ulike utviklingsresultatene som er oppnådd; de tre forskjellige applikasjonene, samt all funksjonaliteten de henholdsvis dekker.

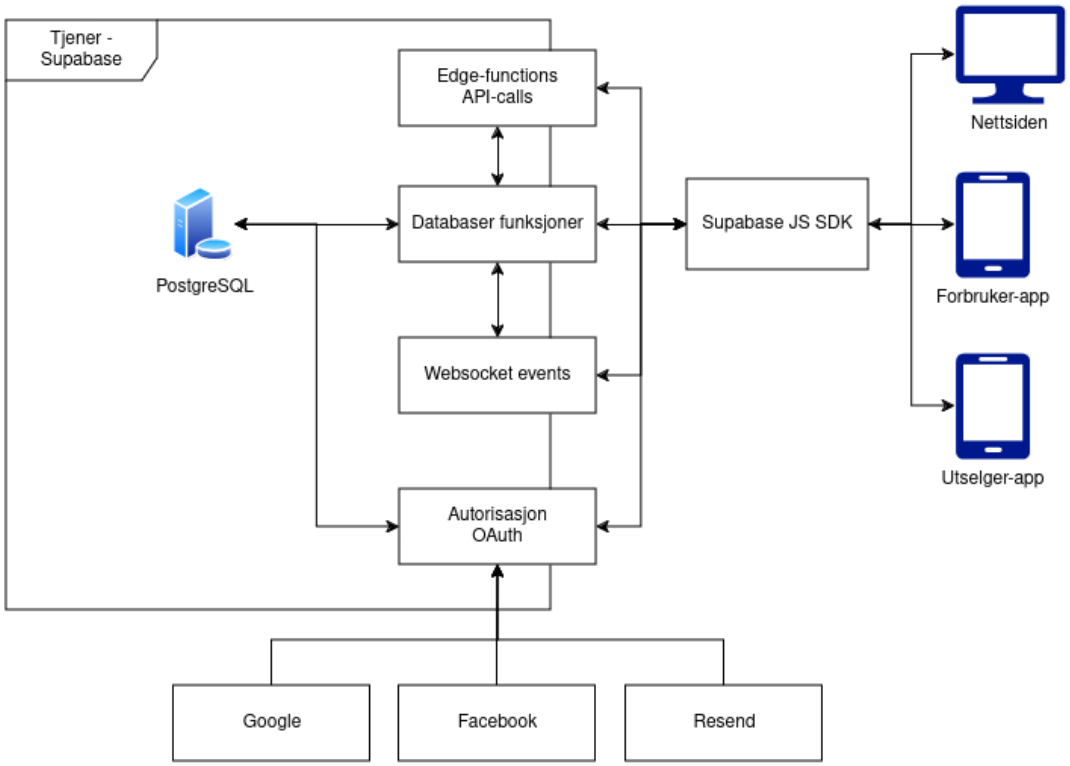
4.2.1 Overordnet systemarkitektur

Systemet består av fire hovedkomponenter: en Supabase-backend i form av en BaaS, to Expo-React Native-applikasjoner, og en nettside utviklet med Next.js og React, som er representert i det følgende arkitekturdiagrammet.



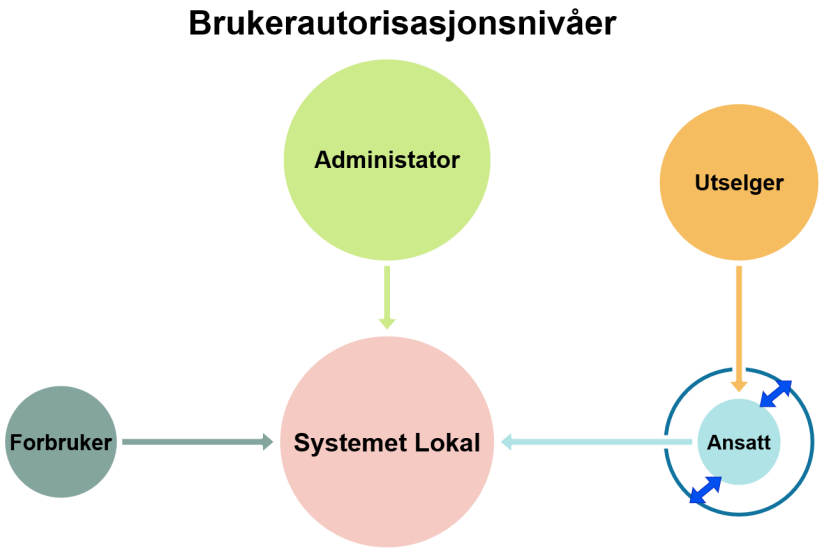
Figur 4.3: Systemarkitektur-diagram for brukere og komponenter som inngår i Lokal som system.

Det finnes ulike kommunikasjonskanaler mellom de forskjellige komponentene, som benytter Supabases sanntidskommunikasjon og Edge Functions. Som illustrert under, går all kommunikasjon via Supabases klientbibliotek, Supabase JS SDK:



Figur 4.4: Klient-tjener-diagram for kommunikasjonsarkitekturen i Lokal.

Videre viser det følgende diagrammet under de ulike autorisasjonsnivåene for en bruker kan ha i systemet. Brukerne er delt inn etter fire kategorier, hvor hver kategori representerer en egen brukerrolle. Størrelsen på hver radius for rollene indikerer brukerens autorisasjonsnivå. Radiusen til sirkelen som representerer en ansatt kan endres, ettersom en utselger har mulighet til å justere tillatelsene for sine ansatte.



Figur 4.5: Representasjon av ulike autorisasjonsnivåer for forskjellige brukertyper i Lokal.

4.2.2 Mobilapplikasjon for forbrukere

Forbrukerapplikasjonen er forbrukernes digitale kanal for å utforske og handle hos lokale utsalg i systemet. Den er utviklet som en kryssplattformløsning, som forklart i kapittel 3.3.2, for å være kompatibel med både Android og iOS-enheter.

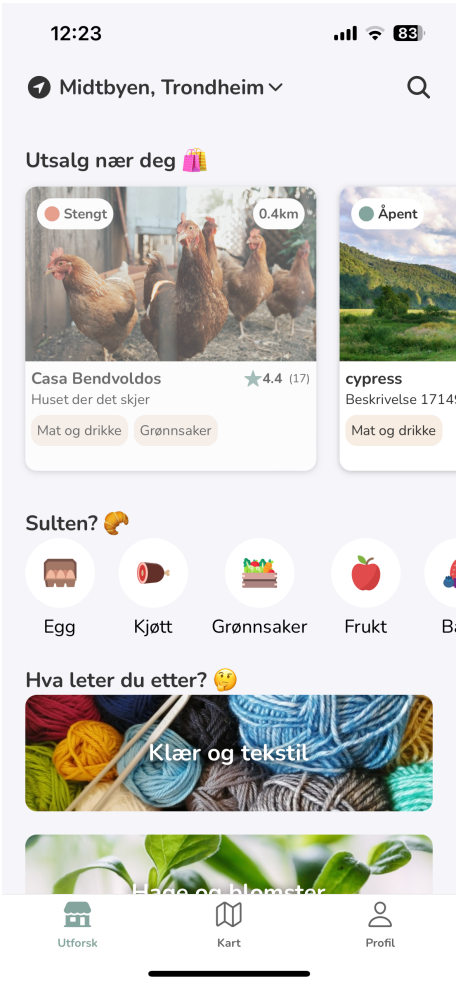
4.2.2.1 Innlogging

Brukere har mulighet til logge inn via tredjeparter, eller med e-post. Når brukere logger inn via tredjeparter som Google eller Facebook, blir de omdirigert til den aktuelle innloggingsportalen og deretter tilbake til applikasjonen. Dette er mulig ved bruk av OAuth, som nevnt i kapittel 3.3.3.6. Ved bruk av e-post mottar brukeren en magic-link, som gjør det mulig å logge inn automatisk i systemet. Applikasjonen anvender dermed et passordløst design.

Figur 4.6: Forsiden for innlogging i forbrukerapplikasjonen, der bakgrunnen er en gjentakende video-snutt.

4.2.2.2 Utforsk

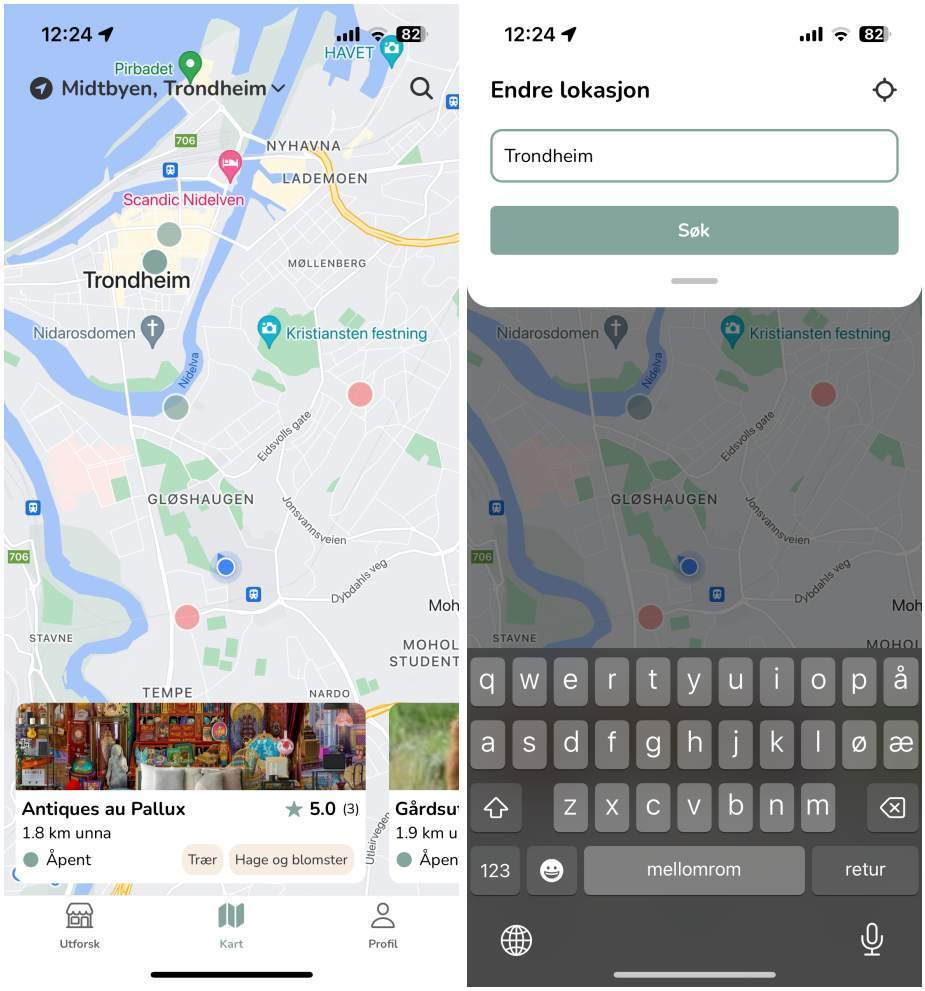
Utforsk-siden er det første brukerne ser når de logger inn i applikasjonen. På denne siden får brukeren en oversikt over lokale utsalg i nærheten, sammen med generell informasjon som avstand til utsalget, åpningstider, vurderinger og hva de tilbyr. Brukeren har også mulighet til å filtrere utsalg basert på produkttype, som for eksempel ”mat og drikke” eller ”hage og blomster”.



Figur 4.7: Utforsk-siden i forbrukerapplikasjonen.

4.2.2.3 Kart

Brukere har mulighet til å se utsalg posisjonert på et kart, fargekodet basert på åpningsstatus. Brukeren kan også søke etter utsalg i andre områder enn der de befinner seg, noe som gjør det mulig å finne utsalg hvor som helst i hele landet.



Figur 4.8: Kart-siden i forbrukerapplikasjonen.

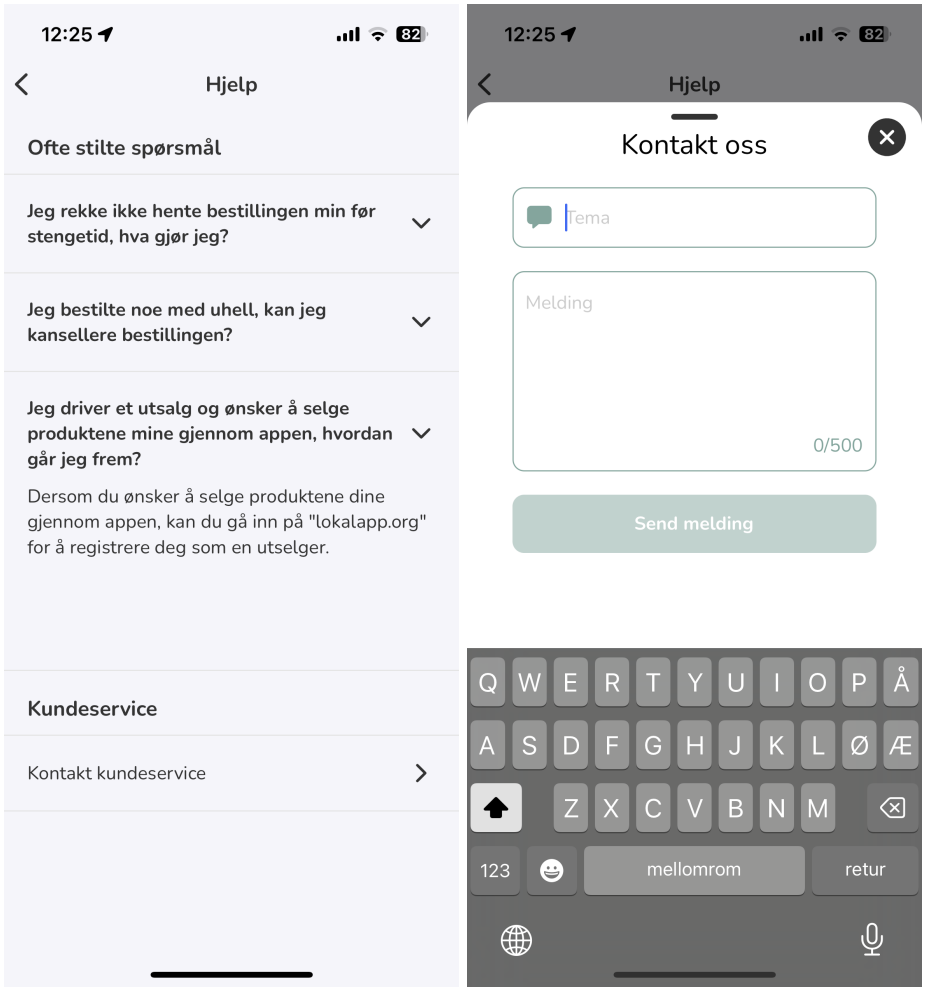
Kart-funksjonaliteten er implementert ved å bruke Google-API, som presentert i kapittel 3.3.5.2.

4.2.2.4 Profil

Brukere har en egen profilside hvor de har tilgang til all ordrehistorikk, innstillinger og brukerhjelp. Brukeren blir møtt med en personlig tilpasset velkomstmelding basert på tidspunktet på dagen.

4.2.2.5 Hjelp

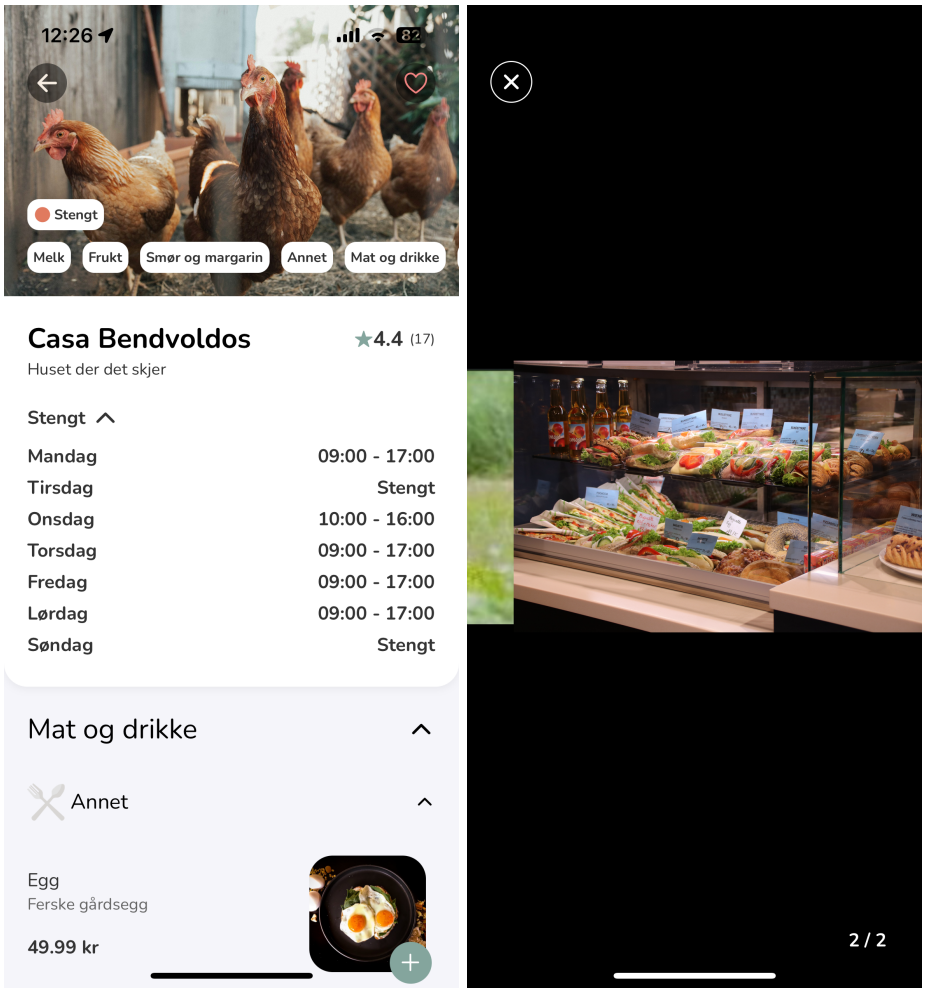
Brukere har tilgang til ofte stilte spørsmål og mulighet til å kontakte kundeservice direkte gjennom applikasjonen. Disse henvendelsene behandles av administratorer i systemet.



Figur 4.9: Brukerhjelp i forbrukerapplikasjonen.

4.2.2.6 Utsalg

Praktisk informasjon som åpningstider er lett tilgjengelig på utsalgssiden, og brukeren kan se hele timeplanen for den aktuelle uken. Det er også mulig å legge til utsalg som favoritter for å få raskere tilgang til dem gjennom utforsk-siden. Forbrukere vil bli informert om den gjennomsnittlige vurderingen av utsalget, men ikke de individuelle tilbakemeldingene. Alle tilgjengelige produkter er listet på siden, og brukeren kan legge produkter i en handlekurv. Et bildegalleri er tilgjengelig av utsalg, dersom utselgeren har flere bilder publisert av utsalget.

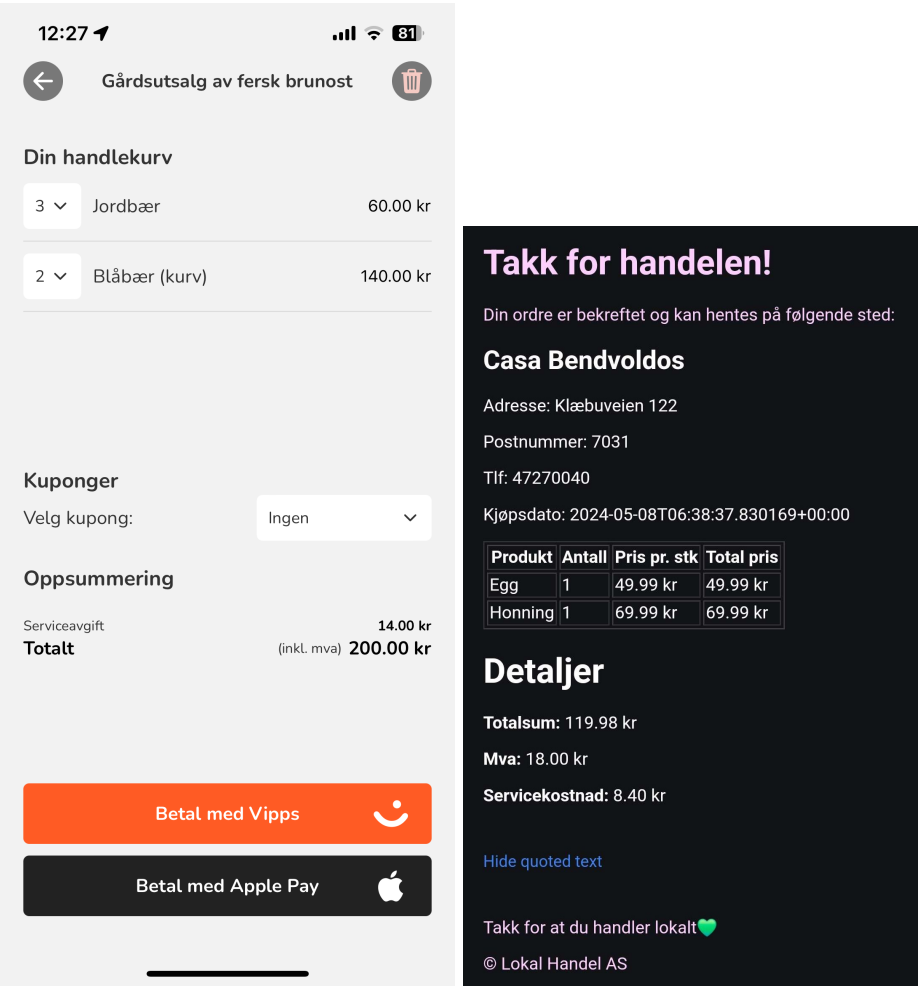


Figur 4.10: Utsalg i forbrukerapplikasjonen.

Det er flere likhetstrekk med den endelige siden i applikasjonen, og prototypen lagd under Design Sprinten (Se figur 3.8).

4.2.2.7 Kasse

Etter at brukeren har lagt til produkter i handlekurven, har de mulighet til å gå videre til kassen for å fullføre ordren. Siden denne applikasjonen ikke skal lanseres umiddelbart, gjennomføres det ingen reell betaling. Brukeren har mulighet til å endre antallet av hvert enkelt produkt, samt benytte eventuelle kuponger de har registrert i applikasjonen.



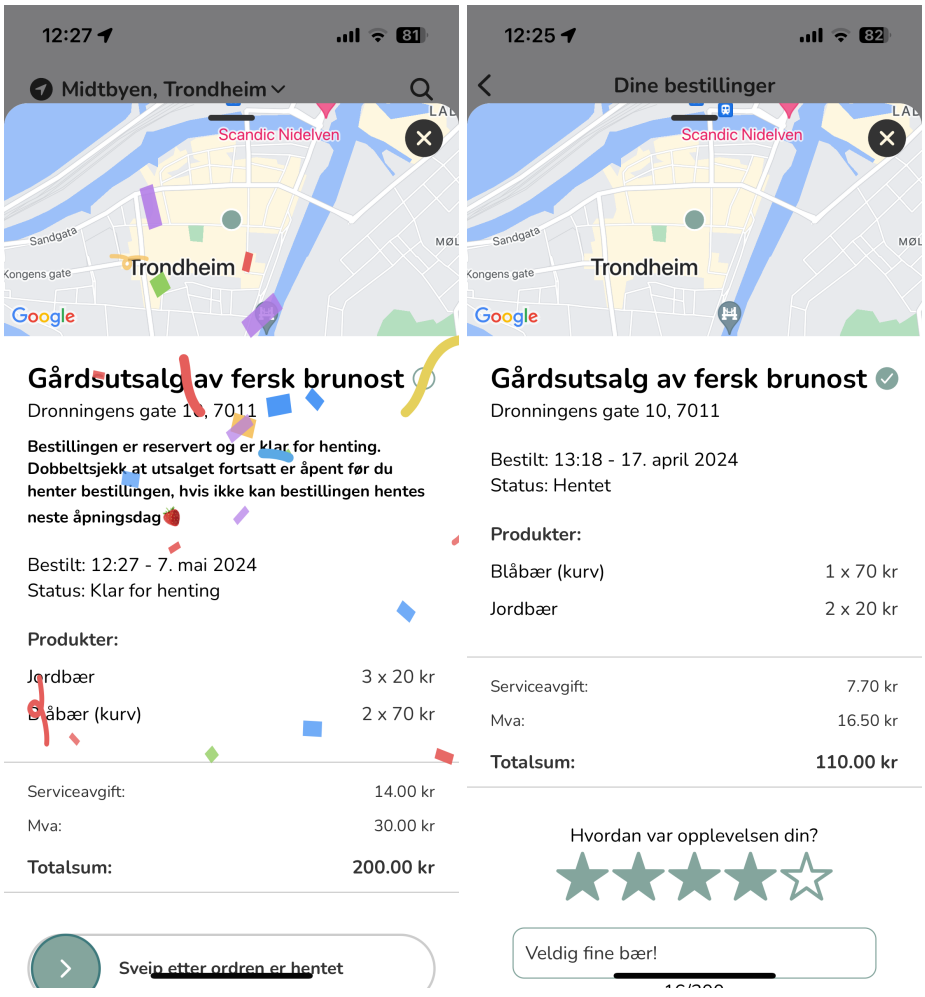
Figur 4.11: Til venstre kassa i forbrukerapplikasjonen, til høyre kvittering på e-post.

Etter at et kjøp er gjennomført, mottar kjøperen en kvittering som bekreftelse via e-post. Dette skjer ved bruk av en Edge Function som aktiveres ved hjelp av databaseutløser og webhook. Dermed blir SMTP-tjeneren implementert. Samme prosedyre anvendes flere steder i systemet, noe som tilbyr flere praktiske muligheter for utsendelse av e-post.

4.2.2.8 Ordre

Umiddelbart etter at et kjøp er gjennomført, vil den aktuelle ordren vises i applikasjonen med en konfetti-animasjon. Brukeren presenteres for praktisk informasjon som navnet på utsalget, adresse, produkter, og en beskrivelse som veileder brukeren. Ved å trykke på kartet kan brukeren åpne veibeskrivelser til utsalget.

Brukeren kan selv bekrefte at ordren er hentet ved å bruke en sveipemekanisme som ligner på den man bruker for å låse opp enkelte mobiltelefoner. Når en ordre er registrert som hentet, blir brukeren presentert med muligheter for å gi en vurdering av kjøpsopplevelsen ved utsalget. Det er denne vurderingen som vises på utsalgssiden og for utselgeren.



Figur 4.12: Ordre i forbrukerapplikasjonen: til venstre en ny uhentet ordre midt i en konfetti-animasjon, til høyre en eldre hentet ordre.

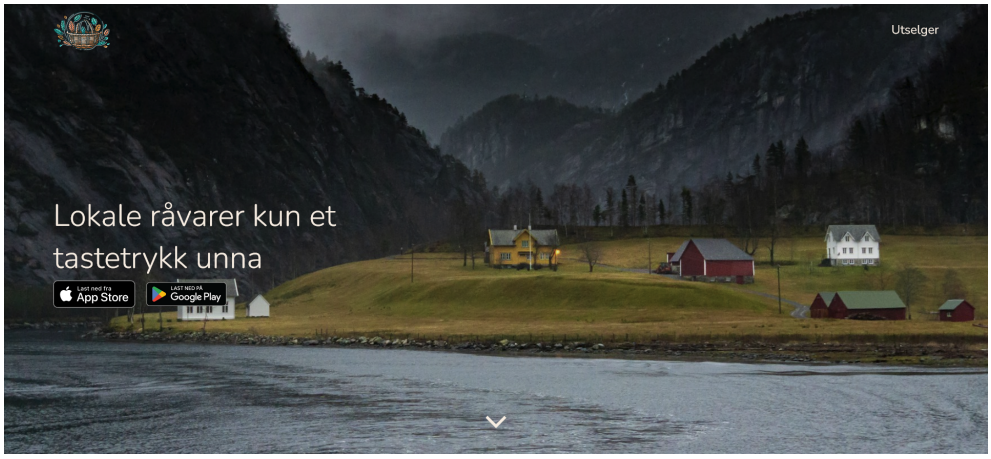
4.2.3 Nettside og mobilapplikasjon for utselgere

Utselgere har to metoder for å interagere med systemet: en nettside med et kontrollpanel for utsalget, og en mobilapplikasjon som tilbyr verktøy for å administrere utsalget. Nettsiden fungerer som en plattform for registrering av nye utsalg og gir brukeren full kontroll over sitt utsalg, samt tilgang til all relevant informasjon om ordre, inventar og lignende.

Mobilapplikasjonen gir nyttig informasjon som sanntidsoppdateringer på innkommende ordre, muligheten til å administrere produkter, varslingstjenester og kontroll over åpningsstatusen til utsalget. Mobilapplikasjonen for utselgere er bygget på kodebasen til forbrukerapplikasjonen, noe som effektiviserte utviklingen.

4.2.3.1 Nettsidens forside

Nettsiden er en viktig kanal for formidling av systemets budskap og rolle. Forsiden inneholder kun den mest essensielle informasjonen om Lokal, som kort beskriver hva systemet innebærer.

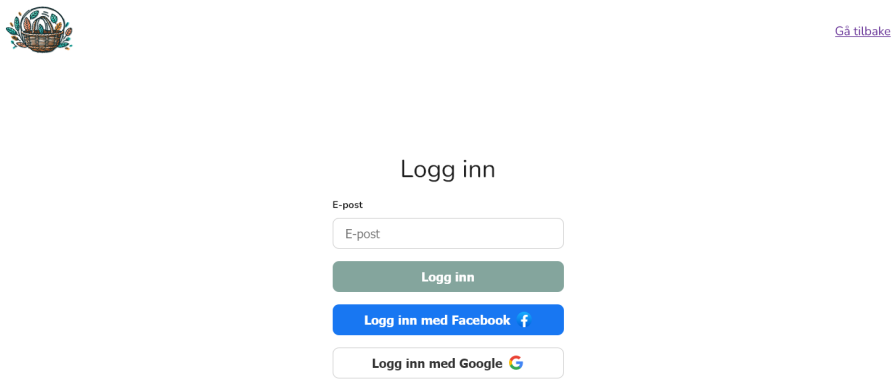


Figur 4.13: Forsiden til nettsiden.

Fra forsiden er det mulig å navigere til utselgerversjonen av nettsiden, som i stor grad er en kopi av den vanlige forsiden, men tilpasset utselgere med informasjon som er relevant for dem. Den gir også mulighet for å logge inn i systemet.

4.2.3.2 Innlogging som utselger

Innloggingsmetoden for utselgere er identisk med den i forbrukerapplikasjonen, hvor det er mulig å logge inn med tredjeparter ved bruk av OAuth, eller med magic-link via e-post. Det er ingen forskjell mellom nye og etablerte utselgerbrukere, noe som gjør prosessen enklere.

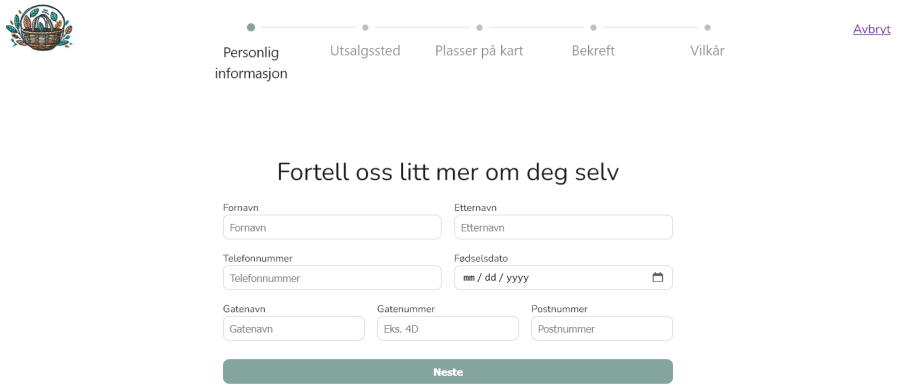


Figur 4.14: Innlogging som utselger på nettsiden.

4.2.3.3 Registrering av utsalg

Dersom den innloggede brukeren forsøker å åpne kontrollpanelet uten å ha et registrert utsalg, blir den møtt med en varsling som spør om han ønsker å opprette et utsalg. Systemet leder dermed de som ønsker å registrere et nytt utsalg til den angitte registreringssiden, og de som ikke ønsker å registrere et utsalg, tilbake til forsiden.

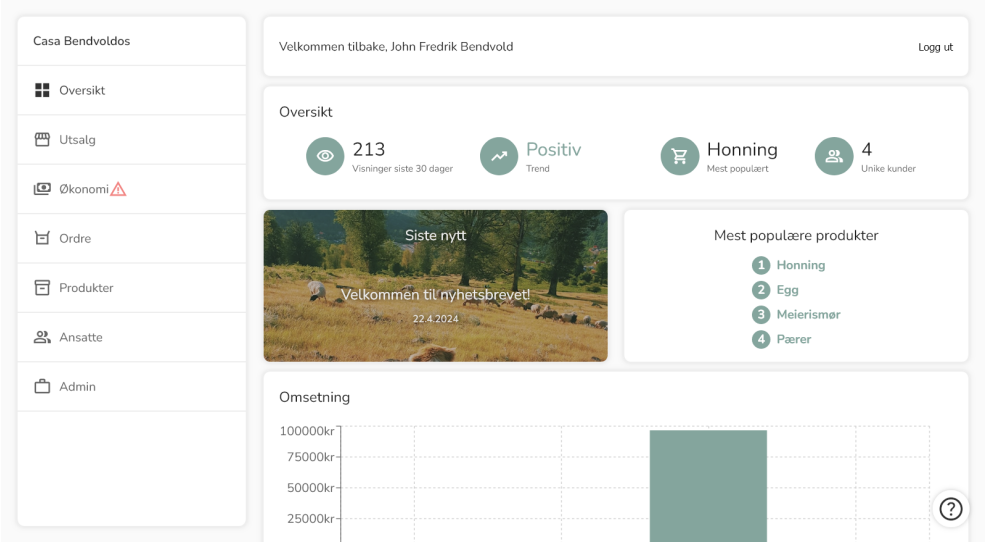
Registreringsprosessen er veiledet, med en visualisering av hvor langt man er kommet i prosessen.



Figur 4.15: Registrering av nytt utsalg på nettsiden.

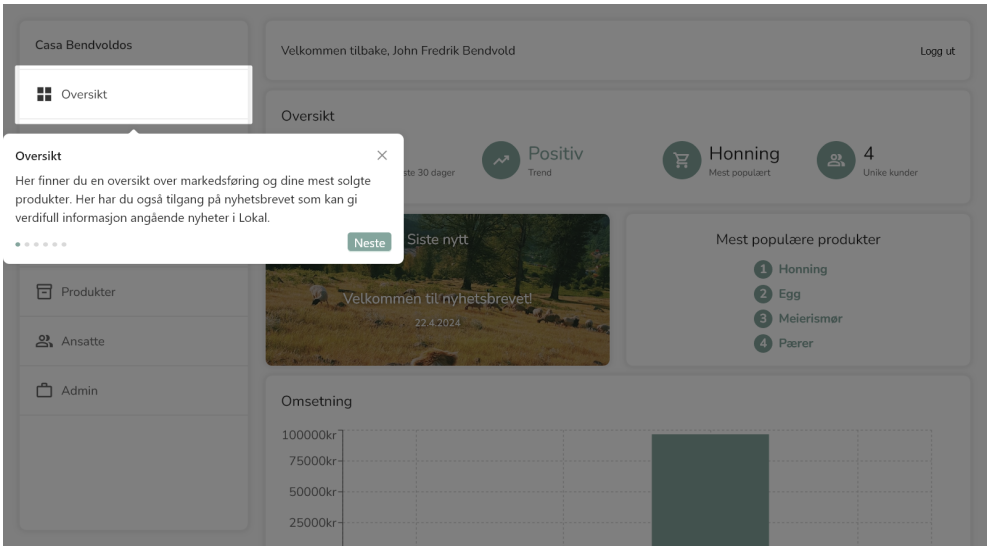
4.2.3.4 Kontrollpanel

Kontrollpanelet er der det meste av administreringen av utsalget foregår. Navigasjonsmenyen er alltid tilgjengelig, noe som tilater rask navigering mellom de ulike sidene.



Figur 4.16: Kontrollpanelet på nettsiden.

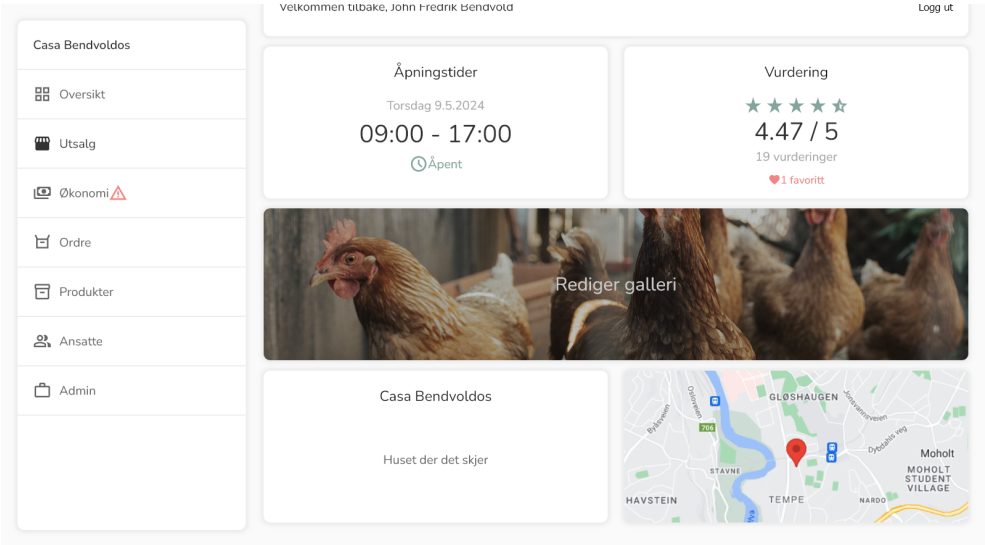
Det er mulig å få en brukerveiledning som forklarer de ulike sidene på kontrollpanelet slik vist under.



Figur 4.17: Brukerveiledning for kontrollpanelet på nettsiden.

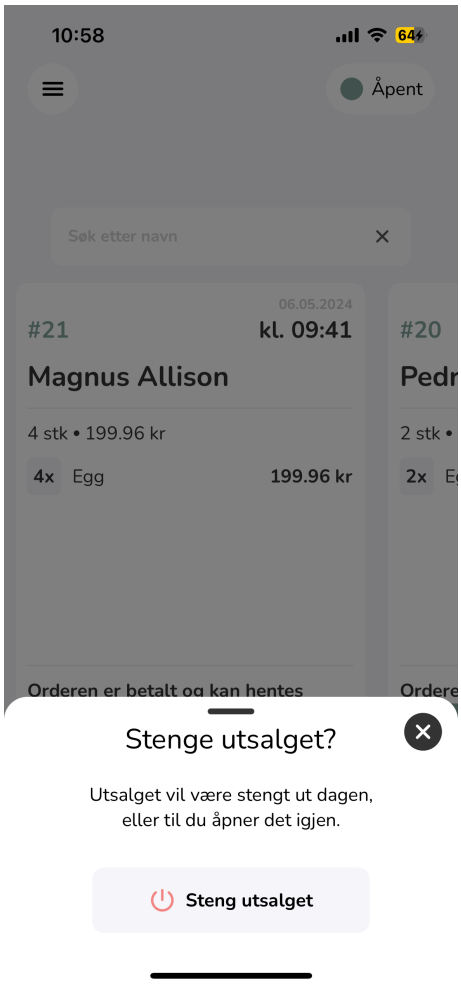
4.2.3.5 Administrering av utsalg

Utselgere har kontroll over informasjonen om sitt utsalg, og kan endre elementer som åpningstider, bildegalleri, utsalgsnavn, beskrivelse og lokasjon. Antallet som har lagt til utsalget som favoritt og vurderinger er kun tilgjengelige for visning. Dette gir innsikt i hva kundene setter pris på eller misliker ved utsalget.



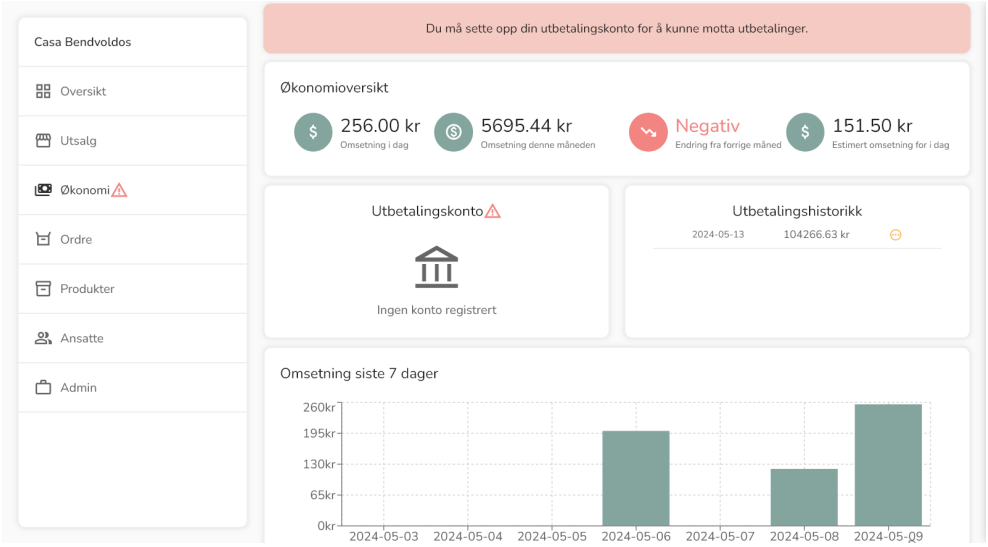
Figur 4.18: Administrering av eget utsalg på nettsiden.

Det er mulig å kontrollere åpningsstatusen fra mobilapplikasjonen, noe Christel understrekte som verdifult (Se vedlegg E). Dette er den eneste formen for administrering av utsalget som er tilgjengelig i mobilapplikasjonen, i tillegg til produkt- og ordreadministrasjon.



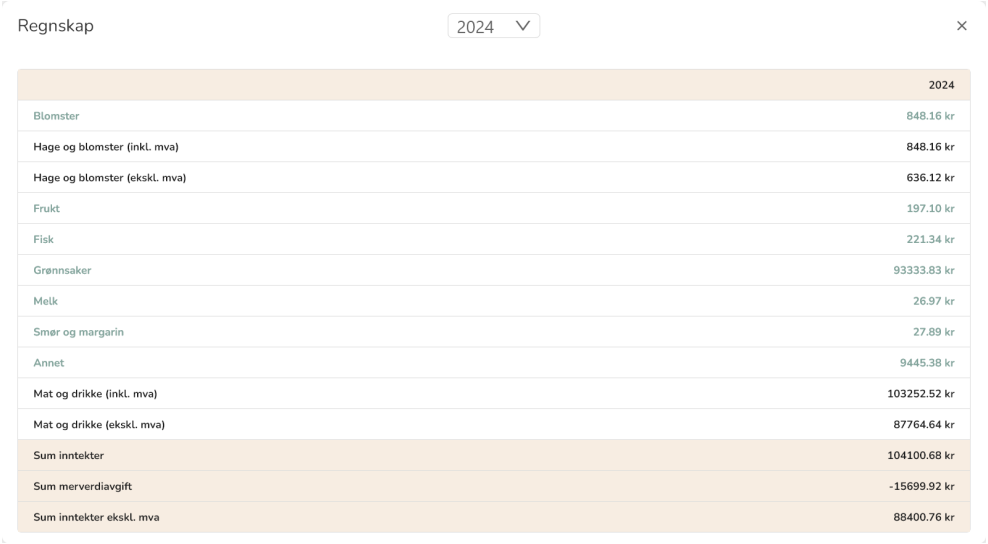
Figur 4.19: Endring av åpningsstatus i utselgerapplikasjonen.

4.2.3.6 Økonomi



Figur 4.20: Økonomioversikt på nettsiden

Det er mulig for utselgere å se et årsregnskap over deres salg ved bruk av systemet.

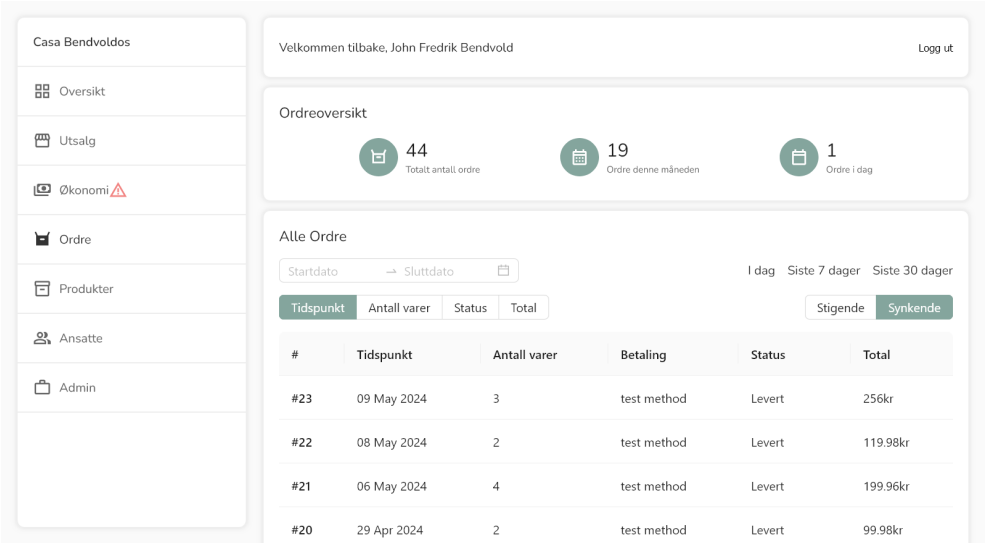


	2024
Blomster	848.16 kr
Hage og blomster (inkl. mva)	848.16 kr
Hage og blomster (ekskl. mva)	636.12 kr
Frukt	197.10 kr
Fisk	221.34 kr
Grønnsaker	93333.83 kr
Melk	26.97 kr
Smør og margarin	27.89 kr
Annet	9445.38 kr
Mat og drikke (inkl. mva)	103252.52 kr
Mat og drikke (ekskl. mva)	87764.64 kr
Sum inntekter	104100.68 kr
Sum merverdiavgift	-15699.92 kr
Sum inntekter ekskl. mva	88400.76 kr

Figur 4.21: Regnskapet for eget utsalg på nettsiden.

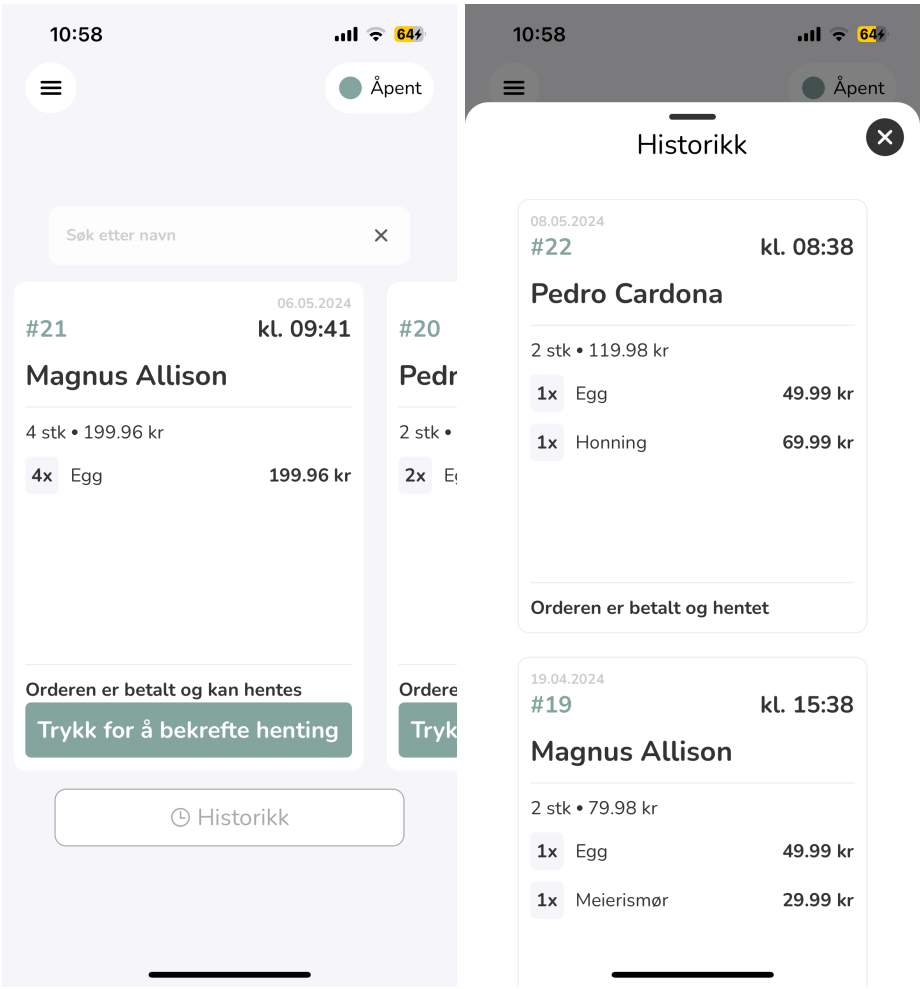
4.2.3.7 Ordreoversikt

Ordreoversikten på nettsiden tilbyr kun informativ funksjonalitet, slik som oversikt over hver enkelt ordre og hvilke produkter som er solgt. Den praktiske funksjonaliteten er tilgjengelig i mobilapplikasjonen.



Figur 4.22: Ordreoversikt på eget utsalg på nettsiden.

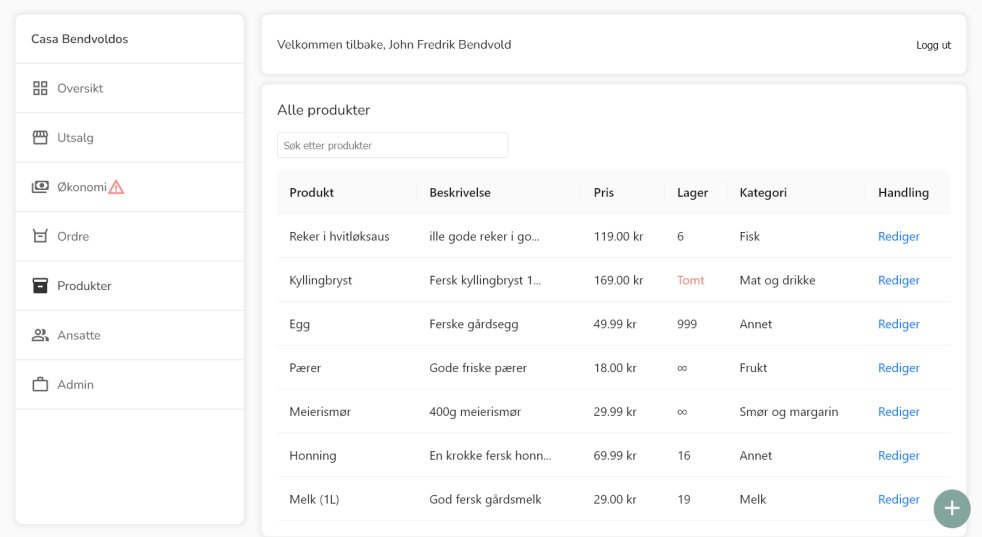
Utselgerapplikasjonen sender ut en push-varsling når den mottar nye ordre, ved bruk av databaseutløser, webhook og en Edge Function til Expo-tjeneren. Dette gjør at utselgeren ikke trenger å ha applikasjonen åpen til enhver tid for å drive utsalget. Utselgeren har også samme mulighet som forbrukeren til å markere ordre som hentet. Sanntidsoversikten, implementert ved hjelp av *Postgres-Changes*, over innkommende ordre utgjør forsiden for innloggede utselgere er presentert av skjermbildet under.



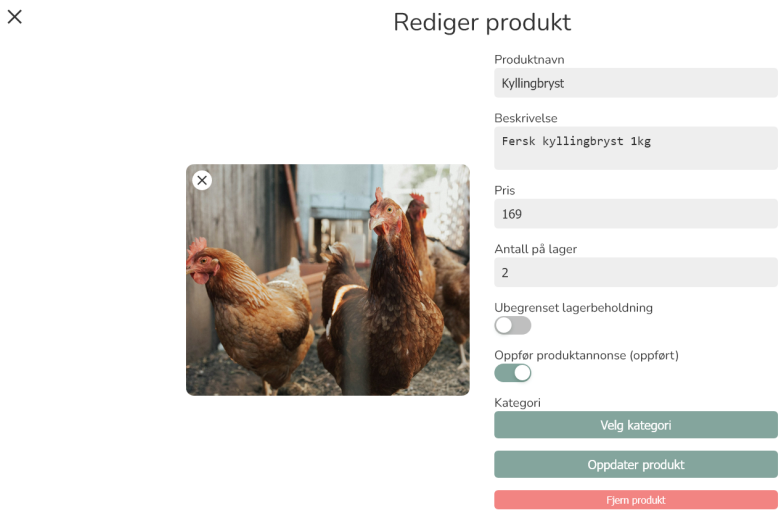
Figur 4.23: Ordre i utselgerapplikasjonen: til venstre er uhentet ordre, til høyre er eldre ordre.

4.2.3.8 Produkter

Nettsiden gir en oversikt over samtlige produkter, og all informasjon som er tilknyttet dem. Den tilbyr også funksjonalitet for å redigere, legge til, skjule og slette produkter fra utsalget.



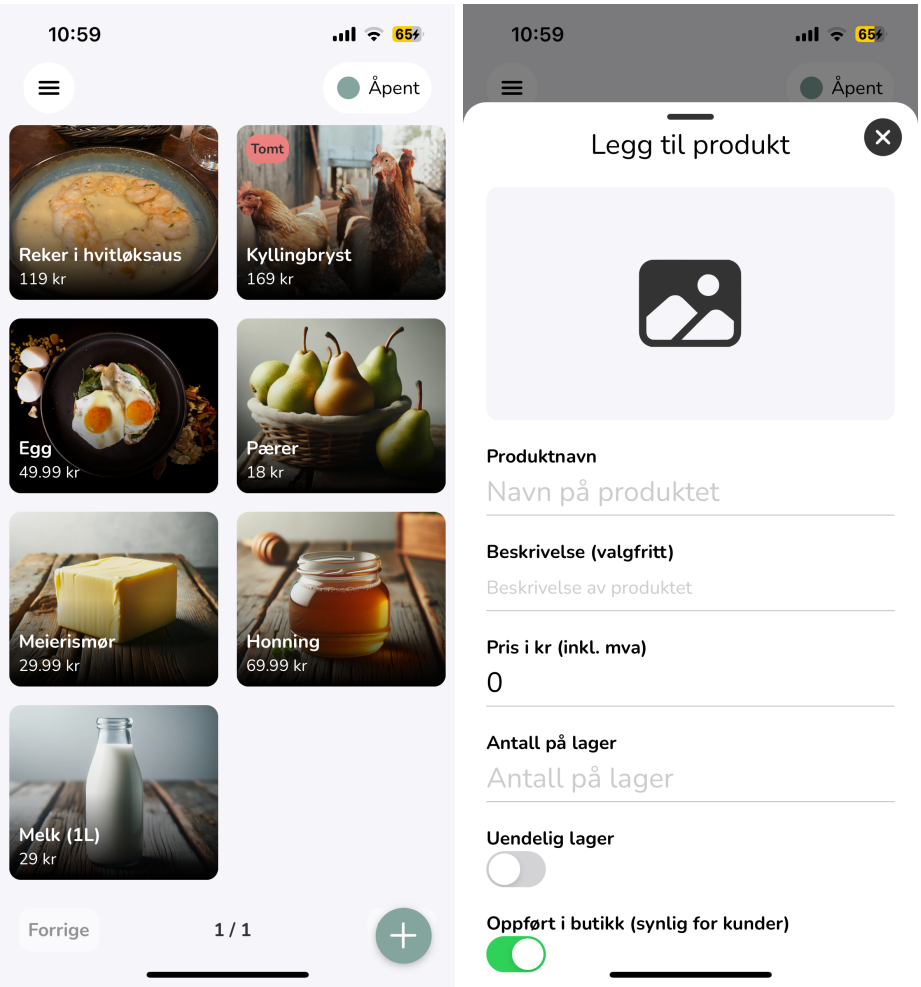
Figur 4.24: Produktoversikt på nettsiden.



Figur 4.25: Redigering av et produkt på nettsiden.

Muligheten til å redigere og legge til nye produkter direkte fra mobilenheten tilbyr et mer praktisk alternativ enn ved bruk av nettsiden, noe som forenkler prosessen.

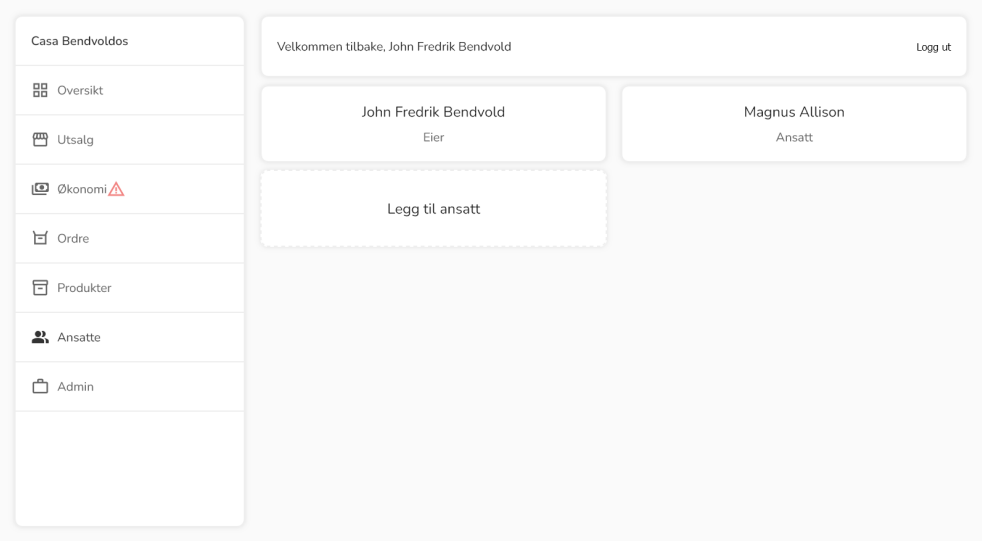
Dette ble bekreftet i brukertester og gjennom tilbakemeldinger fra intervjuobjektene. (Se kapittel 4.1.2)



Figur 4.26: Produkter i utselgerapplikasjonen: til venstre er en produktoversikt, til høyre er registreringen av et nytt produkt.

4.2.3.9 Ansatte

Fra nettsiden kan utselgere registrere nye ansatte til sitt utsalg. De ansatte får tilgang til utsalget gjennom mobilapplikasjonen. Eieren av utsalget har kontroll over hvilke funksjonaliteter de ansatte har tilgang til. For eksempel kan eieren begrense ansattes mulighet til å motta ordre eller redigere produkter.

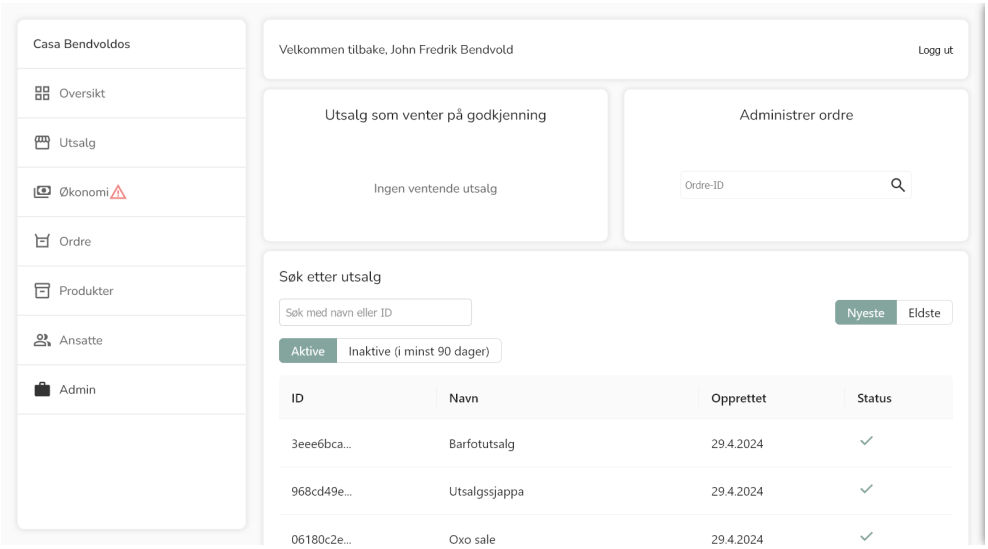


Figur 4.27: Oversikt over ansatte på nettsiden.

4.2.4 Nettside for administratorer

Brukere som er registrert som administratorer vil ha tilgang til admin-funksjoner fra kontrollpanelet på nettsiden. Disse funksjonene er kritiske for systemets helhet og omfatter følgende funksjonaliteter:

- **Verifisering av utsalg**
- **Kundeservice**
- **Refundering av ordre**
- **Deaktivering av utsalg**
- **Utestengelse av brukere**
- **Administrasjon av nyhetsbrev**



Figur 4.28: Administrator-kontrollpanelet på nettsiden.

4.2.5 Funksjonelle krav

Her er en oversikt over alle de funksjonelle kravene til systemet. Disse er resultatet av planlagt funksjonalitet fra visjonsdokumentet (Se vedlegg C). Nærmere spesifikasjon av når de ulike kravene ble oppnådd, er inkludert i prosjekthåndboken (se vedlegg I, seksjon 5).

Krav	Oppnådd	Kommentar
Brukerregistrering	✓	Manuelt med e-post eller via tredjeparter (Google og Facebook)
Brukerinnlogging/-utlogging	✓	Med e-post eller via tredjeparter (Google og Facebook)

Forbruker skal kunne se utsalg i sitt område	✓	Kan enten ses på kart, eller ”listet”
Forbruker skal kunne se alle produkter tilbudt av et utsalg	✓	
Forbruker skal kunne gjennomføre kjøp av produkter	✓	Foreløpig blir ingen reell betaling gjennomført
Forbruker skal få anbefalinger basert på tidligere kjøp	✗	Forkastet på grunn av tidsmangel
Forbruker skal kunne gi vurderinger til utsalg	✓	
Forbruker skal kunne lese ofte stilte spørsmål for brukerhjelp	✓	
Forbruker skal kunne kontakte kundeservice	✓	Meldingen sendes til administratorer
Forbruker skal kunne se ordrehistorikk	✓	
Forbruker skal kunne delta i verveprogram	✗	Forkastet på grunn av tidsmangel
Forbruker skal ha mulighet til å gjennomføre bestillinger uten å være registrert i systemet	✗	Forkastet på grunn av tidsmangel
Forbruker skal kunne se allergener	✗	Forkastet på grunn av tidsmangel, introdusert for sent i utviklingen
Forbruker skal motta kvittering på mail	✓	
Forbruker skal kunne se oppbevaringsanbefalinger for produkter	✗	Forkastet på grunn av tidsmangel, ble introdusert sent under utviklingen
Utselger skal kunne registrere sitt utsalg	✓	Søknad blir sendt til administrator for verifisering
Utselger skal kunne se regnskap for salg utført på utsalget	✓	
Utselger skal kunne registrere nye produkter	✓	Kan utføres på nettside og mobil
Utselger skal kunne redigere sine produkter	✓	Kan utføres både på nettsiden og i mobilapplikasjonen
Utselger skal få varsling ved nye ordre	✓	Får push-varsling fra mobilapplikasjonen
Utselger skal kunne administrere utsalgsinformasjonen sin	✓	

Utselger skal kunne sette åpningstider for utsalget	✓	Full administrering på nettsiden, kun åpning/stenging i mobilapplikasjonen
Utselger skal få en oversikt over alle mottatte ordre	✓	Mulig i mobilapplikasjon og på nettside
Utselger skal kunne legge til arbeidere på utsalget	✓	
Utselger skal kunne administrere arbeidernes tilganger	✓	
Utselger skal kunne endre arbeidsstedet til arbeidere	✗	Forkastet siden utselgere kun har én lokasjon
Administrator skal kunne verifisere registreringssøknader for utsalg	✓	
Administrator skal kunne deaktivere utsalg	✓	
Administrator skal kunne bannlyse brukere	✓	
Administrator skal kunne se alle transaksjoner gjennomført av en bruker	✓	
Administrator skal kunne se alle transaksjoner gjennomført av spesifikke utsalg	✓	
Administrator skal kunne gjennomføre kundeservice	✓	Får alle innsendte meldinger med kontaktinformasjon til brukeren
Administrator skal kunne administrere nyhetsbrev	✓	Kan opprette og fjerne nyheter
Administrator skal få innsikt i tilbakemeldinger hvert utsalg har mottatt	✓	
Administrator skal kunne slette tilbakemeldinger	✗	Forkastet siden tilbakemeldinger ikke er lesbare for andre enn den aktuelle utselgeren
Administrator skal kunne refundere ordre	✓	

Tabell 4.2: Resultat over alle funksjonelle krav, og om de ble implementert i systemet.

4.2.6 Ikke-funksjonelle krav

Resultater for ikke-funksjonelle krav, som definert i visjonsdokumentet ved bruk av "FURPS+"-modellen (se vedlegg C), er:

4.2.6.1 Funksjonalitet (F)

Ved å følge WCAG 2.1, universell utforming og sørge for gode kontraster mellom tekst og bakgrunn, sikrer teamet at all funksjonalitet er tilgjengelig for de fleste brukere, uavhengig av eventuelle funksjonsnedsettelser (se WCAG-rapport for nettsiden, og WCAG-rapport for mobilapplikasjonene).

Supabase tilbyr robuste autorisasjonsmetoder, som sikrer at brukerdata kun bevares hos de relevante brukerne. Videre er all innlogging kryptert. Bruken av RLS sikrer at brukerdata ikke er tilgjengelig for andre parter enn de som er relevante, som forklart i vedlegg J, seksjon 12.

4.2.6.2 Brukervennlighet (U)

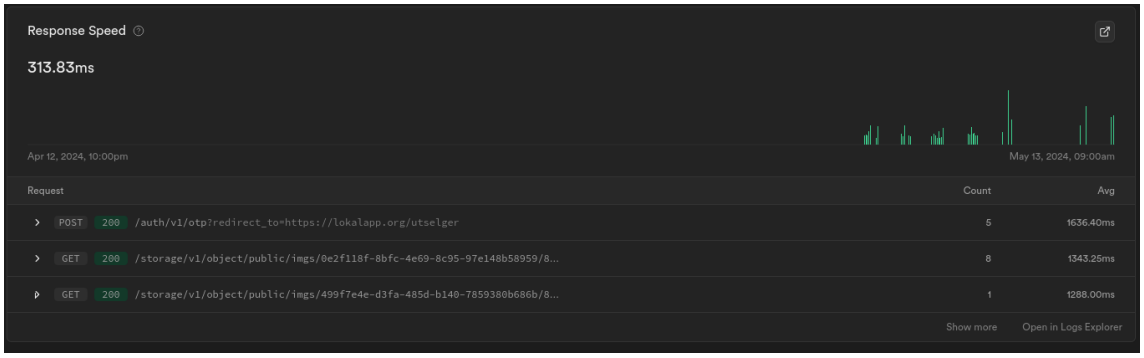
Nettsiden er responsiv og støtter ulike skjermstørrelser, som for eksempel for ulike PC-skjermer, nettbrett eller mobiler. For å sikre en bedre brukeropplevelse, er grensesnittet for mobilapplikasjonene definert til portrettmodus uten mulighet for å rotere mobilen til landskapsmodus.

4.2.6.3 Pålitelighet (R)

Systemet er avhengig av påliteligheten til Supabase. Som vist i Supabase side for oppe-status, har Supabase vært kontinuerlig tilgjengelig fra desember 2023 til mai 2024 uten noen form for nedetid. Derfor har også Lokal vært driftssikker i denne perioden.

4.2.6.4 Ytelse (P)

Gjennomsnittlig lastetid for alle forespørslene til Supabase, inkludert databaseoperasjoner, transaksjoner, Edge Functions, webhooks, databaseutløsere, og sanntidskommunikasjon, er 313.83 ms fra 13.04.2024 til 13.05.2024, som vist i det følgende bildet.



Figur 4.29: Bildet fra Supabase viser gjennomsnittlig lastetid fra 13.04.2024 til 13.05.2024.

Forespørslene som har høyere lastetider er de som å henter statiske objekter som lagret i Supabase som bilder av utsalgene eller produktene.

4.2.6.5 Støtte (S)

Som vist i figur 4.17, inkluderer nettsiden en intuitiv veiledning som forklarer de ulike komponentene av kontrollpanelet. I tillegg til veiledningen på nettsiden, tilbyr forbrukerapplikasjonen en liste over ofte stilte spørsmål samt muligheten til å ta direkte kontakt med kundeservice, som fremgår av figur 4.9.

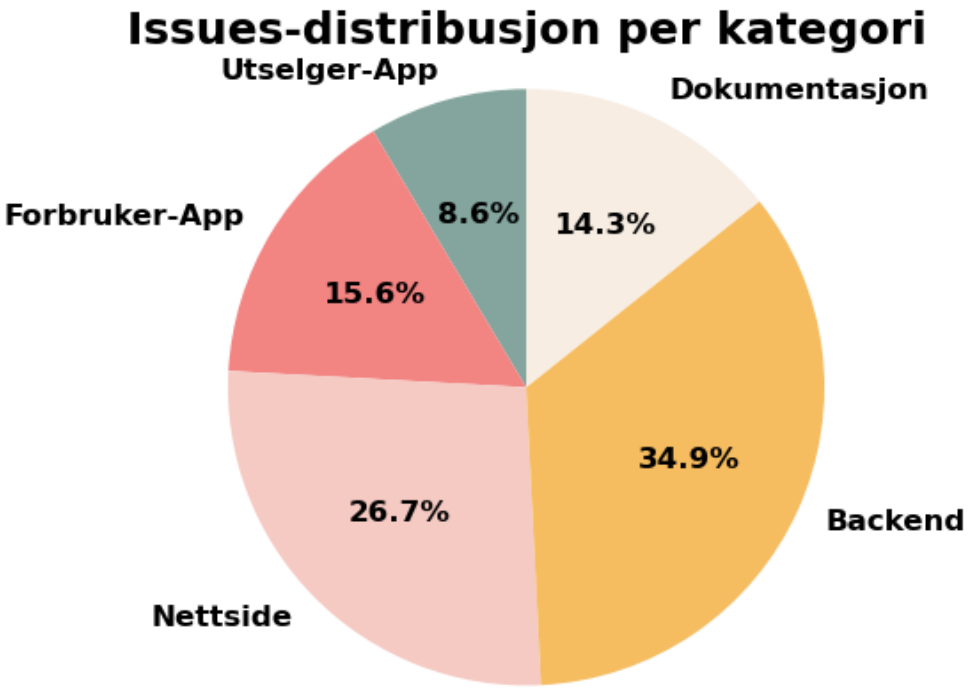
4.2.6.6 Andre krav (+)

I forhold til personvern, følges kravet om at tjeneren er lokalisert i Europa. Dette ettersom Supabase tilbyr å velge hvor tjenestene deres skal befinne seg i verden. For Lokal som system, er dette valgt til å være i Frankfurt. Som videreutvikling, vil det også være naturlig å implementere muligheten til at brukere av systemet kan få tilgang til egne persondata, som diskutert i kapittel 6.2.4.

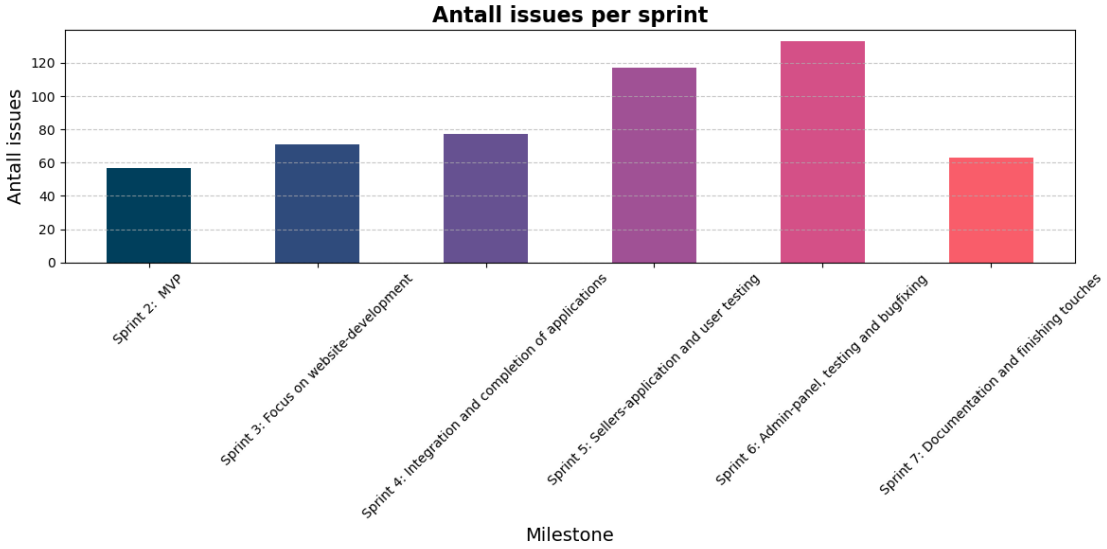
4.3 Administrative resultater

4.3.1 Smidig utvikling

Teamet åpnet totalt 614 issues, hvorav 45 var brukerhistorier som representerte de funksjonelle kravene i Tabell 4.2. Av disse funksjonelle kravene ble 7 markert som *Abandoned*. Av de resterende issues, som ikke var brukerhistorier, ble 547 fullført. Følgende diagrammer viser distribusjonen av issues gjennom prosjektet.



Figur 4.30: Representasjon av issues-distribusjon per kategori som et sektordiagram.

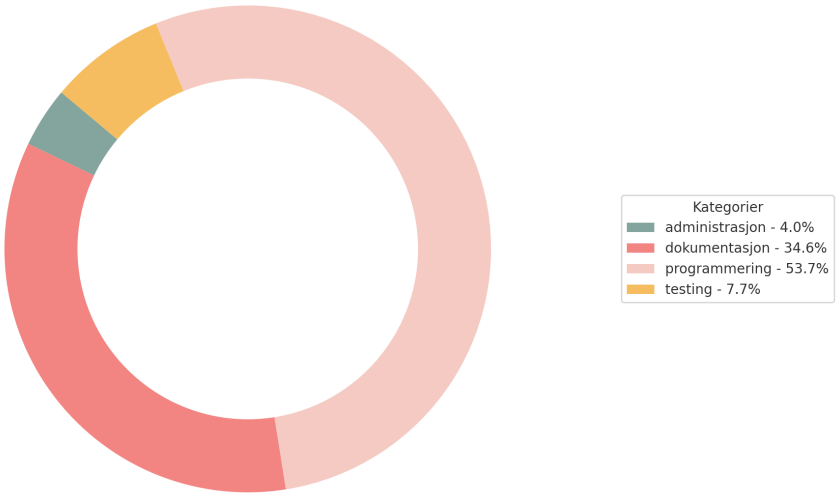


Figur 4.31: Antall issues per sprint.

4.3.2 Prosess

Som vist i figur 3.5, var prosjektet delt inn i syv sprinter og en Design Sprint. Informasjon om hver sprint ble grundig dokumentert på egne sider i vedlegg I, seksjon 5, der mål, arbeidsmetoder, resultater og retrospektiver er beskrevet.

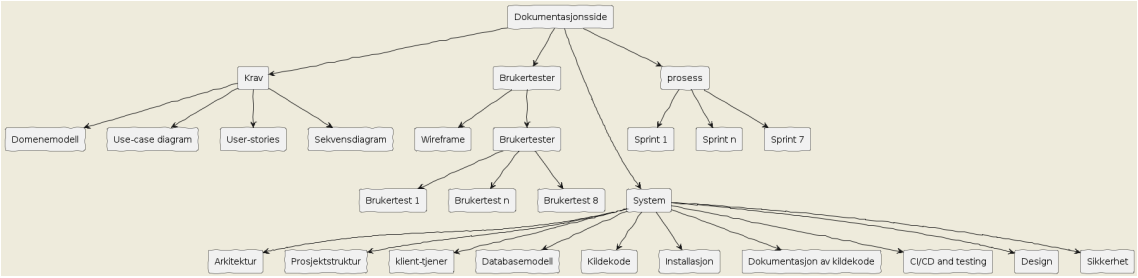
4.3.3 Arbeidsfordeling



Figur 4.32: Teamets arbeidstimer gruppert etter kategori fra timelisten.

4.3.4 Dokumentasjon

I tillegg til produktet, produserte teamet en samling dokumenter som beskriver systemet og andre aspekter relatert til utviklingsprosessen. Disse dokumentene finnes i Wikien til GitLab-prosjektet. Filstrukturen til Wiki er vist i det følgende diagrammet. Disse dokumentene kan også finnes i vedlegg I: Prosjekthåndbok, vedlegg J: Systemdokumentasjon og vedlegg K: Kravdokumentasjon.



Figur 4.33: Strukturen til Wikien.

5. Diskusjon

I dette kapittelet drøftes de aktuelle resultatene. Det trekkes frem diskusjon rundt positive elementer og forbedringspunkter for resultatet.

5.1 Forskningsresultater

Utforskningen for å finne ut om konseptet Lokal er levedyktig som produkt i samfunnet, tar grunnlag i markedundersøkelsen utført i emnet ”Ingeniørfaglig Systemtenkning” (se vedlegg H).

5.1.1 Markedsundersøkelse

Inntrykket fra markedsundersøkelsen som bestod av et strukturert intervju som en Google Forms-spørreundersøkelse og en rekke dybdeintervjuer med bønder og andre relevante personer (se kapittel 4.1.1), er at Lokal kan fungere som et reelt system. Spørreundersøkelsen hadde som mål å samle kvantitative data om folks forhold til handel av lokale varer som kunne brukes til å tolke hvilke funksjonaliteter som var nødvendige å inkludere i Lokal (se kapittel 2.1 for relatert teori).

Resultatene fra spørreundersøkelsen, der noen av de viktigste er presentert i kapittel 4.1.1, indikerer at mange svarer at det er dyrt å handle lokale råvarer. Dette kan være en barriere for implementeringen av Lokal som system, ettersom forbrukeren er kritisk til å bruke ekstra penger på lokale varer. På den andre siden, kan Lokal bidra til at markedet for lokale varer blir mer populært og drevet av konkurranse ved at folk kan vurdere utsalgsstedene. Denne markedskonkurransen kan potensielt senke prisene for forbrukeren, samt øke kvaliteten på produktene. (Busso og Galiani, 2019)

Mange mener også at å handle lokale varer krever mye tid og planlegging. Lokal kan potensielt bidra til å forenkle planleggingen ved å være et system som gir full oversikt over hva som finnes ved et utsalg, uten å trenge å besøke utsalget fysisk. I tillegg kan kartfunksjonaliteten hjelpe med å planlegge reisen ettersom kartet integrerer Google Maps på mobil, som automatisk setter opp en foreslått rute til utsalget. På dette vis kan kartfunksjonaliteten ha positiv innvirkning og være verdifull ettersom relativt mange svarer at ”Lokale råvarer er ikke lett tilgjengelige i ditt [mitt] område” og ”Er ikke klar over hvor man kan kjøpe lokale råvarer”. Samtidig svarer 75.4 prosent at de bor i et område der det er tilgang til gårdsutsalg eller andre utsalg som selger lokale produkter. Det at de fleste oppgir dette samtidig som mange synes lokale råvarer ikke

er lett tilgjengelige, kan tolkes som at kartfunksjonaliteten vil kunne hjelpe folk med å finne ut hvor utsalgene faktisk er.

Få respondenter har oppgitt at det er mangel på interesse for lokale råvarer eller at de tviler på kvaliteten. Disse resultatene betyr at respondentgruppen i liten grad hadde nevnte begrunnelser for hvorfor de ikke handlet flere lokale råvarer. Dette kan tolkes positivt for Lokal, ettersom disse begrunnelsene representerer de sterkeste motargumentene mot bruk av systemet.

Selv om trenden av tilbakemeldinger gjennomgående sett var positiv, og at spørsmålene forsøkte å ikke være ledende, er det viktig å nevne at resultatene fra spørreundersøkelsen kan være påvirket av at brukergruppen som ble spurt kan ha hatt en allerede eksisterende interesse for landbruksmarkedet. Dermed kan innsiktene være mindre representative for samfunnet enn antatt.

Som nevnt i resultater, ble det også utført en rekke omfattende dybdeintervjuer som del av markedsundersøkelsen. Selv om disse ikke var direkte rettet mot problemstillingen i denne rapporten, ga de god innsikt i at flere relevante personer (se kapittel 4.1.1) mente Lokal kunne fungere som et reelt system i samfunnet. Imidlertid ble det også funnet at det eksisterer potensielle konkurrenter til Lokal som system. Likevel fant teamet ut at alternativene ikke hadde etablert et godt system for å bruke mobilapplikasjoner, noe som kan ses som en verdi ved Lokal. I tillegg ble det presentert av oppstarten av Lokal kunne være vanskelig, ettersom det ville kreve en stor brukergruppe. I forhold til dette konkluderte teamet at det ville bli viktig å markedsføre de identifiserte verdiene ved systemet på en god og oversiktlig måte ved reell implementasjon i samfunnet.

5.1.2 Brukertester

Tilbakemeldingene fra brukertestene teamet utførte underveis var til stor nytte for å følge opp problemstillingen med kvalitative data. Brukertestene var en måte for teamet å praktisere Design and Creation, som omtalt i kapittel 3.1.1. Imidlertid, kunne flere slike dybdeintervjuer med reelle utselgere ha bidratt til bredere innsikt i hvilke funksjonaliteter som er verdifulle for dem. Dersom prosjektet hadde hatt en større tidsramme, kunne det også vært verdifult å intervjue utselgere fra andre typer utsalg, som for eksempel loppemarked og bruktutsalg, for å undersøke deres interesse i å benytte et system som Lokal.

Teamet er tilfreds med hvordan systemet ble endret basert på observasjoner fra de ti brukertestene gjennomført underveis. I tillegg mener teamet brukertestene av MVPen var essensielle for å sikre tidlige tilbakemeldinger og ønsker (se tabell 4.1). Fra brukertestene fikk teamet tilbakemelding om at brukergrensesnittet var intuitivt, og hadde et generelt sett godt design. Dette bekreftet for teamet verdien av Design Sprinten, som omtalt i kapittel 2.4.4.1 og 3.2.3. Prototypen produsert i Design Sprinten førte

nemlig til enighet om hvordan designet for MVPen og sluttproduktet skulle se ut, noe som igjen bidro til en mer effektiv og målrettet utviklingsprosess.

5.2 Utviklingsresultater

5.2.1 Mobilapplikasjon for forbrukere

5.2.1.1 Design

Teamet har oppnådd brukergrensesnitt preget av et profesjonelt design, noe som bekreftes og understøttes av brukertestene. Enkelheten i strukturen gjør grensesnittet brukervennlig ved første øyekast, ettersom det er lagt vekt på å ikke overvelde brukeren med for mye informasjon på én gang. For mye informasjon presentert samtidig, kan gjøre at applikasjonen oppfattes rotete og uoversiktlig, som nevnt i kapittel 2.2.1. Et eksempel på dette i applikasjonen er at brukeren ikke får se alle individuelle tilbakemeldinger for hvert utsalg, i stedet presenteres et gjennomsnitt av vurderinger som et tall (se figur 4.10).

Ved å ta i bruk mange symboler og ha veletablerte løsninger som Wolt, Foodora og TooGoodToGo som inspirasjonskilder, forbedres brukeropplevelsen (se kapittel 2.2.1). Dette ettersom brukeren blir introdusert til et system som ikke krever opplæring, siden elementer virker kjente og intuitive. Dette ble tydelig under brukertestene, hvor ingen av testpersonene trengte veiledning for å bruke applikasjonen, noe som er positivt. Troverdigheten og formålet til systemet forsterkes også ved at det har visse likhetstrekk med de nevnte etablerte aktørene, noe som kan bidra til at systemet oppleves mer profesjonelt. På figur 4.7 kan man se flere av disse likhetstrekkene som forbrukerapplikasjonen har med andre applikasjoner som Foodora og Wolt.

Applikasjonen tar i bruk et passordløst design for innlogging, som nevnt i kapittel 2.2.3, kan forbedre brukeropplevelsen. Denne tilnærmingen har vært sentral i systemet, og tilbakemeldingene fra brukertestene har gitt uttrykk for at det har bidratt til en intuitiv og positiv opplevelse. Dette er mulig ved bruk av Supabase sin OAuth-integrasjon, som omtalt i kapittel 3.3.3.6.

5.2.1.2 Funksjonalitet

Som vist i kapittel 4.2.5, ble de aller fleste av de planlagte funksjonalitetene implementert i forbrukerapplikasjonen. Det endelige resultatet var tilfredsstillende funksjonelt sett, hvor funksjonene ble implementert på en god og intuitiv måte. Spesielt kartfunksjonaliteten er noe både teamet og intervjuobjektene ser på som en viktig og kritisk funksjon. Implementeringen viste seg å være viktigere enn først antatt for utselgere, som nevnt i intervjuet med Sissel og intervjuet med Christel.

For forbrukere er det essensielt å tilby effektiv brukerhjelp for enhver utfordring som måtte oppstå, spesielt når problemene involverer økonomiske transaksjoner og fy-

siske produkter (som vist i figur 4.9). Denne funksjonaliteten vil mest sannsynlig bli brukt av en liten del av brukerne, men det er viktig å ha på plass i en digital markedsplass som Lokal. Dette ettersom kundeservice som tilbud kan forbedre inntrykket av systemet, slik at det framstår mer helhetlig og troverdig.

Det er enkelte funksjonaliteter som var planlagt, men som ikke ble implementert. De fleste av disse var det forventet fra starten av at de mest sannsynlig ikke ville bli implementert på grunn av begrenset utviklingstid. Teamet hadde høye ambisjoner og klare prioriteringer mellom de ulike funksjonalitetene, så det faktum at kun noen få ikke ble implementert betraktes av teamet som positivt.

5.2.2 Nettside og mobilapplikasjon for utselgere

5.2.2.1 Design

Etter intervjuene med personer relatert til gårdsdrift, ble det lagt stor vekt på at det ikke skulle være for omfattende å registrere og administrere et utsalg i systemet (se vedlegg D). Teamet har forsøkt å prioritere dette ved å tilby en så enkel brukeropplevelse som mulig. Det modulære designet av kontrollpanelet på nettsiden bidrar i stor grad til dette. Samtlige brukertestere har vært positive til denne tilnærmingen, og det er et designvalg som teamet har lykkes godt med. Det å gjøre brukergrensesnittet modulært på denne måten, bidrar til at informasjonsflyten blir så selektiv og oversiktlig som mulig, som nevnt i kapittel 2.2.1.

Å gjøre kontrollpanelet på nettsiden så oversiktlig som mulig har vært en viktig prioritering under utviklingsprosessen. Dette fordi utselgere vil finne mesteparten av informasjon og verktøy for å administrere utsalget sitt i kontrollpanelet. Brukertester har vist at testpersonene enkelt tar i bruk og lærer seg systemet. For de som eventuelt ikke forstår brukergrensesnittet og ønsker veiledning, er det tilgjengelig via kontrollpanelet, som vist i figur 4.17. Dette øker tilgjengeligheten og forbedrer den generelle brukeropplevelsen.

Utselgerapplikasjonene benytter også det passordløse designet fra forbrukerapplikasjonen, noe som bidrar til en helhetlig brukeropplevelse, samt forenkler registreringsprosessen for nye utselgere.

5.2.2.2 Funksjonalitet

Fra kapittel 4.2.5 er det tydelig at utselgere har fått implementert mange funksjonaliteter for administrering av utsalg. Det er viktig at utselgere opplever at de har nok muligheter til å tilpasse hvordan de ønsker at utsalget deres skal fremstå i systemet, da systemet skal fungere som en kanal for å introdusere nye kunder til utsalget.

Fra brukertesting av utselger-funksjonene ble det tydelig at mange av de implementerte funksjonene er svært nyttige. For de fleste er det tilstrekkelig funksjonalitet for å kunne drive et utsalg. Imidlertid har intervjuer og andre brukertester resultert med

forslag om et par nye funksjonaliteter, som ikke var med i den opprinnelige kravlisten. Disse kan ses i kapittel 4.1.2.

Under redegjøres det for noen av de foreslåtte funksjonalitetene:

- Muligheten til å laste opp et større bildegalleri bestående av seks bilder ble implementert. Under intervjuene var det opprinnelig kun mulig å laste opp ett bilde av utsalget, men det kom tilbakemeldinger fra flere utselgere som ønsket å vise frem utsalget sitt mer, ved å implementere muligheten til å ha et bildegalleri av utsalget.
- Visuell tilbakemelding som viser hvor langt utselgeren er kommet i registreringsprosessen, ble implementert etter tilbakemeldinger fra en brukertest for å gjøre prosessen mer oversiktlig.
- Muligheten for å angi et hentetidspunkt var et gjentagende forslag fra ulike tester og intervjuer. Likevel ble dette ikke implementert i systemet fordi forslaget ble fremmet sent i utviklingsprosessen, noe som førte til at implementeringen ble nedprioritert på grunn av tidspress. Dette er derfor en viktig funksjonalitet for videre utvikling som bør implementeres før en eventuell lansering.

I visjonsdokumentet og fra prosjektets start var det planlagt at utselger skulle ha mulighet til å registrere og administrere flere utsalg. Denne funksjonaliteten ble implementert i MVP-en, men utover i utviklingsprosessen viste det seg at denne funksjonaliteten førte til mer forvirring enn nytte. Derfor ble funksjonaliteten fjernet fra det endelige systemet. I Supabase-databasen er det fortsatt mulig å registrere flere utsalg per utselger, noe som gjør en eventuell fremtidig implementasjon relativt enkel.

Teamet identifiserte også tidlig i prosessen at de fleste utselgere vil være skattepliktige for salg som gjennomføres via Lokal. Å ha et eget regnskap vist i figur 4.26, som oppsummerer hvor mye som er solgt av ulike produkter samt totalsalget fra utsalget, er derfor en nyttig ressurs for føring av skatt og andre juridiske dokumenter.

Mobilapplikasjonen inneholder et mindre utvalg av funksjonalitet enn nettsiden, men applikasjonen er likevel høyt verdsatt av teamet, test- og intervjuobjektene. Dette skyldes at visse aspekter av administreringen er enklere å håndtere fra en mobil enn fra en nettside. For eksempel er det lettere å registrere nye produkter og bestemme om utsalget skal være åpent eller ikke fra mobil, som nevnt i intervjuet med Christel (se vedlegg E). Disse funksjonalitetene, i tillegg til evnen til å motta ordre i sanntid (se kapittel 4.2.3.7), gjør mobilapplikasjonens rolle svært viktig i systemet.

5.2.3 Nettside for administratorer

Administratorfunksjonaliteten på nettsiden var det siste som ble implementert i systemet. Dette ga teamet muligheten til å benytte eksisterende komponenter fra utsel-

gerkontrollpanelet til å raskere skape egne komponenter for administratorer. Dette ble gjort i henhold til sprintplanleggingen, noe teamet i retrospekt syntes fungerte godt for å sikre en effektiv utvikling av administratorfunksjonaliteten.

Det ble lagt lite fokus på å tilby administratorene en unik brukeropplevelse. Dette valget er i stor grad basert på målgruppen som skal bruke administratordelen av systemet. Grunnet gjenbruk av tidligere utviklede komponenter, blir brukergrensesnittet likevel intuitivt og brukervennlig.

I stedet for å designe en tilpasset brukeropplevelse for administratorer, ble det lagt større vekt på å implementere de nødvendige funksjonalitetene for at økosystemet skal fungere som en helhet. Administratorfunksjonene utgjør en kritisk del av infrastrukturen for at alle aspekter av systemet skal fungere optimalt.

5.2.4 Begrensninger ved systemet

5.2.4.1 Kostnader

Supabase tilbyr sine tjenester gratis ved mindre bruk, noe som har vært nyttig gjennom store deler av utviklingsprosessen. Ved oppskalering av systemet vil imidlertid kostnadene øke. Supabases prismodell er basert på forbruk, noe som kan gjøre det kostnadseffektivt hvis serveren konfigureres korrekt (Supabase, 2024d). Det finnes også mulighet for å kjøre Supabase-tjeneren på egen maskin, men dette kan potensielt medføre høyere kostnader.

Google-API tilbyr praktiske funksjonaliteter, som nevnt i kapittel 3.3.5.2. Disse funksjonalitetene anvendes ved flere anledninger i systemet. Bruken av Google-API kan imidlertid medføre betydelige kostnader, spesielt dersom systemet skaleres og bruken øker. Kostnadene varierer avhengig av hvilke funksjoner som brukes og hvor ofte de kalles. Det er derfor viktig å unngå unødvendige API-kall, noe som kan øke kostnadene betraktelig. Under utviklingen har imidlertid bruk av APIet vært kostnadsfritt, da Google tilbyr en månedlig kvote for API-bruk (Google, 2024). Denne kvoten vil mest sannsynlig bli overskredet dersom systemet blir lansert.

5.2.4.2 Ansvarsfraskrivelse

Bruk av tredjepartstjenester som Google-API og Supabase innebærer en ansvarsfraskrivelse fra utviklerens side. Nedetid eller endringer i APIene kan påvirke applikasjonen negativt uten forvarsel, og i verste fall medføre at systemet Lokal slutter å fungere. Det vil derfor være viktig at utviklerne er oppdaterte på eventuelle endringer hos tredjepartene, og raskt responderer på potensielle utfordringer.

5.2.4.3 Utviklingstid

Et omfattende og stort system som Lokal, kunne hatt fordel av lengre utviklingstid enn tildelt i bachelorperioden. Det er enkelte funksjonaliteter som ikke ble implementert,

men som ville ha vært et nyttig bidrag til systemet. Utviklingstiden har imidlertid økt, ettersom ny teknologi som Supabase måtte læres. Selv om resultatet har solide implementasjoner av ulike funksjonaliteter på en effektiv måte ved bruk av Supabase, hadde systemet trengt litt mer utvikling for å bli lansert for reell bruk, som utdypes i kapittel 6.2.

5.2.4.4 Krav om bruk av mobilenheter

En annen begrensning ved systemet er at utselgerapplikasjonen forutsetter at utselgerne har tilgang til en mobilenhet. Dette kan være en barriere for enkelte brukere for å utnytte systemet fullt ut. Selv om det er mulig å administrere sitt utsalg kun fra nettsiden, vil brukeropplevelsen være mindre ideell uten bruk av utselgerapplikasjonen (se vedlegg E). Funksjonaliteten for å markere ordre som hentet er ikke tilgjengelig på nettsiden, og forutsetter bruk av enten forbruker- eller utselgerapplikasjonen.

5.3 Administrative resultater

5.3.1 Smidig utivkling

Bruken av Scrum som rammeverk bidro til en strukturert tilnærming der alle teammedlemmer var klar over sine oppgaver og tidsfrister. Dette bidro til å opprettholde tilfredsstillende utviklingshastighet og sikre at teamet hadde progresjon.

En annen viktig fordel ved at teamet brukte Scrum var kontinuerlige tilbakemeldinger. Gjennom Sprint Reviews og retrospektiv fikk teamet muligheten til å reflektere over hva som fungerte godt og hva som kunne forbedres. Dette førte til fortløpende forbedring underveis, og teamet mener at Scrum dermed har bidratt til en strukturert og godt planlagt prosess.

5.3.2 Prosess

Prosessen som ble fulgt i prosjektet involverte flere nøkkelkomponenter. Forprosjektplanen og visjonsdokumentet la grunnlaget for prosjektets mål og omfang. Gjennom hele utviklingsperioden ble GitLab brukt som versjonskontrollsystem, og dette verktøyet spilte en kritisk rolle i å organisere og spore fremdrift. GitLab Issue-Board ble brukt til å administrere oppgaver (som vist i kapittel 3.2.2.8), og milepælfunksjonaliteten forbedret muligheten til å holde oversikt over milepæler og sprintmål underveis.

Et viktig aspekt ved prosessen var fokuset på brukertesting. Regelmessige brukertester med faktiske brukere, og to verdifulle brukertester med ekte utselgere som ga verdifull innsikt for å justere og forbedre systemet. Dette bidro til å sikre at det endelige produktet møtte behov og forventninger, samt at problemstillingen for bachelorrapporten ble etterfulgt.

5.3.3 Arbeidsfordeling

Arbeidsfordelingen i teamet var avgjørende for å sikre effektivitet og kvalitet i resultatet Lokal som helhetlig produkt. Hvert medlem i teamet hadde spesifikke ansvarsområder basert på deres styrker og ekspertise. For eksempel, hadde en utvikler fokus på frontend-utvikling ved bruk av React og React Native, en annen jobbet hovedsakelig med integrasjon av frontend mot backend og kode-testing, mens den siste tok ansvar for backend-utvikling med Supabase. Denne klare arbeidsfordelingen bidro til å minimere overlapp og konflikt mellom arbeidsoppgaver, og sørget for at alle aspekter av prosjektet fikk nødvendig oppmerksomhet. Dessuten gjorde det teammedlemmene i stand til å spesialisere seg og bli mer produktive innenfor sine respektive områder, noe som støttet prosjektets overordnede utviklingsfokus på rask implementasjon av funksjonaliteter, slik at sluttproduktet ville kunne tilby et stort utvalg av funksjonaliteter for utselgere.

6. Konklusjon og videre arbeid

Lokal som system er utviklet for å implementere funksjonaliteter som faktisk er brukbare og verdifulle for utselgere. Under utviklingen ble det kontinuerlig utforsket hvilke funksjonaliteter ved konseptet som var relevante for å skape et slikt system. Dette kapittelet utdyper konklusjonen for problemstillingen og utviklingen generelt. I tillegg presenteres punkter for videre arbeid.

6.1 Problemstilling og utviklingsfokus

Som svar på problemstillingen, kan det konkluderes at den mest verdifulle funksjonaliteten ved Lokal er kart-oversikten over alle utsalg i nærheten. Dette var mulig å integrere ved å bruke tjenester fra Google-API. Kartfunksjonaliteten støttes som viktig av brukertestene som ble utført underveis. I brukertest 5 og 6 fikk teamet innsikt i arbeidet på bondegård, etter å snakke med Sissel Langørgen (vedlegg D) og Christel Ossletten (vedlegg E), som bekreftet at kartfunksjonaliteten var interessant for å skaffe nye kunder. De så verdi i kartfunksjonaliteten, da de ikke visste om lignende alternativer.

Videre har teamet funnet at å enkelt kunne administrere utsalget sitt gjennom utselgerapplikasjonen, er en styrke ved Lokal som system. Funksjonaliteter som ble bekreftet som verdifulle for utselgeren var å kunne administrere produkter og åpningstid direkte i mobilapplikasjonen. Christel mente dette var en viktig funksjonalitet for utselgeren, da dette forenkler administreringen betraktelig sammenlignet med hennes nåværende løsning. Utover dette ble det også verdsatt at utselgeren kunne lese tilbakemeldinger fra forbrukere som har utført et kjøp hos dem i nettsiden, og at det er mulighet for å laste ned kvittering for transaksjoner.

Fokuset for utviklingen har i tråd med problemstillingen vært å produsere testbare funksjonaliteter så fort som mulig. Målet har vært å skape et helhetlig sluttprodukt med mange implementerte funksjonaliteter. Fremgangsmåten ved å utnytte en Design Sprint og for å kjapt produsere en MVP, fungerte dermed godt for hensikten om å teste funksjonaliteter tidlig i prosessen. Under utviklingen av MVPen og utover i prosessen, ble det tydelig for teamet at å utnytte Supabase som BaaS-løsning til stor grad fasiliterte rask og enkel implementasjon av ny funksjonalitet i systemet. Dette, i tillegg til planleggingen bidro til at teamet klarte å utvikle et system som møtte alle de overordnede kravene som definert i oppgaveteksten (se Oppgavebeskrivelse eller

vedlegg A).

For å skape et gjenkjennbart brukergrensesnitt gikk designet inn for å hente inspirasjon fra kjente applikasjoner som Wolt og Foodora. Teamet mener dette ble utført på en oversiktlig måte, hvor oppsettet av applikasjonene, samt brukeropplevelsen henter verdi ved å være intuitive og lignende det folk er vant til å bruke.

6.2 Videre arbeid

Som nevnt finnes det flere funksjonaliteter som kunne vært implementert i Lokal, for å fullføre systemet og gjøre det mulig å lansere produktet til offentlig bruk.

6.2.1 Allergener og oppbevaringsinformasjon

En tidligere nevnt funksjonalitet som er logisk å implementere, er informasjon om allergener og oppbevaring av solgte produkter i applikasjonen. Dette blir spesielt viktig ved salg av matvarer. Et av intervjuobjektene nevnte denne funksjonaliteten og argumenterte godt for den, så det er en naturlig videre utvikling. (Se vedlegg E)

6.2.2 Betalingsløsning

Det nåværende systemet mangler en implementert betalingsløsning, noe som absolutt må være på plass før produktet lanseres. Betalinger bør foregå så enkelt som mulig for forbrukerne, så det er fornuftig å benytte seg av veletablerte betalingsløsninger. I Norge er Vipps et naturlig valg, mens mobilbetalinger gjennom Apple Pay og Google Pay også er praktiske alternativer for mange brukere av forbrukerapplikasjonen. Å tilby muligheter for å gjennomføre kjøp uten forbrukerapplikasjonen bør også være et mål, ettersom dette kan bidra til å enklere integrere allerede eksisterende kunder hos utsalgene inn i systemet.

6.2.3 Hentetidspunkt

Fra brukertestene og intervjuene ble det diskutert muligheten for å innføre et hentetidspunkt for ordre, noe som kan forenkle gjennomføringen av ordre for både utselger og forbruker. På grunn av systemets struktur og kompleksitet da denne funksjonen ble foreslått, var det ikke gjennomførbart å implementere på så kort tid. Dette er en funksjonalitet som ideelt sett burde være på plass.

6.2.4 Tilgjengeliggjøring av brukerdata

For å oppnå tilfredsstillende personvern er det essensielt at registrerte brukere enkelt kan få tilgang til all data som er lagret om dem i systemet. Til tross for at det nåværende systemet ikke lagrer mye data om hver enkelt bruker, har brukerne ikke tilgang til all lagret informasjon. Dette bør implementeres i fremtiden.

6.2.5 Skalering og optimalisering

Den nåværende infrastrukturen har ikke kapasitet til mye mer enn selve utviklingen av systemet. Det er viktig at den oppgraderes dersom applikasjonen skal publiseres for bruk. Dette kan enten skje ved fortsatt bruk av en tredjepartstjeneste slik som i dag, eller ved å overføre systemet til en egen server. Systemets ytelse under høy brukerbelastning bør undersøkes og forbedres.

Systemet kunne også vært bedre optimalisert på flere områder. For eksempel kunne lagringen av bilder på serveren vært mer effektiv ved å implementere skalering eller andre optimaliseringsmetoder for å redusere lagringsbehovet. Det er lite behov for høyoppløselige bilder i applikasjonen, så dette er definitivt en gunstig videre utvikling.

Samfunnspåvirkning

Lokal som system kan bidra til positiv utvikling i samfunnet. Dette kapittelet omhandler samfunnspåvirkningen som resultat av eventuell implementasjon av systemet.

Et viktig eksempel er at Lokal kan øke folks kunnskap om hvor det går an å handle lokale varer. At folk får informasjon om sine lokale muligheter, kan føre til at det blir mer vanlig å handle lokalproduserte varer. I rapporten *Bærekraftig matforsyning: En direkte forsyningskanal mellom bonde og forbruker* skrevet av teamet i "Ingeniørfaglig Systemtenkning", se vedlegg H, ble det funnet at en rekke norske politiske partier ønsker å heve selvforsyningsgraden i Norge. Lokal, dersom implementert i storskala, kan potensielt skape en trend som gjør at folk handler mer lokalt, og dermed øker samfunnets selvforsyningsgrad. (Vedlegg H)

Implementasjonen av et system som Lokal i samfunnet kan bidra til redusert matsvinn. Dette skyldes at man unngår typiske mellomledd som finnes i forsyningskjeden fra produsent til matbutikk, og man oppmuntrer flere forbrukere til å velge lokale alternativer. Tall fra FN viser at omtrent 14 prosent av all mat går tapt i produksjonskjeden, noe som Lokal potensielt kunne bidratt til å motvirke. (FN, 2023, Vedlegg H)

Som videre diskutert i rapporten *Bærekraftig matforsyning: En direkte forsyningskanal mellom bonde og forbruker* kan et system som Lokal bidra til at samfunnet til større grad etterlever FNs miljø- og bærekraftsmål. Målene som nevnes i rapporten er innenfor bærekraftsmål nummer 12: "ansvarlig forbruk og produksjon" med delmål som omhandler for eksempel å sikre at folk har informasjon om bærekraftig utvikling, å halvere matsvinn, og å fremme lokal kultur og lokale produkter (FN-sambandet, 2024). Selv om Lokal er begrenset av å være et digitalt system, kan budskapet om å handle mer lokalt være en viktig bieffekt ved implementasjon i samfunnet. (Vedlegg H)

På den andre siden, kan også systemet tilby en plattform for illegal handel. Dette kan fremstå som et problem i samfunnet dersom det ikke er kontroll av handelen som utføres via Lokal. Som mottiltak har teamet fokusert på å implementere administratorrollen, som kan benyttes for å regulere og motvirke slike aktiviteter. Likevel er det viktig å vurdere ulovlig handel som en potensiell negativ samfunnspåvirkning ved implementasjonen av Lokal.

For å konkludere, kan Lokal som system ha flere positive innvirkninger i samfunnet. Likevel er det fortsatt viktig å ta hensyn til potensielle negative aspekter, som ulovlig handel, dersom systemet skulle implementeres for reell handel i samfunnet.

Bibliografi

42.8. Transaction Management. (2024 februar). Hentet 30. mars 2024, fra https://www.postgresql.org/docs/12/plpgsql-transactions.html

Alan, B. (2020 april). *Learning SQL: Generate, Manipulate, and Retrieve Data* (3. utg.). O'Reilly.

Alsaqqa, S., Sawalha, S., & Abdel-Nabi, H. (2020). Agile Software Development: Methodologies and Trends [Number: 11]. *International Journal of Interactive Mobile Technologies (iJIM)*, *14*(11), 246–270. https://doi.org/10.3991/ijim.v14i11.13269

Amanuel, A. (2022 mai). Supabase vs Firebase: Evaluation of performance and development of Progressive Web Apps. Hentet 19. februar 2024, fra https://www.theseus.fi/bitstream/handle/10024/771009/Ayezabu_Amanuel.pdf?sequence=2&isAllowed=y

Araújo, C., Santos, I., Canedo, E., & Araújo, A. (2019 juli). Design Thinking Versus Design Sprint: A Comparative Study. https://doi.org/10.1007/978-3-030-23570-3_22

Biehl, M. (2017 desember). *Webhooks – Events for RESTful APIs* [Google-Books-ID: 5j64DwAAQBAJ]. API-University Press.

Bratbergsengen, K., & Mallaug, T. (2024 februar). database. Hentet 14. mai 2024, fra https://snl.no/database

Brindescu, C., Codoban, M., Shmarkatiuk, S., & Dig, D. (2014). How do centralized and distributed version control systems impact software changes? *Proceedings of the 36th International Conference on Software Engineering*, 322–333. https://doi.org/10.1145/2568225.2568322

Busso, M., & Galiani, S. (2019). The Causal Effect of Competition on Prices and Quality: Evidence from a Field Experiment. *American Economic Journal: Applied Economics*, *11*(1), 33–56. https://doi.org/10.1257/app.20150310

Carranza-García, F., Rodríguez-Domínguez, C., Garrido, J. L., & Guerrero-Contreras, G. (2016). BaaS-4US: A Framework to Develop Standard Backends as a Service for Ubiquitous Applications. *2016 15th International Conference on Ubiquitous Computing and Communications and 2016 International Symposium on Cyberspace and Security (IUCC-CSS)*, 23–30. https://doi.org/10.1109/IUCC-CSS.2016.012

Coelho, B. J. T. (2022 februar). Tips fra en UX-designer. Hentet 30. april 2024, fra https://ndla.no/subject:1:1352b19e-e706-4480-a728-c6b0a57ba8ae/topic:9b328b18-c4d4-446c-a3a4-b69dccecda45/resource:e6ff95b6-a21c-453a-b38e-193a8440b9de

Cohn, M. (2013). *User stories applied: for agile software development* (18. print). Addison-Wesley.

Creswell, J. W. (2012). *Educational Research: Planning, Conducting, and Evaluating Quantitative and Qualitative Research* [Google-Books-ID: 4PywcQAACAAJ]. Pearson.

Dalen, M. (2013). *Intervju som forskningsmetode: en kvalitativ tilnærming* (2. utg). Universitetsforl.

Dinku, Z. (2022). React.js vs. Next.js.

Elisabeth Thoresen, O., & Sundbye, L. M. T. (2021 oktober). Observasjon, eksperiment og dybdeintervju - Innovasjon og markedsføring (SR-SRL vg2) - NDLA. Hentet 30. april 2024, fra https://ndla.no/subject:1:a7c337ca-d3b6-492f-ace2-b05c45f54e93/topic:1:1254b264-03b8-406c-b529-e4af3e9182fb/topic:1:76b6c12f-7f63-4a04-aa48-22d07fdc2fa7/resource:35f48ad2-0e29-4201-83fe-1be598982c5b

Expo. (2024). Expo Documentation. Hentet 3. mai 2024, fra https://docs.expo.dev

Fentaw, A. E. (2020). Cross platform mobile application development: a comparison study of React Native Vs Flutter.

FN. (2023 februar). Ansvarlig forbruk og produksjon. Hentet 16. mai 2024, fra https://fn.no/om-fn/fns-baerekraftsmaal/ansvarlig-forbruk-og-produksjon

FN-sambandet. (2024 februar). FNs bærekraftsmål. Hentet 16. mai 2024, fra https://fn.no/om-fn/fns-baerekraftsmaal

Gao, X., Yang, Y., Liu, C., Mitropoulos, C., Lindqvist, J., & Oulasvirta, A. (2018). Forgetting of Passwords: Ecological Theory and Data.

Gillis, A. S. (2024 mai). What is native app? | Definition from TechTarget. Hentet 20. mai 2024, fra https://www.techtarget.com/searchsoftwarequality/definition/native-application-native-app

GitHub, I. (2024 april). About GitHub Pages. Hentet 1. mai 2024, fra https://docs.github.com/en/pages/getting-started-with-github-pages/about-github-pages

Google. (2024). Platform Pricing & API Costs. Hentet 10. mai 2024, fra https://mapsplatform.google.com/pricing/

Grønmo, S. (2023 august). kvalitativ metode. Hentet 30. april 2024, fra https://snl.no/kvalitativ_metode

Haerder, T., & Reuter, A. (1983). Principles of transaction-oriented database recovery. *ACM Computing Surveys*, *15*(4), 287–317. https://doi.org/10.1145/289.291

Hall, A., & Ramachandran, U. (2019). An execution model for serverless functions at the edge. *Proceedings of the International Conference on Internet of Things Design and Implementation*, 225–236. https://doi.org/10.1145/3302505.3310084

Hansen, K. T., & Mallaug, T. (2008). *Databaser* (2. utg) [OCLC: 1028385277]. Gyldendal akademisk.

Hethey, J. M. (2013). *GitLab repository management: delve into managing your projects with GitLab, while tailoring it to fit your environment*. Packt Publ.

Higuchi, M. M., & Nakano, D. N. (2017). Agile Design: A Combined Model Based on Design Thinking and Agile Methodologies for Digital Games Projects. *Revista de Gestão e Projetos*, *08*(02), 109–126. Hentet 14. mai 2024, fra https://www.semanticscholar.org/paper/Agile-Design%3A-A-Combined-Model-Based-on-Design-and-Higuchi-Nakano/66f206c9dd0a183a713023c9f1eba6635816e4cf

Inc., D. L. (2024). Deno, the next-generation JavaScript runtime. Hentet 19. mai 2024, fra https://deno.com/

Initiative (WAI), W. W. A. (2024 mars). WCAG 2 Overview. Hentet 29. januar 2024, fra https://www.w3.org/WAI/standards-guidelines/wcag/

JetBrains. (2024 april). The Six Most Popular Cross-Platform App Development Frameworks | Kotlin Multiplatform Development. Hentet 2. mai 2024, fra https://www.jetbrains.com/help/kotlin-multiplatform-dev/cross-platform-frameworks.html

Joshi, M. (2023 november). Angular vs React vs Vue: Core Differences. Hentet 30. april 2024, fra https://browserstack.wpengine.com/guide/angular-vs-react-vs-vue/

Khleel, N., & Nehéz, K. (2020). Comparison of version control system tools. *10*, 61–69. https://doi.org/10.35925/j.multi.2020.3.7

Knapp, J., & Zeratsky, J. (2024 april). The Design Sprint. Hentet 30. april 2024, fra https://www.thesprintbook.com/the-design-sprint

Kolesnikov, O., Golovko, G., Yastreba, V., & Piatyntsev, Y. (2024). LEVERAGING CLOUD TECHNOLOGIES AND SERVERLESS ARCHITECTURE FOR EFFICIENT WEB DEVELOPMENT: A CASE STUDY FROM REAL-WORLD APPLICATION [Number: 75]. *Системи управління, навігації та зв'язку. Збірник наукових праць*, *1*(75), 98–103. https://doi.org/10.26906/SUNZ.2024.1.098

Krug, S. (2014). Don't make me think, revisited: a common sense approach to Web usability. *Choice Reviews Online*, *51*(11), 257. https://doi.org/10.5860/CHOICE.51-6218

Lee, D., Mao, W., Chiu, H., & Chu, W. W. (2005). Designing Triggers with Trigger-By-Example. *Knowledge and Information Systems*, *7*(1), 110–134. https://doi.org/10.1007/s10115-003-0126-5

Lid, I. M. (2024 april). universell utforming. Hentet 20. mai 2024, fra https://snl.no/universell_utforming

Malt, U., & Grønmo, S. (2023 august). strukturert intervju. Hentet 16. mai 2024, fra https://snl.no/strukturert_intervju

Matiushin, I., & Korkhov, V. (2021). PASSWORDLESS AUTHENTICATION USING MAGIC LINK TECHNOLOGY. *9th International Conference "Distributed Computing and Grid Technologies in Science and Education"*, 434–438. https://doi.org/10.54546/MLIT.2021.89.13.001

McKay, E. (2013 mai). *UI is Communication*.

Meta Platforms, I. (2024 april). Testing · React Native. Hentet 7. mai 2024, fra https://reactnative.dev/docs/testing-overview

Most used web frameworks among developers 2023. (2023 juni). Hentet 1. mai 2024, fra https://www.statista.com/statistics/1124699/worldwide-developer-survey-most-used-frameworks-web/

Mwaura, W. (2021 januar). *End-to-End Web Testing with Cypress: Explore techniques for automated frontend web testing with Cypress and JavaScript* [Google-Books-ID: Er0YEAAAQBAJ]. Packt Publishing Ltd.

Nguyen, P. (2016). Mobile Backend as a Service: the pros and cons of parse [Accepted: 2016-11-23T11:10:34Z Publisher: Lahden ammattikorkeakoulu]. Hentet 2. mai 2024, fra http://www.theseus.fi/handle/10024/117483

Oates, B. J. (2006). *Researching information systems and computing*. London ; Thousand Oaks, Calif. : SAGE Publications. Hentet 6. mai 2024, fra http://archive.org/details/researchinginfor0000oate

Parmar, V., Sanghvi, H. A., Patel, R. H., & Pandya, A. S. (2022). A Comprehensive Study on Passwordless Authentication. *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, 1266–1275. https://doi.org/10.1109/ICSCDS53736.2022.9760934

Paul, A., & Nalwaya, A. (2019). Native Bridging in React Native. I A. Paul & A. Nalwaya (Red.), *React Native for Mobile Development: Harness the Power of React Native to Create Stunning iOS and Android Applications* (s. 165–186). Apress. https://doi.org/10.1007/978-1-4842-4454-8_7

Principles behind the Agile Manifesto. (2024 mai). Hentet 1. februar 2024, fra https://agilemanifesto.org/principles.html

Randolph, W. (2024 januar). sql-docs/docs/relational-databases/security/row-level-security.md at live · MicrosoftDocs/sql-docs. Hentet 6. februar 2024, fra https://github.com/MicrosoftDocs/sql-docs/blob/live/docs/relational-databases/security/row-level-security.md

Resend. (2024a mars). Domain . Resend. Hentet 6. mai 2024, fra https://resend.com/docs/dashboard/domains/introduction

Resend. (2024b april). Pricing · Resend. Hentet 6. mai 2024, fra https://resend.com/pricing

Resend. (2024c mai). Resend. Hentet 6. mai 2024, fra https://resend.com

Rocha, Z. (2023 august). How to configure Supabase to send emails from your domain · Resend. Hentet 6. mai 2024, fra https://resend.com/blog/how-to-configure-supabase-to-send-emails-from-your-domain

Rolstadås, A. (2019 juli). milepæl – prosjektledelse. Hentet 20. mai 2024, fra https://snl.no/milep%C3%A6l_-_prosjektledelse

Sachdeva, S. (2016). Scrum Methodology. *International Journal Of Engineering And Computer Science*. https://doi.org/10.18535/ijecs/v5i6.11

Saldamli, G., Doshatti, A., Kapadia, D., Nyati, D., Bodiwala, M., & Ertaul, L. (2021). Enterprise Backend as a Service (EBaaS). I H. R. Arabnia, L. Deligiannidis, M. R. Grimaila, D. D. Hodson, K. Joe, M. Sekijima & F. G. Tinetti (Red.), *Advances in*

Parallel & Distributed Processing, and Applications (s. 1077–1099). Springer International Publishing. https://doi.org/10.1007/978-3-030-69984-0_78

Salohonka, M. (2020). Automated testing of React Native applications [Accepted: 2020-03-04T12:17:35Z]. Hentet 7. mai 2024, fra https://lutpub.lut.fi/handle/10024/160734

Schwaber, K., & Sutherland, J. (2020). Scrum Guide | Scrum Guides. Hentet 30. april 2024, fra https://scrumguides.org/scrum-guide.html

Shahin, M., Ali Babar, M., & Zhu, L. (2017). Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices [Conference Name: IEEE Access]. *IEEE Access*, *5*, 3909–3943. https://doi.org/10.1109/ACCESS.2017.2685629

Sharif, A. (2022 desember). What Is CRUD? Create, Read, Update, and Delete - CrowdStrike. Hentet 6. april 2024, fra https://www.crowdstrike.com/cybersecurity-101/observability/crud/

Smith, G. (2010 oktober). *PostgreSQL 9.0: High Performance* [Google-Books-ID: OWOAu0GcsqoC]. Packt Publishing Ltd.

Supabase. (2024a mai). Configure a Custom SMTP | Supabase Docs. Hentet 6. mai 2024, fra https://supabase.com/docs/guides/auth/auth-smtp

Supabase. (2024b mai). Database Webhooks | Supabase Docs. Hentet 5. mai 2024, fra https://supabase.com/docs/guides/database/webhooks

Supabase. (2024c mai). Edge Functions | Supabase Docs. Hentet 18. mai 2024, fra https://supabase.com/docs/guides/functions

Supabase. (2024d). Pricing & Fees. Hentet 10. mai 2024, fra https://supabase.com//pricing

Supabase. (2024e mai). Realtime | Supabase Docs. Hentet 5. mai 2024, fra https://supabase.com/docs/guides/realtime

Supabase. (2024f mai). Social Login | Supabase Docs. Hentet 5. mai 2024, fra https://supabase.com/docs/guides/auth/social-login

Supabase. (2024g mai). Storage | Supabase Docs. Hentet 6. mai 2024, fra https://supabase.com/docs/guides/storage

Suto, H., Kawakami, H., & Handa, H. (2007). A Study of Information Flow Between Designers and Users Via Website Focused on Property of Hyper Links. I M. J. Smith & G. Salvendy (Red.), *Human Interface and the Management of Information. Methods, Techniques and Tools in Information Design* (s. 189–198). Springer. https://doi.org/10.1007/978-3-540-73345-4_23

Thakkar, M. (2020). Next.js. I M. Thakkar (Red.), *Building React Apps with Server-Side Rendering: Use React, Redux, and Next to Build Full Server-Side Rendering Applications* (s. 93–137). Apress. https://doi.org/10.1007/978-1-4842-5869-9_3

Varmey, T., Dendukuri, V. P., Higgins, J., & Tavana, D. (2023). Webhooks-as-a-Service: A Custom API Design, 48.

Ventures, G. (2024 april). The Design Sprint — GV. Hentet 30. april 2024, fra http://www.gv.com/sprint

Vercel, I. (2024 april). Learn | Next.js. Hentet 1. mai 2024, fra https://nextjs.org/learn

Vue.js [Page Version ID: 1216191150]. (2024 mars). Hentet 30. april 2024, fra https://en.wikipedia.org/w/index.php?title=Vue.js&oldid=1216191150

Waidelich, L., Richter, A., Kölmel, B., & Bulander, R. (2018). Design Thinking Process Model Review. *2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, 1–9. https://doi.org/10.1109/ICE.2018.8436281

WCAG-standarden | Tilsynet for universell utforming av ikt. (2024 mars). Hentet 26. januar 2024, fra https://www.uutilsynet.no/wcag-standarden/wcag-standarden/86

Zolkifli, N. N., Ngah, A., & Deraman, A. (2018). Version Control System: A Review. *Procedia Computer Science*, *135*, 408–415. https://doi.org/10.1016/j.procs.2018.08.191

Aanesen, K. H. (2020 november). Innsamling av kvantitative data - Sosiologi og sosialantropologi - NDLA. Hentet 14. mai 2024, fra https://ndla.no/subject:1:fb6ad516-0108-4059-acc3-3c5f13f49368/topic:1:860e0dc0-7691-4b90-ba3b-8a00c39c9448/topic:1:6422199b-cd4c-4728-8560-e357482c14d2/resource:02ab786e-689e-4301-b664-477f9da22a7f

A. Opprinnelig oppgavebeskrivelse

Arbeidstittel:	Digitalisering av landbruksmarkedet

Problemstilling
Undersøke hvordan en mobilapplikasjon effektivt kan koble lokale bønder direkte med forbrukere for å fremme salg av lokale produkter.

Kort beskrivelse av oppgaveforslag:
Dagens samfunn er i stor grad preget av digitalisering som påvirker nesten alle aspekter i livet vårt. Imidlertid henger visse områder etter i denne utviklingen, blant annet bondens gårdsutsalg. Ved å utvikle en applikasjon kan man utvide rekkevidden til den lokale bonden, samtidig som forbrukeren får en bedre oversikt over hvilke lokale alternativer som finnes.

Konseptet vil kunne ses som en mulighet for hverdagsbrukeren til å handle mer kortreist mat. Dette styrker ikke bare den lokale økonomien, men fremmer også en mer bærekraftig livsstil ved å redusere matens reiseavstand.

Den foreslåtte applikasjonen ønskes å være en ressurs som reduserer avstanden mellom selger og forbruker. Applikasjonen skal fungere som en digital markedsplass, hvor bonden kan selge direkte til forbrukeren, uten andre mellomledd.

Utfyllende kommentarer til hva oppgaven gjelder:
Overordnet funksjonalitet applikasjonen skal utfylle:
- Selger skal kunne opprette et utsalg og en digital katalog over invetar
- Forbruker skal kunne oppdage utsalg i nærområde
- Forbruker skal kunne gjennomføre kjøp av produkter fra et utsalg

Oppgaven passer for (kryss av de(t) som passer og skriv evt. en kommentar til oss):	- Bacheloroppgave
Hvilket studieprogram og emne passer oppgaven til? (spesifiseres ved bacheloroppgaver)	IDATT2900 – Bacheloroppgave Dataingeniør
Skal oppgaven utføres av bestemte studenter? (der avtalt) Fyll i så fall inn studentenes navn	John Fredrik Bendvold, Pedro Pablo Cardona, Magnus Lutro Allison
Kan oppgavestiller stille arbeidsplass med nødvendig utstyr og programvare:	Nei
Hvis ikke, hva kreves av maskin og programvare:	Egne maskiner
Oppgaven passer best for, antall studenter:	- 3

Opplysninger om oppgavestiller

Er du fra en bedrift/virksomhet eller er du student med en egendefinert/selvlaget oppgave?	- Student (egen oppgave)
Navn på bedrift/organisasjon/student:	John Fredrik Bendvold
Addresse	Ekebergveien 49
Postnummer	1639

Poststed	Gamle Fredrikstad
Navn på kontaktperson/veileder:	John Fredrik Bendvold
Telefon:	47270040
Epost:	j.f.bendvold@gmail.com

B. Forprosjektsplan

Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.

C. Visjonsdokument

Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.

D. Brukertest med Sissel

Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.

E. Brukertest med Christel

Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.

F. WCAG-rapport for nettsiden

Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.

G. WCAG-rapport for mobilapplikasjonene

Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.

H. Bærekraftig matforsyning: En direkte forsyningskanal mellom bonde og forbruker

Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.

I. Prosjekthåndbok

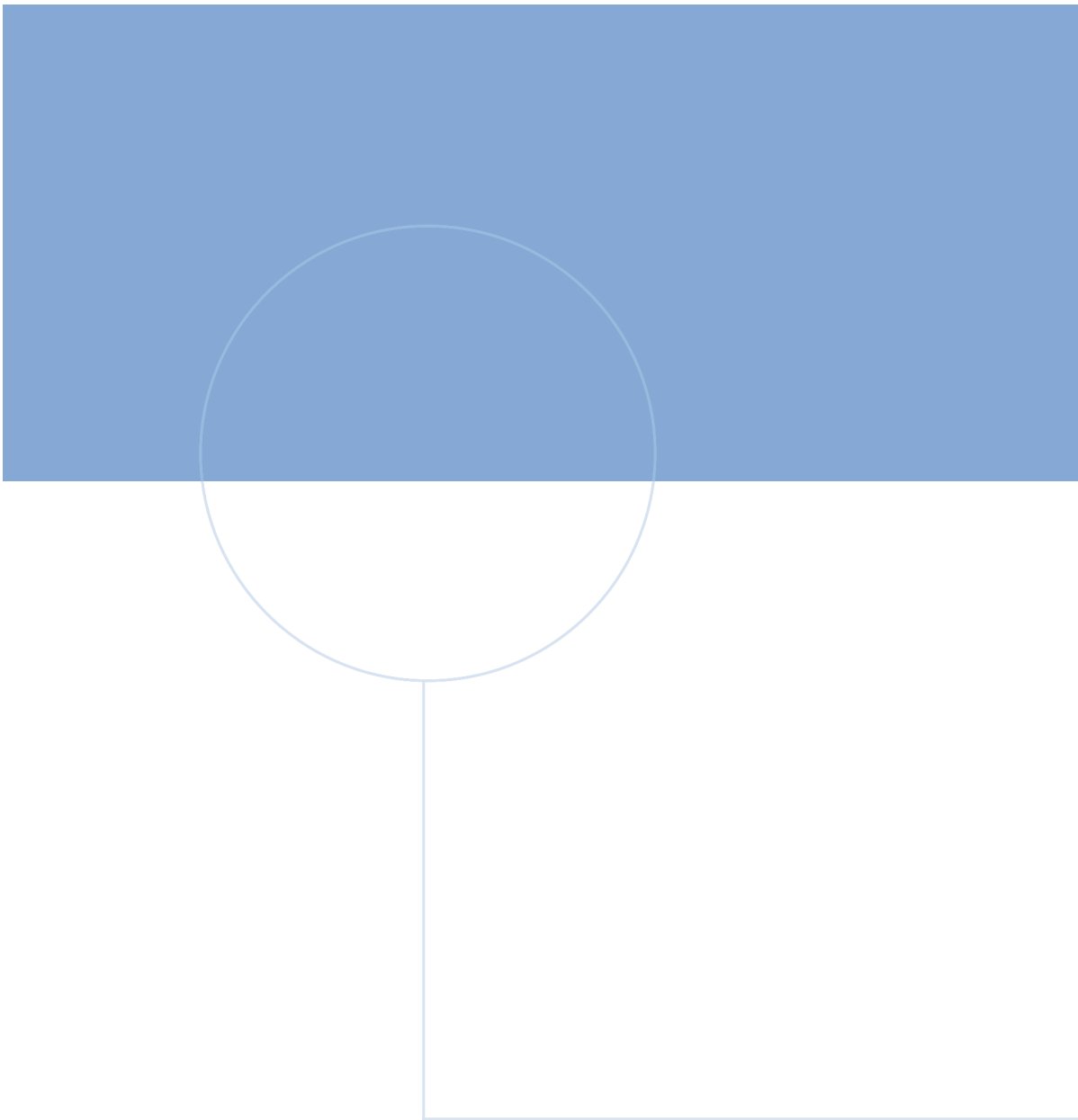
Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.

J. Systemdokumentasjon

Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.

K. Kravdokumentasjon

Dette dokumentet finnes i en ekstern mappe som ble levert ved siden av rapporten.



NTNU

Kunnskap for en bedre verden