Høgskulen
på Vestlandet

BACHELOROPPGAVE

Digitale verktøy for urbant landbruk i Bergen

Digital Tools for Urban Agriculture in Bergen

Rolf Caspar Snähre

David Korsrud Bråten

Eivind Lillestøl Skrede

Randi-Marie Pedersen

Dataingeniør
Fakultet for ingeniør- og naturvitskap
Institutt for data- og realfag

Veileder: Remy André Monsen

03.06.2019

Fakultet for ingeniør- og naturvitskap
Institutt for data- og realfag

TITTELSIDE FOR HOVEDPROSJEKT

Rapportens tittel:	*Dato:*
Digitale verktøy for urbant landbruk i Bergen	03.06.19
Forfatter(e):	*Antall sider u/vedlegg: 67*
Rolf Caspar Snähre David Korsrud Bråten Eivind Lillestøl Skrede Randi-Marie Pedersen	*Antall sider vedlegg: 6*
Studieretning:	*Antall disketter/CD-er:* 0
Dataingeniør	
Kontaktperson ved studieretning:	*Gradering:* (Velg: Ingen eller Begrenset til ddmmåå)
Remy Monsen	
Merknader:	

Oppdragsgiver:	*Oppdragsgivers referanse:*
Bybonden i Bergen/Fylkesmannen i Vestland	
Oppdragsgivers kontaktperson:	*Telefon:*
Ida Kleppe	473 31 590

Sammendrag:

Prosjektet gikk ut på å utvikle digitale verktøy for det urbane landbruket i Bergen. Det ble utviklet en nettside og to mobilapplikasjoner som skal publiseres på App Store og Google Play.

Stikkord:

React Native	Urbant landbruk	Mobilapplikasjon

Høgskulen på Vestlandet, Fakultet for ingeniør- og naturvitskap Postadresse: Postboks 7030, 5020 BERGEN Besøksadresse: Inndalsveien 28, Bergen

Tlf. 55 58 75 00 Fax 55 58 77 90 E-post: post@hvl.no Hjemmeside: http://www.hvl.no

Forord

Denne rapporten er skrevet av Rolf Caspar Snähre, David Korsrud Bråten, Eivind Lillestøl Skrede og Randi-Marie Pedersen våren 2019 og skal dokumentere arbeidet gjort på prosjektet "Digitale verktøy for urbant landbruk i Bergen". Dette er den avsluttende oppgaven på bachelorstudiet dataingeniør ved Høgskulen på Vestlandet, Bergen.

Vi vil gjerne takke vår veileder, Remy André Monsen, for hjelpen og tilbakemeldingene han har gitt oss gjennom prosjektet. Vi ønsker også å takke oppdragsgiveren vår Bybonden i Bergen, Ida Kleppe og Frøydis Lindén ved Fylkesmannen i Vestland for en spennende og utfordrende oppgave, samt et godt samarbeid gjennom prosjektet.

1 Introduksjon

I 2019 er det nærmest blitt et kriterie for virksomheter å tilby digitale løsninger på informasjonskanaler og virkemidler. Om ønsket er å appellere til et bredt publikum er det nødvendig å kunne tilby en løsning som er lett tilgjengelig og som alle kan innhente informasjon om, når de selv ønsker det. I det urbane landbruksmiljøet i Bergen ble det identifisert mangel på en slik digital plattform. Behovet gjaldt ikke bare for allerede etablerte i miljøet, men også for rekruttering av flere av Bergens innbyggere.

Miljøvern og fokus på en bærekraftig utvikling er i frammarsj og stadig flere ønsker å ta del i "det grønne skiftet". I Bergen eksisterer det en del organer som jobber for en grønnere hverdag, og et skritt som har blitt tatt i retning av dette var initiativet for å få en egen Bybonde. Gjennom Bybondeprosjektet skulle fleres oppmerksomhet rettes mot det urbane landbruket i Bergen. Etter Bybonden tiltrådte ble det identifisert et behov for verktøy og kanaler for å sette temaet på dagsorden, samt gjøre det enklere for publikum å engasjere seg.

1.1 Mål og motivasjon

Bybonden i Bergen, Fylkesmannens Landbruksavdeling og Bergen Kommune har sammen tatt initiativ til prosjektet. Hovedtanken bak initiativet var å finne en digital løsning med oversikt over Bergens dyrkningsområder. Flere og flere ønsker å ta del i "selvdyrker"-bølgen, men mangelen på jord er et problem. Idéen var derav å gjøre informasjon om dyrkbar jord lettere tilgjengelig for alle. Samtidig måtte det inkluderes en løsning for kommunikasjon mellom disponenter av jord og personer som har behov for områder å dyrke på.

Formålet med Bybondeprosjektet er å "[...] fremme urbant landbruk gjennom kunnskapsformidling og inspirasjon" (Hordaland Bondelag (2018)). Særlig det å rekruttere flere unge og uerfarne har vært et mål, og det var et ønske om å skape et lavterskeltilbud som gjør det enkelt å komme i gang med dyrking. Det være seg i disponibel hage, eller så enkelt som i vinduskarmen i leiligheten. For å oppnå dette hadde Bybonden i Bergen et ønske om en mobilapplikasjon. Her skulle det kunne deles kunnskap og instruksjoner for å komme i gang med enkle spiselige planter. Løsningen skulle også la brukerne følge løpet til Bybonden gjennom en sesong med dyrking.

Da Bybonden i Bergen var et nyoppstartet prosjekt manglet det en generell informasjonskanal for publikum å få vite mer om Bybonden og hennes arbeid. Målet var at dette skulle løses form av en hjemmeside, hvor det blant annet skulle publiseres oppdateringer og informasjon om arrangementer og tjenester.

Den overordnede motivasjonen bak prosjektet er dermed å skape et økt fellesskap rundt dyrking blant innbyggere i Bergen. Ønsket er at flere skal oppdage gleden og nytten av det å dyrke selv. Dette skal oppnås gjennom å tilby ulike plattformer hvor det skal spres interesse og inspirasjon, samt legge til rette for at flere skal ha mulighet for å dyrke selv. Ved å kartlegge dyrkbar jord i byområder er det også mulig for personer uten egen tilgjengelig plass, å starte med matproduksjon.

For å oppnå dette ønsket ble målet å implementere tre ulike sluttprodukter som en del av den totale plattformen: To applikasjoner, heretter omtalt som "Dyrkingsareal" og "Dyrk med Bybonden", samt en nettside. For å løse problemstillingen rundt kontakt med jordeier var målet å implementere Dyrkingsareal med matrikkelen, hvor det gis data, inkludert eier, for hvert område.

1.2 Kontekst

Urbant landbruk kan kort defineres som dyrking av planter og dyrehold i og rundt byer (RUAF (u.å.)). I dette prosjektet er det i hovedsak den urbane dyrkingen fokuset har ligget på. "I Norge er bare 3 % av landarealet dyrkbart" (Lithun, 2016, s. 39). Dette betyr at om det i framtiden skal være samme matforsyning som det er i dag, må horisonten utvides når det kommer til områder som benyttes til dyrking. Ifølge FN må matproduksjonen dobles og også byene må bidra til økt tilførsel på området. "Digitale verktøy for urbant landbruk i Bergen" er et prosjekt hvor både Bybonden i Bergen og Fylkesmannen i Vestland er involvert. Prosjektet har som mål at flere bergensere selv skal få et direkte forhold til dyrking og at det ikke skal være begrensninger på grunn av manglende plass.

Det er spesielt viktig i bynære områder at tilgjengelige arealer blir benyttet til samfunnsnyttige og bærekraftige formål. I Bergensområdet er det flere tilsynelatende dyrkbare områder, som står urørte, hvor det samtidig ikke er tydelig hvem som disponerer området. Særlig i områder i og rundt byer er ikke beboere på en adresse tilsvarende eier av grunnen på adressen. Det er derfor ikke en like enkel prosess å få tak i grunneiere av slike områder, som det vil være i mer landlige deler av landet.

Ved prosjektstart fantes det ingen løsning for å få direkte kontakt med arealinnehavere. Gjennom Kartverkets nettside "seeiendom.kartverket.no", kan det spores opp rettighetshavere til eiendommer, men det må da foreligge informasjon om veinavn, adresse eller gårds -og bruksnummer (Kartverket (u.å.)). For å få tilgang til informasjon om hver grunn må det logges inn med BankID, hvor det deretter gis tilgang til en PDF-fil med informasjon. Det er flere problemer med denne løsningen, i forhold til den hensikten det skal nytte i denne sammenhengen. Først og fremst er framgangsmåten komplisert og det er få som har kjennskap til den. Et annet poeng er at det ikke skal være et krav om informasjon om arealet. Det skal være mulig å få kontakt med grunneier tilhørende de koordinatene

brukeren befinner seg for øyeblikket, uten at det skal foregå noe mer forundersøking utover dette. Kartverkets løsning inneholder ikke en kartløsning, dermed er informasjonen knyttet til nåværende koordinater ikke tilgjengelig.

I tillegg til det rent praktiske ved å kunne skaffe informasjon om et areal er det i tillegg et aspekt når det kommer til felleskapsånden. Ved å kunne lokalisere andre i nærområdet som driver med dyrking, skapes det et ønske om å ta del i et nærsamfunn og å være en del av noe meningsfylt.

Bergen ansatte sin første Bybonde høsten 2018. Dermed fantes det ikke en allerede etablert plattform for kommunikasjon. Den enkleste måten å starte en informasjonskanal på selv er gjennom sosiale medier som Facebook og Instagram, og det er derfor her dette har foregått. All informasjon om arrangementer i regi av Bybonden har foregått på Facebook, med påmelding via e-post. Betalingstjenesten Tito har blitt benyttet for arrangementer som krevde betaling.

1.3 Begrensninger

Prosjektet finansieres gjennom Bybondeprosjektet. Dette er et samarbeid mellom Hordaland Bondelag og Stiftelsen Lystgården, hvor de økonomiske ressursene er begrensede. Dermed var det avgjørende å prioritere og finne billigst mulige løsninger. Av denne årsaken er det blitt valgt teknologi som følger prinsippet for åpen kildekode, da dette medfølger kostnadsfrie alternativer. Samtidig var det et fokus på å optimalisere spørringer mot database. Dette reduserer prosesseringstid og antall kall, som er en økonomisk fordel.

Et eksempel på besparinger som er foretatt er i forhold til tilgang til 1881 sitt API. API-et skulle benyttes for å kunne ta kontakt med gårdseiere, gjennom applikasjonen Dyrkingsareal. I første omgang var det tenkt at brukerne skulle kunne ta kontakt per telefon direkte fra applikasjonen. For å få denne funksjonaliteten ville det gjennom opplysningstjenester måtte abonneres på API-tilganger, som medfører et månedlig kostnad på 250 NOK, i tillegg til 60 øre per spørring. Priseksemplet er hentet fra en produktoversikt hos 1881, men andre opplysningstjenesters priser var tilsvarende. Den alternative løsningen til dette ble å sende brukere til 1881 sine nettsider med et allerede utført søk på disponenten. På denne måten krever ringe-/meldingsfunkjsonaliteten et ekstra steg, men er uten ekstra omkostninger.

Tidsbegrensningen spilte en avgjørende del under planleggingen og prioriteringen av hva som skulle implementeres. Oppdragsgiver hadde store ambisjoner og ønsker knyttet til sluttresultatet. Det måtte imidlertid settes begrensninger for hva som kunne forventes av sluttproduktet. Til tross for at prosjektgruppen består av fire medlemmer har det fra start av vært klart at deler av ønskelig funksjonalitet var nødvendig å nedprioritere. Særlig

applikasjonen Dyrk med Bybonden har hatt lavere viktighetsgrad. Fylkesmannens Landbruksavdeling var kun involvert i kartapplikasjonen og det var denne delen som var mest nyskapende og etterspurt. Dyrk med Bybonden sitt overordnede mål kunne oppnås gjennom andre kommunikasjonskanaler som Facebook og Instagram, selv om Bybonden hadde et ønske om å en selvstendig og mer personlig applikasjon. For å få benytte tiden best mulig ble det i tillegg besluttet å utforme Bybondens nettside ved hjelp av en nettsidebygger. I tillegg til å være svært tidsbesparende viste dette seg å bli langt mer visuelt estetisk enn det produktet sannsynligvis ville blitt uten hjelp av et slikt verktøy.

En begrensning som ble identifisert før oppstart var kompetanse rundt utvikling av et komplett system. Det har under utdanningsløpet vært flere emner som har inneholdt deler av den nødvendige kunnskapen, men lite undervisning om hvordan alt dette settes sammen. Samtidig var det teknologier som ingen av gruppemedlemmene hadde nevneverdig kjennskap til. Særlig redegjøring og implementasjon av tjenerdelen har vært tidkrevende og det var heller ikke noe utbredt brukernettverk eller -støtte. Kompetansen til oppdragsgiver var også manglende når det kommer til det tekniske aspektet. Det har derfor ikke vært mulig med støtte i forhold til problemstillinger av dette slag og heller ikke tilgang til allerede etablerte rammeverk/teknologier, som ville vært benyttet om oppdragsgiver hadde vært fra IT-bransjen.

1.4 Ressurser

1.4.1 Materielle ressurser

De viktigste materielle ressursene i prosjektet har vært egne datamaskiner for utvikling av nettsiden og mobilapplikasjonene, samt egne mobiler for egen testing underveis, samt brukertesting. Utover dette har oppdragsgiver stilt med de nødvendige økonomiske ressursene for drifting av nettsiden og applikasjonene. I tillegg har oppdragsgiver stilt med arbeidsplass to dager i uken. De resterende dagene har arbeidsplassen vært enten på skolen eller hjemme hos et av prosjektmedlemmene.

1.4.2 Menneskelige ressurser

Av menneskelige ressurser har Remy André Monsen og Frøydis Lindén vært de viktigste. Monsen har vært veileder for prosjektet og har hjulpet med prioriteringer og valg gjennom prosjektet. Lindén holdt kurs i Design Thinking og har hjulpet til med brukertesting.

1.5 Oppbygging av rapporten

Kapittel 1 Introduksjon til rapporten med mål og motivasjon for prosjektet, samt ressurser og begrensninger.

Kapittel 2 Mer om bakgrunnen for prosjektet, kravene til løsningen og initielle forslag til løsning.

Kapittel 3 Mer om mulige måter å løse oppgaven på, mer detaljert om hvilken funksjonalitet som skulle implementeres og valg av verktøy, rammeverk, tjenester og metode for å løse oppgaven.

Kapittel 4 Teknisk om hvordan løsning er bygget opp og satt sammen.

Kapittel 5 Hvordan resultatet har blitt evaluert og resultatet av denne evalueringen.

Kapittel 6 Forskning gjort i prosjektet og resultatet av denne.

Kapittel 7 Beskrivelse av sluttproduktet, og hvilke beslutninger som har vært kritisk for dette, I tillegg til diskusjon rundt hvordan resultatet ville blitt ved andre tilnærminger.

Kapittel 8 Konklusjon rundt resultatet i forhold til målene, og diskusjon rundt hva som kan arbeides videre med i fremtiden.

Kapittel 9 Referanseliste til kilder brukt i rapporten.

Kapittel 10 Risikoliste og Gantt-diagram

2 Prosjektbeskrivelse

2.1 Praktisk bakgrunn

2.1.1 Om oppdragsgiver

Bybonden i Bergen er et nyoppstartet prosjekt og Ida Kleppe tiltrådte stillingen høsten 2018. Det er kun Oslo og Bergen Kommune som har en bybonde i Norge og dette har vært lenge etterlengtet i Bergen. Engasjementet Bybonden er et samarbeid mellom Bondelaget i Hordaland og Stiftelsen Lystgården og skal i første omgang foregå over et løp på fire år. Bybondens oppgave er å spre kunnskap og inspirasjon rundt urbant landbruk og formidle til privatpersoner, skoler og bedrifter, samt allerede etablerte bønder. Kleppe ønsker å kunne nå ut til et bredere publikum og at også folk uten erfaring med dyrking skal bli inspirert til å starte opp.

Stiftelsen Lystgården ble opprettet av Bærekraftige Liv, avdeling Landås. Bærekraftige liv arbeider opp mot bærekraftsmålene definert av FN og verdens fellesplan for bekjemping av fattigdom, ulikheter og klimaendringer. Gjennom dyrking ønsker de å skape et fellesskap hvor mennesker blir knyttet sammen uavhengig av bakgrunn og tilhørighet i samfunnet. Ved å skape en arena hvor det samarbeides om et meningsfylt arbeid, motvirkes ensomhet samtidig som integreringen i samfunnet bedres.

Fylkesmannen i Vestlands landbruksavdeling har som formål å opprettholde og fremme norsk landbruk og matproduksjon. “Det er et politisk mål å opprettholde et levende landbruk i hele landet. Fylkesmannen skal bidra til at den nasjonale landbrukspolitikken blir gjennomført, ved hjelp av informasjon, forvaltning av virkemidler og lokalt tilpassede tiltak.”(Fylkesmannen i Vestland (u.å.)). Det legges også vekt på å veilede om bruksområder, samt øke interesse blant unge, som dermed tydeliggjør rollen i prosjektet.

2.1.2 Initielle kravspesifikasjoner

I utgangspunktet var ideén bak prosjektet å utvikle én applikasjon hvor det kunne aksesseres kartdata. Tanken bak Dyrk med Bybonden oppsto først etter prosjektstart, og ble derfor ikke en del av kravet til sluttproduktet. Det var imidlertid et ønske fra oppdragsgiver at også denne delen ble implementert i løpet av prosjektperioden.

Opphavet til ideen bak prosjektet var å enkelt kunne få tilgang til eiere av areal. Følgende krav, som videre vil bli omtalt som hovedfunksjonaliteter, ble stilt:

- Brukere skal kunne få tilgang til hvem som eier gitte områder ved hjelp av kartdatabaser
- Brukere skal ha en enkel måte å kunne kontakte grunneiere på
- Det skal være mulig for brukere å registrere og foreslå arealer som dyrkbare
- En digital "datingside" for urbant landbruk i Bergen, hvor folk som ønsker å dyrke skal kunne finne ledig areal og arealforvaltere skal få tilgang til personer som ønsker areal

Som en del av den totale plattformen var det behov for en nettside som skulle fungere som Bybondens hjemmeside. Her var ønsket å samle all informasjon rundt Bybondens tjenester på ett og samme sted. Funksjonaliteter som det ble satt krav om var følgende:

- Mulighet for å melde seg på arrangementer
- Betalingsmulighet for arrangementer som krever det
- Kalender med kommende arrangementer
- En generell informasjonsside som Bybonden kan oppdatere fortløpende

Utover disse kravene kom det etterhvert fram at Bybonden hadde et ønske om en kanal hvor det skulle være mulig for brukere å følge dyrkingen av ulike spiselige vekster. I og med at dette oppsto et stykke ut i prosjektet ble det ikke satt strenge krav til denne delen fra start av. Følgende kravspesifikasjoner ble gitt fra Bybonde, heretter omtalt som bifunkjsonaliteter:

- En kanal for Bybonden der brukere skal følge dyrkingen av ulike spiselige planter gjennom en sesong
- Bybonden skal kunne legge ut oppdateringer gjennom sesongen om hva som skal gjøres med den enkelte plante ved ulike anledninger, som f.eks "I dag skal du plante om tomatene, slik gjør du det...", med mulighet for å legge ved bilder
- Brukere skal motta pushvarsler ved nye innlegg fra Bybonden
- Brukere skal kunne kommentere innlegg fra Bybonden, med mulighet for å legge ved bilder

2.1.3 Initielt løsningsforslag

Ved oppstart forelå det ikke et entydig løsningsforslag. I utgangspunktet var oppdragsgiver åpen for forslag til hvordan oppgaven skulle løses, så fremt at kravspesifikasjonene ble overholdt og at løsningen var digital. Som tidligere nevnt var kunnskapen rundt digitale løsninger begrenset hos oppdragsgiver. Dermed var det nødvendig med personer som hadde denne kunnskapen. Først når denne gruppen med personer var etablert kunne det konkretiseres et klart løsningsforslag. Det fantes imidlertid en idé om hvordan målet skulle realiseres.

Delen som var enklest å konkretisere var nettsiden. Kravene til denne instansen var at det skulle utvikles en nettside med de gitte funksjonalitetene. Dermed var det tydelig hva som var løsningsforslaget i forhold til denne delen av oppgaven; sluttresultatet skulle være en nettside som skulle fungere som Bybondens hjemmeside, med de gitte kravspesifikasjonene. Hvordan dette skulle løses rent teknisk fantes det ingen krav om.

Det var imidlertid vanskeligere å identifisere hvilken løsning som var visualisert når det kom til de resterende kravene. Tidlig i oppstartsfasen ble det samtidig brakt fram flere funksjonaliteter, slik at det ble mer uklart hva som var ønsket sluttprodukt og hvilken form dette skulle være i. Innledningsvis var det ønskelig at resultatet skulle utformes som en mobilapplikasjon. Dette var på bakgrunn av at brukere skulle finne eiere av området de befant seg på, og at det dermed var naturlig at dette kunne gjøres ved hjelp av mobile enheter med tilgang til det mobile nettverket. Grunnen til at det var ønskelig med en applikasjon, og ikke en nettside, var begrunnet med at applikasjoner ble sett på som mer brukervennlige og at det i forhold til markedet er mest ettertraktet.

De senere kravspesifikasjonene i forhold til Dyrk med Bybonden hadde krav om pushvarsler i gitte tilfeller. Dermed var det følgelig nødvendig for oppdragsgiver at denne delen skulle utformes som en mobilapplikasjon, om kravene skulle oppfylles.

3 Prosjektdesign

3.1 Mulige tilnærminger

I følgende delkapittel vil ikke alternative tilnærminger for nettsiden diskuteres. Her var kravene tydelige og det var lite å diskutere i forhold til andre løsninger. Det ble derfor fra oppstart ikke vurdert alternative tilnærminger vedrørende denne delen. Ulike varianter ble imidlertid vurdert hva kravene angår utover dette. I alternativene som beskrives under er det derfor kun omtalt de to instansene som ikke omfatter nettsiden.

3.1.1 Én mobilapplikasjon og en webapplikasjon

I første omgang ble det diskutert om den opprinnelige ideén nødvendigvis måtte utvikles som en mobilapplikasjon. Samtlige kravspesifikasjoner kunne implementeres om det hadde blitt valgt å utvikle løsningen i form av en webapplikasjon. På denne måten kunne Bybondens hjemmeside bestått av en del som innebar en kartløsning, samt en markedsplass for dyrkingsareal, i tillegg til de opprinnelige nettside-kravene . Det ble argumentert for at denne løsningen ville gjøre det enklere for brukere å benytte seg av, da det for mange er mer behagelig og ønskelig å utføre ting ved hjelp av en PC. Da kunne det enkelt blitt benyttet mobile enheter når brukere befant seg på aktuelle områder, samtidig som det kunne tas i bruk PC når dette var ønskelig. Tilnærmingen hadde gjort alt innhold tilgjengelig for alle med nettilgang, og det ville ikke vært et krav om nedlastinger og installasjoner. Da hadde det samtidig vært mulig for brukere å finne innholdet ved hjelp av en søkemotor.

Denne løsningen ville innebåret at koordinater hadde blitt hentet ved hjelp av en nettleser. Løsningen ville også betydd at det ikke måtte tas hensyn til operativsystemer. Webapplikasjonen hadde på den annen side vært avhengig av vektlegging på et mobilvennlig og responsivt design.

Selv om applikasjonen Dyrk med Bybonden var av lav prioritet ble det planlagt hvordan denne delen skulle løses om tid og andre ressurser skulle strekke til. Kravene som ble stilt til segmentet innebar at det nødvendigvis måtte utvikles en mobilapplikasjon, både med tanke på pushvarsler og direkte kommunikasjon med brukere. Denne tilnærmingen innebar derfor at løsningen skulle inneholde en mobilapplikasjon i tillegg til den nettbaserte løsningen. Mobilapplikasjonen skulle dermed kun inneholde funksjonaliteter som var spesifisert omhandlende Dyrk med Bybonden.

3.1.2 Én mobilapplikasjon

For å forenkle løsningen var en mulig tilnærming å utvikle kun én mobilapplikasjon. På denne måten ville kun hovedkravene utvikles i første omgang, før eventuelle tilleggskrav ville blitt implementert. Dette ville gjort det enklere å arbeide iterativt i forhold til tilnærmingen med en webapplikasjon. Samtidig ville det være tidsbesparende da det bare ville være nødvendig å skaffe kunnskap om én teknologi og et rammeverk.

Løsningen ville hatt klare skiller mellom funksjonalitetene, men den ville bestått av en felles instans. På denne måten ville det være mer tilrettelagt for direkte kontakt med brukere, samt benytting av plattformavhengig kode. Dette være seg koordinater, kamera, bildebibliotek og pushvarsler. Følgende av dette ville vært økt ytelse, i tillegg til at oppdragsgivers ønske om brukervennlighet og oppfatning av etterspørsel kunne oppfylles.

3.1.3 To separate mobilapplikasjoner

Det ble tidlig identifisert et skille innad i målgruppen. Når det kom til hovedfunksjonalitetene ble brukergruppen ansett å være personer som allerede var etablerte i landbruksmiljøet. Om løsningen skulle innebære én applikasjon ville dermed innholdet i deler av applikasjonen ligge utenfor flere brukeres interesseområde. Det samme ville gjelde for den andre siden av målgruppen; Dyrk med Bybonden ønsker å appellere til personer som er ukjente på dyrkearenaen og som trenger detaljert veiledning og inspirasjon.

Dermed ble det foreslått å separere de to løsningene. På denne måten ville det være tydelig hvem applikasjonene var tiltenkt og avgrense innholdet til hver av de to.

3.1.4 Diskusjon av alternative tilnærminger

For å avgjøre hvilken tilnærming som ville være den beste for å løse oppgaven ble det tatt hensyn til prosjekteier og potensielle brukere. Løsningen med Dyrkingsareal som webapplikasjon ble tidlig utelukket, da dette stred i mot oppdragsgivers ønske. Hoveddiskusjonen lå dermed rundt om hvorvidt det skulle være én eller to applikasjoner. Her kom det tydelig fram hvilke grunnlag de ulike samarbeidspartnerne hadde for å ta initiativ til prosjektet. For Fylkesmannen i Vestland var prosjektinitiativet satt i gang for å kunne tilby en løsning for innbyggere av Bergen når det kommer til å identifisere og kartlegge dyrkbare arealer. Fylkesmannens avdeling hadde ikke samme mål som bybondeinitiativet; avdelingen har i første omgang som formål å opprettholde allerede etablerte innenfor landbruket. Det var dermed et poeng å kunne tilrettelegge for personer som er en del av landbruksmiljøet og på den måten fremme matproduksjonen i fylket.

Bybonden har derimot i tillegg til dette som mål å øke interesse og kunnskap rundt dyrking. Ønsket var å utvikle en personlig applikasjon, som skulle være lokalbasert i Bergen. Det ble sett på som en måte å spre kunnskap om Bybonden i Bergens arbeid, samt skape en dialog med dyrkere i begynnerfasen.

På grunn av dette skillet ble det besluttet å utvikle to forskjellige mobilapplikasjoner, i tillegg til hjemmesiden for Bybonden. Tonen i de to applikasjonene skulle være ulike, samt variere hva målgruppe angår. Dyrkingsareal baserer seg på et brukerdefinert innhold, hvor brukerne er særlig framtredende, og er i den forstand allemannseie. Her er det avgjørende at brukere tar del i utformingen, skal målet bak applikasjonen bli realisert. I Dyrk med Bybonden skal Bybonden i Bergen ha en tydelig rolle og alt innhold bygges på aktivitet fra henne. Her har altså ikke brukerne en like avgjørende rolle og applikasjonen kan bli vellykket selv om brukerne velger å ikke samhandle i samme grad.

3.2 Spesifikasjoner

3.2.1 Dyrkingsareal

Dyrkingsareal var den delen av oppgaven som var mest nyskapende og det forelå etterspørsel etter blant brukerne. Dette var derfor av høyeste prioritet, samtidig som den inneholdt de mest kritiske delene av prosjektet. Her var det nødvendig med en del forarbeid, samt kartlegging av hvordan implementeringen av funksjonalitetene skulle løses. I samråd med oppdragsgiver ble det konkludert at den viktigste funksjonen var delen som skulle implementeres ved hjelp av kartdatabaser og visualiseres i en kartløsning. Her ble følgende spesifikasjoner gitt:

- Matrikkelen skulle benyttes for å aksessere data om eiere av areal
- Et kart-API skulle tas i bruk for å visualisere løsningen, med tilgang til posisjoner
- Brukere skulle på en enkel måte kunne kontakte eiere, foretrukket via telefonnummer
- Brukere skulle ha mulighet for å legge ut informasjon om områder
- Applikasjonen skulle inneholde en oversikt over områder som var tilgjengelige for dyrking

Delen med “datingsiden” for landbruksmiljøet, heretter kalt “markedsplass”, skulle også være av høy prioritet, men dette var ikke den mest kritiske faktoren. Dermed skulle spesifikasjonene gitt over prioriteres før følgende krav angående markedsplassen:

- Brukere skal ha mulighet for å opprette annonser knyttet til en av følgende kategorier:
    - Ledig areal
    - Areal ønskes
    - Dyrkelag søker medlem/-mer
    - Enkeltperson ønsker dyrkelag
- Det skal være enkel tilgang for kontakt mellom annonsør og bruker

3.2.2 Hjemmeside for Bybonden i Bergen

Det ble besluttet at spesifikasjonene gitt under det initielle forslaget skulle stå som gjeldende. Imidlertid ble det et krav om at løsningen skulle utvikles ved hjelp av en nettsidebygger. Dette med hensyn til tid, samt de begrensede kunnskapene hva design angår. Følgende spesifikasjoner ble satt til nettsiden:

- Utvikles ved hjelp av en nettsidebygger
- Implementere Facebook-kalender fra Bybondens Facebookside
- Betalingsløsning ved hjelp av Tito
- Lett tilgjengelig informasjon om foretaket
- Enkel måte for Bybonden å oppdatere og redigere

3.2.3 Dyrk med Bybonden

Da denne delen var av lavere prioritet og det siste som skulle ferdigstilles var det nødvendig at det i utgangspunktet var få funksjonaliteter, og heller mulighet for skalerbarhet. Bybonden hadde mange idéer om hvilke funksjoner applikasjonen kunne inneholde, men dette ble begrenset til kun de grunnleggende tankene bak. Det ble derfor i samarbeid med oppdragsgiver bestemt å prioritere en mobilapplikasjon hvor Bybonden kunne legge ut oppdateringer og instruksjoner for hvert dyrkeprosjekt og at brukere deretter skulle motta pushvarsler.

3.3 Valg av verktøy, rammeverk og tjenester

3.3.1 Rammeverk for klientsiden

Da det skulle utvikles mobilapplikasjoner var det viktig at de skulle kunne brukes på flere typer enheter, også enheter med forskjellige operativsystem. Operativsystemene som ble sett på som viktige var Android og iOS, fordi disse er de mest brukte operativsystemene (Gartner (2017)).

Operating System	4Q16 Units	4Q16 Market Share (%)	4Q15 Units	4Q15 Market Share (%)
Android	352,669.9	81.7	325,394.4	80.7
iOS	77,038.9	17.9	71,525.9	17.7
Windows	1,092.2	0.3	4,395.0	1.1
BlackBerry	207.9	0.0	906.9	0.2
Other OS	530.4	0.1	887.3	0.2
Total	431,539.3	100.0	403,109.4	100.0

Figur 3.1: Utsnitt av skjermdump fra gartner.com. Vlser verdensomspennende salg av smarttelefoner fordelt på de ulike operativsystemene.

Alternativer som ble sett på da rammeverk skulle velges, var Cordova, React Native og Xamarin. Cordova er et rammeverk for kryssplattform-mobilutvikling hvor koden skrives i HTML, CSS og JavaScript. Dette betyr Cordova i praksis fungerer som en nettleser, som leser et bestemt sett med filer. Ytelsen til maskinvaren blir dermed utnyttet for fullt. Det var ingen god støtte for JavaScript-rammeverk for utvikling av brukergrensesnitt som React eller Angular i Cordova, som gjorde at Cordova ikke ble valgt. Xamarin er et rammeverk lagd for kryssplattform-mobilutvikling, hvor det skrives i .NET. Xamarin ble valgt bort fordi Microsoft, som er utvikler bak Xamarin, antar at bare rundt 75% av koden kan brukes på tvers av mobilplattformene og fordi det var ønskelig å skrive i JavaScript, på grunn av Google Maps som har et JavaScript API. Valget falt på React Native, da dette rammeverket er bygget på React og det kodes i JavaScript.

React Native

React Native er et rammeverk for utvikling av plattformavhengige applikasjoner ved bruk av JavaScript-biblioteket React (React Native (u.å.)). Det baseres kun på JavaScript, men lar brukere benytte plattformavhengig kode for optimalisering. Rammeverket bruker de samme fundamentale komponentene som brukes i plattformavhengig kode.

JSX

JSX er en XML-liknende utvidelse av syntaksen til ECMAScript og brukes mye i React Native. (JSX, u.å.). Syntaksen ligner på HTML i form av at det setter opp hvordan brukergrensesnittet skal se ut, samt at det benytter elementer og tagger for å uttrykke hva som skal vises.

```
<Appbar.Header style={styles.header}>
   <Appbar.Content
    title={this.state.objectName}
    titleStyle = {styles.content}
  />
  <AboutDialog desc = {this.state.description}/>
</Appbar.Header>
```

Figur 3.2: Utsnitt av JSX-kode fra Dyrk med Bybonden

3.3.2 Amazon Web Services (AWS)

For utvikling av løsningen var det behov for skytjenester. Alternativer som ble sett på var Microsoft Azure og Amazon Web Services. Valget falt på Amazon Web Services sine vertløse tjenester, fordi dette dekker alle behovene for prosjektet. Det betales bare for brukt prosessortid, samt at det er ikke behov for å tenke på tjenerhåndtering, ettersom AWS håndterer dette. På grunn av at kostnader bare oppstår ved brukt prosessortid, betales det kun for tiden funksjonene kjøres. Tjenestene det var behov for var database, autentisering og autorisering, pushvarsling og REST-tjenester. AWS leverer alle disse tjenestene gjennom AWS DynamoDB, AWS Cognito, AWS SNS, AWS API-Gateway og AWS Lambda. AWS er også lagt til rette for React Native, da det finnes egne AWS-bibliotek for bruk i React Native. Tjenestene som ble tatt i bruk forklares nærmere i de neste avsnittene.

DynamoDB

For lagring av data har DynamoDB blitt tatt i bruk. DynamoDB er en NoSQL-database, og en nøkkel-verdi-basert databasetjeneste fra AWS. I motsetning til en relasjonsdatabase er det ikke faste attributter for hver tabell. I DynamoDB-tabeller er det kun nøkkelattributten som er påkrevd for hvert av objektene i tabellen. Nøkkelattributten består enten bare av en oppdelingsnøkkel, eller både en oppdeling- og sorteringsnøkkel.

NoSQL-databaser skalerer godt horisontalt, som vil si at de skalerer med flere maskiner, i motsetning til vertikal skalering som betyr skalering med kraftigere maskiner. Dette gjør at NoSQL-databaser er ideelle for skytjenester. Ved bruk av DynamoDB betales det bare for benyttet overføring og henting av data. Tjenesten har også mulighet for automatisk skalering. Disse faktorene fører til at det kreves mindre tid og fokus på planlegging av ressursbruk av databasen. På en egen dedikert tjener vil ressursene sannsynligvis ofte ikke bli fullstendig utnyttet. Denne ineffektiviteten vil føre til at en dedikert tjener sannsynligvis vil være dyrere enn en slik skytjeneste, når det er relativt lav utnyttelse av maskinvaren.

API Gateway

API Gateway fungerer som et REST-endepunkt. Tjenesten tar imot REST-forespørsler og sender dem videre til andre tjenester. API Gateway kan utføre enkle operasjoner, blant annet validering av inndata, autentisering og parsing av URL-variabler til data i forespørselens kropp.

Lambda

Lambda er en av tjenestene til AWS. Kode kan lastes opp som funksjoner i AWS Lambda. API Gateway kaller på funksjonene i Lambda, som igjen kan kalle på andre tjenester AWS tilbyr. I denne løsningen har dette blitt benyttet ved at koden kaller på DynamoDB-tjenester. Logikken i tjenerdelen ligger som kode i Lambda.

Cognito

Amazon Cognito er en nettjeneste som tilbyr brukerautentisering, -autorisering og -håndtering for mobil- og webapplikasjoner. Cognito består av to hovedkomponenter: Cognito User Pools og Cognito Identity Pools. Cognito User Pools er en brukerkatalog som håndterer registrering, inn- og utlogging av brukere. Det gir også utviklerne et innebygd brukergrensesnitt som kan tilpasses applikasjonen. Cognito Identity Pools brukes til å autorisere brukerne og gi de tilgang til andre AWS tjenester. (AWS Documentation (u.å.)).

SNS

Amazon Simple Notification Service (SNS) er en publisering/abonnering-meldingstjeneste. SNS brukes sammen med Googles Firebase Cloud Messaging(FCM) for å sende ut varsler til brukere. For å kunne publisere og abonnere på SNS må det opprettes et emne. Klienten kommuniserer med FCM, som igjen mottar og sender meldinger til SNS. SNS har en oversikt over hvilke klienter som har mottatt en publisert melding, slik at klientene ikke mottar samme melding flere ganger. Det er FCM som tar seg av selve sendingen og mottaket av varsler på klientsiden.

3.3.3 Verktøy

Xcode

Ved utvikling av iOS-applikasjoner er det nødvendig å disponere en datamaskin med operativsystemet macOS, samt tilhørende IDE, Xcode. Selv om applikasjonene utvikles i React Native, er det behov for IDE-et for å forsørge emulator, samt for optimalisering av plattformavhengig kode. Om React Native-applikasjonen skal kjøres på en iPhone under testingen må dette også gjøres ved hjelp av Xcode.

Android Emulator

Android Emulator er en del av Android Studio, som brukes til å emulere enheter med Android operativsystem. Dette brukes for å kunne teste ut applikasjonene på flere enheter enn de som er tilgjengelig som reelle enheter. Android Emulator kobles til med Android Debugging Bridge (ADB), som er det samme grensesnittet som brukes mot ekte enheter. Det finnes flere forhåndsdefinerte konfigurasjoner av emulerte enheter i Android Studio, som tilsvarer ekte enheter. Viktige deler av konfigurasjonene er skjermstørrelse, skjermoppløsning og API-nivå. For å kunne være kompatibel med eldre enheter er det nyttig å også kunne teste på et lavere API-nivå. Emulatoren kan simulere sensorer som finnes på den emulerte enheten og virke på disse.

Visual Studio Code

Visual Studio Code er et gratis koderedigeringsverktøy laget av Microsoft. En klient for Git er innebygd i Visual Studio Code, som gjør det enklere å jobbe med Git. Visual Studio Code har verktøyet Intellisense, som hjelper til med å skrive koden. Intellisense har støtte for flere programmeringsspråk, og detekterer hvilket språk det blir skrevet automatisk. Intellisense kommer med forslag til autofullføring og korreksjon av kode. Visual Studio Code ble valgt på grunn av at Intellisense har støtte for JSX og på grunn av den innebygde Git-klienten.

Git

I løpet av prosjektet har Git blitt brukt til å holde kontroll og oversikt over koden, samt gjort det enkelt å samarbeide på felles kode. Bruk av Git har muliggjort at det til enhver tid jobbes med oppdatert kode og har gjort det enkelt å holde oversikt over hvilke oppgaver som skal gjøres og hvem som arbeider med de. For å holde orden på arbeidsoppgavene har de blitt fordelt på grener ut fra en hovedgren. På denne måten har det til enhver tid vært en fungerende hovedgren. Denne har etterhvert har fått flere funksjonalitet etter at arbeidsoppgaver har blitt gjennomført og blitt slått sammen med hovedgrenen. For å gjøre prosessen oversiktlig har programvaren Github Desktop blitt tatt i bruk. Github Desktop gir en enkel og god oversikt over de ulike kodelagrene og deres respektive grener, og gjør det enkelt å slå sammen grener når arbeidsoppgavene er gjennomført.

Postman

I dette prosjektet er Postman et verktøy som brukes for testing av HTTP-tjenester. Postman kan brukes til å opprette, lagre og sende HTTP-forespørsler til HTTP-tjenere. Dette brukes til å sende GET- og POST-forespørsler til tjenerdelen, for å se om tjeneren returnerer forventet svar. Forespørslene opprettes med parametere, topptekst og kropp. På denne måten kan det sjekkes om tjeneren gir forventet svar på en gitt forespørsel. Postman kan også brukes til å etterligne en tjener med REST-API som det kan sendes forespørsler mot. Postman kan

settes opp til å monitorere REST-APIer ved å sette opp planlagte automatiserte tester og sjekke ytelsen til APIet.

Weebly

Weebly er en nettsidebygger som gjør det enkelt å lage nettsider, uten å skrive noe særlig kode. Denne løsningen koster litt mer enn å bygge nettsiden fra bunnen av selv, men ble foretrukket av to grunner. Den første er at at nettsiden skal kunne driftes videre og endres på etter prosjektslutt. Dette legges til rette for med en slik løsning, og det gjør det enkelt å vedlikeholde nettsiden. Den andre viktige grunnen er at på grunn av omfanget av oppgaven, var det ønskelig å bruke minst mulig tid på nettsiden, slik at det ble mer tid til å utvikle mobilapplikasjonene.

Google Drive

I løpet av prosjektet har Google Drive blitt brukt som et felles lagringssted for dokumenter som blant annet møtereferater, innleveringer og diverse UML-diagram. Her har Google Dokumenter blitt tatt i bruk, som har gjort at flere kan arbeide på samme dokument samtidig.

Figma

Tidlig i prosjektet ble det utviklet prototyper av mobilapplikasjonene. Dette ble gjort for å gjøre det tydelig hva som skulle utvikles og vite at det som ble utviklet var det oppdragsgiver ønsket seg. Dette gjorde det også mulig å komme tidlig i gang med brukertesting, slik at det ble gjort rede for om dette var noe brukerne hadde behov for.

For å utvikle disse prototypene ble prototypeverktøyet Figma brukt. Verktøyet gjør det enkelt å lage en skisse for hvordan applikasjonen skal se ut og hvordan det skal navigeres i den. Det er mulig å teste prototypen som en mobilapplikasjon, noe som har vært nyttig for tidlig brukertesting. Dette gjorde det enklere for potensielle brukere å se for seg sluttproduktet og dermed kunne gi nyttige tilbakemeldinger på hva de likte og hva de ikke likte så godt.

I tillegg til fordelen ved testing, ble prototypen et nyttig verktøy for skissering av brukergrensesnittet og navigasjonen i applikasjonene. Dette gjorde prosessen med å komme i gang med utviklingen enklere da brukstilfellene var visualisert og redegjort for, og mye av grunnlaget lagt for utviklingen av applikasjonene.

Cacoo

Cacoo er et verktøy for å lage ulike diagrammer. Cacoo har et stort bibliotek med ikoner, inkludert AWS-ikoner. Diagrammene lages med en dra-og-slipp-funksjonalitet, som gjør Cacoo veldig enkelt å bruke. I dette prosjektet har Cacoo bare blitt brukt for å lage UML-diagram.

AWS Toolkit

Grunnet god kjennskap til språket, ble Java valgt som språk på tjenerdelen. For utvikling ble Eclipse IDE benyttet sammen med AWS Toolkit. AWS Toolkit er et tillegg til Eclipse, som gjør det enklere for utviklere å utvikle, feilsøke, og rulle ut Java-applikasjoner som bruker Amazon Web Services (AWS (u.å.)).

3.4 Prosjektutviklingsmetode

3.4.1 Utviklingsmetode

Under oppstartsfasen ble det bestemt at en godt utarbeidet prototype ville bli viktig for å tidlig kunne fastslå kravene til applikasjonene. Samtidig ville dette bli avgjørende for å kunne forenes rundt funksjonalitet både innad i gruppen, men også mellom prosjektgruppen og oppdragsgiver. Prototypen ble benyttet ved flere anledninger for å demonstrere til potensielle brukere og oppdragsgiver, for deretter å gjøre endringer på design og funksjonalitet. Dette ble en viktig del av prosessen, når det kommer til testing og å bestemme kravspesifikasjoner.

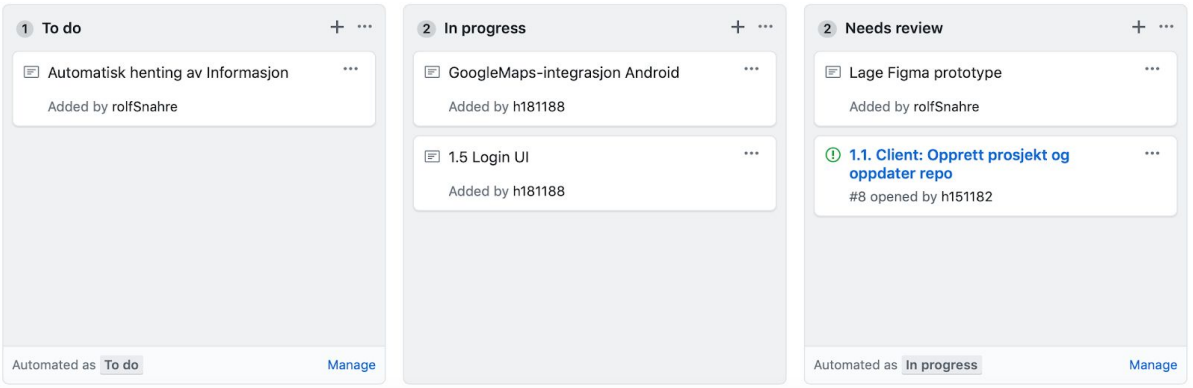
Når det kommer til selve metodikken under utviklingen ble den smidige metoden benyttet. Dette var naturlig da oppdragsgiver ville være tett involvert under hele løpet. Samtidig var det nødvendig for å prioritere funksjonaliteter etter ønske, for å håndtere tidsbegrensningen og andre risiki.

> *Smidige metoder reduserer risiko og øker kvaliteten ved å muliggjøre læring og endring underveis i utviklingen. Når partene samarbeider tettere underveis kan prioriteringer endre seg og ny kunnskap tas med i beregningen. Som regel bedres den gjensidige tilliten i denne prosessen* (Høiseth, 2016).

Det ble uttrykt for Bybonden og representant fra Fylkesmannen i Vestland at det var nødvendig med en klar prioritering over funksjonaliteter, da det på forhånd var vanskelig å forutse hva som var realistisk å kunne ferdigstille innen tidsfristen. Etter hvert som det ble kartlagt hvilke teknologier og tilganger som kunne benyttes ble det tydeligere hvilke funksjonaliteter som det faktisk var kapasitet til og mulighet for å realisere. Listen med oppgaver etter prioritet har derfor stadig blitt oppdatert og redigert.

Et annet aspekt er at selve prosjektet på forhånd ikke hadde tydelige kravspesifikasjoner. Oppdragsgiver hadde en overordnet idé om hva som var målet med prosjektet og hva det var ønskelig å finne løsninger på. Allikevel har det ved hjelp av den smidige metoden blitt utarbeidet hva som faktisk skulle til for å realisere det.

Githubs Kanbantavler ble tatt i bruk under hvert delprosjekt. Dette gjorde det mulig å til enhver tid ha oversikt over påbegynte oppgaver, hva som måtte utføres og hva som var ferdig. Dermed kunne det stadig holdes kontroll på hva de andre gruppemedlemmene foretok seg og hvorvidt tidsskjemaet ble overholdt. Samtidig ble det avholdt statusmøter to ganger daglig; et ved start på arbeidsdagen og et ved slutt. Her ble det henholdsvis gjennomgått hva som var hver enkelt gruppemedlems plan for dagen i forhold til arbeidsoppgaver, hva som hadde blitt utført og hvilke utfordringer som hadde oppstått. På denne måten kunne det diskuteres løsninger og hvilken tilnærming som burde benyttes ved hver oppgave.



Figur 3.3: Skjermdump fra Kanban tavle

3.4.2 Prosjektplan

Under punkt 10.3 ligger et Gantt-diagram som skisserer planen for prosjektet. I første omgang ble nettsiden og kartapplikasjonen prioritert, ettersom det var usikkerhet rundt hvor lang tid det ville ta å utvikle de ulike produktene. Det var ønskelig å heller kunne fullføre to av de tre produktene, enn å ende opp med tre uferdige produkter.

3.4.3 Risikohåndtering

For å håndtere risiki var det viktig at disse ble identifisert så tidlig som mulig, slik at det kunne planlegges tiltak for hver enkelt av de, både for å forhindre at de inntreffer og hvordan de eventuelt skal håndteres om de skulle inntreffe. Med tanke på tidsbegrensningen ville uforutsette problemer kunne få store konsekvenser for sluttproduktet. I punkt 10.2 blir det gjennomgått identifiserte risiki og håndtering for hver enkelt.

3.5 Evalueringsplan

Evalueringsplanen var gjennom den valgte utviklingsmetoden å opprettholde kontakten med oppdragsgiver gjennom hele prosjektperioden. På denne måten ville funksjonalitet og brukergrensesnitt bli evaluert etterhvert som de ble utviklet. Gjennom metoden Design Thinking skulle det være stadig evaluering fra andre interessenter. Det var planlagt å definere en testgruppe som gjennom hele utviklingen skulle få komme med tilbakemeldinger, slik at produktet ville være mest mulig tilpasset brukernes ønsker. Utover dette skulle det utføres brukertester blant målgruppen for de ulike instansene av prosjektet.

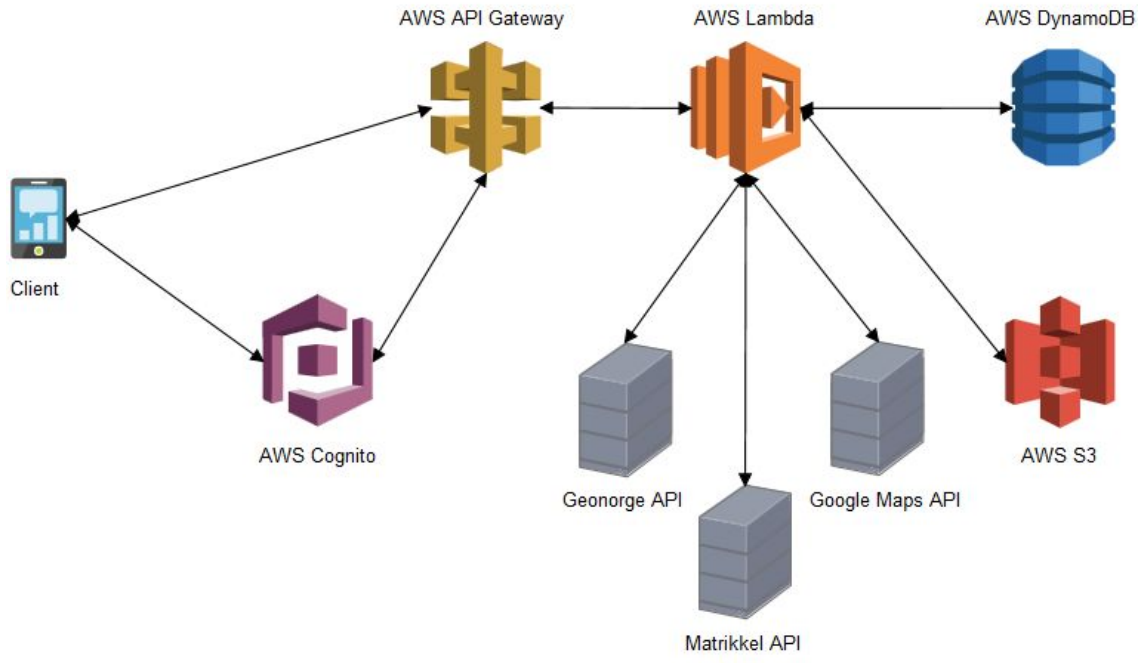
4 Arkitektur og løsning

4.1 Tjener

I både Dyrkingsareal og Dyrk med Bybonden var det nødvendig å håndtere informasjon, inkludert informasjonen som beholdes mellom sesjoner og som deles mellom brukere. Løsningen for dette er en tjenerdel med database for hver av applikasjonene. Klientsiden kan kommunisere med tjenersiden via HTTP-kall. Håndtering av informasjon består blant annet av innsetting, henting og oppdatering av informasjon.

4.1.1 Arkitektur

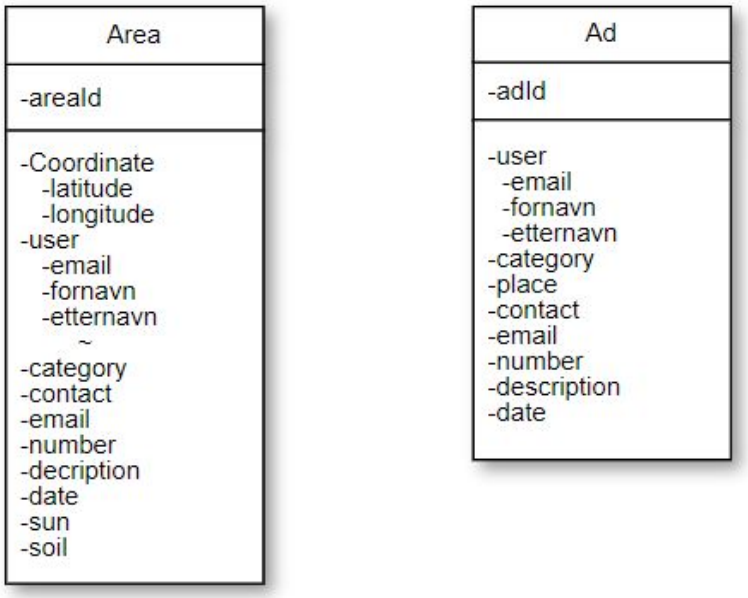
Arkitekturen på tjenersiden består av Amazon Web Services sine tjenerløse løsninger, som betyr at det ikke er noe behov for tjeneradministrasjon. For autentisering kommuniserer klienten med AWS Cognito, som oppretter en sesjon. For å benytte funksjonaliteten på tjenerdelen, sender klienten forespørsler til AWS API Gateway. Dersom autorisasjon kreves, sjekker AWS API Gateway med AWS Cognito om klienten er autorisert. Hvis klienten har autorisasjon til funksjonen den har kalt på, sendes forespørselen til AWS Lambda hvor funksjonene ligger. Funksjoner på AWS Lambda kommuniserer med databasen som består av AWS DynamoDB. For lagring og henting av større elementer brukes AWS S3. AWS S3 er en lagertjeneste og brukes til å lagre bilder til innlegg og kommentarer. AWS Lambda har også funksjoner som gjør forespørsler til Geonorge sitt API for adresser, matrikkelens API og mot Google Maps sitt API. AWS DynamoDB har begrensinger for størrelsen til elementer.



Figur 4.1 :Arkitekturdiagram

4.1.2 Løsning

Databasen er en viktig del av tjenerdelen til applikasjonene. I Dyrkingsareal lagres informasjon om områder og annonser.



Figur 4.2: ER-diagram for Dyrkingsareal

Databasen består av samlingene "area" og "ad", som representerer område og annonse. AWS anbefaler å ha få tabeller i en NoSQL-database som DynamoDB. I dette tilfelle innebærer det å lagre informasjonen om koordinater og brukere i område- og annonseentitetene, istedenfor å ha egne entiteter for disse (AWS, u.å.). Områdeentiteten har derfor ikke bare en fremmednøkkel til en brukers ID, men heller alle attributtene til brukeren. Dette vil føre til duplikasjon av data, men vil også føre til raskere søk. Dette kan skape problemer dersom brukerinformasjonen blir endret ofte, enten ved oppdatering eller sletting. Det antas derimot at dette ikke vil forekomme særlig hyppig i dette systemet, og vil derfor trolig ikke skape mye ekstraarbeid for tjenerdelen.

Logikken på tjenersiden er implementert ved bruk av Java-kode, som funksjoner i AWS Lambda. For hver forespørsel er det en klasse som implementerer "RequestHandler"-grensesnittet. Dette medfører at klassene må overstyre en metode, "handleRequest()", som blir kalt ved HTTP-forespørsler.

For hver entitet finnes det forespørsler for å legge til et nytt element, oppdatere et eksisterende element, hente ut et gitt element, hente ut alle elementer og slette et gitt element. For hver av entitetene er det også en klasse som spesifiserer attributtnavnene i databasen. Denne blir brukt til å hente ut verdier fra inndataene og hjelper til med å få verdier satt til riktige attributter i databasen.

```
public class getMyAds implements RequestHandler<String, Object> {
      @Override
      public Object handleRequest(String input, Context context) {
            DatabaseOperations dops = new DatabaseOperations();
            return dops.getObjectWithKeyValue("Ads", "email", input);
      }
}
```

Det er også en klasse for databaseoperasjoner som gjennomfører fellesoperasjoner for de ulike entitetene. Denne klassen har metoder for de ulike operasjonene som skal utføres, som for eksempel henting og sletting av data. Metodene tar imot navnet på entiteten som operasjonen skal utføres, i tillegg til relevant data. For henting og sletting av data trengs bare en ID for elementet, mens for innsetting av element sendes en liste med parameter-verdi-par for elementet. De funksjonene som har bruk for det tar også imot et parameterobjekt, som spesifiserer navnene til attributtene i entiteten.

Hver operasjon mot databasen består hovedsakelig av tre steg:

1. Hente tabellen som skal brukes.
2. Det lages et spesifikasjonobjekt av typen til operasjonen som skal utføres. Dette er et objekt som styrer hvordan en operasjon utføres. For eksempel vil en skanneoperasjon opprette et spesifikasjonobjekt for skanning med et eventuelt predikat for filtrering. En skanneoperasjon går gjennom alle elementer i en tabell og returnerer enten alle element, eller bare de som oppfyller eventuelle predikat. Eventuelle predikat for skanneoperasjoner blir satt i spesikasjonobjektene.
3. Metoden for operasjonen blir gjennomført med spesifikasjonobjektet som parameter.

I eksempelet under skal alle elementer av en entitet hentes. Da hentes først tabellen, før det opprettes et spesifikasjonobjekt “spec” med spesifikasjoner for en skanneoperasjon. Siden alle elementene returneres er det ingen predikat for filtrering av element. Til slutt blir skannemetoden kalt på tabellobjektet med spesifikasjonobjektet som parameter.

```
public Object getAll(Map<String, Object> input, String tableName,
Params paramObj){
    Table table = DUtil.getTable(tableName);
    ScanSpec spec = new ScanSpec();
    ItemCollection<ScanOutcome> result = table.scan(spec);
}
```

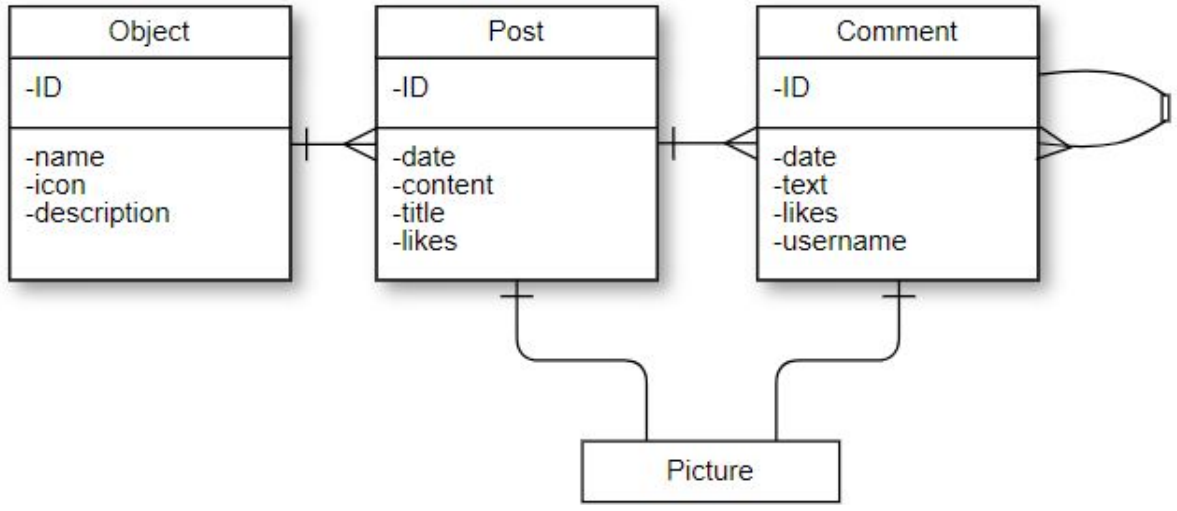
For henting og sletting av et objekt med en gitt ID, lages også et spesifikasjonobjekt. Med metoden withPrimaryKey() settes primærnøkkelen til ID-en. Det kan deretter kalle på hente- eller slettemodtoden til tabellen med spesifikasjonobjektet som parameter.

```
GetItemSpec spec =
        new GetItemSpec().withPrimaryKey(paramObj.getPk(), id);
Item outcome = table.getItem(spec);
```

Her blir også parameterobjektet tatt i bruk. I dette eksempelet brukes det for å hente navnet på primærnøkkelattributten til entiteten. Dette vil enten være “adId” eller “areaId”, avhengig av hvilken entitet som jobbes med. Med metoden “withPrimaryKey()” fortelles spesifikasjonobjektet at henteoperasjonen skal finne et element som har primærnøkkel med navnet parameterobjektet returnerer og verdien id.

Ved innsetting av objekt blir parameterobjektet brukt til å hente verdier fra inndataene og det lages et element med riktige parameternavn som kan settes inn i tabellen. Når en annonse skal legges til i databasen henter tjeneren dagens dato, og legger det til som en attribut i annonseobjektet. Hvis innsendt data mangler parametere sendes en feilmelding.

Applikasjonen Dyrk med Bybonden har en annerledes datastruktur. Entitetene som blir lagret her er: Objekter, innlegg og kommentarer. Denne modellen har en hierarkisk struktur som består av en rekke med en-til-mange-forhold.



Figur 4.3: ER diagram for Dyrk med Bybonden

Som figuren over viser kan en kommentar ha flere kommentarer. Dette er for at brukerne skal kunne svare på hverandres kommentarer i applikasjonen. Ut i fra databasestrukturen kan kommentarer ha nye kommentarer i det uendelige, men dette er begrenset på

klientsiden til at et innlegg kan ha kommentarer som igjen kan ha svar, men ikke noe mer enn det.

Databasen for denne applikasjonen består av bare en tabell. I denne databasen brukes en global sekundærindeks (GSI). Dette betyr at det er to sett med nøkler: En hovednøkkel og en GSI-nøkkel. GSI-nøkkelen består igjen av to nøkler, en oppdelingnøkkel som viser til foreldreelementer, og en sorteringsnøkkel som forteller hvilken attributt de skal sorteres etter.

	Id PK	parentId (GSI-PK)	Sort (GSI-SK)	Andre attributter
Objekt	objectId	"Object"	navn	~
Innlegg	postId	foreldreID	date	~
Kommentar	commentId	foreldreID	date	~

Det kan hentes ut elementer til et foreldreobjekt ved å bruke sekundærindeksen. For å finne alle kommentarer til et innlegg med ID-en "5", blir nøkkelen spesifisert til "foreldreID = 5". Resultatet blir sortert etter soteringsattributten, som i dette tilfelle er dato.

Innsetting av elementer i denne databasen fungerer annerledes enn det gjør i databasen til Dyrkingsareal. I Dyrkingsareal brukes et parameterobjekt med navn på forventede parametre. I databasen til Dyrk med Bybonden brukes alle parameter-verdi par fra innsendt data til å lage elementet. Dette betyr at det ikke er en innebygd garanti på hvilken attributter som har verdier, og hvilken som mangler, for et gitt element.

Fordi objektelementene skal sorteres etter en annen attributt enn de andre elementene, må sorteringsattributten konverteres mellom klientform og tjenerform. For eksempel vil et innlegg lagres slik på tjenersiden:

```
{
    "ID": "372964723652786",
    "parentID": "6754896745867046",
    "sort": "13/02/2019",
    ...
}
```

Når elementet skal sendes til tjeneren må det konverteres til:

```
{
    "ID": "372964723652786",
    "parentID": "6754896745867046",
    "date": "13/02/2019",
    ...
}
```

Som et eksempel vil elementer som sorteres etter dato konverteres til klient form slik:

```
map.put("date", map.get("sort"));
map.remove("sort");
```

I begge applikasjonene blir funksjonene på klientsiden kalt via en API Gateway. Dette er endepunktet til et REST-API som klientsiden av programmet kan kalle for å kommunisere med tjenersiden. Hver av funksjon på klientsiden blir tildelt sin egen URL. Klienten bruker POST-forespørsler når forespørslene inneholder informasjon. Dette inkluderer kall som "/getad", hvor ID-en blir sendt i kroppen til forespørselen, i steden for som parameter i URL-en til en GET. Dette har blitt gjort siden det krever et ekstra steg når GET-forespørsler blir brukt, som unngås med POST.

4.2 Kartimplementasjon

Da implementasjonen av matrikkelen og kartkomponenten er en så stor del av oppgaven og mye arbeid har blitt lagt ned i denne delen, var det naturlig å omtale temaet i et eget kapittel. På forhånd ble dette sett på som den mest kritiske og usikre delen av prosjektet. Av den grunn er det blitt valgt å legge ekstra vekt på emnet i rapporten.

4.2.1 Google maps

Det var nødvendig med funksjonalitet som gjør det mulig å navigere rundt på et kart, bruke posisjonstjenester og kunne plassere ut markører som indikerer definerte områder. Denne funksjonaliteten ligger inne i Google Maps, og det er Google Maps som brukes av React Native sin kartfunksjon som standard. Dermed ble det bestemt at Google Maps skulle brukes. Kartapplikasjonen henter markører fra databasen og plasserer dem på kartet. Disse markørene kan så trykkes på, og vil da vise informasjonen som tilhører den enkelte markør. Når det trykkes på kartet, et sted hvor det ikke finnes noen markør, plasseres det ut en markør på koordinatene hvor det ble trykt. Deretter sendes en POST-forespørsel til tjenerdelen som finner eier for adressen som markøren står på, samt tilhørende teig til adressen. Kartet bruker responsen til å tegne opp teigen på kartet. Det ble brukt et kartoverlegg over standardkartet til Google maps fra kartverket, "Topografisk Norgeskart 4".

4.2.2 Kartverket/Matrikkelen

Applikasjonen Dyrkingsareal var avhengig av å kunne hente ut informasjon om eierforhold til tomter. Frøydis Lindén arrangerte derfor et møte med GIS-koordinator hos Fylkesmannen i Vestland, Trond Rolland. Der ble det oppklart at denne informasjonen kunne fås tak i ved tilgang til matrikkelen. Tilgang til matrikkelen gis bare etter avtale med kartverket, og dette førte til stor tvil om det ville bli gitt tilgang. Det ble likevel gitt tilgang til matrikkelen, og dermed er det tilgang til informasjonen som trengs i applikasjonen.

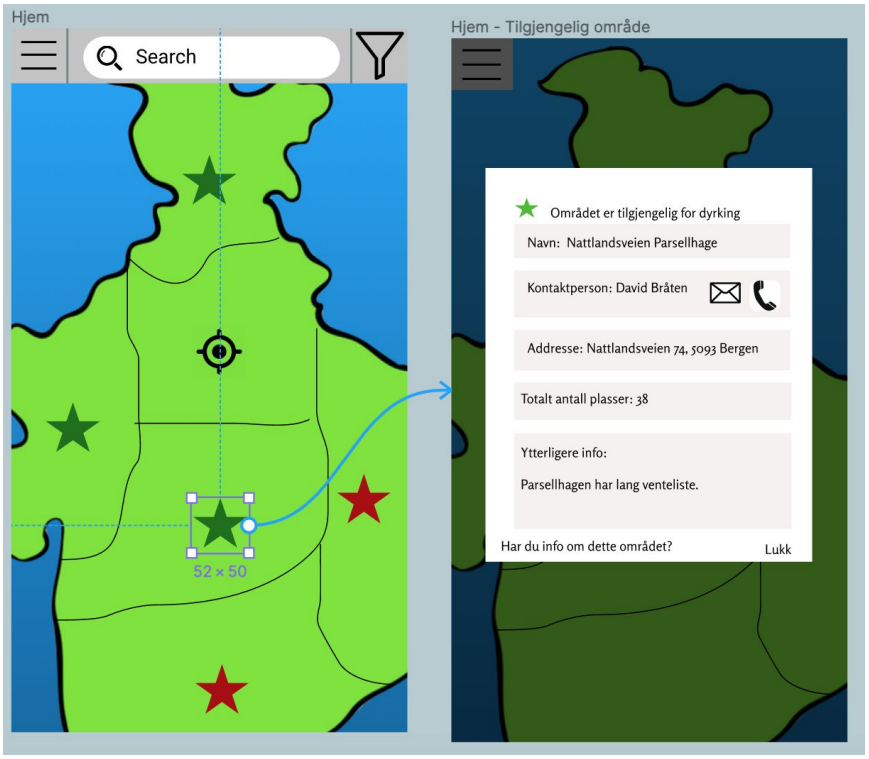
For å hente informasjonen er det opprettet en funksjon i AWS Lambda som tar lengdegrad og breddegrad som parameter. Denne funksjonen gjør et kall mot Geonorge sitt adresse-API, med lengdegrad, breddegrad og radius som parameter. Adresse-API'et bruker det oppgitte punktet og radius til å finne adresser som befinner seg innenfor punktets radius. Hver adresse er definert med et festepunkt. API'et returnerer en liste med adresser, sortert etter avstand fra festepunkt til det oppgitte punktet. Deretter benyttes matrikkelens utvidede service for å hente ut teig for en gitt adresse.

Ettersom hver adresse er definert med et festepunkt, er det ikke nødvendigvis adressen som punktet faktisk ligger på som returneres som nærmeste adresse. For å kunne være sikker på at rett adresse for punktet er funnet, sjekkes det om dette punktet finnes innenfor teigen. En teig er her definert av en liste med koordinater. En algoritme sjekker om punktet er innenfor eller utenfor teigen. Dersom punktet finnes innenfor teigen er rett adresse funnet. Dersom punktet derimot ikke finnes innenfor teigen, sjekkes dette mot neste addresses teig. Hvis punktet ikke ligger på teigen til noen av adressene i listen, gjøres et nytt kall mot adresse-APIet, med utvidet radius, og de nye adressenes teiger blir sjekket. Dette gjentar seg helt til en satt maksradius er nådd. Om korrekt adresse er funnet, brukes matrikkelens basisservice for å hente ut eiere for adressen. Her brukes adressens kommunenummer, gardsnummer og bruksnummer. En adresse kan ha flere eiere, men her returneres første eier. Hvis en eier er funnet for adressen, opprettes en URL til www.1881.no, med eier som søkeparameter. Deretter returnerer funksjonen på Lambda eierinformasjon, sammen med URL-en og koordinatene som definerer teigen.

4.3 Klient

4.3.1 Prototype

Det ble som tidligere nevnt lagt vekt på en godt utarbeidet prototype som var egnet for brukertesting, hvor det samtidig skulle være tydelig hva som var planlagt funksjonalitet.



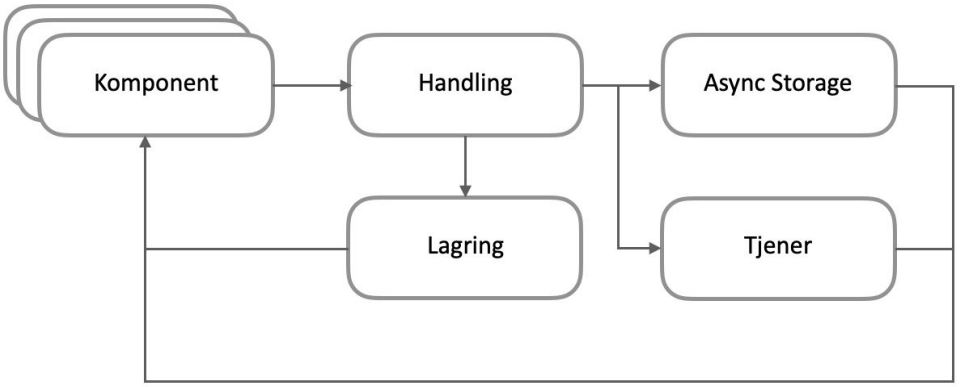
Figur 4.4: Utsnitt fra skjermdump av Figma-modell for applikasjonen Dyrkingsareal

Eksempelet i figuren over er hentet fra prototypen til Dyrkingsareal og illustrerer navigering mellom to skjermer. Den blå pilen illustrerer en hendelse ved trykk på stjernekomponenten, som fører til en dialog åpnes og vises på skjermen.

I prototypene ble all ønskelig funksjonalitet inkludert, til tross for at det ikke var sikkert om det ville være mulig å få implementert alt innenfor tidsrammen som var satt. Det ble derfor viktig å prioritere hvilken funksjonalitet som var viktigst for applikasjonene da implementasjonen av dette ble satt i gang.

4.3.2 Arkitektur

Parallelt med arbeidet med tjenerdelen ble det utviklet et brukergrensesnitt i React Native. Når det kommer til arkitektur er det i sammenheng med React naturlig å følge Flux-prinsippet. Flux er et mønster for å håndtere datastrømmen i en applikasjon (Flux (u.å.)), hvor det viktigste konseptet er at all data strømmer i samme retning. Måten mønsteret har blitt benyttet følger følgende modell:



Figur 4.5: Modell for arkitektur i brukergrensesnitt

Komponenter

Komponentene representerer det som Flux-prinsippet omtaler som "View". Dette kan sammenlignes med "View" i MVC-mønsteret, hvor komponentene er ansvarlige for å vise innholdet fra lagrene. Det er nødvendig for komponentene å abonnere på de ulike lagrene, slik at ved endring av innhold vil tilstanden til komponenten oppdateres og dermed endre hva som vises for brukeren. Dette gjøres ved hjelp av render-metoden som omtales i delkapittel 4.4.3.

Handling

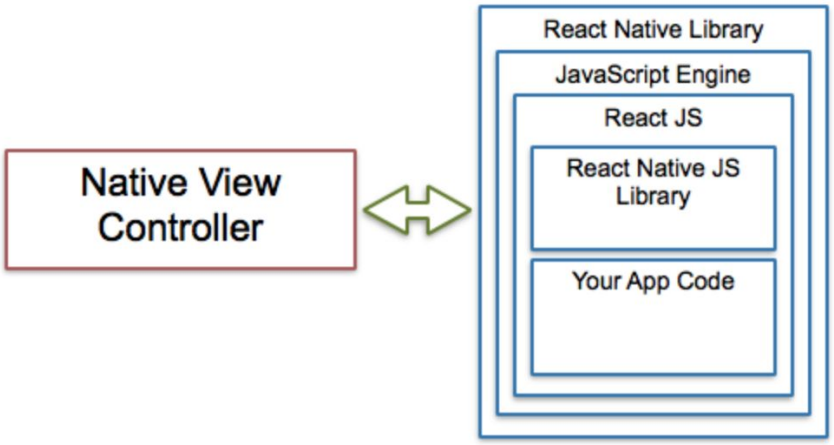
I et React komponent er det innholdet i render() som vises for brukeren. I komponentens klasse utover dette vil imidlertid inneholde metoder som er ansvarlige for handlinger som påvirker data i lagrene. Det er naturlig at en komponent inneholder både render() og andre metoder, da den gitte komponentens tilstand skal benyttes i de ulike metodene, så vel som i render().

Lagre

I modellen representeres lagre ved "lagring", "Async Storage" og "tjener". I hovedsak kan disse omtales på samme måte og de fungerer alle som baser for lagring av data. Lagrene håndterer mer enn en komponents data, i motsetning til "Model"s rolle i MVC (React Native (u.å.)). Innholdet i lagrene blir kun redigert ved påvirkning fra handlinger og sendes ved endringer til abonnerende komponenter. Lagring i modellen er data som inneholder komponentenes tilstand, som beskrives nøyere i avsnittet om "state". Tjener er her en ekstern lagringstjeneste som håndteres ved hjelp av et REST-API. Async Storage forklares også senere i kapittelet.

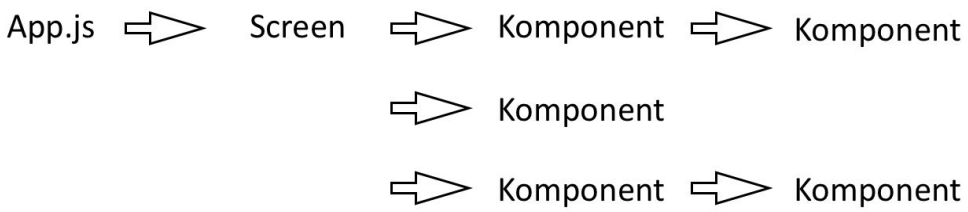
4.3.3 Brukergrensesnitt

Ved hjelp av React Native kunne det utvikles plattformavhengige applikasjoner til både iOS og Android.



Figur 4.6: Oppbyggingen av React Native-applikasjoner (Codebrahma Software Development Service (2018)).

Måten applikasjonene har blitt implementert på er gjennom komponent-konseptet som React Native legger til rette for. På denne måten deles funksjonaliteten opp og komponenter kan gjenbrukes i flere deler av applikasjonen. Hierarkiet her følger et strukturert mønster:



Figur 10 : Hierarki brukergrensesnitt

Rendermetode

Den eneste obligatoriske metoden i en komponent er rendermetoden(React Native, u.å.), som returnerer JSX, som igjen framvises.

App.js

Rutes fra Index.js filen, som er startpunktet i applikasjonen. Innholdet i App.js' rendermetode avgjør hva som visualiseres for brukeren.

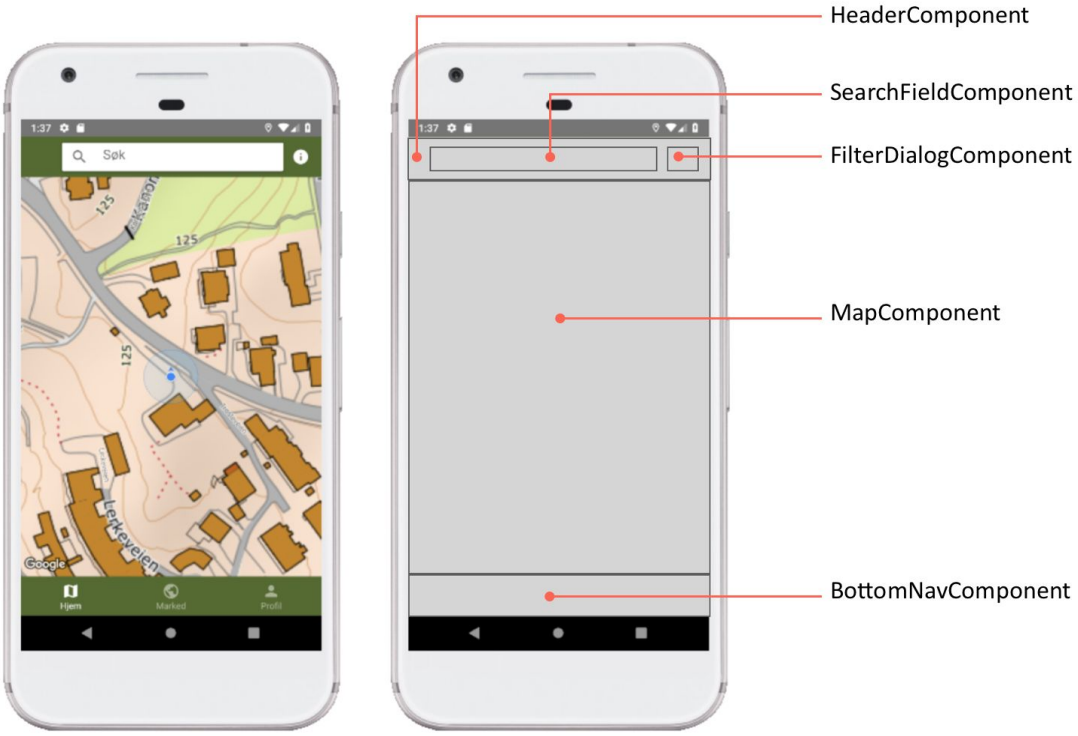
Screen

Strukturen i brukergrensesnittet er basert på at hver konkrete side vist for brukeren representeres av en screen. Hver screen inneholder i utgangspunktet kun komponenter som definerer innholdet. Screens er også subklasser av Reactkomponenten og har samme oppbygging. Det har her imidlertid for praktiske årsaker blitt skilt mellom et screen og en komponent, selv om det i utgangspunktet har samme funksjon.

Komponent

Brukergrensesnittet som skal vises i applikasjonen er i hovedsak satt i komponentene. Alle komponenter er subklasser av Reactkomponenten. Komponenter tar igjen bruk av andre komponenter, for å unngå mye kode i samme komponent, samtidig som dette legger til rette for gjenbruk.

Under følger et eksempel hentet fra Dyrkingsareal hvor det illustreres bruk og oppbygging av komponenter.



Figur 4.7 : Eksempel på bruk av komponenter

I tilfellet over illustreres "MapScreen" og hvordan den er bygget opp. "MapScreen"s oppgave er å frambringe komponenter som skal vises for brukeren. Rendermetoden er dermed som følger:

```
render() {
 return (
   <View style={{flex: 1}}>
     <HeaderComponent />
     <Card style={styles.card}>
       <MapComponent style={styles.map}/>
     </Card>
   </View>
```

```
 );
}
```

Utover dette er det et poeng å videre gi eksempel på komponenter som frambringer komponenter, kalt foreldrekomponenter. I eksempelet under er HeaderComponent en slik komponent, mens "SearchFieldComponent" og "FilterDialogComponent" er barnekomponenter:

```
render(){
 return (
   <Appbar.Header style={styles.top}>
     <SearchfieldComponent />
     <FilterDialogComponent />
   </Appbar.Header>
 );
}
```

I dette eksempelet er det ikke tatt i bruk videreføring av data. Måten dette gjøres på er ved å sende "props" sammen med barnekomponenten. Om dette skulle blitt tatt i bruk i eksempelet over kunne det sett ut som følger:

```
<SearchfieldComponent data={data} />
```

I "SearchfieldComponent" kan dataen hentes ut på denne måten:

```
data = this.props.data
```

Props er på denne måten fastsatt av foreldrekomponenten og er dermed gjeldende gjennom komponentens livssløp (React Native (u.å.)). Om det derimot skal benyttes data som skal kunne endre verdi i løpet av livsløpet skal det "state" brukes. State kontrollerer komponenten ved at den styrer når render skal kalles og dermed hva som vises i applikasjonen. State settes vanligvis i konstruktøren, men kan oppdateres vilkårlig innad i komponenten, også fra rendermetoden. Når setState() kalles vil det forårsake at rendermetoden kalles på nytt og grensesnittet vises med den nye tilstanden. Et viktig moment med state i forhold til andre lagringskanaler er at en komponents tilstand blir "tømt" når applikasjonen asvluttes og omstartes. For illustrasjons skyld følger en forenklet version av en komponent, hvor state settes i konstruktøren og kaller setState() i både rendermetoden og i _getData().

```
class CommentField extends Component {

  constructor(props) {
    super(props)

    this.state = {
        content : '',
        nameUser : ''
    }
     this._getData();
  }

 _getData = async () => {
  try {
    const value = await AsyncStorage.getItem('name')
    if(value !== null) {
      this.setState({nameUser: value})
    } else {
     this.setState({nameUser: 'Anonym'})
  } catch(e) {
    console.log("Cannot access asyncStorage");
  }
}

  render() {
    const { content } = this.state;
    return (
      <View>
        <TextField
          label='Din kommentar'
          value={content}
          onChangeText={ content => this.setState({content})}
          />
      </View>
    )}
}
```

Måten dette er løst på gjør at hver gang brukeren endrer inndata i kommentarfeltet blir dermed staten "content" endret.

Metoden _getData() illustrerer i tillegg til endring av state et viktig moment under utviklingen av Dyrk med Bybonden. Her ble det besluttet å ta i bruk lagring på lokalt nivå når det kom til brukerpreferanser. Brukeren av applikasjonen har ingen form for logg-inn, men lagrer navn og andre variabler lokalt. På denne måten var det ikke behov for autentisering, samt den begrenset behov for lagring i databasen. Det er det nøkkel-verdi-baserte systemet Async Storage ble benyttet for å løse denne delen (React Native Async Storage(u.å.)). Systemet er globalt for hele applikasjonen og er en enkel og effektiv måte å sette, oppdatere og aksessere brukerdefinerte variabler. Systemet ble blant annet benyttet til å sette navn på kommentarer, avgjøre om en bruker har likt en post/kommentar/svar, samt om det skal sendes pushvarsler. Async Storage vil på iOS lagre små verdier i en serialisert liste og større verdier i en separat fil ved bruk av plattformavhengig kode (React Native Async Storage (u.å.)). I tilfeller med Android-enheter vil det på sin side bli tatt i bruk RocksDB eller SQLite, basert på hva som er tilgjengelig.

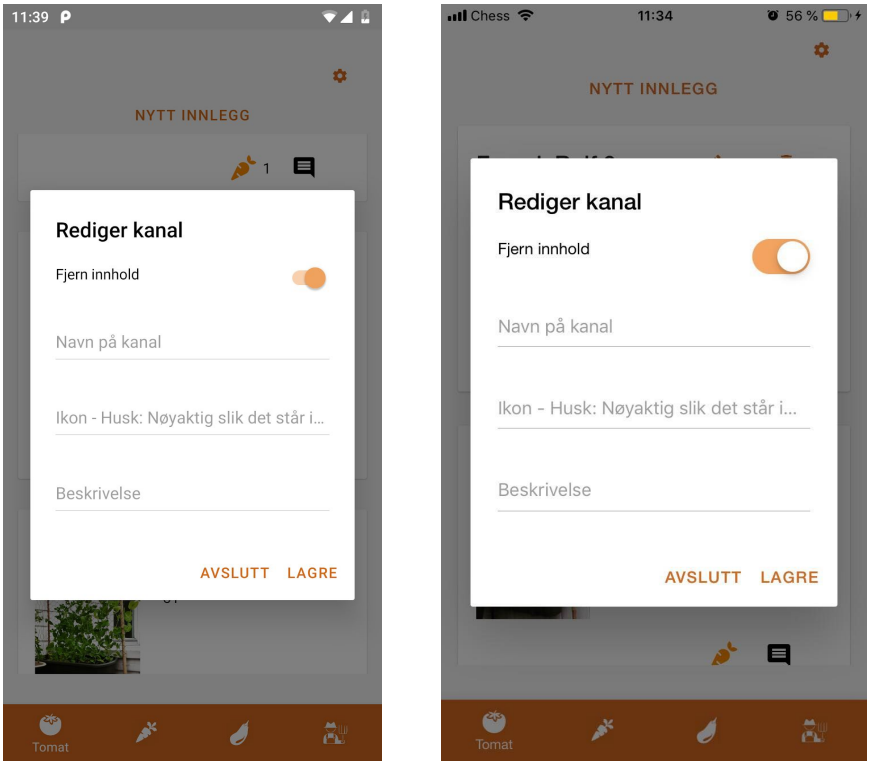
I Dyrkingsareal ble det imidlertid ikke tatt i bruk lokal lagring, da autentisering ble benyttet ved hjelp av Amazon Cognito. Amazon Cognito gir brukeren et symbol(“token”) som blir brukt til identifisering mot tjeneren og databasen. Dette gjør at applikasjonen kan gi brukeren tilgang til å endre sine egne annonser. Det blir også brukt for å identifisere hvem som oppdaterer informasjonen i kartet. Grunnen til at det ønskes å vite hvilken bruker som har lagt inn informasjonen er at det skal være mulig å deaktivere eller blokkere brukere som legger ut feilaktig eller upassende innhold.

Pushvarsler

Det er opprettet emner på AWS SNS, for prosjektene som Bybonden skal sende ut varsler for.
Brukere kan velge hvilke vekster de ønsker å abonnere på, og vil motta meldinger fra emnet som tilhører veksten.

Design

I designet av brukergrensesnittene i applikasjonene har det blitt valgt å følge Googles Material Design-prinsipper. Dette handler om å designe på en måte som er forståelig og forutsigbar for brukeren, med blant annet responsive knapper, intuitive ikoner og skyggelegging (Envato (2016)). For å få til dette har React Native Paper blitt benyttet. Dette er en samling med komponenter med muligheter for tilpasning for React Native. Her følges retningslinjene for Material Design, samt retningslinjer for plattformtilpasning (React Native Paper (2019)). Et eksempel på plattformtilpasning vises på figuren under, hvor bryteren for å skru av eller på pushvarsler er ulik for Android og iOS. Figuren viser også blant annet hvordan bakgrunnen skyggelegges ved åpne dialoger, for å gjøre det tydelig hva som er det aktive området.



Figur 4.8: Skjermdump fra applikasjonen Dyrk med Bybonden, viser forskjell mellom henholdsvis Android og iOS

4.4 Nettside

4.4.1 Weebly

Weebly er en "dra-og-slipp"-nettsidebygger, som betyr at nettsiden bygges opp ved å dra elementer og slippe dem der de ønskes plassert. Disse elementene kan for eksempel være tekst, bilder og knapper. Det er også mulig å legge inn ren HTML og Javascript, dersom de forhåndsdefinerte elementene ikke inneholder ønsket funksjon. Dette ble brukt for å implementere Tito og Facebook-kalender, nærmere forklart under punkt 4.5.2 og 4.5.3 og for legge til "vis mer"- og "vis mindre"-knapper for noe av teksten.

Gjennom diskusjon med oppdragsgiver ble det klart hvilke sider som skulle være på nettstedet. Etter dette ble designet og oppsettet for siden laget, mens oppdragsgiver lagde innholdet. Da alt dette var klart ble innholdet lagt inn og noen siste finjusteringer ble gjort før nettsiden ble lansert.

4.4.2 Implementasjon av Tito

Bybonden i Bergen arrangerer fra tid til annen arrangementer og kurs med en kursavgift for de som ønsker å delta. Et av ønskene til oppdragsgiver var at deltakerne skulle kunne melde seg på gjennom nettsiden. Tito er en betalingstjeneste som gjør det enkelt å ta betaling for påmelding til arrangementer, og Bybonden benyttet seg allerede av denne tjenesten. Tidligere har folk som ønsket å melde seg på kursene måtte sende e-post til Bybonden og Bybonden har måtte håndtert hver enkelt påmelding. Ved hjelp av JavaScript ble Tito sine tjenester implementert i nettsiden og deltakerne kan nå melde seg på direkte gjennom nettsiden.

4.4.3 Implementasjon av Facebook-kalender

Bybonden oppretter arrangementer som folk kan melde seg på, og ønsket å kunne gjøre dette på enklest mulig måte. Tidligere har Facebook blitt brukt for å opprette arrangementer, og det var ønskelig å fortsette med dette. Løsningen ble derfor å vise disse arrangementene på nettsiden. Dette ble løst ved å bruke Facebooks eget API. Nettsiden bruker JavaScript for å hente informasjon fra APIet og viser pågående eller kommende arrangementer, sortert etter dato.

Figur 4.9: Utsnitt av skjermdump fra nettside, viser Facebook-kalenderimplementasjon

Koden for kalenderen:

```
<div id="fb-root"></div>
<script async defer crossorigin="anonymous"
src="https://connect.facebook.net/nn_NO/sdk.js#xfbml=1&version=v3
.2">
</script>

<div class="fb-page"
data-href="https://www.facebook.com/Bybonden-i-Bergen-19821624354
15167/" data-tabs="events" data-width="500" data-height="1000"
data-small-header="true" data-adapt-container-width="true"
data-hide-cover="false" data-show-facepile="true">
<blockquote
cite="https://www.facebook.com/Bybonden-i-Bergen-1982162435415167
/" class="fb-xfbml-parse-ignore">
<ahref="https://www.facebook.com/Bybonden-i-Bergen-19821624354151
67/">Bybonden i Bergen</a></blockquote></div>
```

5 Evaluering

5.1 Evalueringsmetode

5.1.1 Testing

Det har vært lite fokus på enhetstesting gjennom prosjektet. Testing har blitt gjort ved å bruke funksjoner og se om forventet resultat har blitt oppnådd. Denne prosessen har vært gjennomgående for alle komponenter i prosjektet. For brukergrensesnitt har det blitt kontinuerlig testet om komponentene har sett ut, samt oppført seg, som forventet, helt til ønsket resultat har blitt oppnådd. Brukergrensesnittene har deretter blitt testet på flere enheter, både emulerte og reelle enheter. På tjenerdel-funksjonalitetene har det blitt brukt Postman for se at riktige ting blir lagt i databasen, at databasen returnerer riktig data og at funksjoner returnerer forventede verdier. For testing av kartfunksjonaliteten og autorisering har det blitt brukt React Native sin remote debugger. Debuggeren har også blant annet hjulpet med å finne ut av strukturen på sesjoner, og hvordan bruke token. På tjenerdelen har det også blitt benyttet AWS Toolkit til å teste funksjoner, og for å vise tiden på utførelse av funksjoner. Dette har blitt brukt til optimalisering og utbedring av funksjoner.

5.1.2 Evaluering

Helt fra oppstartsfasen har det vært viktig for oppdragsgiver å få innspill fra personer i landbruksmiljøet i Bergen. Dette både fra personer som har tilgang til areal, de som har bruk for areal og generelt personer som har interesse for dyrking, i tillegg til tilfeldig utvalgte. Dyrk med Bybonden og Bybonden generelt har som formål å inspirere folk til å starte opp med dyrking, så vel som å legge til rette for det eksisterende miljøet. Det var som tidligere forklart planlagt å opprettholde kontakt med gruppene og involvere deres meninger og ønsker under hele utviklingsprosessen, samt teste funksjonalitet og brukervennlighet.

Figur 5.1: Bilde fra kurs i Design Thinking, oversikt over tilbakemeldinger

I regi av Fylkesmannen i Vestland ble det avholdt kurs i "Design Thinking", hvorav metoden ble tatt i bruk for å utføre testing blant brukere ved flere anledninger. Design Thinking er en metodikk som gjennom et sterkt brukerfokus skal sørge for et produkt hvor brukerens ideer og faktiske behov spiller en stor rolle (Smart Innovation Norway (u.å.)). Dette gjøres gjennom en såkalt "bottom-up"-prosess hvor det opprinnelige ideen ikke kun skal behandles av initiativtaker og utvikler, men med involvering fra potensielle brukere av løsningen. På denne måten skal det skapes verdi for brukerne og samtidig unngå fremmedgjøring av produktet.

Design Thinking ble en viktig del av utformingen av mobilapplikasjonene, særlig i oppstartsfasen. Det var et stort fokus på å få utarbeide en prototype tidlig, slik at det fra start av kunne inkluderes personer fra målgruppen i prosessen rundt utarbeiding av funksjonaliteter. Ved bruk av Figma fikk testpersoner sett det initielle løsningsforslaget visualisert. Med Figmas mobilapplikasjon ble det samtidig enklere å konkretisere løsningen og å få en følelse av hvordan applikasjonenes brukergrensesenitt ville oppleves.

Prototypen ble først benyttet ved testing med Parsellhagen på Landås. Foreningen består av medlemmer som er godt innarbeidet i det urbane landbruksmiljøet. Her var det derfor i første omgang Dyrkingsareal som var interessant å teste. Gangen i testingen startet med en gjennomgang av applikasjonens formål og innhold. Deretter ble det lagt til rette for at hver enkelt fikk prøve ut prototypen og navigere rundt i den. Samtidig som dette foregikk ble det gitt tilbakemeldinger og stilt spørsmål rundt funksjonalitet. Her ble det etterhvert anledning for å presentere Dyrk med Bybonden og initiativet bak. Også her fikk medlemmer av foreningen testet ut prototypen.

Ved en annen anledning ble prototypen testet blant målgruppen for Dyrk med Bybonden som i hovedsak omfatter personer med liten til ingen kunnskap om dyrking. Dette ble gjennomført ved utspørring tilfeldige personer på Bergen Storsenter. Intervjuobjektene ble spurt om interesse for dyrking og hva som eventuelt skulle til for å skape en interesse. Deretter ble prototypen til Dyrk med Bybonden framvist og intervjuobjektene fikk prøve den ut.

Litt lengre ut i løpet ble det utført en brukertest blant bønder under en samling i Hordaland Bondelag. Bøndene var aktuelle som brukere av Dyrkingsareal, som innehavere av mulig tilgjengelig og dyrkbar jord. Løpet i testingen ble det samme som testing som var utført tidligere ved at det var mulighet for å navigere rundt på Figma-modellen samtidig som det ble gitt tilbakemeldinger og nye ideér rundt konseptet.

Ved prosjektslutt ble det dannet en testgruppe bestående av personer, med lite til ingen erfaring med dyrking, for å teste Dyrk med Bybonden. Her ble det lagt vekt på at testobjektene skulle få oppleve eksempler på hvilket innhold applikasjonen kunne ha. Dette

ble utført ved at Bybonden la ut innhold underveis i testingen, slik at det oppsto en følelse om hva som var aktuelt at ville legges ut av Bybonden. I denne delen av testingen ble det benyttet den faktiske applikasjonen, til tross for at produktet ikke var helt komplett. På denne måten fikk testpersonene oppleve hvordan applikasjonen var ved første oppstart og med egendefinerte valg.

5.2 Evalueringsresultat

5.2.1 Resultat gjennom testing

Gjennom funksjonell testing har det til enhver tid blitt sikret at funksjonaliteten har vært slik som forventet. Den kontinuerlige testingen av applikasjonene etter at nye funksjoner har blitt implementert, har resultert i at nye feil ble oppdaget tidlig og rettet opp i. Dette gjorde at applikasjonene hele tiden hadde en kjørbar versjon.

5.2.2 Resultat gjennom evaluering av prototype

Spesielt når det kommer til Dyrkingsareal har tilbakemelding fra potensielle brukere spilt en avgjørende rolle. Her var det enkelt å identifisere målgruppen og derfor klarere hvilke tilbakemeldinger som var av verdi for produktet. I og med at det kom forespørsler og ønsker om funksjonaliteter etter demonstrasjon av prototypen, var det enkelt å inkrementere dette som en del av løsningen. Særlig når det kom til hvilke data som skulle foreligge om dyrkbare områder kom det en del forespørsler, som ble tatt hensyn til.

Som nevnt tidligere ble det i første instans holdt test blant medlemmer av et parsellag. Her var det stor interesse for Dyrkingsareal og flere påpekte at dette var etterlengtet. I og med at testerne her var medlemmer i et parsellag var det et ønske om å få inkorporert funksjonaliteter som dekker behovet for et lag. Dette behovet utgreies i kapittel 8 - “Videre arbeid”. Utover dette var det fokus på hvilke attributter som skulle lagres for hvert enkelt område. Det viktigste momentet som kom ut av dette var behovet for å få informasjon om forholdene på et gitt område. For flere i landbruksmiljøet er det i tillegg til å finne dyrkbare områder også ønskelig å finne områder som er tilgjengelig for andre formål, som for eksempel hold av insektshotell og midlertidige beiter. I utgangspunktet er Dyrkingsareal forbeholdt, som navnet tilsier, å finne områder for dyrking, men de ulike instansene som er involverte i prosjektet har alle interesse av å legge til rette for all form for urbant landbruk. Dermed ble det bestemt at applikasjonen også kan gi informasjon om områder utover det som er aktuelt i forhold til dyrking. Konkret betyr dette at jordsmonn ble inkludert i listen over informasjon om et område.

Det ble, som nevnt, uttrykt et ønske om et lag- og foreningsbasert element. I forhold til Dyrk med Bybonden kom dette til uttrykk ved at det ble foreslått at flere enn Bybonden skulle kunne legge ut informasjon om dyrking. Ønsket var at alle innehavere av applikasjonen

skulle kunne opprette "kanaler" som andre brukere igjen kunne følge. På denne måten kunne også et dyrkelag som en gruppe ha opprettet kanaler og der flere skulle administrere samme konto. Ved å gjøre det mulig for andre brukere å legge ut innhold, kunne det samtidig være ulike nivåer hva dyrkingsferdigheter angår. Dyrk med Bybondens initielle formål var å vekke interesse hos uerfarne personer. Ved å inkludere alle brukere som kunnskapsformidlere, ville derimot målgruppen få større omfang. Denne implementasjonen blir videre diskutert i kapittel 8.4 - "Videre arbeid".

Testing av prototyper på Bergen Storsenter ga bekreftelse på at det var behov for applikasjonene, og at funksjonene var implementert på en måte som var intuitiv fra et brukerperspektiv. Testingen ga ingen tilbakemeldinger som resulterte i at det ble gjort noen endringer i applikasjonene.

Gjennom arbeidet med bøndene fra Hordaland Bondelag ble det identifisert aspekter fra eiere av jordområder. Også her ble det uttrykt ønsker om mer informasjon på et område. Spesifikt var ønsket å få kjennskap til hvert områdes solforhold. Et viktig moment når det kommer til dyrking er nettopp hvor området er plassert i forhold til skygge og sol. Innspillet førte til at denne egenskapen ble inkorporert som informasjon om et område.

Som innehavere av jord var bøndene opptatt av at om det skulle være mulig for brukere å ta direkte kontakt med eiere, måtte det også være mulig med en form for reservasjon mot dette. Dette førte til at det ble implementert en advarsel ved forsøk på å kontakte eiere av jord som er oppført som "ikke tilgjengelig". I og med at Dyrkingsareal benytter UGC-prinsippet, kan i utgangspunktet hvem som helst oppdatere informasjon om et område. Det er ikke mulig for brukere å identifisere seg som eiere av et område, og derfor heller ikke mulig å reservere seg. Dermed ble den beste løsningen å advare brukere mot å kontakte innehavere av ikke-tilgjengelige områder.

5.2.3 Resultat gjennom arbeid med testgruppe

I og med at testingen av Dyrk med Bybonden foregikk ved bruk av den faktiske applikasjonen, var det i høy grad mulighet for å gi tilbakemelding i forhold til brukergrensesnitt og -vennlighet. Det kunne samtidig gis direkte respons til Bybonden om hvilket innhold som var ønskelig fra hennes side. Resultatet ble her at det ble foretatt enkelte endringer i forhold til utseende og plassering av komponenter. Utover dette ble det diskutert hvilke hendelser som skulle utløse sending av pushvarsler. I utgangspunktet var det planlagt å kun sende ut pushvarsler ved nye innlegg fra Bybonden. Det ble imidlertid gitt uttrykk for at det var behov for varsler i flere situasjoner, for å gjøre applikasjonen mer interaktiv. Dermed ble det lagt til rette for at brukere skal kunne velge å motta pushvarsler i følgende tilfeller:

- Ved nye innlegg fra Bybonden i de kanalene brukeren ønsker
- Ved “likes” eller kommentarer på et innlegg brukeren selv har lagt ut, eller har ”liket”/kommentert
- Når en kanal oppdateres med ny beskrivelse/nytt prosjekt

Som nevnt ble det gitt respons i forhold til hvilket innhold applikasjonen måtte ha for at den skulle være attraktiv for aktuelle brukere. Dette angår i hovedsak Bybonden, da hun skal stå for håndteringen av dette. Det var imidlertid nyttig å forstå hvilken måte testpersonene så for seg å bruke Dyrk med Bybonden.

6 Forskningsspørsmål

6.1 Kostnadsestimat for tjenertjenester

Som tidligere nevnt var det begrensede ressurser når det kom til den økonomiske delen. Det var derfor viktig å finne fram til det alternativet som ville dekke behovet for tjenerdelen, til lavest mulig pris.

Det ble gjort et kostnadsestimat for de forskjellige typene løsninger, for å komme fram til en løsning som var mest mulig kostnadseffektiv. Amazon sitt alternativ til sine tjenerløse tjenester, er Amazon EC2. Ved bruk av Amazon EC2 betales det en fast timespris for en eller flere virtuelle servere. En t3.micro-instans med to virtuelle CPUer og 1GiB ram, sees på som et minimum for å dekke tjenerbehovet. Om én t3.micro-instans benyttes, koster det $0.0118 per time. Med 720 timer i måneden vil dette koste $8.496 per måned (AWS (u.å.)).

	vCPU	Minne	Pris per time
t3.nano	2	0.5 GiB	$0.0059
t3.micro	2	1 GiB	$0.0118
t3.small	2	2 GiB	$0.0236
t3.medium	2	4 GiB	$0.0472
t3.large	2	8 GiB	$0.0944
t3.xlarge	4	16 GiB	$0.1888
t3.2xlarge	8	32 GiB	$0.3776
t2.nano	1	0.5 GiB	$0.0066
t2.micro	1	1 GiB	$0.0132
t2.small	1	2 GiB	$0.026
t2.medium	2	4 GiB	$0.052
t2.large	2	8 GiB	$0.1056
t2.xlarge	4	16 GiB	$0.2112
t2.2xlarge	8	32 GiB	$0.4224
m5.large	2	8 GiB	$0.111

Figur 6.1: Tabell over noen mulige EC2 konfigurasjoner med pris.

Ettersom at denne konfigurasjonen sees på som et minimum, er dette den billigste mulige EC2-løsningen for dette prosjektet. Kraftigere konfigurasjoner koster mer, dermed er dette minstepris for en EC2-løsning. Dette gjorde at Amazon sine tjenerløse tjenester ble valgt.

AWS API Gateway

AWS API Gateway har et gratisnivå, som inkluderer 1 000 000 forespørsler til REST-API (AWS (u.å.)). Det er beregnet å ligge godt under dette og AWS API Gateway vil dermed ikke koste noe.

AWS Lambda

Den månedlige prosessortidprisen er $0.00001667 per gigabytesekund(GB-s) og gratisnivået til AWS gir 400 000 GB-s gratis. Det er beregnet at funksjonene på Lambda bruker i gjennomsnitt omtrent 1 sekund. Funksjonene er tildelt 512MB hver. Hvis det antas at det blir gjort 1 000 000 eller færre forespørsler, gir dette følgende regnestykke (AWS (u.å.)):

Total antall sekunder = 1 000 000 * (1s) = 1 000 000 sekunder
Total GB-s = 1 000 000 * 512MB/1024 = 500 000 GB-s
500 000 GB-s - 400 000 GB-s = 100 000GB-s
$0.00001667 * 100 000 GB-s = $1.667

Kostnadene til Lambda vil med disse parameterne havne på $1.667 per måned.

AWS DynamoDB

DynamoDB er gratis hvis det ikke leses mer enn 1GB data fra databasen per måned, at mengden lagret data ikke overstiger 25GB og at det ikke er mer enn 2 500 000 leseforespørsler per måned (AWS (u.å)). Med utgangspunkt i dette, vil ikke DynamoDB koste noe for dette prosjektet.

AWS Cognito

Med 50 000 eller færre månedlig aktive brukere(MAU) fra Cognito User Pools, koster det ingenting for AWS Cognito. Oppdragsgiver antar at antall MAU vil ligge under 1 000 i første omgang, og vil ikke overstige 20 000. Dermed vil det ikke påbeløpe noen kostnader på AWS Cognito (AWS (u.å.)).

AWS SNS

De første 1 million publikasjonene og leveransene av meldinger til SNS, er med i gratisnivået til SNS. Bybonden regner med at det på det meste vil bli publisert én melding om dagen. Med 20 000 MAU og 1 melding som skal leveres til hver, blir dette 620 000 leveranser av meldinger per måned. Dette havner innenfor gratisnivået, og vil ikke koste noe (AWS (u.å.)).

6.2 Personvernregler - GDPR

En viktig betraktning gjennom prosjektet har vært å passe på at det som blir utviklet overholder personvernregler, og da spesielt den nye personvernforordningen, også kjent som GDPR. Her er det en del hensyn som må tas i forhold til hvilke data som samles inn, behandles og lagres, og hvordan dette gjøres. Informasjon om hvilke hensyn som må tas, samt gjeldende lover og regler, ble kartlagt gjennom møte med Fylkesmannen i Vestland v/Frøydis Lindén.

Et viktig i punkt i disse reglene er at det ikke skal innhentes mer personlig informasjon enn det som trenges. På grunn av dette samles det ikke inn personlig informasjon i applikasjonen Dyrk med Bybonden. Det eneste som brukes for å identifisere brukerne her er et brukernavn. Dette gjør at de fleste av de andre reglene hovedsakelig påvirker applikasjonen Dyrkingsareal.

Et av punktene som må overholdes er at det må skaffes eksplisitt samtykke fra brukeren til å behandle personlig informasjon. Det er også krav om at dette samtykket til enhver tid skal kunne bevises. Navn, e-post og telefonnummer er alle eksempler på personlig informasjon som blir behandlet i applikasjonen Dyrkingsareal. For å sikre at applikasjonen overholder denne regelen, kan brukere av applikasjonen bare opprette en brukerprofil dersom de samtykker til at denne informasjonen blir lagret og behandlet. Reglene sier også at brukeren har rett på å vite hva informasjonen som er samlet inn om brukeren skal brukes til. Dette blir brukerne opplyst om når de oppretter sin brukerprofil.

Et annet krav er at brukerne skal ha tilgang til hvilken informasjon som er lagret om dem. Dette vil være informasjonen brukeren oppga ved oppretting av brukerprofilen, samt eventuell informasjon brukeren har lagt inn gjennom oppretting av annonser. Brukeren vil alltid kunne se alt dette på profilsiden i applikasjonen. Et siste punkt som er verdt å nevne er at brukeren har rett til å bli glemt. Dette vil si at brukeren kan kreve at all informasjon knyttet til brukeren skal slettes. Dette er løst ved å gi brukeren en “Slett brukeren min”-knapp på innstillinger-siden. Om brukeren trykker på denne, slettes brukeren og all tilknyttet informasjon, inkludert annonser og informasjon brukeren har lagt inn i kartet. Om brukeren skulle ha spørsmål rundt dette, kan de finne kontaktinformasjon til eieren av applikasjonen på denne siden.

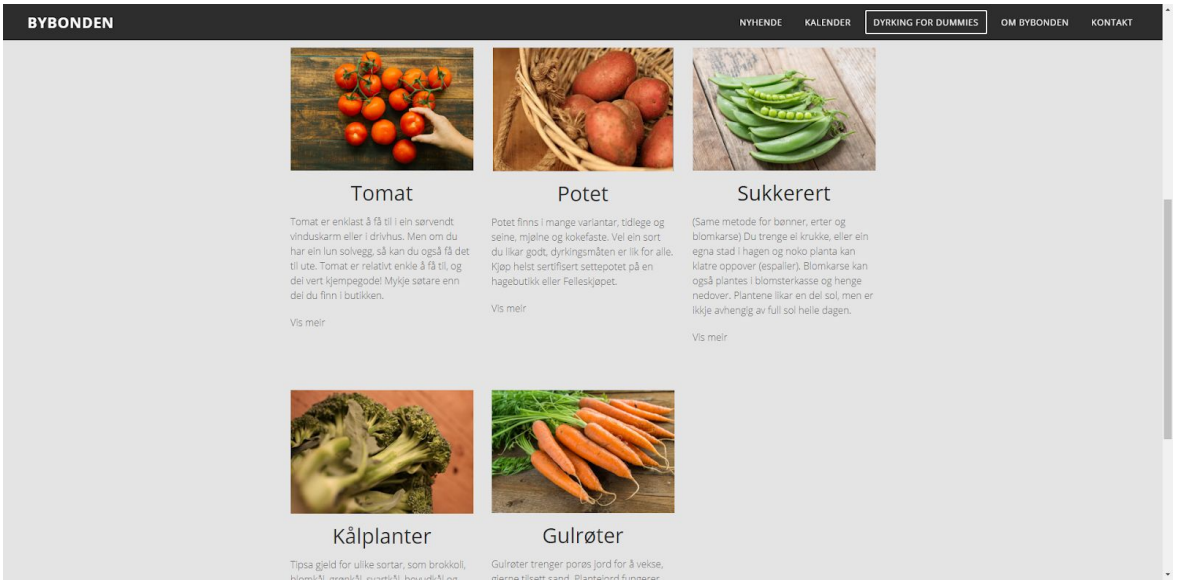
7 Diskusjon

7.1 Resultat

Det endelige resultatet ved prosjektslutt var tre ulike instanser: En nettside og to uavhengige mobilapplikasjoner. Nettsiden ble publisert tidligere i løpet, mens mobilapplikasjonene skal lansereres i løpet av juni. Dyrk med Bybonden og Dyrkingsareal blir å finne på Google Play og App Store.

7.1.1 Nettside

Resultatet ble et nettsted bestående av seks hovedsider: Forsiden, nyheter, kalender, "Dyrking for dummies", om Bybonden og kontakt. Hele nettsiden kan sees på www.bybondenibergen.no.



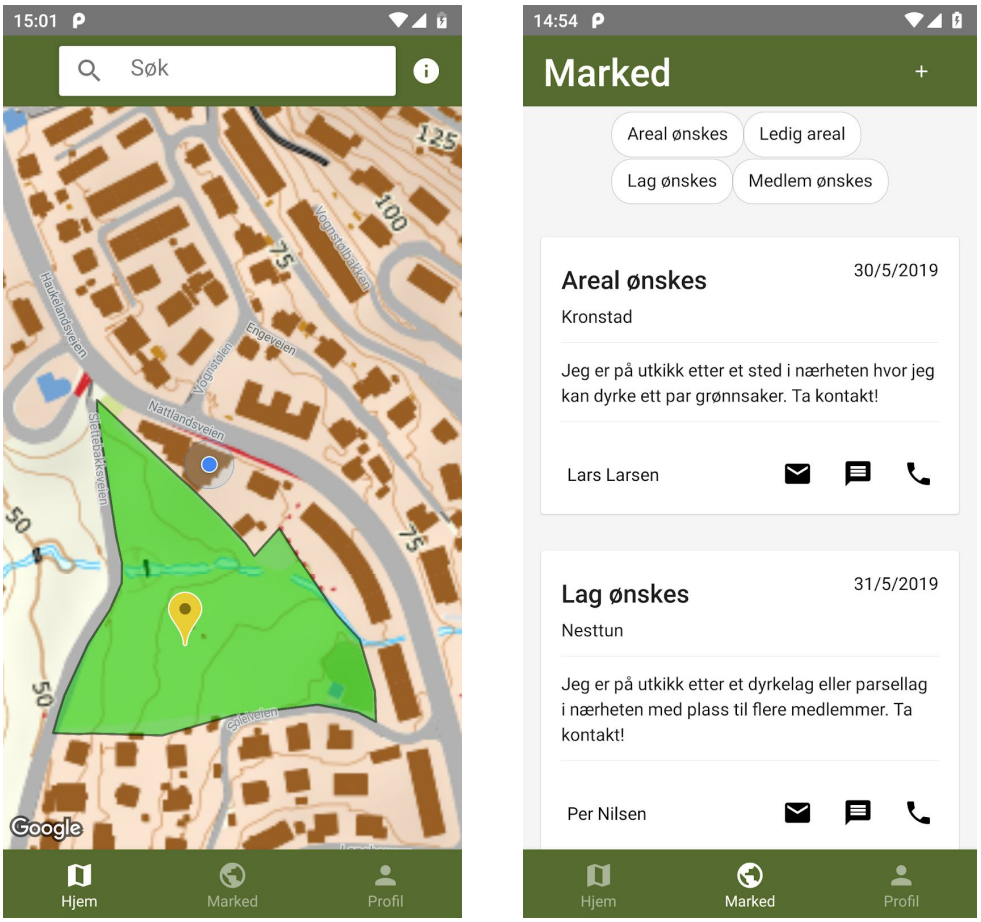
Figur 7.1: Skjermdump fra nettside

Nyhetssiden er en bloggbasert side hvor Bybonden kan legge ut innlegg og oppdatere leserne om hva hun driver med. Her kan også leserne gå inn og legge igjen kommentarer. På kalendersiden vises arrangementene Bybonden har opprettet. Disse hentes automatisk fra Facebook, og legges til i kalenderen når hun oppretter nye arrangementer der, og fjernes fra kalenderen når arrangementene er avsluttet. Ved siden av kalenderen er en påmeldingstjeneste til de arrangementene som det er avgifter på. Dette er implementert med en modul fra Tito, og oppdateres automatisk mot Titoprofilen til Bybonden. Når et arrangement er fylt opp, stenges påmeldingsmuligheten og de som ikke fikk plass kan da eventuelt sette seg på en venteliste.

"Dyrking for dummies" er en side hvor leserne finner dyrkingstips, både generelt og for noen spesifikke vekster. På en annen del av nettsiden kan det leses mer om Bybonden, det hun driver med og hvilke tjenester hun tilbyr. Til slutt er der en side med kontaktinformasjon og et kontaktskjema for eventuelle henvendelser.

7.1.2 Dyrkingsareal

Dyrkingsareal ble utviklet som en mobilapplikasjon med formål om å gjøre det enkelt for folk å få tak i dyrkingsareal og komme i kontakt med eiere, samtidig som det skal være enkelt for eiere som ønsker å leie ut områder å finne andre som er interessert i dette. Første gang brukeren starter applikasjonen blir de bedt om å logge inn eller opprette en bruker, dersom de ikke har en. Når dette er gjennomført vises kartet med brukerens posisjon og området rundt brukeren. Dersom det finnes markører med informasjon om områder, vil disse bli lastet inn og vist på kartet. Ved trykk på kartet plasseres det en markør som viser hvor det har blitt trykt. Dersom det finnes en teig for området, blir denne tegnet opp. Ved trykk på den opptegnede teigen, åpnes en dialog som viser informasjon om eieren til området. Dersom brukeren har informasjon om dette området kan det legges inn her, og det vil da opprettes en markør på kartet. Markører som har blitt lagt inn av brukere på kartet, kan trykkes på og vil da åpne en dialog med informasjonen som har blitt lagt inn. Alle brukere har mulighet til å redigere informasjonen til markører, samt slette markører. Brukerne har også mulighet til å filtrere hvilke markører de ønsker å se i kartet, for eksempel slik at de bare ser markører med kategorien "Parsellhage". Det er samtidig mulig å søke på adresser, stedsnavn eller områder. Ved søk på gyldig adresse eller stedsnavn, flytter visningen av kartet seg til denne lokasjonen.



Figur 7.2: Skjermdump fra Dyrkingsareal: Kartsiden og Markedsplassen

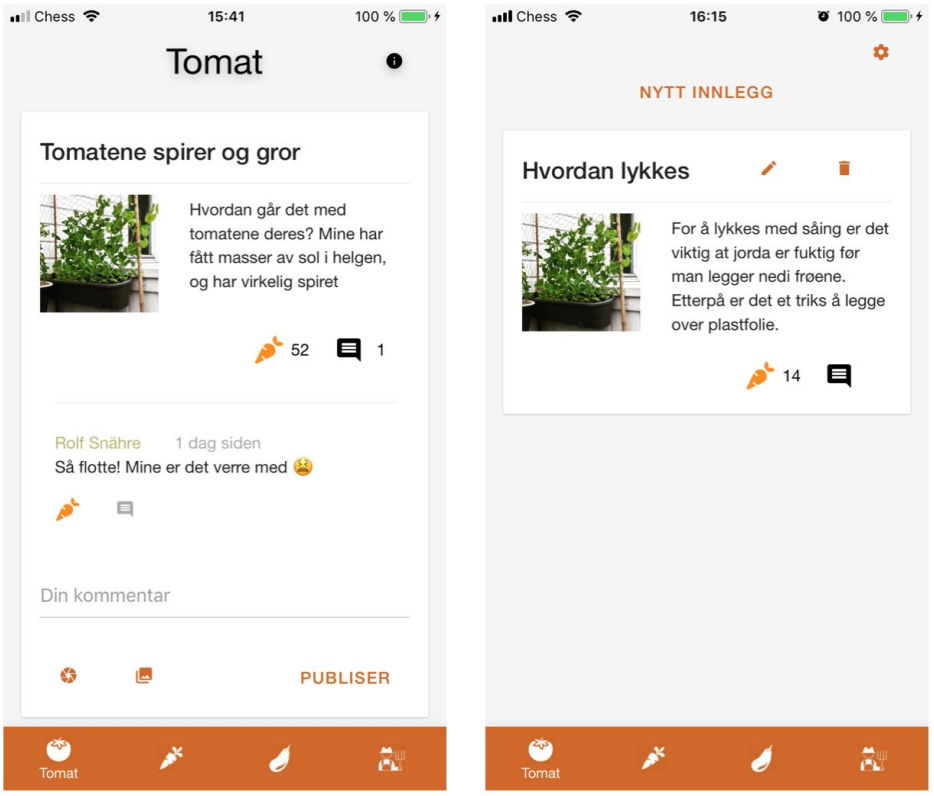
I applikasjonen finnes også en markedsplass, hvor brukerne kan legge ut annonser og se andre brukeres annonser. Her kan brukerne legge ut annonser om at de ønsker dyrkingsareal, at de har ledig dyrkingsareal, at de ønsker å bli med i et dyrkelag/parsellag, eller at et dyrkelag/parsellag har plass til flere medlemmer. Av denne grunn deles annonsene inn i fem kategorier: Areal ønskes, areal tilgjengelig, lag ønskes, medlem ønskes og udefinert. Det har blitt delt inn i disse kategoriene for å gjøre det mulig å filtrere når en bruker ser gjennom markedsplassen, og disse kategoriene virker å dekke de fleste behovene for markedsplassen. Når brukerne legger ut en annonse velger de kategori og legger inn sted, kontaktinformasjon og eventuell annen informasjon.

Til slutt har applikasjonen en profilside hvor brukeren kan redigere og slette annonsene sine. Her kan de også logge seg ut, slette brukeren sin og finne mer informasjon om applikasjonen.

7.1.3 Dyrk med Bybonden

Mobilapplikasjonen Dyrk med Bybonden var Bybondens hjertebarn, og det var høyt ønskelig at også denne skulle bli ferdigstilt. For å kunne tilby en løsning, ble fokuset å lage en helt grunnleggende applikasjon.

Alt i alt består løsningen av tre kanaler, hvor det gjennom en sesong vil være en fastsatt vekst å følge. Alle vekster vil være av den spiselige sorten og relativt enkle for uerfarne å mestre.



Figur 7.3: Skjermdump fra Dyrk med Bybonden, henholdsvis versjonen som skal distribueres og administrator-versjonen

I utgangspunktet var planen å tilby Bybonden en form for verifisering, hvor det deretter ble gitt tilgang til muligheter som å legge ut nye innlegg og slette kommentarer. Det ble derimot bestemt at det ikke skulle benyttes autentisering i denne applikasjonen, og beste løsning var derfor å utvikle en egen mobilapplikasjon med administratormuligheter. Denne applikasjonen skal ikke distribueres, kun bygges på Bybondens mobil. Figuren over illustrerer forskjellen på de to applikasjonene. Alt innhold er det samme, sett bort i fra administrastor-versjonens mulighet til å slette/redigere innlegg og kommentar, publisere nye innlegg og redigere kanaler.

Fra brukeren sin side vil det være mulig å abonnere på de ulike kanalene, hvor det mottas pushvarsler ved nye innlegg. Utover dette kan innlegg kommenteres, med mulighet for å legge ved bilde, samt å svare på kommentarer. Samtidig er det en "like"-funksjon på alle innlegg, kommentarer og svar.

Gjennom en sesong med dyrking vil Bybonden legge ut veiledninger om hvordan plantingen, og videre behandling, av vekstene skal utføres. Tanken er at applikasjonen skal være en interaktiv og lite høytidelig plattform, hvor brukere kan spørre etter råd, briljere med bilder av sine vekster og tilegne seg basiskunnskaper ved selvdyrking.

Utover de tre vekstkanalene inneholder applikasjonen en profil-side, hvor brukere kan oppgi/oppdatere navn, samt lese informasjon om applikasjonen og Bybondeprosjektet. Informasjonen som ligger på denne siden er mulig å endre på i administrator-applikasjonen.

7.2 Kritiske beslutninger

Beslutningen om å bruke en nettsidebygger til å utvikle nettsiden var en av de mest kritiske i prosjektet. Dette sparte mye verdifull tid, som ble viktig for å få tid til å utvikle mobilapplikasjonene. Om det hadde blitt valgt å lage nettsiden fra bunnen av ville det blitt mer tidkrevende og det hadde sannsynligvis blitt nødvendig å droppe den ene mobilapplikasjonen. Dette ville også ført til at nettsiden ville vært mye vanskeligere å drifte og vedlikeholde for Bybonden etter prosjektets slutt. At det var Weebly som ble benyttet var nok ikke like kritisk, da det finnes flere lignende nettsidebyggere som sannsynligvis hadde gitt et lignende resultat.

Et annet kritisk valg var valget om å bruke React Native til utvikling av mobilapplikasjonene. Som nevnt tidligere var det viktig for oppdragsgiver at applikasjonene skulle være tilgjengelig for både iOS og Android, men om applikasjonene skulle utvikles i til begge plattformene hver for seg. Det ble derfor nødvendig å finne en måte å utvikle applikasjonene til begge plattformene fra samme koden. Her var det flere mulige løsninger, og at det var akkurat React Native som ble valgt var nok ikke like kritisk som valget om å ikke utvikle applikasjonene med egen kode for iOS og egen kode for Android.

Valget om å bruke skytjenester var også en viktig beslutning i prosjektet. At det var AWS som ble valgt er ikke like kritisk, da det finnes flere lignende løsninger, blant annet Microsoft Azure. Valget om å bruke en slik skyløsning for tjenersiden førte til at all tjenerfunksjonalitet ble samlet på ett sted og fungerer godt sammen. Med den forventede bruken av applikasjonene blir dette også en billig løsning, som samtidig skalerer godt dersom bruken skulle øke.

Valg av hvilken funksjonalitet som skulle prioriteres i applikasjonene ble også en avgjørende beslutning for sluttproduktet i prosjektet. Det var en del funksjonalitet i applikasjonen Dyrk med Bybonden som måtte nedprioriteres, til fordel for funksjonalitet i Dyrkingsareal. I diskusjon med oppdragsgiver ble det enighet om at denne funksjonaliteten var den viktigste, og ble derfor prioritert.

At fokuset lå på å ferdigstille Dyrkingsareal ble avgjørende i prosessen. På denne måten ble det først etter at hoveddelen av arbeidet var ferdigstilt, satt i gang med utviklingen av Dyrk med Bybonden. Hadde de to applikasjonen på den annen side blitt utviklet parallelt, ville muligens Dyrk med Bybonden inneholdt flere funksjonaliteter, mens dette hadde gitt en negativ innvirkning på Dyrkingsareals kvalitet.

7.3 Alternative tilnærminger

Når det sees tilbake på mulige tilnærminger og hvilket alternativ som vant fram, er oppfatningen at det var den beste løsningen å utvikle to separate løsninger. Ved å dele de to applikasjonene ble det ryddigere og oversiktlig, samt et tydelig skille mellom målgruppene.

Det kan derimot diskuteres hvorvidt alternativet hvor Dyrkingsareal hadde vært i form av en webapplikasjon, ville vært en bedre løsning. Denne tilnærmingen ble utelukket etter oppdragsgiver ønske, samtidig som at det var foretrukket å utvikle en mobilapplikasjon. I ettertid er det derimot tvil rundt om at dette faktisk var den beste tilnærmingen. Om applikasjonen hadde vært i form av en webapplikasjon ville det først og fremst vært mindre arbeid for brukere å ta del i den. Det kan diskuteres hvor mye arbeid som kreves for å laste ned en mobilapplikasjon, men realiteten er at det krever plass, noe som ikke alltid er like lett å sørge for. En annen side er muligheten for å kunne benytte applikasjonen ved hjelp av en PC. Det er en god del som er av den oppfatning at dette er mer oversiktlig og enklere å håndtere. Samtidig kunne applikasjonen blitt inkorporert i Bybondens hjemmeside, og dermed forenklet markedsføringsprosessen.

For å ha tilgang til å publisere applikasjoner i Google Play og App Store kreves en avgift, på henholdsvis 25 og 100 dollar. I tillegg til dette viste det seg at å prosessen for å bli Apple-utvikler var nokså omfattende. Dette innebar verifisering i form av telefonsamtaler, utredning av juridisk ansvarlige, organisasjonen og bruksområde, samt anskaffelse av internasjonale organisasjonsnummer. I og med at det uansett skulle utvikles en annen mobilapplikasjon måtte tilgangen uansett blitt anskaffet. Det er allikevel mulig at etter utviklingen av nettstedet ville også den andre instansen blitt løst på denne måten.

8 Konklusjon og videre arbeid

8.1 Initielle mål

Som et overordnet mål hadde prosjektet et ønske om å øke interessen og tilrettelegging for det urbane landbruket i Bergen. Dette skulle gjennomføres ved å utvikle effektive og brukervennlige digitale verktøy. Konkret skulle disse verktøyene inkludere tre instanser med følgende spesifikasjoner:

Kartløsning/Markedsplass

- En kartløsning med mulighet for å finne eiere av jord i Bergen, samt kontaktinformasjon. Kravet besto etterhvert i å implementere løsningen ved hjelp av tilgang til matrikkelen.
- Kartløsningen skulle omfatte mulighet for brukere å legge til informasjon om områder
- En markedsplass hvor brukere kan finne hverandre ut i fra behov: Trenger område/lag eller ledig område/lag

Nettside

- En nettside hvor all relevant informasjon om Bybonden i Bergen skulle samles
- Mulighet for påmelding til arrangementer i regi av Bybonden
- Betalingsmulighet for arrangementer som krever det
- Oversikt over kommende arrangementer

Dyrkingshjelp

- En mobilapplikasjon hvor Bybonden skulle dele tips og kunnskap gjennom en sesong
- Mulighet for opplasting av bilder, både fra Bybonden og brukere
- Brukere skal motta pushvarsler ved gitte hendelser

8.2 Måloppnåelser

Hvorvidt de utviklede verktøyene vil bidra til inspirasjon og økt kunnskap rundt dyrking blant innbyggere i Bergen, vil det ikke kunne konkluderes med i skrivende stund. Applikasjonene blir lansert en god stund etter oppstart av dyrkesesongen, og det blir ikke før neste vår at den eventuelle effekten vil kunne oppleves. Samtidig gjenstår det en god del arbeid rundt markedsføring og informasjonsformidling, som oppdragsgiver må håndtere, før den tid. Det som derimot kan måles per i dag er i hvilken grad spesifikasjonene gitt de ulike delene er oppfylt.

Når det kommer til delen med kartløsning og markedsplass var dette fra start av høyeste prioritet. Slik det framgår i 7.1 - "Resultat" foreligger det en mobilapplikasjon som består av en vel utarbeidet kartimplementasjon, løst ved hjelp av Topografisk Norgeskart og Google Maps. Løsningen gjør det enkelt for brukere å finne informasjon om området de befinner seg på. En viktig del var samtidig at det i tillegg til informasjon om området skulle være lett tilgjengelig kontaktinformasjon. Ved hjelp av tilgangen til matrikkelen er informasjonen tilgjengelig for hvert registrerte område, og det er samtidig mulighet å utføre søk på 1881.no via applikasjonen.

Ved bruk av UGC-prinsippet ble kravet rundt mulighet for brukere å legge ut informasjon om områder løst. På denne måten kan alle brukere, så langt de er autentisert, opprette og endre informasjon om et område. Hva markedsplassen angår er dette implementert på en måte som gjør det mulig for alle autentiserte brukere å legge ut annonser. Samtidig er det implementert en sorteringsløsning som enkelt gjør det mulig å finne den formen for annonse det ønskes å se.

Dermed er det rimelig å påstå at målet angående løsningen med en kartimplementasjon og markedsplass er nådd. Både oppdragsgiver og prosjektgruppen opplever at applikasjonen oppfyller de initielle kravene og ønskene, og det er spesielt tilfredshet rundt tilgangen til matrikkelen. Denne delen var som tidligere nevnt den mest kritiske delen av utviklingen og var avgjørende for å få et optimalt resultat.

Nettsiden var den minst tidkrevende delen av prosjektet. På grunn av valget om å ta i bruk en nettsidebygger ble den i tillegg en visuelt bedre løsning, enn om den skulle blitt utviklet fra bunn av. Bybonden var opptatt av at det ikke skulle føre med seg dobbeltarbeid, som kunne oppstått ved at det måtte legges ut informasjon om arrangementer på Facebook og nettsiden, samt Tito. Implementasjonen med Facebook-kalenderen og Tito-modulen har på denne måten ikke ført med seg noe ekstra arbeid fra Kleppes side. Den har derimot forenklet prosessen med arrangementshåndtering. Utover dette har nettsiden blitt en fin og oversiktlig samling av tjenester som Bybonden tilbyr.

Et viktig moment for Bybonden var at nettsiden skulle være enkel i drift, og uproblematisk for henne å håndtere uten teknisk støtte. Dette aspektet ble det også en god løsning på ved hjelp av Weebly. Nettsidebyggeren tilbyr en enkel og forståelig løsning, også for personer uten vide tekniske kunnskaper. Weebly gir mulighet for å enkelt bytte innhold og omrokkering av komponenter.

Delen som var av laveste prioritet under prosjektet var den som omfattet deling av informasjon om dyrking fra Bybonden. Dermed er det stor tilfredshet at denne delen også ble ferdigstilt til prosjektslutt, til tross for en nokså enkel versjon. Mobilapplikasjonen Dyrk med Bybonden gjør det mulig for Kleppe å oppdatere brukere om tre vekster gjennom hver

sesong. Hver oppdatering kan inneholde bilder hentet direkte fra mobiltelefonen, det samme gjelder kommentarer fra brukere. Kravet om pushvarsler ble også oppfylt, og brukere kan velge å motta dette fra hver enkelt kanal.

Ut i fra det initielle målet og kravspesifikasjoner føles det riktig å påstå at samtlige mål har blitt oppfylt. Målet rundt Dyrk med Bybonden inngikk ikke i det opprinnelige prosjektinitiativet, men også disse spesifikasjonene har blitt en del av den endelige løsningen. Oppdragsgiver er fornøyd med de tre løsningene og det er av gruppens egen oppfatning at hvert delmål ble utført etter beste evne.

8.3 Konklusjon

Fokuset på digitalisering av systemer og verktøy vokser stadig. Særlig om det ønskes å appellere til yngre er det så og si et krav at det tilbys en form for digital løsning. For at et tema skal settes på dagsorden er det derfor en klar fordel å tilrettelegge for teknologiske svar på tidligere løsninger. Samtidig med å være praktisk, er dette også regnet som det mest miljøvennlige. Teknologi har til alle tider frigjort bruk av materielle ressurser (Næringslivets Hovedorganisasjon (2018)), som er et av de viktigste momentene verdens industri står overfor.

En av utfordringene under prosjektet var oppdragsgivers mangel på konkretisering. Oppdragsgiver besto av flere instanser og det var tydelig fra start av at det ikke var dannet en enighet om hva ønsket resultat var. Det overordnede målet var på plass, men det eksisterte ikke en klar tanke om hvordan dette skulle løses. I hovedsak kan dette skyldes mangel på teknologisk kunnskap og hvilke muligheter prosjektet hadde. Samtidig var det viktig at produktet skulle være noe det faktisk var behov for, og ønsket var å involvere målgruppen i prosessen. Dette var også en grunn for at funksjonaliteter og brukstilfeller ikke var fastsatt, da ønsket var at brukerne selv skulle identifisere dem.

Til tross for at mangel på konkretisering var en stor utfordring, var det samtidig det mest spennende med prosjektet. Det er svært vanlig at initiativtaker ikke har tydelige spesifikasjoner på forhånd, og derfor noe som må påregnes i arbeidslivet. På denne måten var det ved prosjektstart fremdeles mye som gjensto, også i forhold til ting som ikke omhandlet det tekniske aspektet. Dermed handlet prosessen om mer enn bare utvikling, men om hele konteksten. Ved å ta i bruk den smidige utviklingsmetoden ble det dannet prioriteringer av implementasjon, som ble oppdatert etterhvert som ny teknologi og tilganger kom på banen. Metoden viste seg å være svært nyttig i et prosjekt hvor det på forhånd var uklart hvilke funksjonelle krav som skulle stilles, i tillegg til å være godt forenlig med Design Thinking.

Gjennom arbeidet med prosjektet har det blitt tilegnet tverrfaglige kunnskaper. De mest åpenbare er de som omhandler nye teknologier og rammeverk. Det har fra dag én blitt arbeidet iherdig med utarbeiding for å identifisere de beste teknologiske løsningene, og hvordan de fungerer. Mange timer med nettkurs og tolkning av dokumentasjon har blitt lagt ned for å få en forståelse for hva som faktisk trengs ved utvikling av et system. Mesteparten av verktøyene og rammeverk som er benyttet var nye for samtlige gruppemedlemmer, hvor opplæring dermed krevde sin del av tidsressursene.

Den viktigste lærdommen som det teknologiske arbeidet har medført er verdien av planlegging og forarbeid. Ved oppstart var det stor entusiasme etter å komme i gang med utviklingen og det ble forsøkt utviklet funksjonaliteter i ulike rammeverk. Når det sees tilbake på dette oppleves det som om mange timer kunne blitt spart om det hadde blitt utført et grundigere forarbeid. På den annen siden var prosessen nødvendig for å forstå hvilke teknologier som passet prosjektets behov og hvilke rammeverk som opplevdes mest intuitive og forsørget gode dokumentasjoner. Samtidig har kunnskapen rundt ulike teknologier og rammeverk, til tross for at det ikke ble tatt i bruk, blitt etablert.

Læringsutbyttet gjennom det teknologiske aspektet av prosjektet har vært stort. Det har også blitt endring i oppfatning hva tid angår. Alle deler av et prosjekt krever tid og selv den enkleste ting viste seg å kreve mer enn forutsett. I ettertid vil verdien av tidseffektivisering ikke undervurderes og det er ikke vanskelig å forstå hvorfor planlegging og utarbeiding av tidsplaner blir vektlagt. Det ble opprettet et Gantt-diagram ved oppstart, hvor det etterhvert ble innsett at planen var vel optimistisk. Dette kunne med fordel blitt utarbeidet bedre, samt oppfulgt bedre.

Verdien av egenskaper som tilegnes i et prosjektarbeid er tydelig. I tillegg til å forholde seg til gruppens andre medlemmer, samt oppdragsgiver, har det stadig vært andre interessenter som skulle inkluderes i prosessen. Gjennom prosjektperioden har det vært mye fokus på å opprettholde kontakten med aktuelle brukergrupper. For å oppnå ønsket effekt ble Design Thinking benyttet, noe som førte med seg en utviklingsprosess hvor det var høyt fokus på brukervennlighet og hvilke funksjonaliteter det faktisk var behov for. Dette gjorde prosessen spesielt lærerik og det er en klar fordel å komme tettere på brukere. Ved å ta i bruk denne metoden skapte det samtidig et engasjement og eierskap blant brukergruppene. På denne måten vil det allerede være etablert grupper som vil ta i bruk mobilapplikasjonene ved utgivelse og som kan ta del i den videre markedsføringen.

I tillegg til lærdommer knyttet til teknologi og prosjektarbeid, har det også oppstått en interesse for dyrking og hvilke grønne valg som kan tas i hverdagen. Gjennom oppdragsgiver og andre personer i landbruksmiljøet har det blitt utviklet et annet syn på betydningen av å dyrke selv. Ikke bare fører det til selvforsyning av mat, men er med på å forbedre matkulturen i samfunnet. Dette er også særlig viktig med tanke på en bærekraftig utvikling,

og det er derfor nødvendig at tankesettet spres til barn og unge. Dyrking handler om mer enn bare gulrøtter. Som med andre aktiviteter, handler det samtidig om en fellesskapsfølelse og en tilhørighet. Gjennom å tilrettelegge for et digital nettverk for landbruksmiljøet oppleves det at prosjektet har bidratt til å gjøre Bergen litt grønnere.

8.4 Videre arbeid

Slik det framgår i punkt 5.2 - "Evalueringsresultat", har mobilapplikasjonene utbedringspotensiale. Dette gjelder i første omgang funksjonaliteter og muligheter for mobilapplikasjonene. Underveis i løpet har det dukket opp flere egenskaper applikasjonene kunne inneholdt, både fra oppdragsgiver og andre hold. Prosjektet var i utgangspunktet nokså ambisiøst og skalerbarheten viste seg å være uten ende. Konkret hvilke funksjonaliteter dette innebærer diskuteres i delkapitlene under.

Utover hvilke ekstra funksjonaliteter mobilapplikasjonene kan inneholde, er det nok rimelig å si at utførelsen av den tekniske delen trolig har forbedringspotensiale. Selv om det kan påstås at løsningen er god, er likevel mangelen på kunnskap reell. Samtlige gruppemedlemmer er relativt uerfarne når det kommer til utvikling av systemer, og det kan derfor diskuteres hvorvidt det er de beste valgene som er tatt på dette området. Ved et eventuelt videre arbeid kan det derfor være aktuelt å forbedre ikke-funksjonelle egenskaper.

8.4.1 Dyrk fra Vitenparken

I løpet av prosjektet ble det kjent at det ble utviklet en mobilapplikasjon som lignet noe på applikasjonen Dyrk med Bybonden ved Vitenparken. Vitenparken er et viten- og opplevelsessenter lokalisert på Ås ved NMBU. Vitenparken setter fokus på grønn verdiskapning og bærekraftig utvikling, og skaper grønne kunnskapsopplevelser gjennom blant annet arrangementer, utstillinger og diverse skoletilbud (Vitenparken (u.å.)). Vitenparken startet tidligere i år med utvikling av en mobilapplikasjon for iOS, hvor det skal tilbys hjelp med dyrking på et generelt nivå. Brukeren skal ha mulighet for å oppgi hvilke forhold det tilgjengelige dyrkingsarealet har, som lys, jord, dimensjon osv. Applikasjonen skal kobles opp mot en database hvor det ligger faktabasert informasjon om hvilke tiltak som må tas ved de ulike forholdene.

Det ble opprettet kommunikasjon med Vitenparken for å diskutere et eventuelt samarbeid og mulige synergier ved arbeidet som ble gjort med applikasjonene Dyrk fra Vitenparken og Dyrk med Bybonden i dette prosjektet. Etter samtaler med Vitenparken ble det tydelig at de to applikasjonene ikke kom til å bli så like som først antatt likevel, og det ble konkludert med at det ikke var noe særlig å samarbeide om der.

Vitenparken var mer interessert i applikasjonen Dyrkingsareal og kunne gjerne se for seg at det skulle bli en del av applikasjonene deres. Det endte med at applikasjonen ble utviklet som planlagt, med mulighet for at Vitenparken kunne ta over applikasjonen etter at prosjektet er avsluttet, og få ansvar for å drifte den videre. Ved prosjektslutt var det enda ikke kommet til enighet om en eventuell overtakelse.

8.4.2 Dyrkingsareal

Som nevnt i punkt 5.2.2 var det ønskelig med funksjonalitet for administrering av parsellag. Dette innebærer blant annet ventelister for medlemskap, inn- og utmelding og deling av relevant informasjon. Dette er noe som gjerne skulle blitt implementert, men ble ikke prioritert i denne omgang.

En annen funksjonalitet som kan vurderes å legges til ved en senere anledning er at en eier av et område skal kunne overstyre informasjonen som legges inn om områder. Dette ville gjort at andre ikke kunne lagt inn feilaktig informasjon om et område, enten det hadde vært forsettlig eller ikke. Det hadde samtidig lagt til rette for reservasjon mot å bli kontaktet gjennom applikasjonen.

En tredje funksjonalitet som gjerne kunne vært en del av applikasjonen er at annonsene kan kobles opp mot kartet. Dette kunne fungert begge veier, altså at brukeren ble sendt til det tilknyttede området i kartet ved trykk på annonsen, samt at annonsen vises med en markør på kartet når kartvisningen er oppe.

Utover hvilke ekstra funksjonaliteter som applikasjonen kan utvides med, kan kartløsningen med tilgang til matrikkelen benyttes i andre bransjer og sammenhenger. Dyrkingsareal kan dermed benyttes til inspirasjon og videreutvikling i andre prosjekter for eksempel i byggebransjen, til landmåling og identifisering av tomteareal.

8.4.3 Dyrk med Bybonden

Som presisert ved flere anledninger ble det identifisert en del brukstilfeller både før og underveis i prosjektet angående Dyrk med Bybonden. Det ble imidlertid begrenset til de ytterst nødvendige under utviklingen, grunnet mangel på tid. Om det ved en anledning skal videreutvikles er det dermed mange idéer og påtenkte funksjonaliteter å ta tak i. Under følger en liste med de identifiserte utvidelsene:

<u>Andre enn Bybonden kan opprette kanaler med vekster</u>
Om applikasjonen skal appellere til andre enn uerfarne dyrkere bør det være mulig for andre, i form av grupper, bedrifter, lag og enkeltpersoner å kunne publisere egne vekster. På denne måten blir det samtidig skapt en form for et sosialt media, noe som igjen vil øke interessen blant andre.

Mer dyrkingsinformasjon

Slik funksjonaliteten til applikasjonen er per i dag vises det kun informasjon om pågående vekster. Det var imidlertid et forslag fra Kleppe å i tillegg inkludere generell informasjon om flere vekster. Dette ble forestilt løst ved hjelp av en sidemeny hvor det kunne navigeres til de ulike informasjonssidene.

Integrasjon av arrangementer

Fra Bybondens side var det et ønske om å få integrert arrangementer, i regi av henne, i mobilapplikasjonen. Ønsket var at at dette skulle fungere på samme måte som i nettsiden, hvor det kan betales via Tito. Dette elementet hadde vært naturlig å hatt som en del av sidemenyen.

Forsinkede pushvarsler

Slik applikasjonen er formet per i dag vil det ikke være optimalt om brukere tar del i prosjekter etter oppstart. Det vil kun mottas pushvarsler som oppstår direkte, og det finnes ingen løsning for å "ligge etter". Ideelt bør det være mulig for brukere å melde seg på et prosjekt etter ønske, for deretter å motta pushvarsler basert på tidsintervaller. På denne måten ville brukere også kunne følge prosjekter fra tidligere sesonger. Dette ville forårsake at interaktiviteten mangler, men allikevel være greit for dem som kun er ute etter konkrete veiledninger.

Utvidede innleggsfunksjonaliteter

Innleggene som skal legges ut av Bybonden under hver kanal er svært grunnleggende. Det kan skrives tekst, overskrift og legges til et bilde fra mobilkameraet. Her er det mulighet for utvidelser som for eksempel å legge ut videoer og opprette meningsmålinger.

8.4.4 Nettside

I utgangspunktet hadde oppdragsgiver planer om å ha en side med informasjon om dyrkingsareal i Bergen på nettsiden, men da det aldri ble klart hva innholdet her eventuelt skulle være ble ikke dette en del av sluttproduktet. Etter lansering av nettsiden ble det også diskutert om det skulle vært en egen side som presenterer hvilke tjenester Bybonden tilbyr, men her heller var det ikke klart hva innholdet på siden skulle være og ble dermed ikke implementert i løpet av prosjektet. Når mobilapplikasjonene har blitt lansert på Google Play og App Store, vil det være naturlig å informere og linke til disse på nettsiden. Siden Weebly har blitt brukt for å lage nettsiden er disse tingene noe som i etterkant kan legges til, uten store vanskeligheter.

9 Referanser

AWS(u.å.) *Amazon API Gateway pricing* [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://aws.amazon.com/api-gateway/pricing/> [Lest 29. mai 2019]

AWS(u.å.) *Amazon Cognito pricing* [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://aws.amazon.com/cognito/pricing/> [Lest 29. mai 2019]

AWS(u.å.) *Amazon DynamoDB* [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://aws.amazon.com/dynamodb/> [Lest 02.juni 2019]

AWS(u.å.) *Amazon DynamoDB pricing* [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://aws.amazon.com/dynamodb/pricing/> [Lest 29. mai 2019]

AWS(u.å.) *Amazon EC2 pricing* [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://aws.amazon.com/ec2/pricing/on-demand/> [Lest 29. mai 2019]

AWS(u.å.) *Amazon Lambda pricing* [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://aws.amazon.com/lambda/pricing/> [Lest 29. mai 2019]

AWS(u.å.) AWS Toolkit for Eclipse [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://aws.amazon.com/eclipse/> [Lest 31. mai 2019]

AWS(u.å.) *Cloud computing with AWS* [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://aws.amazon.com/what-is-aws/> [Lest 29. mai 2019]

AWS(u.å.) *noSQL Design for DynamoDB* [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/bp-general-nosql-design.html#bp-general-nosql-design-vs-relational> [Lest 02.juni 2019]

AWS Documentation (u.å.) *What Is Amazon Cognito?* [Internett]. Seattle: Amazon. Tilgjengelig fra: <https://docs.aws.amazon.com/cognito/latest/developerguide/what-is-amazon-cognito.html> [Lest 20. mai 2019]

Codebrahma Software Development Service (2018) *Awesome React Native: Building Android App with JavaScript* [Internett]. Tilgjengelig fra:

<http://codebrahma.blogspot.com/2018/07/awesome-react-native-building-android.html> [Lest 03. mai 2019]

Flux (u.å.) *flux-concepts* [Internett]. California: Facebook. Tilgjengelig fra: <https://github.com/facebook/flux/tree/master/examples/flux-concepts>[Lest 20. mai 2019]

Flux (u.å.) *In Depth Overview* [Internett]. California: Facebook. Tilgjengelig fra: <https://facebook.github.io/flux/docs/in-depth-overview.html#content>[Lest 20. mai 2019]

Fylkesmannen i Vestland(u.å.) *Landbruk og mat* [Internett]. Bergen: Fylkesmannen i Vestland. Tilgjengelig fra: <https://www.fylkesmannen.no/nb/vestland/landbruk-og-mat/> [Lest 29.03.19]

Gartner(2017) *Gartner Says Worldwide Sales of Smartphones Grew 7 Percent in the Fourth Quarter of 2016* [Internett]. Stamford: Gartner. Tilgjengelig fra: <https://www.gartner.com/en/newsroom/press-releases/2017-02-15-gartner-says-worldwide-sales-of-smartphones-grew-7-percent-in-the-fourth-quarter-of-2016> [Lest 27.mai 2019]

Hordaland Bondelag (2018) Bybonden i Bergen [Internett]. Kokstad : Hordaland Bondelag. Tilgjengelig fra <https://www.bondelaget.no/bybonden/category7730.html> [Lest 21. mai 2019]

Hordaland Bondelag (u.å.) *Fagorganisasjon for bønder* [Internett]. Kokstad: Hordaland Bondelag. Tilgjengelig fra: <https://www.bondelaget.no/om-hordaland-bondelag/category7710.html> [Lest 28.juni 2019]

Høiseth, Y.(2016) *Smidig vs. fossefall: fordeler og ulemper.* [Internett]. Skien: eZ. Tilgjengelig fra: <https://ez.no/no/Blogg/Smidig-vs.-fossefall-fordeler-og-ulemper> [Lest 15. mai 2019]

Intersoft Consulting (2016) *General Data Protection Regulation* [Internett]. Hamburg: Intersoft Consulting. Tilgjengelig fra: <https://gdpr-info.eu/> [Lest 27.mai 2019]

JSX (u.å.) *Draft: JSX Specification* [Internett]. California: Facebook. Tilgjengelig fra: <https://facebook.github.io/jsx/> [Lest 31. mai 2019]

Kartverket(u.å.) *Finn informasjon om eiendom* [Internett]. Hønefoss: Kartverket. Tilgjengelig fra: <https://seeiendom.kartverket.no> [Lest 21.mai 2019]

Kartverket (2018) *Hva er matrikkelen?* [Internett]. Hønefoss: Kartverket. Tilgjengelig fra:

<https://www.kartverket.no/eiendom/eiendomsinformasjon/matrikkelen/> [Lest 21. mai 2019]

Lithun, G.(2016) *Stikk fingeren i jorda*. ***Bærekraftige liv***, s. 39.

Næringslivets Hovedorganisasjon (2018) *Digitalisering* [Internett]. Majorstuen: NHO. Tilgjengelig fra: <https://www.nho.no/publikasjoner/p/naringslivets-perspektivmelding/digitalisering/> [Lest 21. mai 2019]

React Native (u.å.) *Async Storage* [Internett]. California: Facebook. Tilgjengelig fra: <https://github.com/react-native-community/react-native-async-storage> [Lest 21. mai 2019]

React Native (u.å) *Learn the Basics* [Internett]. California: Facebook. Tilgjengelig fra: <https://facebook.github.io/react-native/docs/tutorial> [Lest 03. mai 2019]

React Native Paper (2019) *Making your React Native apps look and feel native* [Internett]. Wrocław: Callstack. Tilgjengelig fra: <https://reactnativepaper.com> [Lest 19.mai 2019]

React Native (u.å.) *React Native, Build native mobile apps using JavaScript and React* [Internett]. California: Facebook. Tilgjengelig fra: <https://facebook.github.io/react-native/>[Lest 01.04.19]

React Native (u.å.) *State* [Internett]. California: Facebook. Tilgjengelig fra: <https://facebook.github.io/react-native/docs/state> [Lest 26. mai 2019]

Revold, M.K. (2017) *Stort flertall eier boligen.*[Internett]. Oslo: Statistisk sentralbyrå. Tilgjengelig fra <https://www.ssb.no/bygg-bolig-og-eiendom/artikler-og-publikasjoner/stort-flertall-eier-boligen> [Lest 21. mai 2019]

Rocheleau, J. (2016) *What is Google's Material Design?* [Internett]. Melbourne: Envato. Tilgjengelig fra: <https://envato.com/blog/introduction-material-design/> [Lest 01.juni 2019]

RUAF Foundation (u.å.) *Urban agriculture: what and why?* [Internett]. Leusden : RUAF. Tilgjengelig fra: <https://www.ruaf.org/urban-agriculture-what-and-why> [Lest 21. mai 2019]

Smart Innovation Norway (u.å) *Design Thinking: Prosess og samarbeidsmetode* [Internett]. Halden: Smart Innovation Norway. Tilgjengelig fra: <https://www.smartinnovationnorway.com/design-og-visualisering/design-thinking-prosess-og-samarbeidsmetode/>[Lest 31. juni 2019]

SuperOffice (2019) *Hva er GDPR, og hva betyr det for din bedrift?* [Internett] Oslo: SuperOffice. Tilgjengelig fra: <https://www.superoffice.no/ressurser/artikler/hva-er-gdpr/> [Lest 27.mai 2019]

Vitenparken (u.å.) *Om oss - Vitenparken* [Internett]. Ås: Vitenparken. Tilgjengelig fra: <https://vitenparken.no/contact/om-oss/> [Lest 31.mai 2019]

Wikipedia, bidragsytere.(2019) *Apache Cordova* [Internett]. Wikipedia, The Free Encyclopedia. Tilgjengelig fra: <https://en.wikipedia.org/wiki/Apache_Cordova> [Lest 01.04.19]

10 Appendix

10.1 Akronymer og ordliste

Akronymer

ADB	Android Debug Bridge
API	Application Programming Interface
AWS	Amazon Web Services
FN	Forente Nasjoner
FCM	Firebase Cloud Messaging
GDPR	General Data Protection Regulation
GIS	Geografisk Informasjonssytem
GSI	Global Sekundærindeks
HTML	Hyper Text Markup Language
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Enviroment
JSX	JavaScript XML
MAU	Monthly Active Users
NMBU	Norges miljø- og biovitenskapelige universitet
MVC	Model-view-controller
REST	Representational State Transfer
SNS	Simple Notification Service
UGC/UCC	User-generated content / User-created content
UML	Unified Modeling Language
URL	Uniform Resource Locatorre
XML	Extensible Markup Language

Ordliste

Amazon Web Services	Amazon sin plattform for skytjenester, leverer over 165 tjenester globalt. (AWS (u.å))
Android Studio	IDE for utvikling av applikasjoner for Android.
Angular	Rammerverk for utvikling av mobil- og webapplikasjoner.
App Store	Distibusjonsplattform for applikasjoner til iOS-enheter.
Arealinnehaver	En person som eier et område, som regel dyrkbart i denne konteksten.
Async Storage	Et enkelt, asynkront nøkkel-verdi-lagringssystem som er globalt for applikasjonen.
Cordova	Rammeverk fra Adobe for utvikling av kryssplattformapplikasjoner.
Design Thinking	En iterativ utviklingsprosess som fokuserer på å forstå brukeren og dens behov.
Det grønne skiftet	Uttrykk for miljøvennlig forandring.
ECMAScript	Scriptspråkspesifikasjon som JavaScript følger.
Emulator	Et program som kan kjøre et annet program på en maskinvare/programvare programmet ellers ikke kunne blitt kjørt.
Gantt-diagram	Et diagram for å planlegge aktiviteter i forhold til tid, mye brukt i prosjektledelse.
GDPR	Personvernforordning som skal styrke personvernet ved behandling av personopplysninger i EU. Trådte i kraft 25. mai 2018 og kan føre til store bøter dersom reglene brytes.
Geonorge	Nasjonalt nettsted for kartdata og annen geografisk stedfestet informasjon.
Git	Versjonkontrollsystem. Muliggjør at flere kan arbeide på samme prosjekt samtidig.
Github Desktop	Programvare for bruk av Git.
Google Play	Distibusjonsplattform for applikasjoner på Android-enheter.

Intellisense	Koderedigeringsfunksjon for blant annet autofullføring og stavekontroll.
Kanbantavle	Verktøy for å holde oversikt over og effektivisere utviklingsprosessen.
Matrikkelen	"Matrikkelen er Norges offisielle register over fast eiendom, herunder bygninger, boliger og adresser." (Kartverket (2018))
Microsoft Azure	Microsofts nettskyplattform.
React Native Paper	En samling med komponenter som følger prinsippene for Material Design.
RocksDB	En nøkkel-verdi -basert database som er innebygd i en applikasjon, uten noe klient/tjenerforhold.
SQLite	En relasjonsdatabase som er innebygd i en applikasjon, uten noe klient/tjenerforhold.
Teig	Areal som er omsluttet areal tilhørende andre eiendommer.
Tito	En nettjeneste for organisering og håndtering av arrangementer og billetter.
UGC / UCC	Et fellesuttrykk for all form av innhold, som for eksempel tekst, bilder eller videoer, som har blitt publisert av brukere på nettplattformer eller -applikasjoner. Eksempel på dette er sosiale media som Facebook, Instagram og Twitter, wikisider som Wikipedia og nettforum som Reddit.
Visual Studio Code	IDE for utvikling med blant annet JavaScript
Xamarin	Rammeverk fra Microsoft for utvikling av kryssplattformapplikasjoner.
Xcode	IDE for utvikling av applikasjoner for iOS.

10.2 Risikoliste

Risiko	Sannsynlighet 1-5	Alvorlighetsgrad 1-5
1. Ikke få tilgang til matrikkelen	4	5
2. Oppgaver tar lenger tid enn planlagt	4	3
3. Problemer med tilkobling til cloud-server	2	4
4. Utvikling av lignende løsninger	4	2
5. Sykdom	2	3

1. I starten av prosjektet var det usikkerhet rundt om det var mulig å få tilgang til informasjonen i matrikkelen. Dette var viktig, fordi det ikke finnes andre måter å få tak i denne informasjonen på, og manglende tilgang ville fått store konsekvenser for sluttproduktet.

   Dersom tilgang til matrikkelen ikke hadde blitt gitt, ville alternativet være å tilby informasjon om hvem som bor på eiendommene, i steden for eierne av eiendommene. Siden rundt 75% av nordmenn eier egen bolig vil dette i mange tilfeller bli det samme, men ikke alltid, spesielt ikke i byområder (Statistisk sentralbyrå (2017)). Brukerne av applikasjonen måtte da hatt muligheten til å kunne endre denne informasjonen, slik at den blir riktig. Heldigvis ble ikke dette et problem, da tilgang til matrikkelen ble gitt.

2. Grunnet manglende erfaring med slike prosjekter, var det en risiko for at oppgavene kunne ta lengre tid enn det som var planlagt. Dette var også en risiko på grunn av at samtlige gruppemedlemmer hadde manglende kjennskap til teknologien og rammeverkene som ble brukt. Dette kunne ført til uforutsette problemer, som igjen kunne ført til at prosjektet ikke ble fullført innenfor tidsfristen og at sluttproduktet ikke ble ferdigstilt i tide.

   For å håndtere dette ble det valgt å fokusere på en hovedoppgave om gangen. På denne måten ble det sørget for at om det ikke skulle bli tid til å ferdigstille alle tre hovedoppgavene, så ville det i hvert fall være tid til å fullføre en eller to ferdige produkter, heller enn tre halvferdige. Nettsiden ble laget først, fordi dette var en rimelig enkel og lite tidkrevende oppgave, men samtidig viktig for oppdragsgiver. I tillegg ble det satt av tid i starten av prosjektet til å bli kjent med de nye teknologiene og rammeverkene, for å redusere antall uforutsette problemer i løpet av prosjektet.

3. En annen risiko var eventuelle problemer med å koble klientsiden opp til tjenersiden. Om dette skulle blitt et problem, kunne det krevd mye tid å ordne opp i, og ville gitt mindre tid til andre oppgaver.

   Dette ble håndtert ved å få kommunikasjonen mellom klientsiden og tjenersiden opp så tidlig som mulig, slik at det på et tidlig tidspunkt var klart at dette fungerer, og det var tid til å fikse det dersom det skulle oppstå problemer.

4. Tidlig i prosjektet ble det gjort kjent at Vitenparken skulle utvikle mobilapplikasjonen Dyrk som lignet noe på Dyrk med Bybonden, som skulle utvikles i dette prosjektet. Dersom applikasjonene ble veldig like ville verdiskapingen ved utviklingen av Dyrk med Bybonden bli redusert.

   For å unngå at applikasjonene skulle bli for like ble det avholdt møter med Vitenparken og det ble klart hvordan applikasjonene skulle skille seg fra hverandre. Det ble klart at applikasjonen Dyrk skulle være mer som et informasjonssenter om dyrking av ulike vekster, mens Dyrk med Bybonden skulle være en mer personlig applikasjon, hvor det er mulig å følge Bybonden sin dyrking og interagere med henne og andre brukere.

5. Sykdom i prosjektgruppen ville ført til mangel på arbeidskraft og gitt mindre tid til å gjennomføre de planlagte oppgavene.

   Ved sykdom innad i gruppen ville de resterende medlemmene måtte ta på seg en større arbeidsmengde for en periode. Ved langvarig sykdom måtte eventuelt målene for prosjektet justeres i forhold til den tilgjengelige arbeidskraften, og det ville blitt nødvendig å prioritere hvilken funksjonalitet som var viktigst.

10.3 Gantt-diagram

Uke	11	12	13	14	15	17	18	19	20	21	22	23	24
Kartlegging av oppgave													
Prototyper													
Arkitektur													
Statusmøte m/ veileder													
Utvikling av nettside													
Forprosjektrapport													
Forprosjektpresentasjon													
Dyrkingsareal kartfunksjonalitet													
Dyrkingsareal brukergrensesnitt													
Dyrkingsareal tjenerdel													
Statusrapport													
Dyrk med Bybonden brukergrensesnitt													
Dyrk med Bybonden tjenerdel													
Utkast til rapport													
EXPO-poster													
Sluttrapport													
Presentasjon													
EXPO													