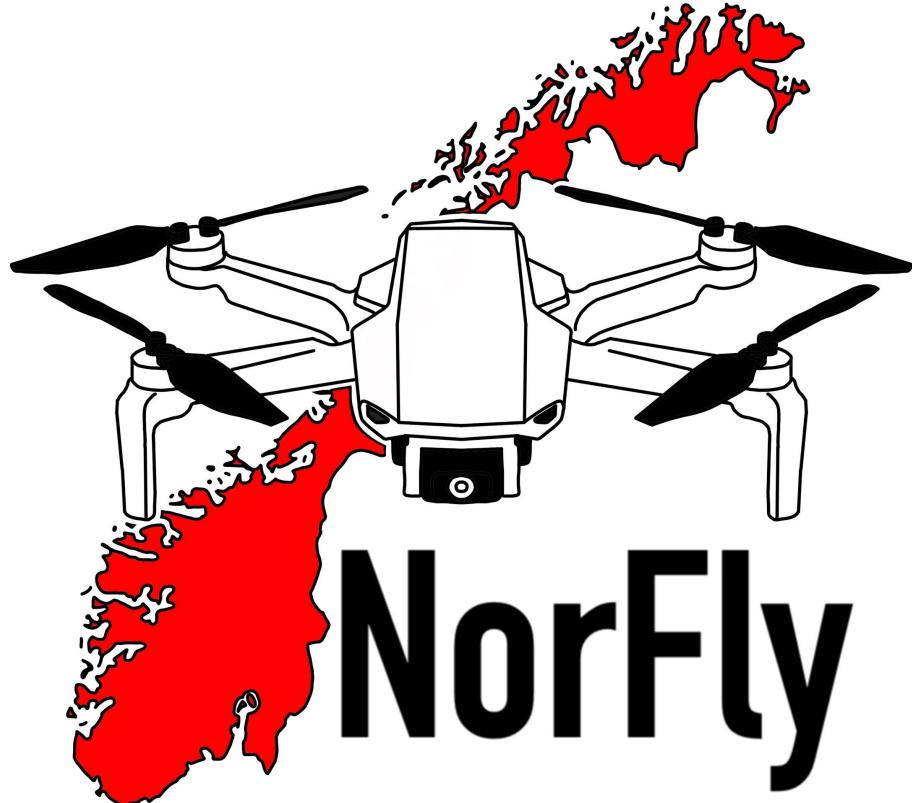


# *Droner og luftrom*

Gruppe 40



# NorFly

Aksel Fagernæs (akselkf)

Boye Molteberg (boyemm)

Christian Lande (chrlan)

Filip Felberg (filipf)

Håvard Nybråten (haavanyb)

Luis Gonzalez (luisdg)

Veileder: Vetle Utvik

<b>Introduksjon</b>	<b>4</b>
<b>Teamet</b>	<b>5</b>
<b>2 Brukerdokumentasjon</b>	<b>7</b>
<b>2.1 Kart</b>	<b>7</b>
<b>2.2 Væroppsett</b>	<b>8</b>
<b>2.3 Søkefunksjon</b>	<b>8</b>
<b>2.4 Innstillinger</b>	<b>9</b>
<b>2.5 Trykkbare ikoner</b>	<b>10</b>
<b>2.5.1 Tilbakestill posisjon</b>	<b>10</b>
<b>2.5.2 Forbudt sone grunnet flyplass</b>	<b>10</b>
<b>3 Kravspesifikasjon</b>	<b>11</b>
<b>3.1 Funksjonelle krav</b>	<b>11</b>
<b>3.2 Ikke-funksjonelle krav</b>	<b>12</b>
<b>3.3 Modellering</b>	<b>13</b>
<b>3.3.1 Use-case diagram</b>	<b>13</b>
<b>3.3.2 Klassediagram</b>	<b>14</b>
3.3.2.1 Klassediagram for: se egen posisjon på kart	14
3.3.2.1 Klassediagram for: sjekke værforhold	15
<b>3.3.3 Sekvensdiagrammer</b>	<b>16</b>
3.3.3.1 Sekvensdiagram for: se egen posisjon på kart	16
3.3.3.2 Sekvensdiagram for: sjekke værforhold	17
3.3.3.3 Sekvensdiagram for: sjekke om det finnes en flyplass	18
<b>3.4 User stories</b>	<b>19</b>
<b>4 Produktdokumentasjon</b>	<b>19</b>
<b>4.1 API</b>	<b>19</b>
<b>4.2 Brukerens lokasjon</b>	<b>21</b>
<b>4.3 Kart</b>	<b>22</b>
<b>4.4 Ulovlige soner</b>	<b>23</b>
<b>4.4.1 Flyplasser</b>	<b>23</b>
<b>4.4.2 NSM sine forbudssoner</b>	<b>24</b>
<b>4.5 Væroppsett</b>	<b>25</b>
<b>4.6 Søkefunksjon</b>	<b>26</b>
<b>4.7 Innstillinger</b>	<b>27</b>
<b>4.8 Trykkbare ikoner</b>	<b>27</b>
<b>5 Testdokumentasjon</b>	<b>28</b>
<b>5.1 Integrasjonstesting</b>	<b>28</b>

<b>5.2 Enhetstesting</b>	<b>29</b>
<b>6 Prosessdokumentasjonen</b>	<b>30</b>
<b>6.1 Informasjonsinnhenting</b>	<b>32</b>
<b>6.1.1 Ekspertvurdering av prototyper</b>	<b>33</b>
<b>6.1.2 Brukertesting av app</b>	<b>34</b>
<b>6.2 Prototyper</b>	<b>35</b>
<b>6.2.1 Prototype 1</b>	<b>36</b>
<b>6.2.2 Prototype 2</b>	<b>37</b>
<b>6.2.3 Prototype 3</b>	<b>39</b>
<b>6.2.4 Prototype 4</b>	<b>41</b>
<b>6.2.5 Prototype 5</b>	<b>42</b>
<b>6.3 Værinformasjon</b>	<b>43</b>
<b>7 Universell utforming</b>	<b>46</b>
<b>8 Refleksjon</b>	<b>47</b>
<b>9 Kilder/Vedlegg</b>	<b>50</b>
<b>Kilder:</b>	<b>50</b>
<b>Vedlegg:</b>	<b>50</b>

# 1 Introduksjon

Denne rapporten er en del av utviklingsprosjektet i faget *IN2000 - Software Engineering og prosjektarbeid*. Fagets vurdering består av to deler, et gruppeprosjekt og en skriftlig eksamen mot slutten av semesteret. I årets gruppeprosjekt, blir studentene delt inn i grupper og får som oppgave å utvikle en Android-applikasjon. Rapportens hensikt er å gi en grundig beskrivelse av utviklingsprosessen til applikasjonen fra start til produktet er ferdigutviklet. Gruppene fikk muligheten til å velge fritt mellom seks forskjellige caser.

Vår gruppe valgte å gå for majoritetens ønske, case nummer 5: *Droner og luftrom*.

Oppgavebeskrivelsen lyder som følger:

*Droner blir mer og mer vanlig i norsk luftrom. Bruken kan variere fra hobby-flygning og moro, til å løse avanserte arbeidsoppgaver relatert til vedlikehold av strømmaster. I dette prosjektet skal dere utvikle en applikasjon som hjelper droneflygere til å få informasjon om værforhold knyttet til droneflygning. Hvordan er sikten? vindforhold? Fuktighet/regn? Følger værforholdene kravene til droneflygning som er satt av det norske lovverket?*

Casebeskrivelsene var relativt åpne slik at vi kunne bestemme selv appens funksjonalitet og design. Vår løsning er en applikasjon som gir brukeren muligheten til å sjekke om værforholdene egner seg for droneflygning gitt en spesifikk lokasjon eller brukerens egen posisjon i Norge. Ved hjelp av et kart klarer applikasjonen å vise data og hvor det er lov å fly etter luftfartstilsynets retningslinjer. Vårt hovedmål er å gi en oversiktlig fremvisning av værforholdene der brukeren har tenkt til å fly, mer spesifikt legger vi mest vekt på nedbør, tåke og vind.

## Teamet

### Håvard Nybråten

Håvard er 21 år gammel, og går 4. semester på Bachelor i Informatikk; Programmering og Systemarkitektur. Håvard liker å utfolde seg kreativt, og finner informatikk som en ypperlig kombinasjon av kreativitet og realfaglig kompetanse. Tidligere kompetanse innebærer internship hos Dimension10, hvor han jobbet med grafisk skalering av modeller i VR.



### Boye Molteberg

Boye er 22 år gammel og går 4. Semester på Bachelor i Informatikk; Design, Bruk og Interaksjon. Etter å ha fullført 3 årig realfag på videregående og å ha prøvd seg på en bachelor i Informatikk, elektronikk og teknologi fikk han sansen for programmering.



### Aksel Fagernæs

Aksel er 29 år gammel, er fra Stavern og går 4. semester, på bachelorgraden Informatikk: Programmering og systemarkitektur. Aksel jobber som utvikler/konsulent og synes bachelorgraden fra UiO er ypperlig faglig påfyll.



### Christian Lande

Christian er 21 år gammel og går 4. Semester på bachelorgraden Informatikk: Programmering og systemarkitektur. Christian startet med programmering på videregående skole, og siden den gang har interessen for informatikk bare vokst seg større.



## [Luis Gonzalez](#)

Luis Gonzalez er 21 år gammel og går 4. semester på bachelorgraden Informatikk: Programmering og systemarkitektur. Luis har stor interesse for utviklingsprosjekter, programmering og databaser.



## [Filip Felberg](#)

Filip er 25 år og går siste semester på bachelorgraden Informatikk: Programmering og systemarkitektur. Filip har i store deler av livet hatt stor interesse for alt som handler om data, og trives godt med å jobbe med programmering, databaser, kretslogikk osv.



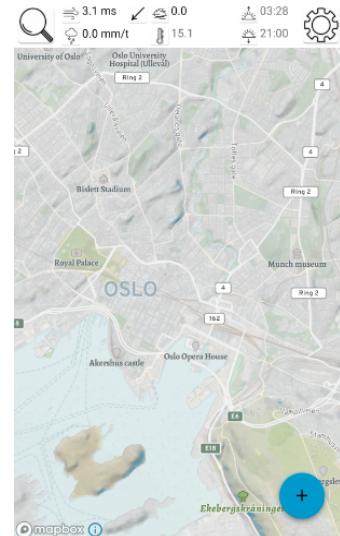
## 2 Brukerdokumentasjon

Applikasjonen skal kunne kjøre på smarttelefoner som har Android som operativsystem. Disse må ha API-nivå 23 eller høyere. Grunnen til at vi valgte et lavt API-nivå er for at flest mulig skal kunne bruke systemet. Vi hadde heller ikke behov for å bruke et høyere API-nivå siden vi ikke tok i bruk avanserte funksjonaliteter. Som en spesifikk målgruppe har vi valgt droneflygere uavhengig om de flyr kommersielt eller som hobby.

Applikasjonen gir brukeren en oversikt over værdata som er viktig for droneflygning der brukeren befinner seg, gir en god og presis oversikt over soner der droneflygning er forbudt etter norsk lov og lar brukeren sette sine egne grenser for flygning. Gjennom hele utviklingen har teamet fokusert på brukervennlighet og derfor egner systemet seg like godt for erfarne droneflygere som for nybegynnere.

### 2.1 Kart

For å kunne representere forbudte soner og spesifisere brukerens posisjon på en god måte, valgte vi å bruke et kart som applikasjonens hovedside. I tillegg til dette, viser kartets oppsett fjellområder som kan være interessant for brukeren å vite om. Kartet fungerer mer eller mindre som mange applikasjoner som bruker kart. Brukeren kan zoome inn og ut, se sin egen posisjon som blir markert med en markør, flytte på kartet for å se andre andre steder og trykke seg tilbake til opprinnelig posisjon.



Kart med layout, som viser Oslo

## 2.2 Væroppsett

Vær Informasjonen blir presentert på en enkel og oversiktlig måte øverst på kartet.

Væroppsettet består av to knapper og syv forskjellige verdier som er aktuelle for droneflygere. Disse verdiene er følgende:

-  Vindhastighet (m/s)
-  Nedbørsmengde (mm/t)
-  Vindretning
-  Tåke/Sikt (%)
-  Temperatur (°C)
-  Soloppgang (Klokkeslett)
-  Solnedgang (Klokkeslett)



De forskjellige verdiene blir vist ved siden av et ikon som beskriver den respektive værinformasjonen. Verdiene oppdateres fortløpende dersom brukeren endrer lokasjon eller oppdaterer egen posisjon. Fargen på disse verdiene kan variere etter brukerens preferanser. Denne funksjonaliteten blir beskrevet under [innstillinger](#). Dersom brukeren befinner seg i områder der det er umulig å hente værdata eller det ikke finnes en spesifikk verdi, blir verdiene satt til enten "n/a" eller "ukjent".

## 2.3 Søkefunksjon

Søkefunksjonen gir brukeren muligheten til å søke opp andre steder enn nåværende posisjon og sjekke værdata på disse stedene. Denne funksjonaliteten lar brukeren bevege seg utenfor Norges grenser, men vil da ikke kunne vise værdata i væroppsettet.



## 2.4 Innstillinger

Som beskrevet over, egner applikasjonen seg godt for både nybegynnere og erfarne droneflygere. Likevel ønsket vi å legge til en funksjonalitet som gjorde at erfarne droneflygere kunne få nytte av. Innstillinger knappen er representert med et tannhjul, og gjør at brukeren åpner en ny aktivitet med diverse funksjoner.



Innstillinger lar brukeren selv justere toleransen for tre viktige verdier: vindstyrke, nedbørsmengde og sikt. Etter at brukerne har bestemt seg for verdier, kan de trykke på "godkjenn" og deres toleranseverdier blir lagret lokalt på enheten. De kan dermed lukke applikasjonen, åpne den igjen og likevel kunne se og bruke deres lagrede verdier. Med denne funksjonaliteten kan erfarne brukere etter eget ansvar sette grenser på verdiene og velge om de vil fly eller om de lar være å fly. Dersom brukeren setter vindtoleransen på for eksempel 0.5 og vindhastigheten på det nåværende stedet overstiger denne verdien, vil vindhastighetsverdien i væroppsettet bli rød for å signalisere overskridelsen.

### Innstillinger

Justeringer	
Vindtoleranse	0.5
Regntoleranse	1.8
Sikttileranse	1.7



AVBRYT

GODKJENN

## 2.5 Trykkbare ikoner

Nederst til høyre har vi laget en knapp som utvider seg og gir tilgang til to nye knapper. Disse “forsvinner” igjen dersom den samme knappen blir trykket på. Dette gjorde vi slik at knappene ikke tok for mye plass og var et hinder for kartets funksjonalitet. Knappene er brukervennlige og det ble tatt hensyn til fargeblinde brukere da fargevalget ble tatt.



### 2.5.1 Tilbakestill posisjon

Denne knappen lar brukeren gå tilbake til egen posisjon dersom brukeren nавигerer seg bort fra den. Vi forsto raskt at denne knappen er essensiell for en applikasjon som bruker kart. Knappens symbol er også ganske selvforklarende og gjør at det ikke nødvendig å spesifisere hva knappen gjør på andre måter.

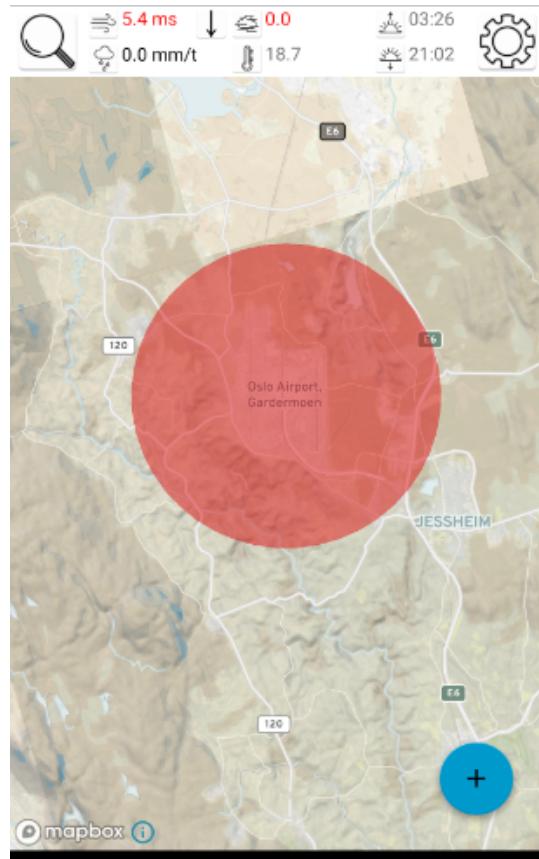


### 2.5.2 Forbudt sone grunnet flyplass

Denne knappen gir brukeren en oversikt over flyplasser i Norge og viser en rød sirkel med en radius på fem kilometer som etter norsk lov er grensen på hvor nærmeste droner kan være en flyplass. Denne er en ekstra funksjonalitet som vi mener er viktig for droneflygere i tillegg til værinformasjon. Når knappen blir trykket på kan brukeren enkelt se om de befinner seg innenfor denne røde sirkelen.



Funksjonaliteten kan “skrus av” ved å trykke på den samme knappen.



Forbudssone ved Gardermoen, vist ved rød sone

## 3 Kravspesifikasjon

### 3.1 Funksjonelle krav

- Systemet skal ha et brukervennlig og drone egnet kart.
- Brukeren skal kunne justere grenser for maks vindhastighet, regnmengde og tåke/sikt.
- Brukeren skal kunne sentrere kartet tilbake til egen posisjon.
- Brukeren skal kunne se sin egen posisjon på et kart så lenge brukeren befinner seg i Norge.
- Brukeren skal kunne sjekke om det finnes flyplasser rundt egen eller ønsket posisjon.
- Brukeren skal kunne sjekke værforholdene gitt en posisjon i Norge.

- Bruker skal kunne velge hva slags informasjon som vises på kartet.
- Systemet skal varsle brukeren dersom vind/regn/sikt-toleransen er nådd.

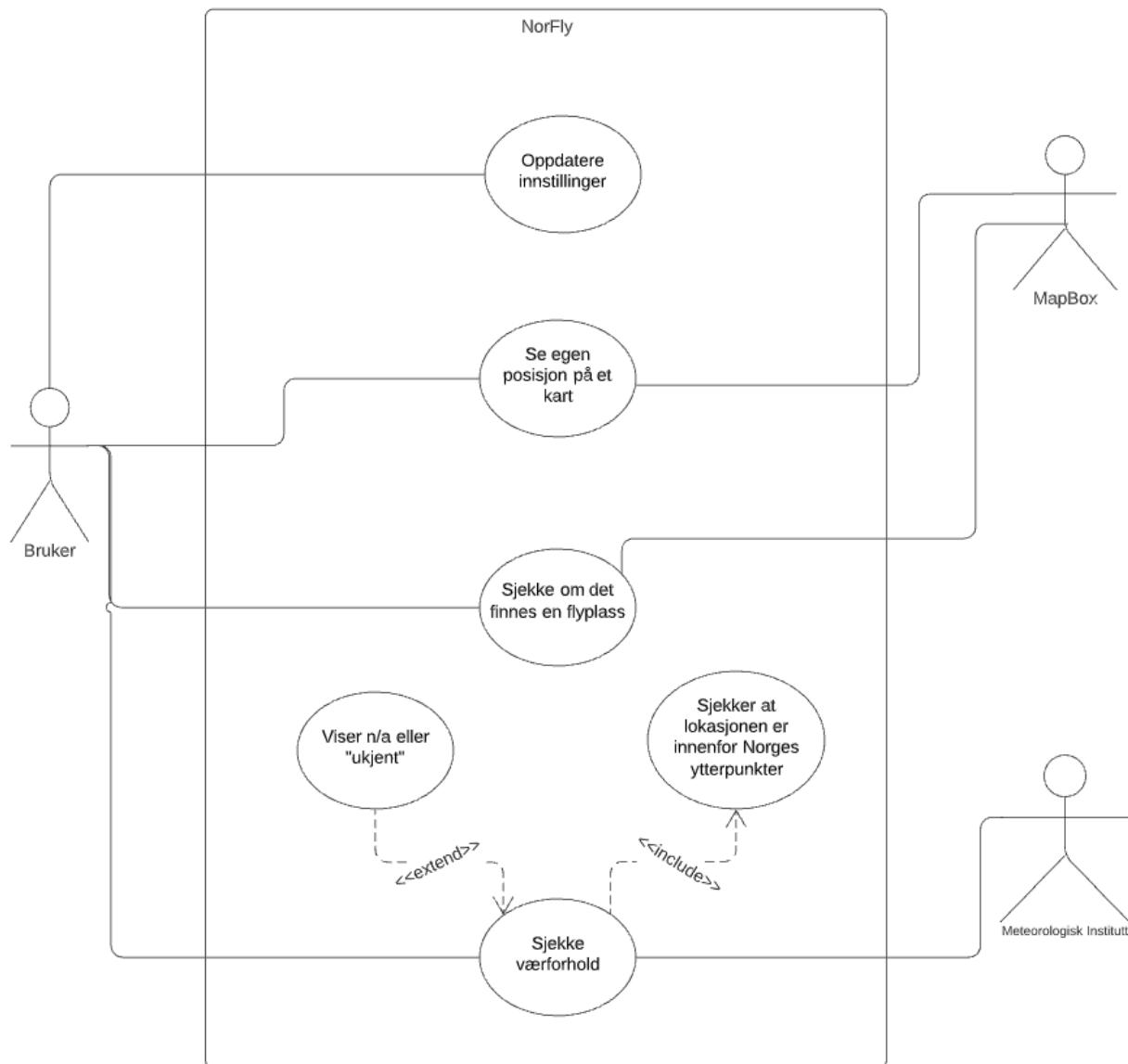
### 3.2 Ikke-funksjonelle krav

- Systemet skal bruke et API for å hente riktig værdata.
- Systemet skal vise riktig posisjon og værdata i sanntid.
- Systemet skal være lett å forstå/bruke for enhver bruker uavhengig av teknisk bakgrunn.
- Systemet skal kunne lagre preferanseverdier lokalt på brukerens enhet.
- Systemet skal ha få knapper slik at kartets funksjonalitet ikke forstyrres.
- Det skal ikke ta mer enn ett sekund å aksessere de forskjellige aktivitetene som systemet består av.
- Systemet skal kunne brukes av fargeblinde brukere.
- Systemet skal gi oversiktlig feilmeldinger dersom brukeren prøver å gjøre noe som applikasjonen ikke støtter.

## 3.3 Modellering

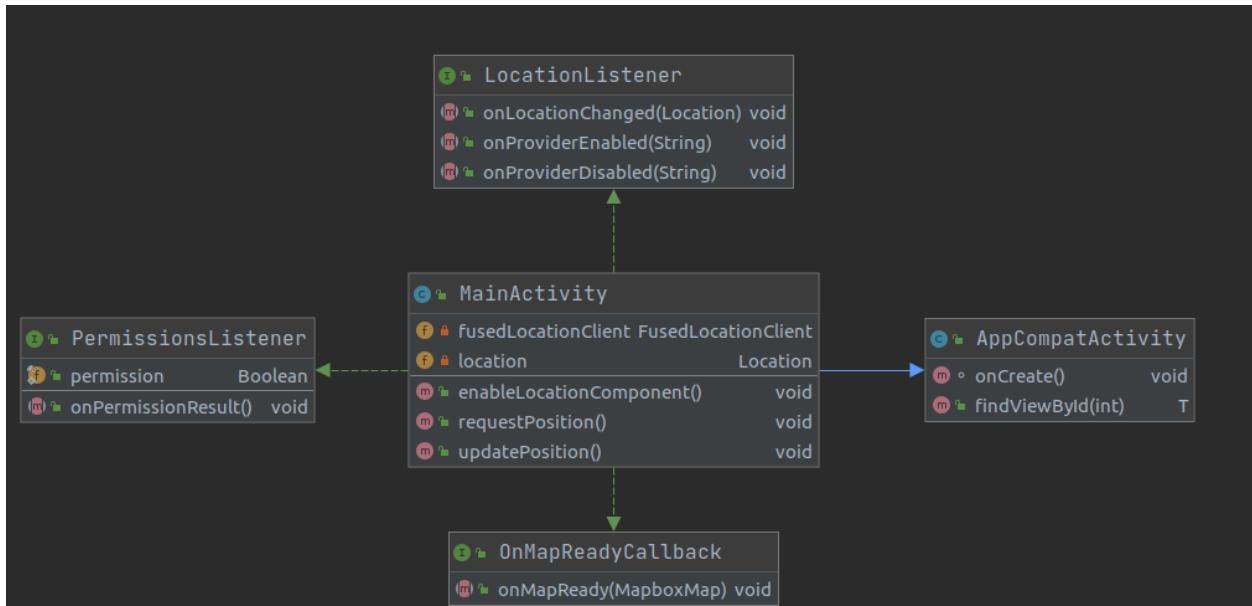
### 3.3.1 Use-case diagram

Use-case diagram for applikasjonens viktigste funksjonaliteter

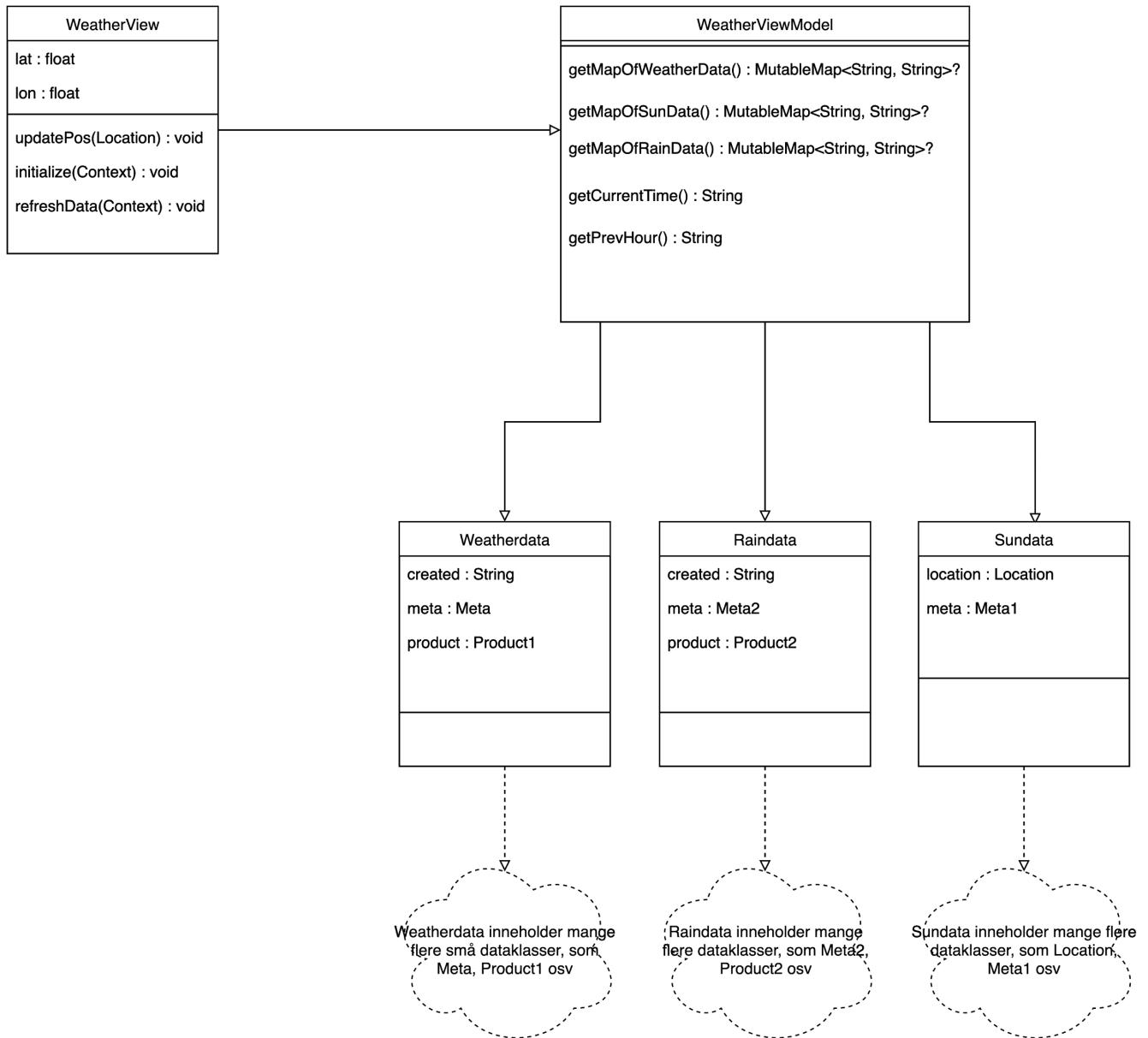


### 3.3.2 Klassediagram

#### 3.3.2.1 Klassediagram for: se egen posisjon på kart

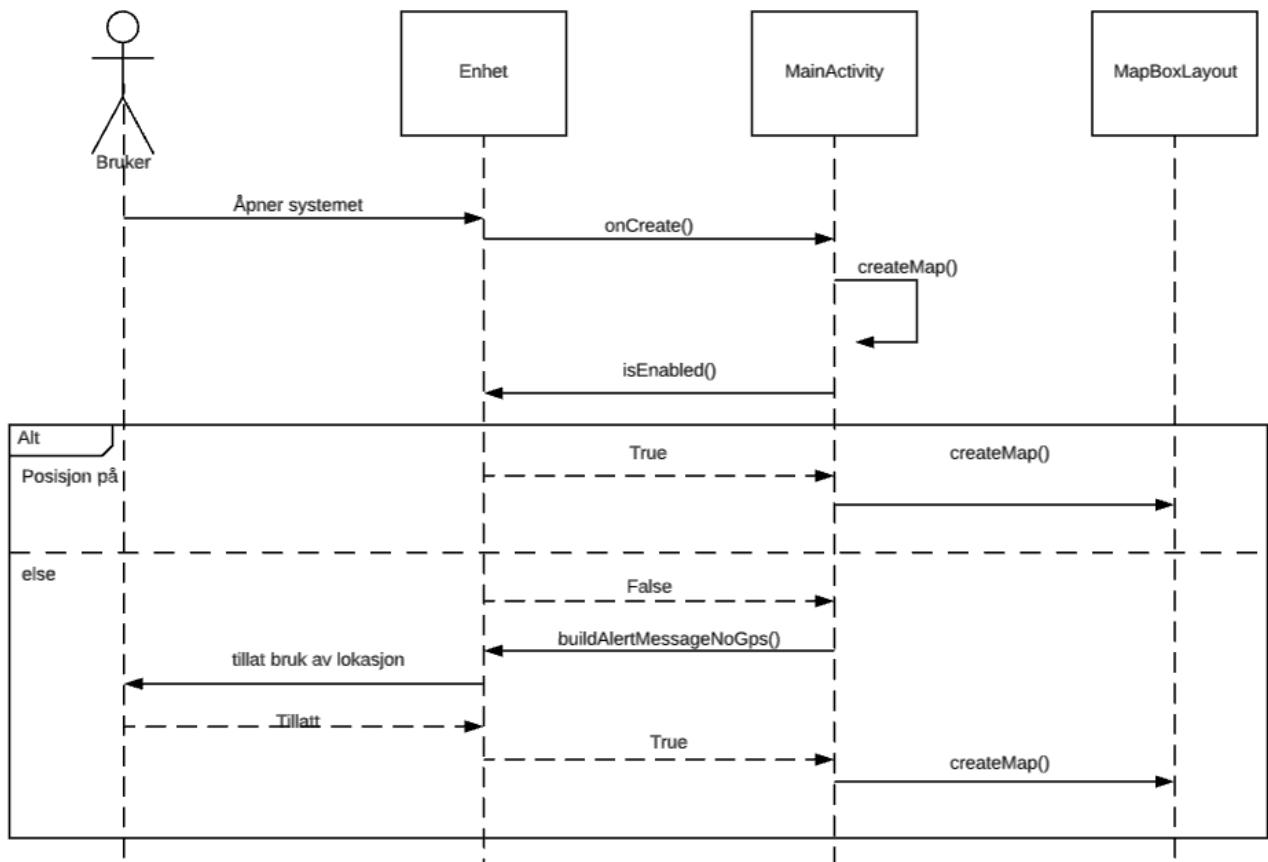


### 3.3.2.1 Klassediagram for: sjekke værforhold

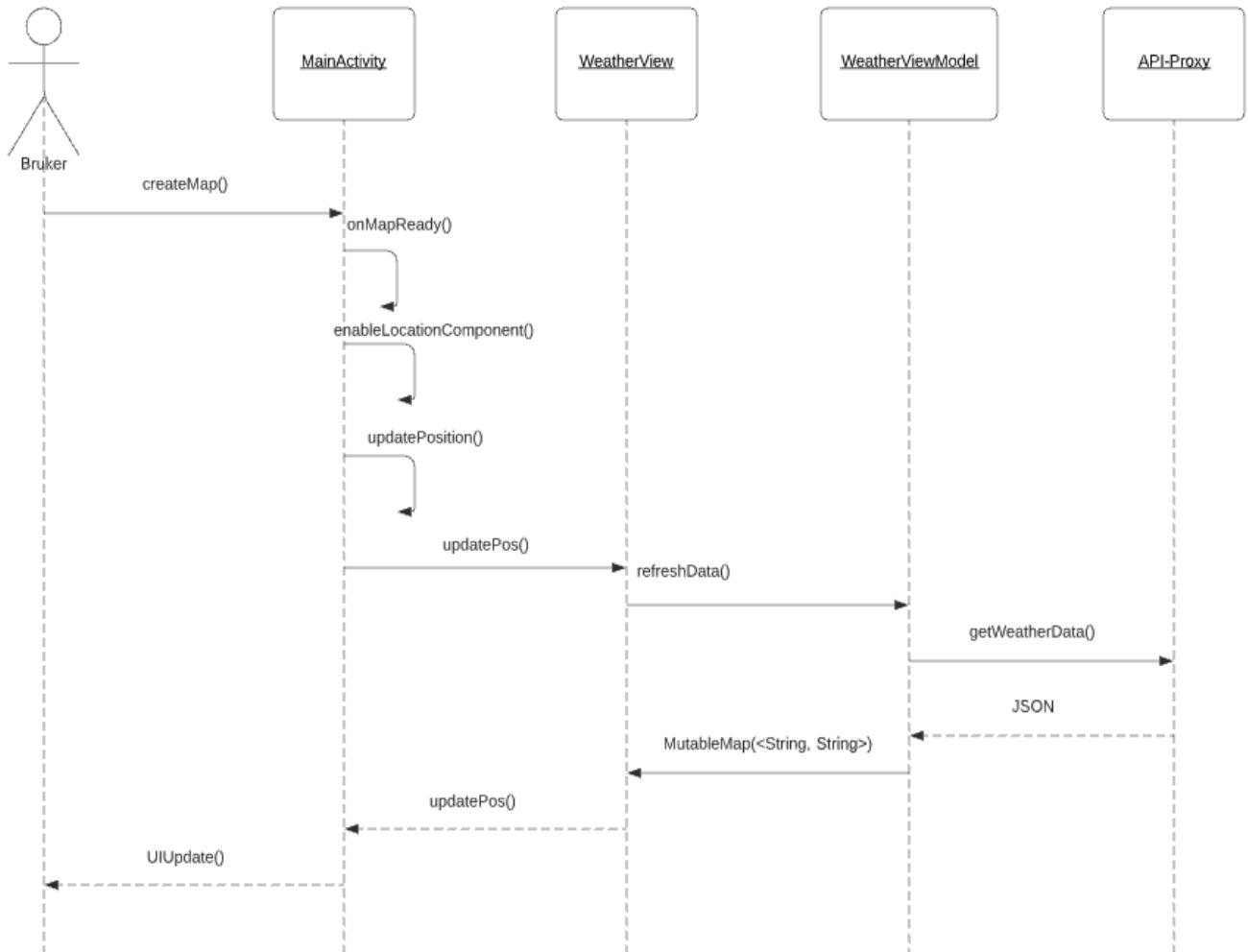


### 3.3.3 Sekvensdiagrammer

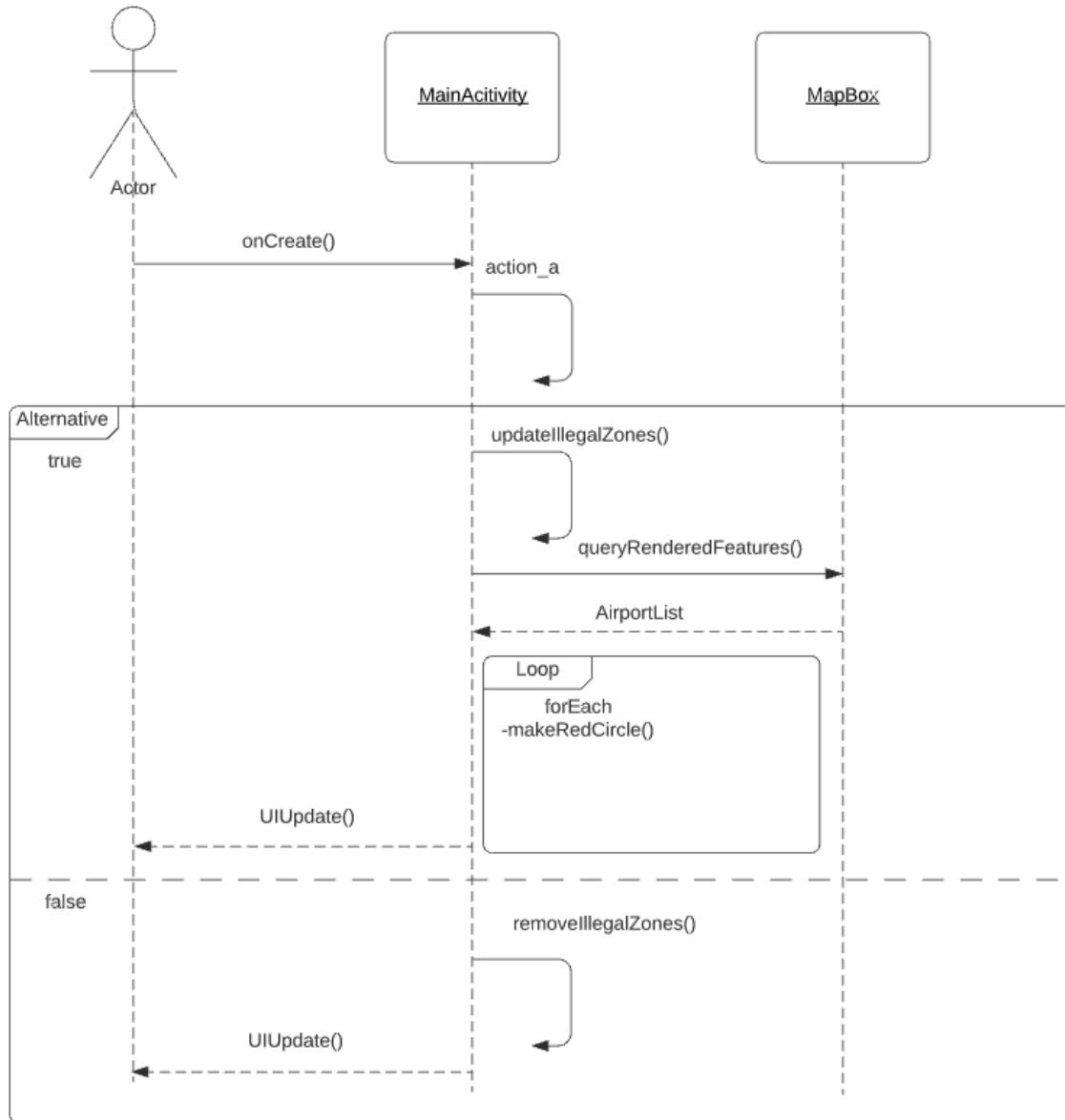
#### 3.3.3.1 Sekvensdiagram for: se egen posisjon på kart



### 3.3.3.2 Sekvensdiagram for: sjekke værforhold



### 3.3.3.3 Sekvensdiagram for: sjekke om det finnes en flyplass



## 3.4 User stories

1. Som bruker ønsker jeg å kunne se områder hvor det er ulovlig å fly drone, markert med rødt på kartet.
2. Som bruker ønsker jeg å sjekke værforhold som er relevant for droneflygning på et hvilket som helst område i Norge.
3. Som bruker ønsker jeg å bli opplyst dersom vindhastigheten, nedbørsmengden eller siktforholdene overstiger en terskelverdi satt av meg.
4. Som bruker av denne appen vil jeg kunne sette begrensninger for vær, slik at jeg kan fly i værforhold jeg er komfortabel med.
5. Som bruker av denne appen ønsker jeg muligheten til å sentrere kartet på min nåværende posisjon, samt få oppdatert værdata fra dette punktet.

## 4 Produktdokumentasjon

### 4.1 API

For å få tak i den nødvendige værdata, tok vi i bruk Gson biblioteket, tre ulike vær API-er og dataklasser for å lagre all dataen. Vi valgte å bruke JSON på de ulike API-ene våre, og Gson biblioteket gjorde det mulig å konvertere JSON-strings til Kotlin-objekter. Dermed blir informasjonen fra API-ene lagret i dataklassene vi har opprettet ved hjelp av Gson.

Når det kommer til API-ene har vi valgt å ta i bruk tre forskjellige. Grunnen til dette er at vi har bruk for ulike typer data, og alt finnes ikke i et og samme API. Det første vi tok i bruk var “locationforecast 1.9”, og dette gir oss mye generell informasjon om nåtid værdata på en gitt posisjon. Noen eksempler på ulike parametere vi tar i bruk herfra er

vindhastighet, fåkeprosent og temperatur. Dette API-et er veldig dekkende og har mange parametere vi ikke fikk bruk for. Dermed valgte vi å fjerne mange dataklasser slik at vi ikke lagret masse unødvendig informasjon.

Det andre API-et vi tok i bruk var “sunrise 2.0”. Når man flyr droner, er det et krav at man alltid skal kunne se den ihht. [avinors retningslinjer](#)<sup>1</sup>. Det er lovlig å fly droner i mørket dersom de er utstyrt med riktig lys, men vi ser på det som veldig hensiktsmessig å gi bruker oversikt over når solen går opp og ned. API-et inneholder informasjonen vi trenger om soloppgang og nedgang på et gitt koordinat og tidssone, men også mye annet som når månen går opp og ned, og dens posisjon. Ettersom at vi kun har bruk for sol-dataen, valgte vi i ettertid å fjerne det andre.

Det tredje og siste API-et vi valgte å bruke var “nowcast 0.9”. Dette tok vi i bruk for å få nedbørsmengden på et gitt koordinat. Vi har andre API-er som tar hensyn til vind og sikt, men nedbør er også en viktig faktor for droneflygere. Størrelsen på dronen man flyr kan påvirke i hvor stor grad man bør tenke på dette, men det er en faktor man uansett må ta høyde for. Fra dette kallet får vi informasjon om mm nedbør i timen gitt en lengde og breddegrad. Vi fikk akkurat den informasjonen vi hadde bruk for, og har derfor ikke hatt behov for å fjerne noe data.

Siden vi har tatt i bruk tre ulike API-er, hadde vi også bruk for tre forskjellige dataklasser for å lagre all dataen vi tok inn. Som et hjelpemiddel for å opprette dataklassene, tok vi i bruk verktøyet json2kotlin<sup>2</sup> for å generere dem. Dette var et godt verktøy som fungerte ganske bra, men enkelte feil forekom i noen klasser. Dermed var vi nødt til å feilsøke og rette opp i de klassene og variablene det gjaldt. Vi har brukt mappestrukturen model-view-view-model (mvvm), og derfor er alle dataklassene lagret i “model” mappen, og alle api-kall gjøres i viewModel.

---

<sup>1</sup> <https://avinor.no/konsern/pa-flyplassen/droner/generelt>

<sup>2</sup> <https://www.json2kotlin.com>

## 4.2 Brukerens lokasjon

For å vise brukeren sin lokasjon, endte vi opp med å bruk en location component<sup>3</sup>. Etter å ha testet ut både Google Maps og Mapbox landet vi på mapbox, og dermed ble en naturlig løsning på dette problemet å bruke nettopp en location component. Vi har lagd en funksjon (enableLocationComponent) som til å starte med sjekker om bruker har gitt appen tillatelse for stedstjenester, og spør bruker om tillatelse dersom det ikke er tilfellet. Deretter oppretter den et location component ved hjelp av to variabler som tilpasser og aktiverer det. Komponenten blir også satt til å følge brukeren. Det er også i denne funksjonen vi får tak i brukeren sin posisjon med lengde- og breddegrader. Ved hjelp av en LocationManager lager vi et location objekt. Dersom det blir opprettet, oppdaterer vi hvor kartet zoomer og værdataen i weatherviewet.

Appen sjekker også om bruker har posisjon påskrudd ved hjelp av funksjonene “isEnabled”, “buildAlertMessage” og “createMap”. isEnabled sjekker om posisjon er på og returnerer true dersom den er det, eller false om det er skrudd av. Denne informasjonen brukes i onCreate for å bestemme hvilken metode som blir kalt på videre. Dersom posisjon er avskrudd og isEnabled returnerer false, blir det gjort et kall på buildAlertMessage. Denne metoden spør bruker om å skru på stedstjenester. Dersom man ikke tillater, kommer det opp en toast som sier at appen trenger dette for å fungere, og appen avsluttes. Trykker man derimot på “Ja” og tillater, blir man sendt til lokasjonsinstillinger slik at man kan skru det på. Hvis isEnabled returnerer true, blir det gjort et kall på createMap. Vi tok i bruk “fusedLocationClient” for å få tak i bruker sin siste posisjon. Vi tar tak i bredde- og lengdegradene til lokasjonskomponenten for å så sende dem inn “updatePosition” for å endre kameraet sin zoom til der bruker befinner seg. Lokasjonskomponenten blir også sendt inn i weatherview for å oppdatere

---

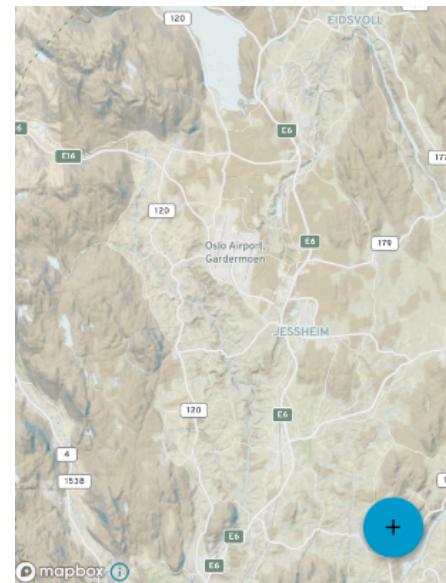
<sup>3</sup> <https://docs.mapbox.com/android/maps/overview/location-component/>

værdataen i viewet. Her blir det også sjekket at koordinatene befinner seg innenfor Norges ytterpunkter<sup>4</sup>, og man får en feilmelding dersom dette ikke er tilfelle.

## 4.3 Kart

For å kunne gi brukeren en oversikt over egen lokasjon og forbudte flyområder, tok vi i bruk Mapbox. Mapbox er en kartløsning for apputvikling som er gratis å bruke inntil man når 25.000 brukere eller 100.000 API requests. Siden vi i vårt prosjekt ikke kommer til å nå disse tallene er det en god løsning for oss. Vi laget en egen bruker for å administrere Mapbox, og det ble generert en egen “access token” for brukeren vår. Denne bruker vi til å integrere Mapbox i appen ved å lage en instans av kartet.

Mapbox har en del ferdiglagde Styles som kan brukes til å endre utseende på appen. Styles er en JSON-fil som beskriver hvordan Mapbox skal vise de forskjellige dataene som finnes i Datasets og Tilesets. Stylen vi valgte vises på bildet, og heter Cali-Terrain<sup>5</sup>



Eksempel på kart brukt i appen

<sup>4</sup> [https://no.wikipedia.org/wiki/Lista\\_over\\_norske\\_geografiske\\_ytterpunkter](https://no.wikipedia.org/wiki/Lista_over_norske_geografiske_ytterpunkter)

<sup>5</sup> <https://www.mapbox.com/gallery/#cali-terrain>

## 4.4 Ulovlige soner

I denne delen av programmet skal det vises røde sirkler rundt ulovlige flysoner for droner i henhold til lufttilsynets droneguide<sup>6</sup> og militære forbudssoner. I guiden står det at det er ulovlig å fly nærmere enn 5 km fra flyplasser/lufthavner, og programmet skal derfor lage en rød sirkel med radius på 5 km rundt lufthavner.

### 4.4.1 Flyplasser

For å få til å vise ulovlige flyplass-soner med Mapbox må vi ta i bruk “Styles” og metoden “queryRenderedFeatures”

«Style» i MapBox er den delen som viser det grafiske på kartet. Vi henter ut style-en som blir brukt i kartet vårt ved å bruke «map.getStyle». Stylen består av forskjellige style-layers, som inneholder og det er mulig å enten endre eksisterende layers eller lage nye.

Det er ingen ferdiglagde metoder i MapBox som kan lage sirkler ut ifra en konstant størrelse, så vi skal i dette programmet manuelt lage en polygon som skal bli den røde sirkelen. Dette gjøres i metoden polygonCircleForCoordinate (modifisert versjon av denne metoden<sup>7</sup>, som tar inn koordinater i form av LatLng (breddegrad og lengdegrad) og radius i form av Double. Metoden returnerer en polygon som beskriver en 5 km sirkel i form av en liste av LatLng.

Når vi har laget en liste av slike polygoner i form av List<List<LatLng>>, kan vi bruke FillOptions for å endre fargen og gjennomsiktighet, og legge til de ferdige sirklene i Style-en med FillManager. Men for å generere slike polygoner trenger vi koordinater som bestemmer midtpunktet av sirkelen.

---

<sup>6</sup> <https://avinor.no/konsern/pa-flyplassen/droner/generelt>

<sup>7</sup> <https://github.com/mapbox/mapbox-gl-native/issues/2167>

Disse koordinatene kan bli hentet ut ved å bruke metoden «queryRenderedFeatures», som brukes på variabelen “map” som hentes av mapView.getMapAsync og sendes med rectF som definerer arealet av hva som synes på skjermen og sourceLayerId-en «airport label» som parameter. Denne metoden vil gi en liste av alle «features» som matcher «airport label». En feature er et spesifikt punkt i kartet, og kan f.eks. være midtpunktet i flyplassen Gardermoen, og et slikt punkt inneholder et sett med koordinater i form av LatLng. Ved å gå gjennom alle features ved hjelp av en løkke kan vi hente alle koordinatene fra flyplassene og generere polygoner ved hjelp av metoden beskrevet over.

En svakhet ved å bruke “queryRenderedFeatures” får å få tak i koordinatene fra flyplassene, er at bare de sourceLayerId-ene som er rendret og synes på kartet kan bli hentet. Dette betyr at hvis kartet er zoomet så langt ut at navnene på flyplassene ikke synes, så vil heller ikke metoden kunne fange opp nye flyplasser og lage sirkler. Dette er en svakhet i Mapbox og vi har dessverre ikke funnet en måte å komme forbi denne svakheten på.

#### 4.4.2 NSM sine forbudssoner

NSM prøvde å hjelpe oss å anskaffe dataene for forbudssonene deres. Problemet er imidlertid at det ikke var et API men en link til en XML fil som var nøstet med andre nettsider. Måten dette var lagt opp på ga oss liten mulighet for å enkelt hente ut relevant informasjon om kordinater til forbudssonene. Etter manuelt å ha gått gjennom en rekke av lenkene i dette oppsettet klarte vi ikke å lokalisere kordinater og valgte å gi opp søken ettersom NSM selv allerede har en kartløsning for dette formålet. Ettersom det finnes en løsning for militære forbudssoner som man ikke kan unngå, fokuserte vi på sivile forbudssoner.

## 4.5 Væroppsett

Værinformasjon fra nåværende posisjon er fremstilt i et oversiktlig felt øverst på skjermen. Oppsettet er en kombinasjon av linearlayouts, knapper, imageviews og textviews. Oppsettet baserer seg på en fordeling av den overordnede weightsum, slik at informasjonen og oppsettet er skalerbart og basert på hvor mye plass den har. Dette ser man tydelig dersom man har appen i en liggende stilling. Hovedlayouten til værinformasjonen er en horisontal linear layout, med en weightsum på 14. Under denne ligger det da først en knapp, søke-knappen på venstre side. Denne knappen er så etterfulgt av tre nye linear layouts, hvor disse har en vertikal fremstilling. Til slutt kommer en ny knapp, som tar deg videre til innstillinger. Innenfor de tre vertikale linear layouts i midten, har man igjen to nye horisontalt fokuserte linear layouts i hver av dem. Disse viser henholdsvis informasjon som blir hentet fra diverse tidligere nevnte API, i form av et ikon som beskriver informasjonen, og en oppdaterbar tekst som viser informasjonen. Eneste unntaket er vind informasjonen, som har et ikon etter teksten som viser vindretningen på gitt lokasjon. Alle linear layouts i oppsettet har en weightsum, samt at alle direkte underliggende layouts og views har en tyngde som representerer en brøkdel av summen til dens forelder. Denne måten å sette opp informasjonen på fungerer bra siden delene alltid forholder seg til hverandre og baserer seg på hvor mye plass som er tilgjengelig, slik at det er skalerbart uten at noen av delene er basert på en spesifikk avstand eller størrelse. Spesifikasjon av størrelser er helt og holdent basert på diverse bruk av "match\_parent", "wrap\_content", "weightSum", "layout\_weight", samt at bildene bruker "scaleType= "fitCenter"" og "adjustViewBounds=true"" for å skalere bilde til tilgjengelig plass. Xml er vedlagt.



Væroppsett med søker og innstillinger, og blueprint av oppsettet

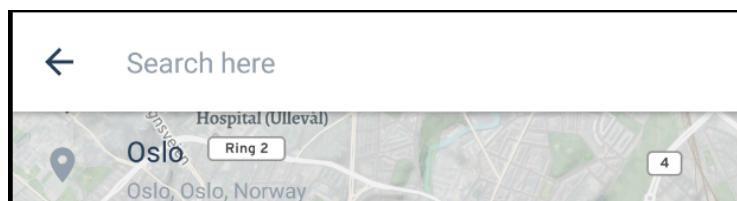
## 4.6 Søkefunksjon

Denne funksjonen bruker «Places» i MapBox, og det må derfor importeres i gradle med følgende linje: implementation

'com.mapbox.mapboxsdk:mapbox-android-plugin-places-v9:0.12.0'. Dette gir oss flere ferdiglagde metoder som gjør det lettere å implementere en søkefunksjon.

Vi skal lage søkefunksjonen som en fragment, og lager derfor en instans av den ferdiglagde fragment-klassen «PlaceAutocompleteFragment», som vi kaller «autocompleteFragment». Vi bruker så «supportFragmentManager» til å lage en transaction som legger til fragmentet i layoutet med id-en «fragment\_container», og holder styr på fragmentet ved hjelp av tag-en «fab\_location\_search».

Vi legger så til en «PlaceSelectionListener» på fragmentet som utfører en funksjon når brukeren har søkt etter et sted. I denne funksjonen hentes koordinatene ut fra en instans av klassen «CarmenFeature», som inneholder informasjon om feature-en som brukeren har søkt på. Som forklart i ulovlige soner dokumentasjonen er en feature et bestemt punkt på kartet i MapBox. Når vi har fått disse koordinatene sender vi de inn i et kall på metoden “updatePosition”, som oppdaterer kartet til å vise stedet brukeren har søkt på.



Søkevindu ved trykk på "søk"-knapp i væroppsett

## 4.7 Innstillinger

Innstillinger-funksjonaliteten tillater brukeren å sette egne toleranser for vær og vind.

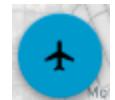
Mer spesifikt er disse delt opp i vind, regn og tåke ettersom dette kan være begrensende faktorer for pilotene.

Tanken bak dette var å kunne vise regulerbare risikoområder, slik at piloten selv kunne begrense hvor mye vind, regn og sikt som var tolererbart for flygning. I utgangspunktet var planen å ha toleransen for værinformasjon samt endring av diverse farger. Etter forskjellige problemer mot slutten med fargevelgeren ble det bestemt å holde seg til kun å justere værtoleranse. Dette ble oppnådd ved å bruke “seekbar”, som er en progresjon bar som lar deg sette øvre maks verdi. Maks verdi ble satt til verdien multiplisert med 10, og senere multiplisert igjen med 0.1, for å få desimaltall med ett desimal. Ved hjelp av metoden “onProgressChanged” fikk vi oppdatert tekst på siden av hver progresjonsbar med foreløpig progresjon. Hvis brukeren nå trykker på avbryt, så vil verdiene til de diverse progresjons barene bli satt tilbake til default verdi eller siste lagrede endring. Om brukeren trykker på godkjenn så vil verdiene i sharedpreferences bli satt lik progresjonen til de diverse barene, og toleransen vil bli oppdatert. Denne verdien vil også være den samme neste gang brukeren går inn på appen, siden sharedpreferences lagrer informasjonen lokalt på brukerens enhet.

## 4.8 Trykkbare ikoner

Applikasjonen gir brukeren muligheten til å trykke på en “pluss-knapp” som viser to andre knapper med diverse funksjoner. Grunnen til at vi valgte dette oppsettet var for å ha så få knapper som mulig.

- Fly: Denne knappen har som funksjon å gi en oversikt over ulovlige flysoner knyttet til flyplasser. Trykker man på denne, kaller vi på metoden: updateIllegalZones/removIllegalZones



som toggler et filter med røde sirkler på kartet over alle flyplasser.

Sentrer: Denne knappens funksjon er å sentrere kartet slik at brukeren lett kan finne tilbake til sin egen posisjon etter å ha utført søk eller flyttet på kartet. Knappen kaller på metoden center.



## 5 Testdokumentasjon

Som testmetode har vi gjennom hele utviklingen kjørt egentesting og feilsøking etter hver vesentlig endring i både frontend og backend. Til dette ble det som oftest brukt emulatorer med forskjellige API-nivå og mobiltelefoner som hadde Android som operativsystem. Vi har hatt som prinsipp at før hver eneste merge kan gå fra en lokal branch til den universelle masteren, må minst én annen i gruppen ha testet endringene på sin emulator/mobil.

For å teste brukervennligheten til de forskjellige xml-layoutene våre ble flere prototyper sendt ut til brukergruppen slik at vi kunne få konkrete tilbakemeldinger på disse. Etter ulike svar fra brukergruppen, ble vi enige om hvordan appens endelige versjon skulle se ut.

### 5.1 Integrasjonstesting

Til integrasjonstesting har vi valgt å bruke Android Studios innebygde verktøy Espresso. Espresso gjør det mulig å gjøre enkle opptak av interaksjoner med appen, og oversetter dette til kode som vi selv kan sette assertions på.

Appen vår har 2 integrasjoner vi har ansett som interessante å teste:

1. Datahenting fra in2000-apiproxy.ifi.uio.no/\*

## 2. Søketjenesten etter byer som Mapbox gir oss.

Vi har da lagd en test som går ut på at vi søker opp en annen by enn den man opprinnelig står i, forflytter seg dit - og deretter sammenligner værdata fra den opprinnelige lokasjonen, med værdata fra den nye lokasjonen.

Data som alltid bør være ulikt gitt to ulike posisjoner sør for polarsirkelen, er klokkeslettene for når sola går opp og ned; så ved å sammenligne klokkeslettene fra de to lokasjonene kan vi verifisere:

1. Dersom vi har klokkesletter å sammenlikne betyr det at vi har kontakt med [in2000-apiproxy.ifi.uio.no/](http://in2000-apiproxy.ifi.uio.no/)
2. Dersom de er ulike betyr det at søketjenesten og lokasjon oppdateringene fra Mapbox har fungert

## 5.2 Enhets testing

Da vi skulle til å skrive enhetstester oppdaget vi at vi har innkapslet all logikk inn i større og tyngre metoder, uten enkle inn og ut parametre som enkelt kan bli testet og verifisert. For å få skrevet enhetstester kreves det da mye omskriving av koden, som vi mot slutten av prosjektet ikke ville begi oss ut på. Denne avgjørelsen tok vi siden vi har integrasjonstester som får testet en del av appens logikk, og forsåvidt de viktigste komponentene i den.

Vi har erfart at det er lurt å begynne så tidlig som mulig med enhetstester, slik at man ender opp med testbar kode, og tester som kan gjennomføres kjapt.

## 6 Proseszdokumentasjonen

Prosessen som ledet oss til det endelige produktet har stort sett gått veldig bra, men det var en del utfordringer vi måtte takle underveis og som vi har lært mye av. Det var ikke alle som kjente hverandre fra før, derfor bestemte vi oss for å møtes og kickstarte prosjektet med å gjøre noe sosialt sammen slik at vi ble bedre kjent.

Vi var tidlig ute med å velge arbeidsmetodikk, fordele arbeidsroller og bestemme hvilke verktøy som skulle brukes under utviklingen av prosjektet. Vi gikk for en balansert kombinasjon av Kanban og Scrum (Scrumban). Dette tenkte vi at egnet seg best for et slikt prosjekt med tanke på tiden vi hadde på utviklingen, antall teammedlemmer og arbeidsmengde. Ut ifra erfaring, preferanse og kunnskap delte vi ut fem ansvarsroller. Design-ansvarlig, frontend-ansvarlig, backend-ansvarlig, backlog/git-ansvarlig og rapportansvarlig. Disse rollene fastslo ikke hvem som skulle gjøre hva, men de ansvarlige skulle ha en viss oversikt over de ulike delene av prosjektet, tildele oppgaver og finne løsninger dersom det oppsto konflikter. Alle teammedlemmene skulle jobbe med de ulike delene av prosjektet, derfor trengte vi diverse verktøy som alle kunne ha tilgang til og som var godt egnet for samarbeid og kommunikasjon i et team. Alle kom med forslag til digitale verktøy og argumenter til hvorfor de passet inn i prosjektet. Vi ble enige om å ta i bruk følgende verktøy:

- Jira
- Slack
- Discord
- Github
- Google drive
- Google docs

Under prosjektet var prosessen todelt. Med dette mener vi at noen jobbet mest med de generelle funksjonene og andre fokuserte på XML, oppsett og brukervennlighet. Med generelle funksjoner mener vi her ting som kart, søkefunksjon og vær-API. Dette ble lagt inn i en POC activity og testet. Når en funksjon var god nok ble den hentet fra POC og lagt inn i UI. Fordelen med dette var at vi kunne bruke POC-activity som en slags test-grensesnitt.

Bildet under viser hvordan Jira-boardet vårt så ut midt i prosessen. Her ser vi bruk av flere utviklingsstadier hvor gruppemedlemmene valgte eller ble bedt om å gjøre oppgaver, som de da fikk tildelt på oppgavespesifikasjonen inne på boardet. “Backlog” viser til ledige oppgaver. Videre ble oppgavene plassert i “Under Arbeid” og ble tildelt til en av medlemmene, som arbeidet videre på oppgaven. Om man så satt seg fast på oppgaven var alle i gruppen veldig åpne for å hjelpe til. Etter oppgaven er utført, ble den plassert i “Til Testing” som ofte betydd at man hadde sendt inn en pull request, hvor man satt diverse andre gruppemedlemmer opp til gjennomgang, tilbakemelding og eventuelle forslag til forbedringer. Når oppgaven hadde blitt gjennomgått av andre gruppemedlemmer og gitt klarsignal, ble det merget til master og plassert i “Utført” i Jira-boardet. I senere tid la vi også inn “Forkastet”, for oppgaver og idéer som vi gikk bort i fra.

Jira-boardet underveis i prosessen

## 6.1 Informasjonsinnhenting

Vi var tidlig ute med å opprette kontakt med brukergruppen. For å opprette kontakt sendte vi ut en brukerundersøkelse hvor vi lette etter androidbrukere som fløy droner. I tillegg til å sjekke om de kunne delta i prosjektet, samlet vi også inn andre apper de eventuelt brukte. Vi fikk til slutt en fokusgruppe bestående av 6 dronepiloter.

På dette punktet ble et generelt samtykkeskjema sendt ut. Dette var et skjema som inneholdt de planlagte kvantitative undersøkelsesmetodene som vi hadde skulle benytte, samt hva og hvordan vi ville bruke denne informasjonen. Vi har konkludert med

å kun dele sitat fra mail og ingen mailadresser, navn eller stedsnavn. Ut av hensyn til deres personvern er det kun et team medlem som har tilgang på mailadresse og navn.

Ut ifra de seks samtykkeskjemaene vi sendte ut ble det kun sendt et svar.

Designansvarlig gikk så gjennom alle fokusgruppe deltakerne og prøvde å få kontakt med dem. Ut ifra de 6 som originalt hadde meldt interesse var det nå kun to som svarte ved den tredje henvendelsen. Av disse to fikk vi inntrykket av at coronaviruset hadde ført til at mange ikke ønsket å delta, samt flyrestriksjoner i Troms og Finnmark var et hinder for andre.

### 6.1.1 Ekspertvurdering av prototyper

Originalt var planen å sende ut en rekke low-fidelity prototyper til våre seks brukere i fokusgruppa og følge dette tett opp. Det viste seg imidlertid at pandemien satte en stopp for dette. Av de seks som meldte interesse var det bare to som svarte på vårt infoskriv med samtykkeerklæring. Av disse sa en ja øyeblikkelig, sistemann valgte å trekke seg fra prosjektet. Etter mye korrespondanse klarte vi å avtale med sistemann at han skulle holde seg i prosjektet.

Vi hadde imidlertid flaks i uflaksen. Begge disse pilotene var erfarne. Siden vi nå hadde klart å holde på en erfaren hobby-pilot og en erfaren kommersiell pilot kan vi behandle dem som ekspertvitner.

Slik nevnt i 6.3.5 var det litt blandede inntrykk av de ulike prototypene. Prototype 4 var desidert best i rangert. Det ble trukket frem stort kart med lett tilgjengelig værinformasjon, mulighet til å velge om man skal ha forbudssoner samt justerbare toleransegrenser som positive trekk.

De trakk også frem prototype 3. Denne mente de var veldig oversiktlig og enkel, de likte spesielt den tydelige søkefunksjonen. Men den hadde den ulempen at det ikke var tydelig at det var forbudssoner på kartet sågar som navigasjon var knotete. En deltaker sa at "#3 ser veldig oversiktlig og enkel ut, så den tenker jeg kommer på en god andre plass, så lenge den også inneholder oversikt over plasser jeg ikke kan fly." mens den andre deltageren hevdet at "Prototype 1 og 3. Virket litt mer rotete, men veldig greit å ha søkefunksjonen tilgjengelig på "fremsiden"". Vi tolket dette til at denne prototypen ikke hadde nok affordance i den visuelle representasjonen.

Prototype 2 og 1 ble trukket frem som mer avanserte enn de to andre og generelt dårligere enn resten. Vanskeligheter med navigasjon ble blant annet trukket frem.

### 6.1.2 Brukertesting av app

Dersom vi hadde hatt mulighet til å sette oss ned med en bruker å teste appen på samme telefon ville dette latt oss teste appen i kontrollerte omgivelser før vi sender den rundt til fokusgruppen.

For å få gjennomgått brukertesting sende vi ut et apk, norfly.apk, via mail. Denne ble imidlertid stoppet av mailtjenesten. Vi gikk derfor over til en løsning som resulterte i en dropbox link<sup>8</sup>. Dette så ut til å løse problemet, og noe vi i tillegg merket i ettertid, lot oss oppdatere appen underveis. Første tilbakemeldingen vi fikk var delt positiv og negativ. "Har ikke opplevd at appen har krasjet til nå, og syntes dere har truffet flott med designet på den. Men den har dessverre store mangler for meg, noe du også nevner. Må nok fortsatt bruke <https://www.safetofly.no/> for å føle meg på den sikre siden. I tillegg bruker jeg UAV Forecast." Det viser seg at selv om appen er viser værinformasjon på en god måte og er lett og intuitiv å bruke, så er det ikke nok. Ettersom vi ikke fikk til å bruke datasettet til

---

<sup>8</sup> <https://www.dropbox.com/s/dvi0u1tonkc76iz/norfly.apk?dl=0>

NSM så var det enklere å bruke safetofly. I tillegg kan vi ikke forutse værforhold. Derfor vil den ikke være nyttig i en forberedende rolle.

Dette sier oss at hvis vi skulle fortsatt med utvikling ville vi måtte prioritere forbudssoner og vær. Det er mulig det er en måte å vise vær fremover tid ved å basere seg på APK til meteorologisk institutt. I tillegg skal det være mulig å implementere NSM sine forbudsjoner, forbudssoner rundt vei og bebygget område. Dessverre var dette noe vi ikke rakk å gjennomføre. Spesielt NSM sine forbudssoner ville tatt en lang tid å implementere på en god måte. Ikke bare må selve laget implementeres, men det er også et lovmessig rammeverk rundt bruken av dette datasettet.

## 6.2 Prototyper

Vi så for oss å lage tre, wizard of oz type, wireframes som prototyper. Disse skulle sendes til en fokusgruppe. Poenget med dette blir da for brukergruppen å gi oss tilbakemelding på prototypene. Med denne tilbakemeldingen kan vi diskutere og eventuelt gjenbruke det som fungerer og det som ikke fungerer.

Det ble to større endringer fra planen. Først lagde vi fire prototyper istedenfor tre. Etter dette forsvant 4/6 deltagere av fokusgruppen. Dette gjorde at vår allerede lille fokusgruppe ble enda mindre. For å veie opp for dette behandlet vi disse to gjenværende deltakerne som ekspertvitner heller enn fokusgruppe. Dette kommer av to grunner. Nummer en for å representere dronepiloter i Norge trenger vi en god del forskjellige mennesker. Dette fører til at informasjonen ikke kan generaliseres til en majoritet. Nummer to, de to gjenværende er en amatør og en kommersiell pilot, begge er også ganske erfarte. Derfor, ved å behandle disse som ekspertvitner vil vi kunne etablere en god antagelse av hva brukeren ønsker.

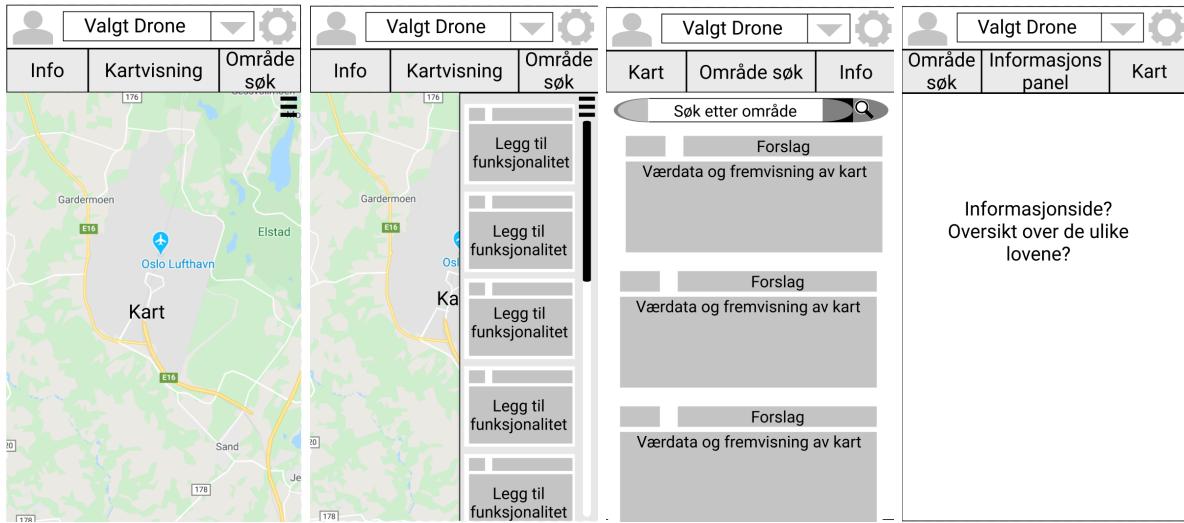
I utviklingen av de fem prototypene ble verktøyet figma benyttet.

### 6.2.1 Prototype 1

Prototype 1 var vårt første utkast av hvordan vi så for oss at layouten på appen kunne være. Hovedvinduet var satt sammen av et menyfelt på toppen, med brukerens profil, brukerinnstillinger og en dropdown-menü med diverse droner. Under dette var det lagt en meny over hvilket view som skulle vises på forsiden. Tanken bak denne menyen var å rotere mellom 3 forskjellige sider, hvor menyens oppsett oppdaterte seg når man kom til en ny side. Nåværende side skulle vises i midten, og gjenstående sider skulle vises i feltene til høyre og venstre. Bilde 1.1 viste kart over området man befant seg i, og dette skulle være siden man startet på når man åpnet appen, etter at man hadde godkjent brukervilkår. Fra denne siden kunne man også åpne en funksjonalitetsmeny ved å trykke på symbolet øverst i høyre hjørne av kartet, dette er vist i bilde 1.2. Ved å trykke på høyre felt av view menyen, kunne komme til siden som utførte områdesøk, som er vist i bilde 1.3. Den gjenværende siden var tenkt som et informasjonspanel, hvor brukeren kunne gå inn og lese informasjon om området og en oversikt over de forskjellige lovene og reguleringene som brukes som utgangspunkt av evalueringen av området. På bilde 1.5 ser man siden for brukerens profil, hvor man også har en oversikt over droner som er lagt til, samt en knapp for å opprette en ny drone profil. Her kan brukeren legge inn informasjon om dronen. Om brukeren huker av for at dronen har andre sensorer eller funksjonaliteter enn de man kan ha på sivile uregistrerte droner, vil brukeren få informasjon om dette og kunne bli sendt til Nasjonal Sikkerhetsmyndighets nettsider for registrering.

Link til prototype 1:

<https://www.figma.com/proto/zRqEwsusFqoenrE9R6eOhy/Prototype-1?scaling=scale-down&node-id=1%3A3>



1.1: Hovedside

1.2: Funksjonalitet meny

1.3: Område søk

1.4: Informasjons side

The image shows two screenshots of the 'Legg til drone' (Add drone) page:

- Left screenshot:** Shows fields for 'Navn' (Name) and 'Hjemsted' (Homeplace). Below is a section titled 'Legg til drone' with three groups of fields:
  - Logo, Drone navn, Endre innstillinger for Droner, Vekt, Maks takoff vekt, Flygetid
  - Logo, Drone navn, Endre innstillinger for Droner, Vekt, Maks takoff vekt, Flygetid
  - Logo, Drone navn, Endre innstillinger for Droner, Vekt, Maks takoff vekt, Flygetid
- Right screenshot:** Shows fields for 'Navn' and 'Type'. Below is a section titled 'Legg til drone' with three groups of fields:
  - Vekt, Maks takoff vekt, Flygetid
  - Kan ta foto, Kan ta video, Kan ta IR, Har andre sensorer
  - Vekt, Maks takoff vekt, Flygetid
  - Kan ta foto, Kan ta video, Kan ta IR, Har andre sensorer

Both screenshots include a 'Tilbake' (Back) button at the bottom left and a red 'Avbryt' (Cancel) and green 'Opprett' (Create) button at the bottom right.

1.5: Side for å legge til drone, med info om registrering av droner som har funksjonaliteter av militært nivå

## 6.2.2 Prototype 2

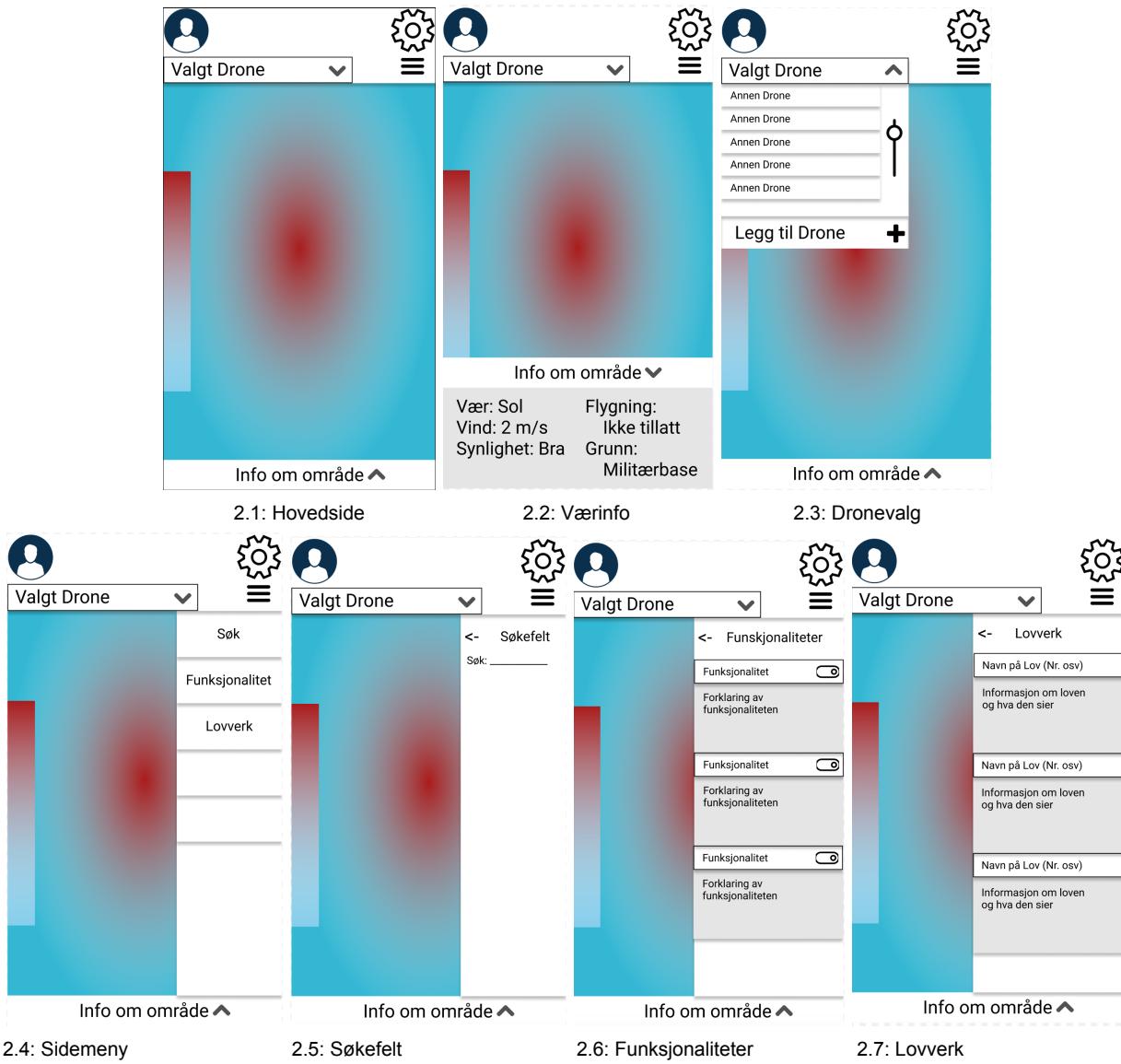
I Prototype 2 var tanken å ha kartet sentralt til enhver tid, i tillegg til å bruke farger, hvor blå viser til godt område og rød viser til dårlig område<sup>9</sup>. Menyen i denne prototypen i kontrast med forrige er uten den tredelte side-fordelingen, men denne har også en

9

brukerprofil og en innstillingsside. Bilde 2.1 viser hovedsiden, hvor det blå og røde området i midten skal forestille kartet med fargevariasjon, og med en bar med fargespekteret på venstre side. På bilde 2.2 Vises værinformasjon og eventuell restriksjon på området ved et pop-up vindu fra bunnen. Ved denne fremstillingen kunne man da se kartet samtidig som værinformasjonen. I denne prototypen hadde vi også med videre funksjonaliteten at brukeren kunne velge og legge til droner, men gjorde den mindre fremtredende og la funksjonaliteten ved å legge til droner på samme sted. I denne prototypen ville vi få med like mye funksjonalitet, men fremstille det slik at brukeren kan få det inn fra siden ved samme type meny i hjørnet som på prototype 1, for å frigjøre mer plass på hovedsiden, slik at kartet skulle få mer oppmerksomhet. Dette oppsettet var inspirert av funksjonalitet vinduet i prototype 1. Denne menyen ser man på bilde 2.4. Her ble det i prototypen lagt inn tre felter; søk, funksjonaliteter og lovverk. Søkefeltet ser man på bilde 2.5, hvor tanken var mye av den samme som på prototype 1. På bilde 2.6 ser man funksjonalitets vinduet som er den samme som i prototype 1, med unntak av at man her aktiverte funksjonaliteten med en av-på bryter i stedet for å “Legge til” funksjonalitet. I bilde 2.7 ser man et oppsett til lovverk lignende tanken om informasjons siden i prototype 1, men her i en cardview format.

Link til prototype 2:

<https://www.figma.com/proto/uACq9heYkPdLncnVTcHrOH/Prototype-2?scaling=min-zoom&nod e-id=1%3A4>



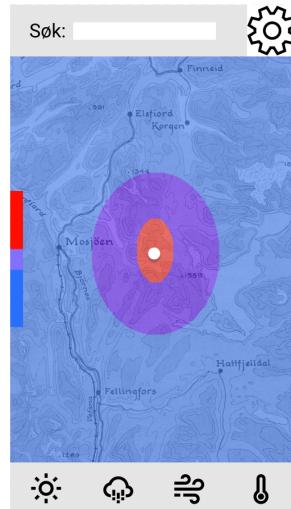
### 6.2.3 Prototype 3

I prototype 3 gikk vi helt bort fra tanken om ekstra funksjonaliteter og prøvde å fremstille appen så enkel og direkte som mulig. Bilde 3.1 viser hovedsiden til appen, hvor fargebruken er lignende prototype 2, bare her med klarere grenser for områdene, og med et kart under sånn at man klart ser hva det skal være, i tillegg til en hvit sirkel som skal vise til brukerens posisjon. I denne prototypen har vi beholdt søkefeltet, som er plassert tydelig øverst på skjermen, sammen med en knapp til innstillingens side. For å

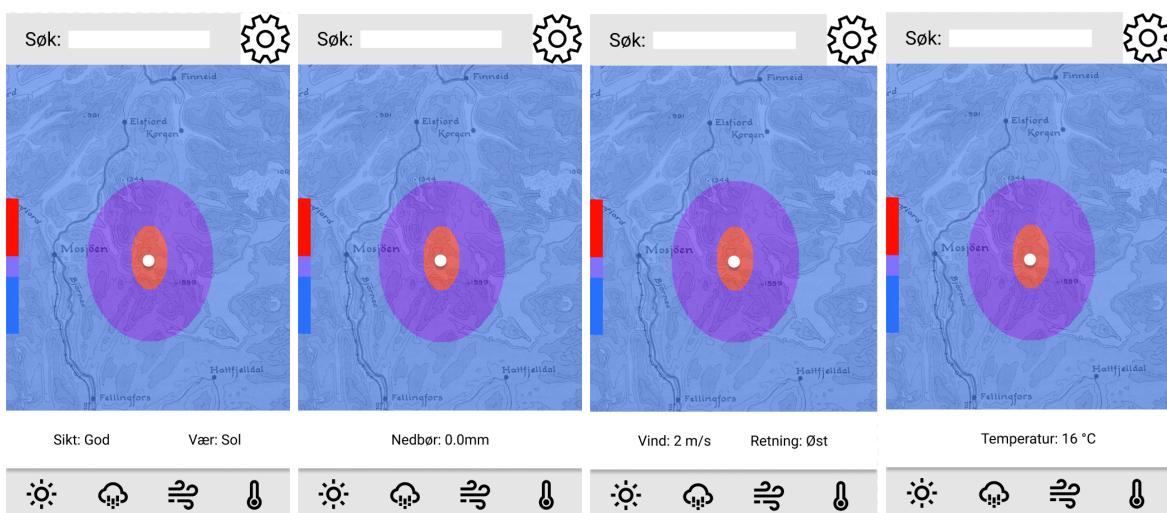
fremstille værdata fra brukerens område la vi i denne prototypen inn en rekke intuitive knapper i bunnen av skjermen. Bilde 3.2 viser informasjon om sikten i området og hva slags vær det er, om den sier at sikten er dårlig kan det for eksempel skyldes tåke eller tett regn. Bilde 3.3 viser nedbør i mengde. Bilde 3.4 viser vindhastighet samt vindens retning. Bilde 3.4 viser temperaturen i området.

Link til prototype 3:

<https://www.figma.com/proto/0RHiyqobtwGAktRulyrs4i/Prototype-3?node-id=1%3A2&scaling=scale-down>



3.1: Hovedside

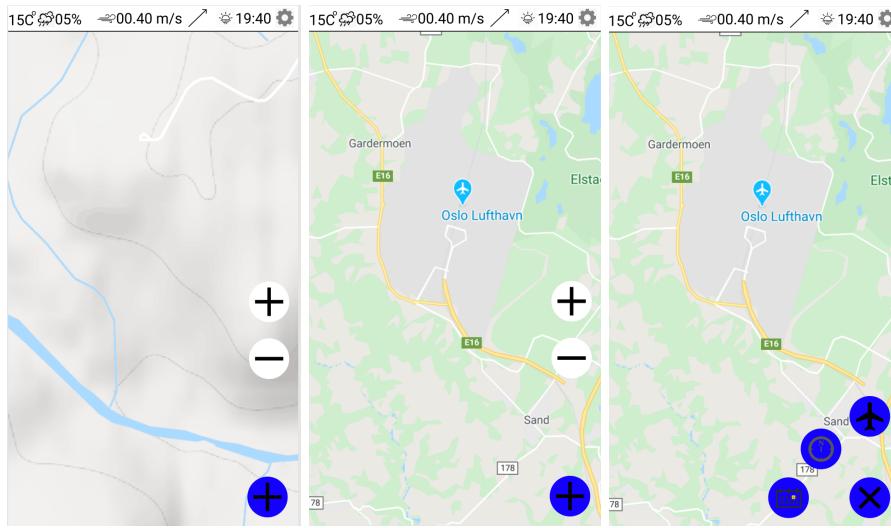


## 6.2.4 Prototype 4

I prototype 4 ble det fokusert på restriksjon av brukerens muligheter til å interagere med appen, samtidig som de få mulighetene skulle vise mye informasjon. På bilde 4.1 og 4.2 ser man forsiden av denne prototypen, hvor 4.1 er zoomet inn, og 4.2 er zoomet ut, som man kunne styre med pluss og minus knappene på høyre side. I denne prototypen gjorde vi også værinformasjonen enda lettere tilgjengelig, ved å legge dette i et informasjonsfelt øverst på skjermen, hvor det er vist informasjon om temperatur, nedbør, vindhastighet, vindretning og solnedgang. Hvor vi i prototype 3 fjernet både funksjonaliteter og lovverk sidene, fjernet vi her det samme i tillegg til søkefunksjonen, igjen for å ha begrenset brukerinteraksjon. Den blå knappen med plussstege nede i høyre hjørne åpnet en knappemeny, som man ser i bilde 4.3. Denne menyen består av tre knapper: no-fly, vanlig kart og schedule. No-fly knappen aktiverte laget som viste ulovlige flysoner på kartet, samt risikoområder, som man ser på bilde 4.4. Schedule knappen åpnet et felt for informasjon om området, samt steder av interesse i nærheten eller hvor brukeren har brukt appen tidligere, som vist i bilde 4.5. I bilde 4.6 ser man en side for innstillinger, hvor brukeren kan justere preferert toleranse for vind, tåke og flygetid, fra null opp til det lover og reguleringer sier er forsvarlig å fly. Det var også lagt opp til at brukeren kunne endre farge på knapper, ulovlig sone og risiko sone. Denne innstillingssiden er tilgjengelig via en knapp øverst i høyre hjørne.

Link til prototype 4:

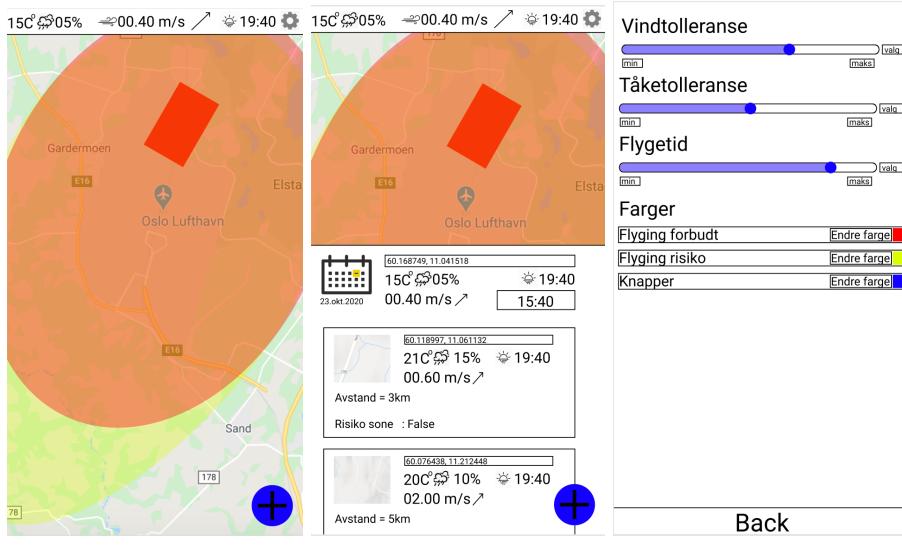
<https://www.figma.com/proto/JmVrlHdfzsdk7qtGobvBY/Prototype-4?scaling=contain&node-id=6%3A586>



4.1: Hovedside zoomet inn

4.2: Hovedside zoomet ut

4.3: Åpnet knappemeny



4.4: No-flight sone

4.5: Info om området

4.6: Innstillingar

## 6.2.5 Prototype 5

Vi benyttet tilbakemeldingen vi fikk av de to resterende medlemmene av fokusgruppa til å trekke inn hovedpunktene fra de prototypene. Ved å samle sammen hva som var positivt og hva som var negativt fikk vi opprettet prototype 5 som da på dette punktet er en miks av prototype 4 og 3.

Endringene fra prototype 4 var et søke ikon tydelig på informasjons panelet øverst på appen. Dette var noe det var enighet om at var en nyttig egenskap. Vi baserte derfor denne funksjonen på vindu 4.5 fra prototype 4 (se over) ettersom denne allerede var opprettet og utenkt. Dette vinduet kan derfor sees på figur 5.1



5.1 Søkevindu

## 6.3 Værinformasjon

Gjennom prosessen med diverse prototyper var en av hovedpunktene hvordan vi skulle fremstille værinformasjon. I de forskjellige prototypene har vi derfor prøvd på forskjellige fremstillinger og visninger av denne informasjonen, for igjen å få tilbakemelding fra brukergruppen om hvilken fremstilling de likte best.

I prototype 1 hadde vi ikke visning av værinfo på hovedside, og heller ikke på området man fikk fra brukerens lokasjon, men i stedet kunne få informasjon gjennom områdesøk. På dette fikk vi tilbakemelding fra brukergruppen at det var svært uoversiktlig og ikke visste hvordan de skulle få info om sitt område, så denne fremstillingen ble forkastet i videre utvikling av pre-alfa (prototype 5). Fremstillingen vist på bilde 6.1.

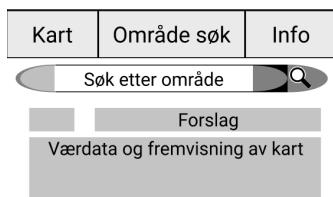
I prototype 2 var tanken å flytte informasjonen til et mer sentralt og lett tilgjengelig felt på skjermen. Tanken var der å ha et pop-up vindu nederst på skjermen som skulle vise værinformasjon for område gitt ved brukerens lokasjon, som vist på bilde 6.2. Det var blandet tilbakemelding for denne måten å fremstille informasjonen på fra brukergruppen. Generelt fikk vi tilbakemelding på at det kunne funke og var greit å ha det tilgjengelig på forsiden, men at det ikke helt kunne måle seg med fremstillingen fra prototype 4.

I prototype 3 hadde vi hovedsakelig samme tanke som i prototype 2, at man skulle ha informasjonen tilgjengelig på startsiden, men ikke påtrengende. Dette var også en fremstilling som vi kom fram til etter å ha diskutert prototype 2 litt innad i gruppa. Her var det i kontrast tanken at brukeren kunne trykke på konkrete ikoner nederst på skjermen for å få ut spesifikk værinformasjon, som vist på bilde 6.3. I prototypen ble det dette vist med ikoner for sikt, regn, vind og temperatur, hvor man kunne få informasjon om gitte typer værinformasjon ved å trykke på ikonet, bytte ved å trykke på et annet ikon, og fjerne ved å trykke på ikonet igjen. Brukergruppen hadde mer eller mindre samme tilbakemelding til denne fremstillingen som til prototype 2, hvor denne ble sett på som litt for avansert for fremstilling av enkel informasjon.

I prototype 4 gikk vi bort ifra å ha informasjonen relativt lett tilgjengelig men fortsatt skjult, til å vise den direkte på toppbaren, som vist i bilde 6.4. Tilbakemeldingene bra brukergruppen var svært positive til dette, grunnet at informasjonen alltid var lett tilgjengelig og de ikke trengte å gjøre noe for å hente den ut.

I prototype 5, pre alfa, utviklet vi videre fremstillingen fra prototype 4, siden den fikk desidert best tilbakemeldinger. Vi lagde toppbaren tykkere, for å få mer plass til informasjon, og for at det skulle se mer oversiktlig ut. I tillegg til dette la vi inn en knapp for søkefunksjonen i toppbaren, i venstre hjørne, som skulle speile posisjonen til knappen for innstillinger. Dette vises i bilde 6.5.

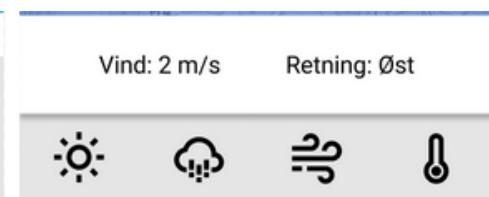
I den endelige versjonen la vi inn mer informasjon, i form av soloppgang og solnedgang. Det ble også lagt inn eget tegnede ikoner for all informasjon, innstillinger og søker. Dette er vist i bilde 6.6.



6.1: prototype 1, via område søker



6.2: prototype 2, via pup-up nederst



6.3: prototype 3, via ikoner nederst



6.4: prototype 4, via toppbaren



6.5: prototype 5, pre alfa, videreføring av prototype 4, med søker

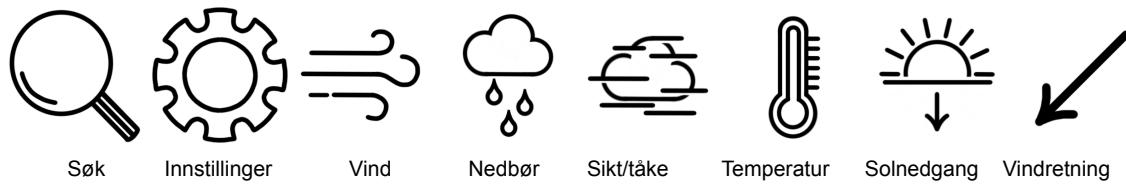


6.6: endelig resultat, med egne ikoner

Vi var usikre på om vi kunne bruke diverse gratis bilderessurser i prosjektet vårt. Ifølge University of Notre Dame, Australia<sup>10</sup>, så kan man bruke bilder hentet fra internett om det bare er ment for studenter og universitetets ansatte. Ved publisering kan det forekomme kostnader for bruk også trenger man tillatelse fra parten som har opphavsrett, og parten som har de morale rettighetene. På den andre siden kan man også bruke bilder og lignende som er offentlige, hvor opphavsrett ikke er et problem. Vi tenkte å være på den sikre siden, og lagde våre egne ikoner for værdata, søker og

<sup>10</sup> <https://library.nd.edu.au/copyright/works/imv/images>

innstillinger. Ikonene ble lagd på Ipad, med programmet Autodesk SketchBook og en disk-penn.



Etter alle disse implementasjonene ble gjort sendte vi denne fungerende prototypen til fokusgruppen for testing av funksjonalitetene. Ettersom gjennomføring av brukertesting gjennom “wizard of oz” metode ikke var et alternativ grunnet helsemessige forutsetninger ble dette den første brukertestingen.

## 7 Universell utforming

Universell utforming er en form for tilrettelegging av hovedløsningen i de fysiske forholdene slik at denne kan brukes av flest mulig. Helt siden vi begynte med utviklingen av prosjektet har vi hatt fokus på universell utforming. Vi mener at vår applikasjon kan brukes av de fleste brukere uavhengig av teknisk bakgrunn og fysiske evner.

Vi innså tidlig at å innføre flere constraints og gjøre grensesnittet mer minimalistisk kan vi gjøre bruken av appen mye enklere. I begynnelsen gjorde vi antagelsen at å ha en dedikert zoom inn og knapp vil gjøre zoom lettere med en hånd. Dette ville tillate brukeren å beholde kontroll over en eventuell drone mens han/hun fløy. Etter å ha implementert en rekke funksjoner oppdaget vi at mapbox (vår kartjeneste) har dobbeltklikk funksjonalitet. Dette gjør en dedikert zoomknapp overflødig. Mulig at det ville gi større “affordance”, men siden vi er ute etter et minimalistisk grensesnitt virket det som en logisk løsning. Dette er ytterligere forsterket av det faktum av at vi ikke fikk implementert forbudssoner fra NSM. På dette punktet er det ingen informasjon som krever at brukeren skal betjene appen under flyvning. Med dette påpeker vi at all informasjon appen gir under flyvning vil lett observeres uten å berøre telefonen. Derfor

konkluderte vi med å gå bort fra zoom knappene som kan observeres på noen prototyper.

Vi ser også at ved å ha ulike grader og typer fare og forbudssoner, gir det mening å separere disse visuelt ved å utheve dem i ulik farge. For å forenkle bruk for fargeblinde, valgte vi å se bort ifra statisk definerte fargevalg og heller la brukeren selv justere fargene i appen. Dessverre frafalt dette ved frafall av NSM sine forbudssoner.

For den gjenværende sonen ble rød valgt ut ettersom den er universalt sett på som en farge for farlige soner og områder.

I innstillingsmenyen kan piloten justere toleranser for vind, regn og tåke. Dette gjorde vi for at piloten skal kunne tilpasse appen etter sitt ferdighetsnivå.

Selv utformingen er ikke spesialisert for funksjonshemmede. Dette er til dels på bakgrunn av vår antagelse av brukerens behov. Vår antagelse av appen dekker brukerens behov for selvrealisering, her basert på Maslows behovspyramide<sup>11</sup>.

Vi gjorde tidlig en antagelse som gikk ut på at personer med alvorlig synsnedsettelse ikke har noe nytte av en app som værdata for dronepiloter. Norske myndigheters krav for å operere en drone er at piloten skal ha oversikt og kunne se dronen til en hver tid. Vi har derfor ikke gjort noen tilpasninger i appen vår, for de svaksynte.

## 8 Refleksjon

Korona-tiden har skapt litt utfordringer med prosjektet. Vi fikk bare til å møtes fysisk en gang før universitetet ble stengt, og det er naturlig å tenke seg at hvis man hadde møttet fysisk flere ganger så hadde samarbeidet blitt enda mer effektivt. Men på tross av det, føler vi at vi har fått til et godt samarbeid. Vi har hatt ukentlige møter på Discord,

---

<sup>11</sup> [https://no.wikipedia.org/wiki/Maslows\\_behovspyramide](https://no.wikipedia.org/wiki/Maslows_behovspyramide)

kommunisert bra på slack, og benyttet oss av Jira for å fordele arbeidsoppgaver. Vi føler at alle har bidratt og gjort det de skal.

Utviklingsprosessen gikk relativt bra. Vi jobbet uten strenge tidsfrister frem til slutten av mars. Dette gjorde at teammedlemmene ble godt kjent med hverandre, ble klare over ferdighetsnivå og kode/skrive-stil. Fra slutten av mars / starten av april begynte vi med ukentlige sprinter, hvor vi satt nye mål for hver uke. Det ble også satt et mål om å bli helt ferdig med prosjektet før den 12. mai.

Grunnet en stor hjemmeeksamen for 4 av de 6 medlemmene i slutten av april og starten av mai. Dette gjorde at fokuset på appen ble litt nedprioritert i den perioden, og målet om å være ferdig til 12 mai ble utsatt med en uke. I løpet av den siste uken jobbet vi mye sammen for å løse en rekke små, men kritiske problemer som gjenstod å fikse for å gjøre appen bra nok til å kunne publisere appen til fokusgruppa for testing.

Vi oppdaget også en annen utfordring. Vi var ikke klar over var at vi ikke var så flinke til å si ifra dersom man satt seg fast og trengte hjelp. Etter at vi diskuterte dette i gruppa fikk vi jevnet ut noen av disse problemene. Dette førte til at vi ble mye bedre etterhvert, etter at vi opprettet en slack-kanal for "direkte-hjelp" og begynte å bruke den. Etter et par uker var vi mye bedre på kommunikasjon. Dette førte til mer aktivitet i discord-kanalen vår, og det ble mer vanlig at det alltid var noen på og kodet i fellesskap slik vi gjorde før alt stengte ned.

Innimellom retrospektive møtene benyttet vi standuply<sup>12</sup> for daglige stand up oppdateringer fra mandag til søndag. Ettersom man ikke jobbet med faget hver dag så rapporterte man kun dersom man hadde noe nytt å melde. Denne sviktet imidlertid rundt starten av mai. Vi fikk likevel uformelle, men informative, standup møter. Ettersom

---

<sup>12</sup> <https://standuply.com/>

vi snakket daglig på discord. Derfor ble det et naturlig retrospektivt møte for å finne hvilke problemer vi skulle takle den dagen.

I de siste ukene før innlevering ble det et enda mer intenst fokus på rapportarbeid, app og fokusgruppe. Dette førte til en nærmest daglig kontakt over discord og slack. Det begynte med et retrospektivt møte, dette gikk naturlig over i en diskusjon om dagen oppgaver. Vi snakket sammen mens en av oss delte skjerm og diskuterte oppgaven. Med øvelse ble dette nesten en like god arbeidsmetode som å jobbe felles i grupperom. Disse arbeidsintervallene var ofte på mellom seks til åtte timer med konsentrert arbeid og diskusjon. Vi hadde, ikke overraskende, en mer skunkworks<sup>13</sup> rettet arbeidsmåte ved å jobbe så tett og intensivt.

Denne arbeidsperioden var kanskje den mest effektive. Mye av det skyldes vår kunnskap om hverandres styrker og hvordan vi kunne på dette stadiet fordele oppgaver mer effektivt. Vi merket at vi ble mer synkroniserte for hver dag, vi kunne derfor lene oss på hverandres kunnskap. Retrospektiv møtene ble satt i gang enkelt og sømløst.

Dersom vi skulle fortsette å arbeide videre sammen på dette punktet vil nok denne tilliten og kommunikasjonen forbedre seg ytterligere. På et vis har vi vært heldige med pandemien, ettersom det har satt en så stor last på samarbeidsmulighetene at vi måtte stadig innovere. Til tross for disse vanskelighetene, viste vi evnen til god kommunikasjon, samarbeid og fikk fastslått viktigheten av kommunikasjon innad i en gruppe under et slikt samlet prosjekt. Det hjalp spesielt retrospektive møter, noe vi merket lot oss oppdage problemområder lett og raskt.

---

<sup>13</sup>

<https://www.lockheedmartin.com/en-us/who-we-are/business-areas/aeronautics/skunkworks/kelly-14-rules.html>

## 9 Kilder/Vedlegg

Kilder:

<https://www.w3.org/TR/WCAG21/>  
<https://lovdata.no/dokument/SF/forskrift/2015-11-30-1404>  
<https://lovdata.no/dokument/LTI/forskrift/2018-12-05-1807>  
<https://luftfartstilsynet.no/dronelek/>  
<https://www.datatilsynet.no/personvern-pa-ulike-områder/overvaking-og-sporing/droner---hva-er-lov/droner/>  
<https://avinor.no/konsern/pa-flyplassen/droner/generelt>  
<https://library.nd.edu.au/copyright/works/imv/images>  
<https://www.lockheedmartin.com/en-us/who-we-are/business-areas/aeronautics/skunkworks/kelly-14-rules.html>  
<https://docs.mapbox.com/android/maps/overview/>  
<https://www.geeksforgeeks.org/android-how-to-request-permissions-in-android-application/>  
<https://docs.mapbox.com/android/maps/overview/camera/#center-the-camera-within-a-map-area>  
<https://docs.mapbox.com/android/maps/overview/location-component/>  
<https://developer.android.com/training/location/retrieve-current.html>  
<https://developer.android.com/about/versions/oreo/background-location-limits>  
<https://docs.mapbox.com/android/maps/overview/query>  
<https://docs.mapbox.com/android/plugins/overview/places/>  
<https://github.com/mapbox/mapbox-gl-native/issues/2167>  
<https://docs.mapbox.com/vector-tiles/reference/mapbox-streets-v8>  
<https://www.journaldev.com/337/android-seekbar-using-kotlin>  
<https://developer.android.com/guide/topics/ui/floating-action-button>

Vedlegg:

Spørreskjema:

<https://docs.google.com/forms/d/1UEIY59cIn0XvGCxkCZBXT7JJ38Si3xcaCy8PVvvvcqUg/edit>

Eksempler på kildekode:

Xml for WeatherView:

<https://docs.google.com/document/d/18Afqqiv3ISd8gCLKhJ7QLHZZ5eqPLwagrH-i1WD5dmY/edit?usp=sharing>